CAR: Context-aware Adaptive Routing for Delay Tolerant Mobile Networks

Mirco Musolesi, *Member, IEEE Computer Society,* and Cecilia Mascolo, *Member, IEEE Computer Society*

***Abstract*— Most of the existing research work in mobile ad hoc networking is based on the assumption that a path exists between the sender and the receiver. On the other hand, applications of decentralised mobile systems are often characterised by network partitions. As a consequence delay tolerant networking research has received considerable attention in the recent years as a means to obviate to the gap between ad hoc network research and real applications.**

In this paper 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 delivery. The choice of the best carrier is made using Kalman filter based prediction techniques and utility theory. The large scale performance of the CAR protocol are evaluated using simulations based on a social network founded mobility model, a purely random one and real traces from Dartmouth College.

***Index Terms*— Wireless communication, routing protocols**

I. Introduction

Mobile ad hoc network research [1] has often assumed that a connected path exists between a sender and a receiver node at any point in time. This assumption reveals itself unrealistic in many decentralized mobile network applications such as vehicular networks, wildlife monitoring sensor networks, deep space communication systems and emergency operations networks. To answer this dichotomy, delay tolerant networking (DTN) [2] has received considerable attention from the research community in recent years as a means of addressing exactly the issue of routing messages in partitioned networks.

DTNs span very challenging application scenarios where nodes (e.g., people, wild animals) move around in environments where infrastructures cannot be installed (e.g., emergency operations, military grounds, protected environments). Some solutions to routing have been presented also for these cases, starting from the basic *epidemic routing* [3], where messages are blindly stored and forwarded to all neighbouring nodes generating a flood of messages. The drawback of epidemic dissemination lies in the very high number of messages which are needed to obtain successful delivery to the right recipient. Other 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 [4]–[6] or on probabilistic approaches [7], [8].

Mirco Musolesi is with the Institute for Security Technology Studies at the Department of Computer Science, Dartmouth College, USA, and Cecilia Mascolo is with the Computer Laboratory, University of Cambridge, UK.

In this paper we present the Context-aware Adaptive Routing (CAR) protocol, an approach to delay tolerant mobile ad hoc network routing which uses prediction to allow the efficient routing of messages to the recipient. A host willing to send a message to a recipient, or any host in the multi hop path to it, uses a Kalman Filter prediction and multi-criteria decision theory [9] to choose the best next hop (or carrier) for the message. The decision is based on the mobility of the host (a highly mobile host is a good carrier as it meets many hosts) and its past colocation with the recipient (we implicitly assume that past colocation indicates that the host will meet the recipient again in the future). CAR does not assume any previous knowledge of the routes of the hosts like other approaches, such as the Message Ferrying project [5], 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 [7], 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.

An earlier version of the protocol with a limited evaluation has been presented previously in a symposium paper [10]: we have extended that paper with additional protocol details, an implementation and a thorough performance evaluation with a new mobility model validated using real traces provided by Intel [11]. Our approach can be considered the first one exploiting forecasting techniques for carrier selection founded on analytical prediction models.

This paper is organised as follows. Section II presents the design of the CAR protocol. The performance evaluation of the protocol is discussed in Section III. A comparison with the state of the art is presented in Section IV. Section V summarises the contribution of this work.

II. Design of the CAR Protocol

A. Overview

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 describe the general steps of the protocol. Secondly, we analyse the prediction theory and its foundation algorithms. Thirdly, we discuss the protocol implementation, focussing on the management of routing information for synchronous and asynchronous delivery.

As first step, we introduce the assumptions underlying 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.

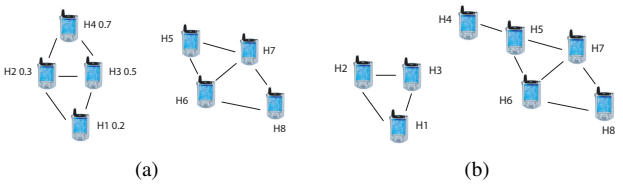
The design goal of CAR is to support communication in intermittently connected mobile ad hoc networks. The key problem solved by the protocol is the selection of the carrier. Our solution is based on the application of forecasting techniques and utility theory for the evaluation of different aspects of the system that are relevant for taking 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 when there are no network partitions between sender and receiver) and asynchronously (i.e., by means of a store-and-forward mechanism when there are partitions). 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 protocol.

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(a). As in our implementation, let us assume that Dynamic Destination-Sequenced Distance-Vector (DSDV) [12] 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 1(a). 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 1(b)). Since a connected path between H_4 and H_8 now exists, the message is delivered to its intended recipient. Using DSDV, for example, 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*. 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 rate 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

[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.



(a) (b)

Fig. 1. Two connected clouds, with associated delivery probabilities for message transmission between H_1 and H_8 (Fig. a). Then, H_4, carrying the message, joins the second cloud (Fig.b).

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.

B. Prediction and Evaluation of Context Information

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

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 calculation of utilities for each attribute describing the context. Then the future values of these utilities are predicted (see Section II-H.1) and composed using multi-criteria decision theory [9] (see Section II-C) in order to estimate an overall delivery probability. 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

[2]We will discuss the management of this set of probabilities in Section II-F.

