Adaptive Content-based Routing for Delay-tolerant Mobile Ad Hoc Networks

Paolo Costa
Dipartimento di Ingegneria Elettronica e Informazione
Politecnico di Milano
Piazza L. Da Vinci 32
Milano, Italy
costa@elet.polimi.it

Mirco Musolesi
Department of Computer Science
University College London
Gower Street
London WC1E 6BT, United Kingdom
m.musolesi@cs.ucl.ac.uk

Cecilia Mascolo
Department of Computer Science
University College London
Gower Street
London WC1E 6BT, United Kingdom
c.mascolo@cs.ucl.ac.uk

Gian Pietro Picco
Dipartimento di Ingegneria Elettronica e Informazione
Politecnico di Milano
Piazza L. Da Vinci 32
Milano, Italy
picco@elet.polimi.it

ABSTRACT

Content-based routing fosters a sharp decoupling between data producers and consumers. Therefore, it is key in highly dynamic scenarios like mobile ad hoc networks (MANETs), where it can provide the building block for higher-level programming abstractions, e.g., publish-subscribe and query-advertise. A few proposals exist for content-based routing on MANETs, addressing the issues concerned with the dynamic network topology. However, these approaches do not take into account that the network is often partitioned in smaller portions with rather stable topology, enjoying only intermittent connectivity thanks to carrier hosts traveling across partitions. This mobility pattern is frequent among human beings, and can be regarded as a form of delay-tolerant network.

In this paper we propose an adaptive content-based routing approach addressing this problem. The protocol takes into account information about host mobility and connectivity changes to produce estimates enabling a more accurate message forwarding. These include the identification of potential carrier hosts, therefore maximizing message delivery despite network partitions and intermittent connectivity. We compare the performance of our protocol against others, using a mobility model validated with real-world traces.

1. INTRODUCTION

Content-based routing differs from classical routing paradigms as messages are routed based on their *content* rather than their destination address. This form of implicit, multipoint communication fosters a high degree of decoupling, since the communicating parties are not necessarily aware of each other, and can therefore change dynamically without affecting the rest of the system. These ideas are at the core of several implemented systems, including publish-subscribe and event notification systems [15], distributed databases [3], peer-to-peer applications [19], and data collection in wireless sensor networks [20]. In these middleware layers, the programmer is provided with considerable expressive power, by enabling the data consumer (the *subscriber*) to dynamically specify the filtering of relevant data based on its content, instead of letting the source (the *publisher*) bind it explicitly to some static notion of group or topic, as in conventional multicast and topic-based approaches. This flexibility is enabled precisely by the content-based routing layer, which routes published messages towards subscribers, often using application-level routers. This is the case in content-based publish-subscribe, which is perhaps the most popular incarnation of content-based routing, and therefore we use hereafter as a reference in our discussion. Examples of well-known content-based routing systems are Siena [6], Jedi [13] and Gryphon [36].

Most of the work in the field has been devoted to improving the scalability of these systems, e.g., to tolerate high message loads through efficient filtering and forwarding algorithms [8,17], or to reduce the network overhead through efficient routing protocols [4,7]. Recently, approaches enabling content-based routing in more dynamic scenarios, including MANETs, have begun to appear [1,12,27]. MANETs pose challenging requirements, most notably due to the dynamicity of the network. As the network topology changes, the content-based routing infrastructure must change accordingly, e.g., to properly reconfigure the subscription information exploited to route messages towards interested nodes. In the aforementioned systems, however, the focus is on dealing with the dynamicity of the network, but not with the fact that the network itself may become partitioned. In traditional systems, network partitioning translates into impossibility to communicate. Instead, MANETs enable forms of disconnected communication where information may be carried by a mobile node and forwarded opportunistically across partitions, therefore allowing communication between areas of the network that are never connected by an end-to-end path.