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.

C. Local evaluation of context information

Each host calculates its delivery probability locally, given observations related to the various context attributes. Therefore, the key problem is to *measure* and *combine* the attributes. The delivery probabilities are calculated by evaluating the utility of each host as potential carrier for a message.

There are several techniques for assigning an overall utility given the multiple dimensions of the context. A possible 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 so that the optimum value of a higher priority goal is never degraded by a lower priority goal [13].

However, this technique is too simplistic because, in general, our decision problem involves multiple conflicting objectives [9]. 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.

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 case of mutually preferentially independent attributes $X_1, X_2, ...X_n$, 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:

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

where U_i is a utility function over X_i.

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. To solve this problem, we apply the so-called *Weights method* [9]. 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.

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

D. 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 [14]. 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 II-H.1. 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) [15], we also consider an attribute describing the battery level, a fundamental aspect for that specific application domain. Other possible context dimensions are memory availability or group membership (i.e., two hosts of the same social group are more likely to be colocated).

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.

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.

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

[4]In the remainder of this article, 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, $U_{cdc_h}(x_{cdc_h}(t)) = U_{cdc_h}(t)$.

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 Section III 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.

E. 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 [16]. 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.

a) 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.

b) 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 [17].

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

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 II-H.

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.

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

Formally, to date, we have implicitly assumed that a static set of attributes is defined. However, 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).

F. 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.

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, dist, 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.

These routing tables are used both for synchronous and asynchronous delivery: 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 [18]. 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).

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 [19]. This technique alleviates the CAR scalability issue in terms of routing table size.

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 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 note that the size of the routing table is limited, since to every entry is associated a Kalman filter based predictor that has to be updated periodically.

3) 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. 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; 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 (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. The update interval of the Kalman filter is set to the routing table transmission interval in our work.

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 Section III.

G. Message Delivery

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 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 II-G.3.

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). In order to avoid the propagation of stale routes, we use sequence numbers for the routing tables like in DSDV.

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 II-G.3).

3) (Re-)transmissions: 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 III-B.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 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.

H. Prediction of the Context Information Attributes using Kalman Filter Prediction 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 II-B. We used Kalman filter forecasting techniques [14] 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.

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*[5].

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 [19]. 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 *addendum* of this paper [20], 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. We also discuss how the Kalman filter model that we use in CAR can be recast and studied using alternative theoretical frameworks, namely EWMA, ARMA and Bayesian forecasting models [19].

2) 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 [2]. 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; there are situations, however, where context cannot be predicted.

[5]These equations are linear and the complexity is proportional to the number of hosts in the routing tables like in PRoPHET [7].

Variables

- $z(t)$: time series of residuals (prediction errors)
- r_k: autocorrelation coefficients
- k_{max} max lag
- N: number of samples
- $maxError$: max acceptable prediction error

boolean **isPredictable** $(U(t-NT)...U(t), \widehat{U}(t-NT)...\widehat{U}(t))$ {
for all $\tau \in [t-nT, t]$ **do**
  calculate $z(\tau) = \widehat{U}(\tau) - U(\tau)$
for all $k \in [1..K_{max}]$ **do**
  calculate r_k of $z(t)$
if ((90% of $\{r_k\} \in [\frac{-2}{\sqrt{N}}, \frac{2}{\sqrt{N}}]$) and $(0.1z(t) \leq maxError)$) **then**
  return true;
else
  return false;
}

Fig. 2. Algorithm for verifying the predictability of the time series.

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 presented in [21]. The technique that we adopted is predicated on the analysis of the time series representing the context information and, more specifically on residual analysis [19]. Given a certain number of measurements of the predictability of the 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. The algorithm used by the prediction component to calculate the predictability of the time series is shown in Figure 2. In our experiments, we consider the predictability of the time series of the colocation between pair of hosts; every sample of the time series is evaluated for the calculation of the predictability level. If the predictability level is under a given threshold, alternative protocols can be used, for example epidemics-inspired approaches [3].

III. Performance Evaluation

A. 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.

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 to 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. We implemented our simulation scenario using OMNet++ [22].

1) Simulation Scenarios Parameters: We considered three simulation scenarios composed of 50 in 1 $km \times 1\ km$, 50 in 2 $km \times 2\ km$, 100 in 2 $km \times 2\ km$ areas in order to study the performance of the protocols with scenarios characterised by different degrees of sparseness and number of hosts.

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 Waypoint mobility

model [23], generate purely random movements that are very different from the ones observed in the real world and produces meaningless colocation patterns[6].

For this reason, we have used a mobility model based on social network theory, the Community based mobility model [11], [24]. 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.

The key problem is the generation of a synthetic social network with realistic characteristics in terms of clustering and average path lengths between the members of communities (i.e., clusters of nodes present in the social network). Our approach is based on the so-called Caveman model proposed by Watts in [25] to generate a social network characterised by a realistic clustering degree. The social network is built starting from a certain number of 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. A weight modelling the importance of the relationship between two individuals is associated to each link (i.e., edge of the graph) of the network; edges between individuals of the same community are higher than the others.