Recently, this kind of opportunistic forwarding scenarios became popular in the research area investigating *delay tolerant networks* (DTN) [16], i.e., networks characterized by long delay paths and frequent (and often unpredictable) disconnections and network partitions. Examples are intermit-

.

tently connected mobile ad hoc networks [18], interplanetary and satellite communications [21] and mobile systems to provide transitive connectivity to isolated villages in rural areas [32]. In the Data Mules project [34], for instance, data from sensor nodes are collected by a device (the "mule") traveling among them. DakNet [32] aims at providing intermittent connectivity to the global Internet to rural areas of India and Cambodia. Villagers access services such as e-mail in e-kiosks: messages are collected and transported by buses to (and from) an Internet gateway in the nearest town. Buses are equipped with wireless technology, enabling the download and upload of messages among the e-kiosks and the Internet gateways. In all the aforementioned scenarios, mobile nodes enable indirect data exchange among disconnected portions of the overall network, typically using a store-and-forward approach and some form of opportunistic forwarding. Interestingly, a study by Intel Research using real-world traces of connectivity of people recently showed how similar mobility patterns emerge naturally from human behavior [9] (e.g., people moving across different social groups) therefore making the potential applicability of these concepts and techniques even wider.

In this paper, we present an adaptive content-based routing approach designed to deal with the aforementioned scenarios. Our protocol complements the subscription information necessary to content-based routing with information about the changes in the *context* observed by nearby nodes. Our framework is general enough to encompass a broad definition of context, e.g., including a node's residual energy, physical location, or application-specific data. For the purpose of this paper, however, we restrict ourselves to the contextual changes that are more relevant to our goals, i.e., mobility patterns and topology changes. Kalman filter forecasting techniques [2] are used to predict the future evolution of these parameters, based on previous observations. This enables the computation of estimates about which nodes are potentially good message carriers, i.e., may enable indirect connectivity by moving into partitions containing subscribers. These estimates are built by aggregating information (including subscriptions) collected only in the proximity of each node, to reduce overhead, and are used to "steer" messages towards good carriers during the routing process.

This paper is organized as follows. Section 2 presents the core concepts of our protocol. Section 3 discusses the details of the prediction mechanism used for opportunistic, content-based, message forwarding. Section 4 describes additional optimizations of the protocol. Section 5 evaluates our protocol in realistic mobility scenarios, showing that our approach is able to provide good delivery and low overhead in the presence of partitions and intermittent connectivity. Section 6 discusses related work. Finally, Section 7 contains brief concluding remarks.

2. BASIC PROTOCOL OPERATION

In this section we describe the base concepts of our protocol. Our reference setting is a MANET where hosts can subscribe to different interests in a content-based fashion. Note how this implies that the same message may match multiple subscription filters and/or multiple subscribers, and therefore, at some point, must be duplicated during forwarding. Partitions may occur as, in general, nodes have different degrees of mobility. The goal of our protocol is to enable content-based routing despite partitions and intermittent connectivity.

2.1 Overview

Our approach is inspired by a semi-probabilistic content-based routing protocol described in [12]. There, deterministic information about subscriptions is disseminated only in the vicinity of a node, therefore reducing the likelihood of loops and yet providing accurate—albeit limited—information for routing messages. On the other hand, in areas where this localized information is unavailable, messages are forwarded towards a randomly chosen subset of neighbors. Therefore, this approach performs routing by combining probabilistic and deterministic decisions: the former is resilient to change and therefore addresses dynamicity, while the latter reduces indiscriminate propagation by guiding messages towards the subscribers. The approach, however, does provide any mechanism to deal with network partitions.

In this paper, this simple idea is extended by considering context information (e.g., mobility, connectivity with others interested in the same content, or residual energy) to help the routing decision. In the semi-probabilistic approach, subscriptions are propagated up to a *subscription horizon* of ϕ hops from the subscribers. In our approach, we complement this with the propagation of additional information about the context changes. This information is aggregated by each node into a *utility function*, which enables to determine whether a host is a "good carrier" for a given message, therefore allowing us to cope with intermittent connectivity and network partitions. The dissemination of context information only within a neighborhood of ϕ hops provides reasonable accuracy while keeping the overhead under control, since ϕ is typically small. The actual computation of utility functions is inspired by work on a unicast scheme for delay-tolerant MANETs [28], and is described in more detail in Section 3.

Forwarding is governed by a *message forwarding threshold* τ, $0 < \tau < 1$, which prevents flooding the network with too many messages. A message is forwarded to $\lceil \tau L \rceil$ of the L available links towards direct neighbors. Links associated with deterministic information are used first. However, the estimates associated to utility functions are used next. If also this information is lacking, probabilistic forwarding is finally used. In addition, a store-and-forward approach is also exploited, to enable messages to persist for longer in the network, possibly while being carried by mobile nodes. Interestingly, store-and-forward is modeled as a special case of forwarding: the decision about whether to store a message locally is based on the same predictions about "being a good carrier" used for forwarding a message towards neighbors.