The simulation area is divided into a grid formed by a certain number of squares. Each group detected using the clustering algorithm is then placed in one of these squares. Each host moves following the Random Way Point model inside each square, until it reaches its goal. The next goal is chosen inside the square associated to the group of hosts that exerts the highest "social" attraction towards it (including the current one). This group attraction is calculated by summing the values that express the intensity of the relationship between the hosts and the members of the community (extracted from the matrix that describes the social network).

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 Waypoint model in the same simulation scenarios, with host speeds in the range $[1 - 6]$ m/s and stop times equal to 0 seconds. 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.

[6]However, 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 likelihood of establishing communication with the recipient of the message.

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 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, 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. 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.

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

2) Choice of Parameters of CAR: We implemented all the features of CAR described in Section II 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 w_{cdc_h} and $w_{col_{h,i}}$ for all the pairs 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 III-B.4.

Each message has a *time to live* field that is decreased each time a message is transferred 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. The buffer for each node was set to 100 messages, unless otherwise specified.

The number of retransmissions 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.

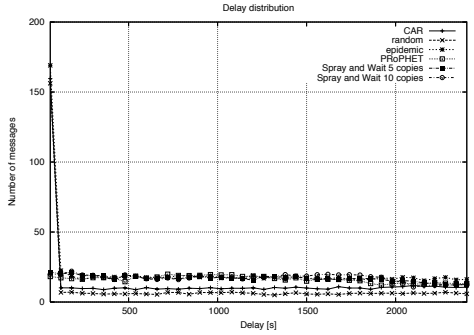


Fig. 3. Delivery delay distribution (scenario with 50 hosts, 1 $km \times 1$ km area and Community based mobility model).

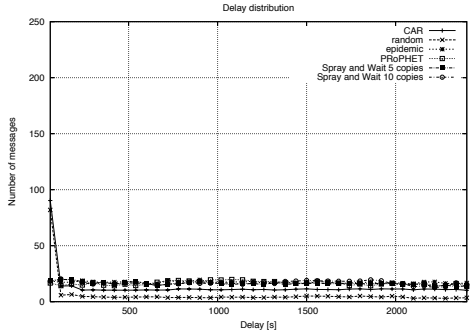


Fig. 4. Delivery delay distribution (scenario with 100 hosts, 2 $km \times 2$ km area and Community based mobility model).

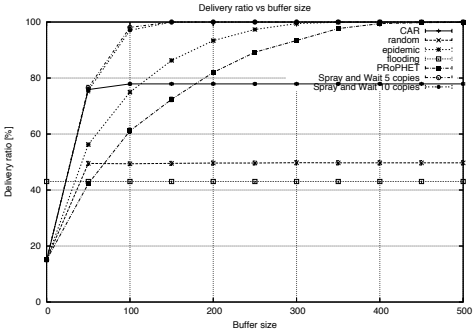


Fig. 5. Delivery ratio vs buffer size (scenario with 50 hosts, 1 $km \times 1$ km area and Community based mobility model).

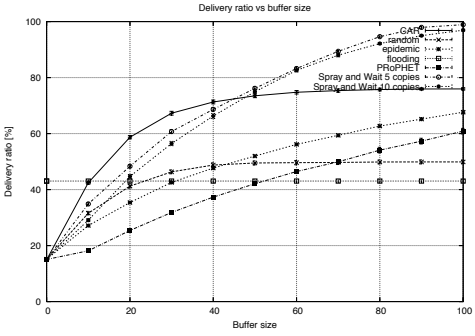


Fig. 6. Delivery ratio vs buffer size – analysis with small buffers (scenario with 50 hosts, 1 $km \times 1$ km area and Community based mobility model).

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 [20]). 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 [26]. The values that we selected are appropriate and general enough for the range of inputs (and its variations) that we are considering.

3) 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) [3] 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.

a) Flooding: We elected to compare our approach with flooding. 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 the CAR algorithm. In order to evaluate the difference in delivery rates that results from the ability of the different algorithms to handle partial connectivity, it is therefore essential to compare against flooding, the synchronous protocol with optimal delivery ratio in case of infinite buffers.

b) Epidemic Routing: The implementation of the epidemic protocol follows the description presented in [3]. The epidemic approach represents an example of an asynchronous protocol and, in particular, 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).

c) 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 (i.e., it is similar to a random walk). 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.

d) PRoPHET: PRoPHET is a semi-epidemic approach based on the calculation of the probability of replicating a copy of a message based on the frequency of encounters between pairs of hosts. Details of the protocols can be found in [7]. We used the values of the parameters suggested by the authors in their paper.

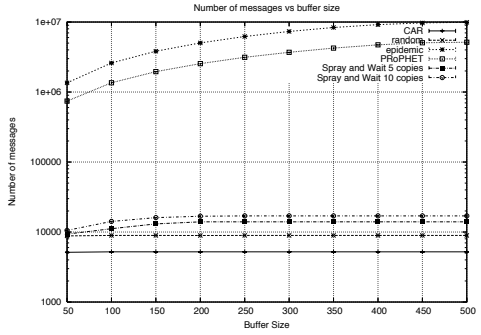


Fig. 7. Number of messages vs buffer size (scenario with 50 hosts, 1 km $\times$ 1 km area and Community based mobility model).