The details of our routing protocol are described next, together with the mechanism for disseminating and aggregating utilities.

2.2 Utility Dissemination and Aggregation

Utility functions are used to make decisions about message forwarding and local storing.

2.2.1 *Host Utilities*

Every host h calculates its own *host utility* $U_{h,i}$ for each interest i it knows of. $U_{h,i}$ measures the willingness of h to receive messages matching the interest i. Utilities can

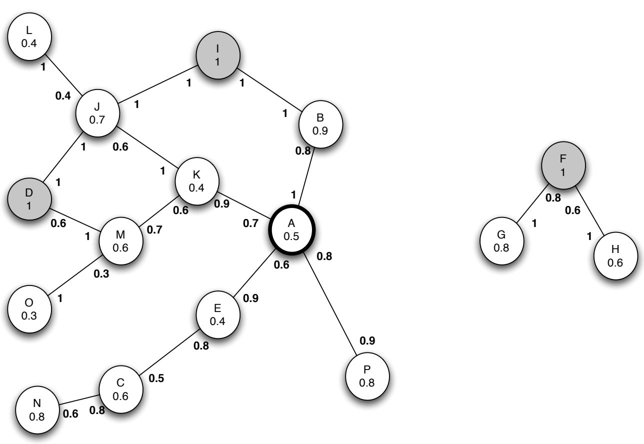


Figure 1: An example of a partitioned MANET. Host A is the publisher, gray hosts are subscribers to interests matching A's message. Numbers on hosts represent host utilities. Numbers on the links are forwarding utilities, as seen from the closest host to it. ϕ is assumed equal to 2.

assume values in the range $[0, 1]$, and are computed as

$$U_{h,i} = \begin{cases} 1 & \text{if } h \text{ is a subscriber for } i; \\ U^{rec}_{h,i} & \text{otherwise} \end{cases} \quad (1)$$

If h is a subscriber for i, its utility is the highest possible (i.e., it is equal to 1). If h is not a subscriber, it can still act as receiver in the forwarding process, with $U^{rec}_{h,i}$ representing its effectiveness either as a forwarder or a *carrier* of messages matching interest i. The mechanics of the computation of this utility using Kalman filters are explained in detail in Section 3. For now it suffices to say that a host utility combines information on the attributes of the context the host is immersed in.

2.2.2 Forwarding Utilities

Utilities are broadcast periodically by each host to the extent defined by the subscription horizon parameter ϕ. This information is aggregated by each receiving host on a per-link basis. More precisely, we define the set $R_{h,l,\phi}$ as the set of all the hosts at a distance less than or equal to ϕ hops from h, and are reachable through a path beginning at h with link l. The host h maintains a set of *forwarding utilities* $F_{h,l,i}$ for each link l and each subscription interest i, computed as:

$$F_{h,l,i} = \max(U_{x,i}) \qquad x \in R_{h,l,\phi} \quad (2)$$

Therefore, at host h, the forwarding utility associated to the pair (l, i) is equal to the maximum host utility for i, received along l.

2.2.3 Example

To illustrate how our approach works we introduce a running example, which we also use to illustrate message routing in Section 2.3. Consider the network depicted in Figure 1. Links represent connections between hosts: the network is temporarily partitioned in two parts. The gray hosts denote subscribers for some interest i, matching the message published by A. A subscription horizon $\phi = 2$ is assumed.

The number on each host is its utility, which has value 1 for subscribers. The number on the links are instead the values of the forwarding utilities $F_{h,l,i}$, computed by aggregating on a per-link basis the host utilities from nodes in the neighborhood at ϕ hops. For instance, in Figure 1, the forwarding utility for i at node A along link AE is 0.6. This value is computed from (2) where $R_{A,AE,2} = \{C, E\}$, and therefore $F_{A,AE,i} = \max(U_{C,i}, U_{E,i}) = 0.6$.

The picture shows a number of cycles. We note that cycles which are longer than 2ϕ create no problems as information on host utility is not propagated beyond ϕ: this is the case of cycle $ABIJDMKA$. Instead, in cycles shorter than 2ϕ host utilities can reach a host through more than one link. For instance, in cycle $DMKJ$ the host utility of D reaches K through link JK and link MK. Duplicates are easily discarded using timestamps associated to host utilities. Let us assume K discards the one which arrives through link MK. In this case $F_{K,JK,2} = 1$, based on the host utility of D which came through JK. Instead, $F_{K,MK,2} = 0.6$, i.e., it is the maximum between the host utilities of M and O, without considering the host utility of D.