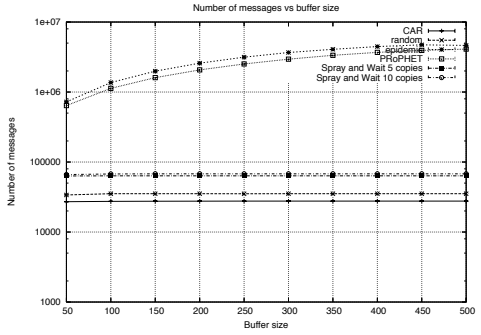


Fig. 9. Number of messages vs buffer size (scenario with 50 hosts, 2 km $\times$ 2 km area and Community based mobility model).

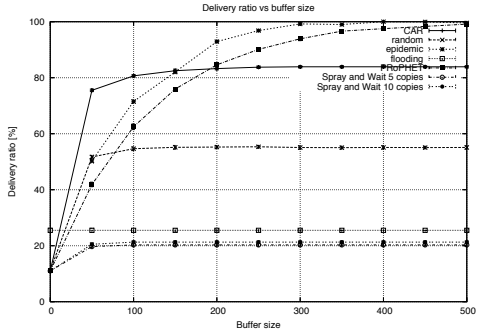


Fig. 8. Delivery ratio vs buffer size (scenario with 50 hosts, 2 km $\times$ 2 km area and Community based mobility model).

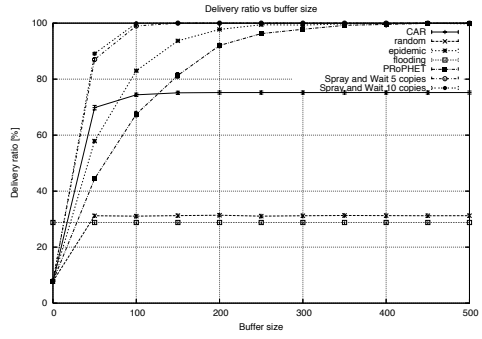


Fig. 10. Delivery ratio vs buffer size (scenario with 100 hosts, 2 km $\times$ 2 km area and Community based mobility model).

e) Spray and Wait: Spray and Wait [8] is based on the initial replication of a certain number of copies of the message (5 and 10 in our experiments). Then, these copies are not replicated further and are only forwarded to the recipient of the message. Each node advertises its presence every 30 seconds using a beacon message.

B. Simulation Results

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

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 only data messages. The number of control messages is equal for all the protocols since we use the same transmission interval for these messages.
- **Predictability level** 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 prediction error is under a certain margin, i.e., the prediction component returns true. We consider a prediction error of 0.15 as the maximum acceptable error in the current evaluation. This value was chosen considering the fact that the value of the variance of the observation $Y(t)$ in the prediction model is 0.1.

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 3 and 4 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 time passes. We assume a delay due to local processing (such as calculation of the next best hop) 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, PRoPHET and Spray and Wait in a synchronous way is lower than CAR. In fact, the message is replicated to the neighbouring nodes at each replication step that is equal to the retransmission interval of CAR. The synchronous delivery

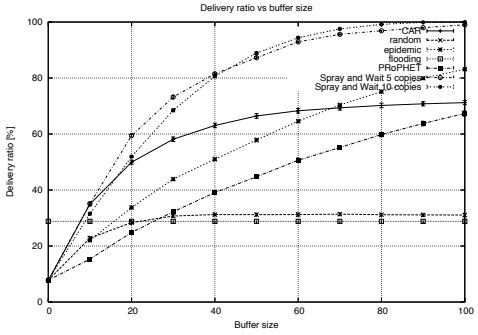


Fig. 11. Delivery ratio vs buffer size – analysis with small buffers (scenario with 100 hosts, 2 km × 2 km area and Community based mobility model).

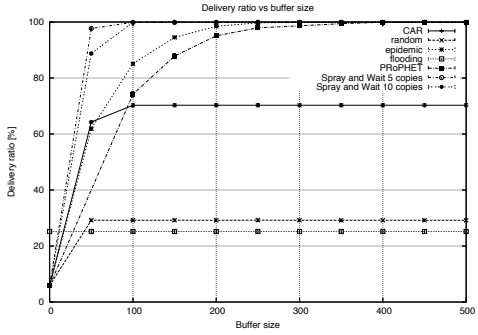


Fig. 13. Delivery ratio vs buffer size (scenario with 100 hosts, 2 km × 2 km area and Random Waypoint model).

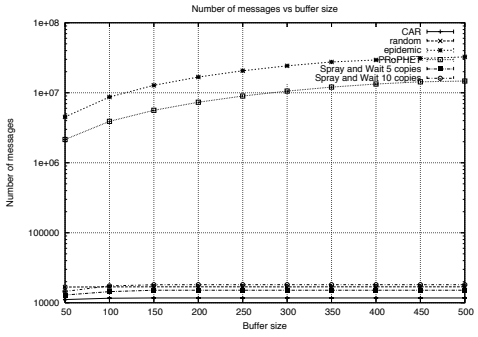


Fig. 12. Number of messages vs buffer size (scenario with 100 hosts, 2 km × 2 km area and Community based mobility model).

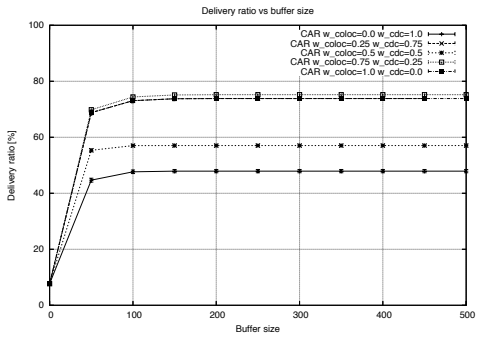


Fig. 14. Influence of the weights of the utility function on the delivery ratio (scenario with 100 hosts, 2 km × 2 km area and Community based mobility model).

mechanism in CAR allows for a more efficient routing in presence of connectivity, since a message is delivered immediately to the destination if a known connected route to it exists.

3) Influence of the Buffer Size: Another critical aspect we investigated is the size of the buffers of the hosts. Figures 5 and 10 show the impact of the buffer size on the delivery ratio in rather dense scenarios composed of 50 nodes in a 1000 m × 1000 m and 100 nodes in a 2000 m × 2000 m area respectively. 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 6 and 11.

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

Let us consider the scenario composed of 50 hosts in Figure 5, 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. CAR also outperforms Spray and Wait with buffer size lower than 20. This is due to the fact that the latter is a multiple copy routing scheme. At the same time, the overhead of Spray and Wait is almost double in terms of messages sent.

In fact, another key 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 7 and 12 respectively. CAR shows the best performance with respect to the other protocols. The number of forwarded messages by CAR is lower since it is a single-copy routing protocol. For this reason, the comparison with the Random Choice is the most significant one.

PRoPHET also suffers by the limitations of buffer size. We also note that the performance of PRoPHET are worse than epidemic; this is probably due to the fact that, since the message is not always retransmitted to the neighbours like in the epidemic protocol, the probability of infection is not sufficient to guarantee the spreading of the copies of the messages to the recipients.

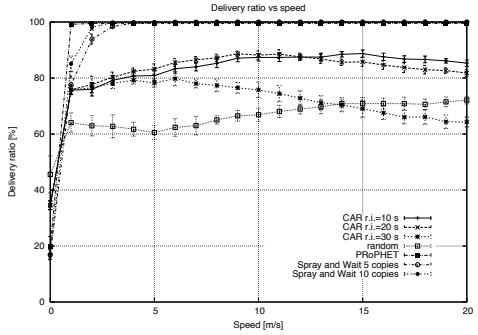


Fig. 15. Influence of the host speed on the delivery ratio (scenario with 50 hosts, 1 $km \times 1\ km$ area and Community based mobility model).

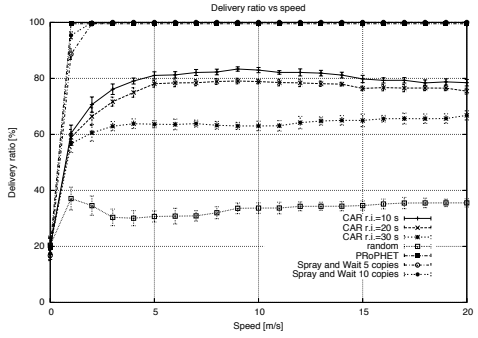


Fig. 16. Influence of the host speed on the delivery ratio (scenario with 100 hosts, 2 $km \times 2\ km$ area and Community based mobility model).

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 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 of the colocation time series for this scenario is about 85%. Moreover, as far as the CAR protocol is concerned, by analysing Figure 6, 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 10. 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 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. With respect to the performance of PRoPHET and Spray and Wait, we observe a similar trend, with CAR outperforming these two protocols in case of small buffers. The predictability level of this scenario is about 82%, a value close to the one measured for the 50 hosts scenario. In Figure 11 we can observe an asymptotic

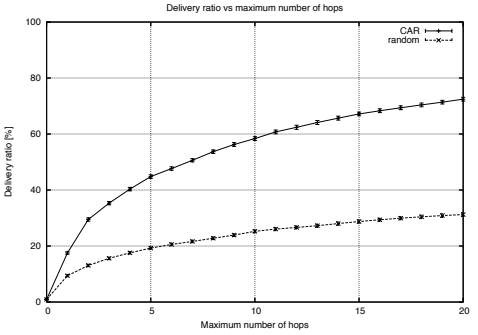


Fig. 17. Influence of the number of retransmissions on the delivery ratio (scenario with 100 hosts, 2 $km \times 2\ km$ area and Community based mobility model).

behaviour for values of the buffer size higher than 70 and 30 for CAR and the Random Choice protocol respectively.