2.3 Message Routing

In this section we describe how messages are routed, based on the dissemination of utility information we just described.

2.3.1 Forwarding Messages to Neighbors

As we discussed in Section 2.1, the forwarding mechanism relies on a *message forwarding threshold* τ that specifies the exact fraction of links that must be used when forwarding a message to neighbors. If L is the number of available outgoing links[1], the number of links used for forwarding is $\bar{f} = \lceil \tau L \rceil$. Then, the forwarding rule is as follows:

1. Deterministic information is exploited, if available:
   (a) a message m is always sent to *all* the subscribers at 1-hop distance whose interest i matches m, regardless of the value of τ. This ensures that subscribers in direct communication receive the message.
   (b) If $f < \bar{f}$ links have been used at this point, forwarding occurs on the $\bar{f} - f$ remaining links using deterministic information concerning subscribers at $1 < x \leq \phi$ hops, starting with subscriptions coming from subscribers that are closer. This information is obtained for free by storing in the subscription tables nodes that broadcast a host utility equal to 1. The process ends when $\bar{f}$ links have been used, or there is no more deterministic information available.
2. If still $f < \bar{f}$, the predictions $F_{h,l,i}$ based on utility functions are exploited. Clearly, only those for an interest i matching message m are used. Utility values are sorted in decreasing order; those with higher values are selected first, and forwarding occurs on the link l associated to the entry. The process ends when $\bar{f}$ links have been used, or there are no more predictions available.

[1] A message is never forwarded onto the link it came from. Therefore, $L = n - 1$, with n being the number of outgoing links towards neighbors. For a publisher, however, $L = n$.

3. If still $f < \bar{f}$, probabilistic forwarding is used. The message is simply forwarded along $\bar{f} - f$ links chosen at random.

2.3.2 Storing Messages

Content-based routing schemes typically forward messages as they arrive, without ever storing them. This approach is reasonable when connectivity is available, but not when hosts may experience partitions and disconnections—as in our case. Therefore, the base forwarding strategy we just described is complemented by a store-and-forward approach. Interestingly, its operation is also based on utility functions. After forwarding of a message m has occurred at a host h, m is also stored in h's buffer if, for at least one of the matching interests i, the host utility is $\sigma \leq U_{h,i} < 1$. Forwarding of buffered messages is attempted periodically, according to the aforementioned rules, until an associated lease expires.

The constraint $U_{h,i} < 1$ prevents a message to be unnecessarily stored (and later re-forwarded, by causing unnecessary traffic) by subscribers. This limits store-and-forward only to hosts that are good carriers. Indeed, the storing rule above can be equivalently rewritten as $U^{rec}_{h,i} \geq \sigma$. The only exception to this storing rule occurs when a node does not have *any* neighbors. In this case, since there is no one to forward the message to, it is always stored locally. The *storing threshold* σ is, like τ and ϕ, a protocol parameter and its impact is evaluated in Section 5.

2.3.3 Example

Consider the example in Figure 1 and assume a message forwarding threshold $\tau = 0.75$ and a storing threshold $\sigma = 0.6$. The publisher A has $L = 4$ outgoing links, therefore, the message must be forwarded along $\bar{f} = \lceil \tau L \rceil = 3$ links. There are no subscribers at distance 1 from A, therefore step 1 of the forwarding rule cannot be applied. However, A has information (from the dissemination of host utilities) about subscriber I at distance $\phi = 2$ from A, and therefore forwards the message onto link AB. To satisfy the constraint about $\bar{f}$, the message needs be forwarded onto two more links. This is accomplished using predictions as per step 2 of the forwarding rule. The message is sent also on the links with the highest forwarding utility, i.e., AP and AK. The message is then forwarded analogously along the subsequent hops in the left partition. Clearly, a message may be duplicated along different paths: for instance, the message can reach subscriber D along the route through IJ, KM or KJ. However, some amount of redundancy is beneficial in the dynamic environment we target, and the parameter τ allows us to keep overhead under control, as shown in Section 5.