Figures 8 and 9 show the performance in terms of delivery ratio and overhead of the protocols taken into consideration in a more sparse scenario composed of 50 nodes in 2000 $m \times 2000\ m$ with the same underlying social network structure. In this scenario, CAR outperforms Spray and Wait since in the latter the carriers are able to deliver their messages only directly and not by means of a sequence of message forwarding as in CAR. In other words, Spray and Wait has very poor performance in very disconnected areas where the carrier of a message could never meet the final recipient of the message. These results are also confirmed in the scenario based on the Dartmouth traces discussed in Section III-B.8.

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.

In Figure 13 we show the performance of CAR, Random Choice, Epidemic protocol, Spray and Wait and PRoPHET using the Random Waypoint model in the 100 hosts scenario. Quite interestingly, the routing mechanisms at the basis of CAR are still useful also in presence of this mobility patterns. 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. We also observe that the performance of Spray and Wait improve considerably in this scenario since the probability of reaching any host is higher and uniform compared with the 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) 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. We report the results for the 100 hosts scenario in Figure 14. We obtained similar results for the 50 hosts scenario. 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 is not used.

5) Influence of the Speed of the Hosts and Routing Table Transmission Interval: In Figures 16 the influence of the speed of the hosts on the delivery ratio for the 50 and 100 scenarios is shown considering increasing retransmission intervals. In these experiments all the nodes have the same speed and we assume infinite buffer size. 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[7], but also because the quality of the prediction deteriorates as the mobility of the nodes increases. 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 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 become stale increases[8] 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. As expected the performances of PRoPHET and Spray and Wait improve as the speed increases, since the probability of getting in reach with all the hosts increases. The retransmission interval used for these two protocols is 30 seconds.

[7]This is a well know problem of distance vector protocols for mobile ad hoc networks [12].

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

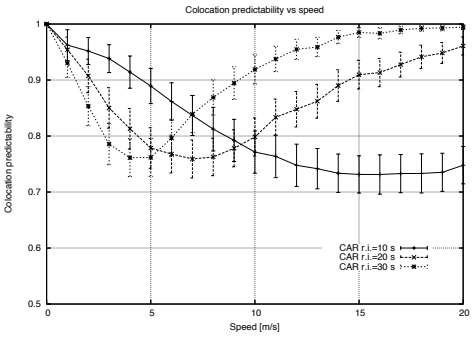


Fig. 18. Influence of the host speed on the colocation predictability (scenario with 100 hosts and Community based mobility model).

6) Influence of the Maximum Number of Hops: The influence of the maximum number of hops (i.e., the Time-To-Live in terms of number of hops) on the delivery ratio for the 100 hosts scenario is shown in Figure 17. A small maximum number of hops impacts on the possibility of transferring messages from a carrier to another one with a higher delivery probability. This saturation of the performance is more visible for the 50 hosts scenario, where the impact of imposing a limited number of hops is less evident. Clearly, the impact of a limited maximum number of hops 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).

7) Predictability Level and Protocol Performance: An analysis of the prediction level for both scenarios with different routing table transmission intervals is reported in Figure 18. 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 $[0, 9]$ for the 100 hosts scenarios. Then, we note an increment of the prediction level. This is due to the fact that with high speeds the node goes undetected, since the routing tables are also used to detect the presence of the neighbours. An undetected node is considered always not colocated, and, consequently the colocation prediction error is very low. As speed increases, the number of undetected nodes increases and, therefore, the overall predictability level increases. This is a limitation of the proposed protocol in case of very dynamic environments, since it needs frequent updates of colocation attributes to be able to detect correctly all the changes in terms of connectivity.

We also tested the predictability level of the colocation attribute of a real set of traces, the ones from the Haggle Project (more specifically, the Intel dataset [27]) applying the same prediction framework that we used for the simulations. Within a 10% error margin, the predictability level is 97%.

	Delivery ratio	Number of Messages
Flooding	9.6%	39890
Epidemic	62.7%	262456
CAR	49.9%	15189
Random	28.5%	23961
Spray and Wait 5 copies	5.7%	2735
Spray and Wait 10 copies	6.5%	3180

TABLE I
PERFORMANCE EVALUATION USING DARTMOUTH COLLEGE TRACES.

8) Evaluation using Dartmouth traces: Finally, we ran our simulation code using real connectivity traces from Dartmouth College. We used a selection of traces from [28], considering the period from 01/04/2001 to 30/06/2004. These traces record connections and disconnections of users at a number of access points in the Dartmouth campus; in particular, the data available include MAC addresses, locations of access, and timestamps. Two hosts are considered colocated if they are registered to the same access point. We generate traces for 8 hours considering 200 hosts mirroring the behaviour of Dartmouth traces in terms of connectivity patterns in the period 19 April to 19 May 2004, a period without holidays, considering the time interval from 9am to 5pm. During the simulation we send 1000 messages in these 8 simulated hours. The messages are sent from randomly chosen senders to randomly selected recipients at the beginning of the simulation. We use a buffer size equal to 1000 (i.e., infinite) for the store-and-forward protocols.

The results of this evaluation, in particular the delivery ratio obtained using flooding, show that the topology is rather sparse in this scenario. As it can be observed in Table I, the maximum achievable delivery ratio is 62.7% (i.e., the upper bound provided by the epidemic protocol). CAR is able to show better performance than other multi-copy approaches considering the trade-off between delivery ratio and number of messages sent. In fact, it outperforms Spray and Wait in terms of delivery ratio even if the overhead is larger due to the number of retransmissions of the messages and the higher number of messages actually delivered to the recipients. Instead, in the case of Spray and Wait, the overhead is in large part only associated to the initial creation and forwarding of the copies that are never delivered to the final destination. It is interesting to note that the hosts are not able to create all the copies (5 and 10 respectively); moreover, we did not observe any improvement with a number of copies higher than 10. This means that each carrier is not able to encounter more than 10 different nodes (i.e., it is not able to create more than 10 copies) in this setting.

IV. RELATED WORK

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 [29]. 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.

We give now an overview of the most interesting algorithm for routing in delay tolerant networks [30]. We have extensively described the epidemic routing in Section III: this is the simplest way of enabling communication in intermittently connected networks by means of replicating messages on all the hosts (epidemic algorithms) or in a certain number of them (semi-epidemic algorithms). Examples of semi-epidemic approaches are Spray&Wait [8] based on the generation of a given number of copies sent to a random set of neighbours and PRoPHET [7] where the probability of replication is a function of the period of colocation of the host with the message recipient in the past. With respect to 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.

Many more refined approaches have been proposed in the recent years [30]. For example, in [4] 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. In [31] the authors study the DTN routing problem from a resource allocation problem perspective. Zhao et alii in [5] 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. With respect to the existing 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 use lightweight mechanisms able to be implemented on potentially small devices such as sensors [15]. Our techniques could be integrated with these approaches, since they address orthogonal aspects of the problem.

CAR has inspired the design of other protocols based on the study of mobility patterns such as [32], where different metrics for the evaluation of host colocation are taken into the consideration for the selection of the best carriers, and [33] where machine learning techniques are applied to extract social patterns among the individuals carrying the devices.

V. CONCLUSIONS

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 connected 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.

The simulation experiments have shown that CAR is able to guarantee good performance with a limited overhead in terms of number of messages sent, in comparison to the other single-copy and multiple-copy protocols taken into consideration. More specifically, CAR is able to outperform other multi-copy protocols such as Spray and Wait in scenarios characterised by sparse connectivity and very small buffers.

ACKNOWLEDGMENT

The authors would like to thank Jon Crowcroft, Anders Lindgren and Niki Trigoni for the useful suggestions and comments on the content of this paper. The authors would also like to acknowledge the support of EPSRC through project CREAM. Mirco Musolesi is currently supported from a research program in the Institute for Security Technology Studies at Dartmouth College, under award 60NANB6D6130 from the U.S. Department of Commerce. The statements, findings, conclusions, and recommendations are those of the authors and do not necessarily reflect the views of the National Institute of Standards and Technology (NIST) or the U.S. Department of Commerce.

REFERENCES

[1] C. E. Perkins, *Ad Hoc Networking*. Addison-Wesley, 2001.
[2] K. Fall, "A delay-tolerant network architecture for challenged internets," in *Proceedings of SIGCOMM'03*, August 2003.
[3] 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.
[4] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," in *Proceedings of SIGCOMM'04*, August 2004.
[5] V. Zhao, M. Ammar, and E. Zegura, "A message ferrying approach for data delivery in sparse mobile ad hoc networks," in *Proceedings of MobiHoc'04*, May 2004.
[6] M. P. N. Sarafijanovic-Djukic and M. Grossglauser, "Island Hopping: Efficient Mobility Assisted Forwarding in Partitioned Networks," in *Proceedings of IEEE SECON'06*, September 2006.
[7] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," *Mobile Computing and Communications Review*, vol. 7, no. 3, July 2003.
[8] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *Proceeding of WDTN'05*. New York, NY, USA: ACM Press, 2005, pp. 252–259.
[9] R. L. Keeney and H. Raiffa, *Decisions with Multiple Objectives: Preference and Value Tradeoffs*, ser. Wiley Series in Probability and Mathematical Statistics. John Wiley and Sons, 1976.
[10] M. Musolesi, S. Hailes, and C. Mascolo, "Adaptive routing for intermittently connected mobile ad hoc networks," in *Proceedings of WoWMoM'05. Taormina, Italy.* IEEE press, June 2005.
[11] M. Musolesi and C. Mascolo, "Designing mobility models based on social network theory," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 11, no. 3, 2007.
[12] C. E. Perkins and P. Bhagwat, "Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers," in *Proceedings of SIGCOMM'94*, August 1994.
[13] H. A. Taha, *Operations Research: An Introduction*. Prentice Hall, 1996.
[14] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME Journal of Basic Engineering*, March 1960.
[15] B. Pasztor, M. Musolesi, and C. Mascolo, "Opportunistic Mobile Sensor Data Collection with SCAR," in *Proceedings of MASS'07*. Pisa, Italy: IEEE Press, October 2007.
[16] 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.
[17] C. Chatfield, *The Analysis of Time Series An Introduction*. Chapman and Hall CRC, 2004.
[18] G. Malkin, "RFC2453 – RIP Version 2," 1999.
[19] P. J. Brockwell and R. A. Davis, *Introduction to Time Series and Forecasting*. Springer, 1996.
[20] M. Musolesi and C. Mascolo, "CAR: Context-aware Adaptive Routing for Delay Tolerant Mobile Networks –Additional Material," May 2007. [Online]. Available: http://www.cs.ucl.ac.uk/staff/m.musolesi/papers/tmc-car-addendum.pdf
[21] ——, "Evaluating context information predictability for autonomic communication," in *Proceedings of ACC'06*. Niagara Falls, NY: IEEE Computer Society Press, June 2006.
[22] A. Varga, "The OMNeT++ discrete event simulation system," in *Proceedings of ESM'01*, Prague, June 2001.
[23] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in ad hoc wireless network," in *Mobile Computing*, T. Imielinski and H. Korth, Eds. Kluwer Academics Publisher, 1996, ch. 5, pp. 153–181.
[24] M. Musolesi and C. Mascolo, "A Community based Mobility Model for Ad Hoc Network Research," in *Proceedings of REALMAN'06*. ACM Press, May 2006.
[25] D. J. Watts, *Small Worlds The Dynamics of Networks between Order and Randomness*, ser. Princeton Studies on Complexity. Princeton University Press, 1999.
[26] J. Durbin and S. J. Koopman, *Time Series Analysis by State Space Methods*. Oxford University Press, 2001.
[27] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, and A. Chaintreau, "CRAWDAD data set cambridge/haggle (v. 2006-01-31)," Downloaded from http://crawdad.cs.dartmouth.edu/cambridge/haggle, Jan. 2006.
[28] D. Kotz, T. Henderson, and I. Abyzov, "CRAWDAD trace set dartmouth/campus/movement (v. 2005-03-08)," Downloaded from http://crawdad.cs.dartmouth.edu/dartmouth/campus/movement, Mar. 2005.
[29] Q. Li and D. Rus, "Sending messages to mobile users in disconnected ad-hoc wireless networks," in *Proceedings of MobiCom'00*. New York, NY, USA: ACM Press, 2000, pp. 44–55.
[30] Z. Zhang, "Routing in Intermittently Connected Mobile Ad Hoc Networks and Delay Tolerant Networks: Overview and Challenges," *IEEE Communications Surveys and Tutorials*, vol. 8, no. 1, January 2006.
[31] A. Balasubramanian, B. N. Levine, and A. Venkataramani, "DTN routing as a resource allocation problem," in *Proceedings of SIGCOMM'07*, August 2007, pp. 373–384.
[32] J. Leguay, T. Friedman, and V. Conan, "Evaluating Mobility Pattern Space Routing for DTNs," in *Proceedings of INFOCOM'06*, Barcelona, Spain, April 2006.
[33] J. Ghosh, S. J. Philip, and C. Qiao, "Sociological Orbit aware Location Approximation and Routing (SOLAR) in MANET," *Elsevier Ad Hoc Networks Journal*, vol. 5, no. 2, pp. 189–209, March 2007.

Mirco Musolesi is a Postdoctoral Researcher at the Department of Computer Science, Dartmouth College, NH, USA. He is a Fellow of the Institute of Security Technology Studies at Dartmouth. Previously, he has been a Research Fellow at University College London, University College London (2005-2007). He also spent a research period at INRIA Rocquencourt, France in 2003. He holds a PhD in Computer Science from University College London, United Kingdom (2007) and a MSc in Electronic Engineering from the University of Bologna, Italy (2002). His research interests include delay tolerant networking, mobile networking and systems, wireless sensor systems, mobility modelling and social network based systems. More information about his profile and his research work can be found at `http://www.cs.dartmouth.edu/~musolesi`.

Cecilia Mascolo is an EPSRC Advanced Research Fellow and a University Lecturer in the Computer Laboratory, University of Cambridge. Prior to this, she was with the Department of Computer Science, University College London. She holds an MSc (1995) and a Ph.D. (2001) in Computer Science from University of Bologna (Italy). She has been a visiting fellow in Washington University in St. Louis in 1998. She has published extensively in the areas of opportunistic mobile network routing, realistic mobility models exploiting social theory, mobile sensor networks, middleware for pervasive and context- aware systems. Dr. Mascolo is currently working on EPSRC, EU and industry funded projects on opportunistic routing for mobile and sensor networks and embedded systems middleware, with applications in wildlife monitoring, emergency rescue operations and vehicular information dissemination. Dr. Mascolo has served as a Technical Programme Committee member in many mobile and sensor system, delay tolerant networks, middleware and software engineering ACM and IEEE conferences and workshops. More details of her profile are available at `http://www.cl.cam.ac.uk/~cm542`.