Moreover, a copy of the message sent by A is also stored at B and P, but not at A and E since their utility is lower than σ. Similar storing occurs in later forwarding steps (e.g., on J), but not on subscribers (e.g., I). Storing messages on nodes with a high utility enables to bypass partitions, as shown in Figure 2. Here, host P has moved from one partition to the other, in order to become co-located with G. This causes the update of P's host utility (e.g., because it has observed a change in connectivity) as well as of the forwarding utilities of other nodes in the right partition. However, more importantly, since each node periodically attempts to forward the messages in its buffer, it is likely that P will be able to deliver the message to G. Indeed, the utility of P

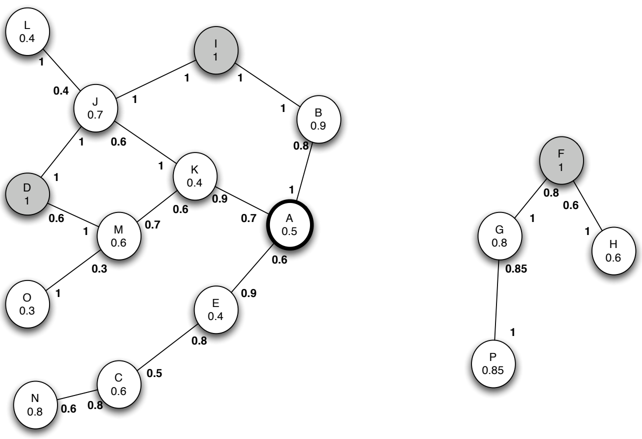


Figure 2: The network of Figure 1 after host P has migrated from the left to the right partition. P can therefore be exploited as an information carrier.

may have been high, precisely because in the recent past P was co-located with subscriber G and became temporarily disconnected. Our approach, as discussed in the next section, computes utilities based on context information and, combined with our routing strategy, enables the identification and exploitation of the best message carriers.

3. PREDICTING CONTEXT EVOLUTION

We now present the technique we use for calculating the host utility function $U^{rec}_{h,i}$. As we discussed in Section 2.2, it represents the utility of a host h to act as a forwarder or carrier for a message matching an interest i. In our approach, this utility function enables efficient forwarding by providing a prediction of the evolution of the context information perceived by a given host.

Context attributes and their composition. Many parameters affect the ability of a host to be a good receiver and, particularly for our goals, a carrier. For instance, a host with a high *change degree of connectivity* frequently changes its set of neighbors (e.g., because the host is moving, or is stable but in an area where many are moving around), and therefore has more options for forwarding. Instead, the probability of *subscriber co-location* can be used as a direct indicator of the likelihood of a host to meet a subscriber for a given interest i, therefore enabling direct delivery of matching messages. The host *residual energy* determines whether it has enough power to stay alive long enough to meet other hosts, possibly after travelling from one partition to the other, and disseminate the message further. Finally, the host *free buffer space* is a direct measure of the ability of the host to carry the message altogether.

Knowledge about the current values of these context attributes is helpful, but only to a limited extent. Instead, what really matters are the values that these attributes are likely to assume in the future. Knowledge of these future values increases the effectiveness of forwarding decisions, which can indeed steer messages towards the hosts that are most likely to be effective in delivering messages. This greatly improves the performance of routing, as demonstrated by our evaluation in Section 5.

Our approach makes available these estimates about the future values of context attributes as follows. First, we model each context attribute $a_1, a_2, ...a_n$ with a separate utility function U_{a_k}. Then, we compose these utilities into a single utility function by using results from multi-criteria decision theory [23]. Formally:

$$U(a_1, a_2, ..., a_n) = \sum_{k=1}^{n} w_{a_k} \widehat{U}_{a_k} \tag{3}$$

In this expression, the relative importance of each context attribute estimates is defined by using the weights w_{a_k}. Their values depend on the application scenario at hand, and how to set them is beyond the scope of this paper. In Section 5 we use values determined through simulation. The focus of this paper is instead on $\widehat{U}_{a_k}$, which denotes the *prediction* of the future value of the utility U_{a_k} associated with attribute a_k.

Kalman filters. The estimated values $\widehat{U}_{a_k}$ are determined using prediction techniques based on Kalman filters [22], originally developed in automatic control systems theory. Kalman filters are a technique for 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, by a set of *prediction recursive equations*. Our prediction problem can be expressed as a state space model: a time series of observed values represents context information, from which we can derive a prediction model based on an inner state represented by a set of vectors. Formally, given the current input observed value $\mathbf{Y}_t$ and the current state $\mathbf{X}_t$, a predictor based on Kalman filters is able to provide an estimate for the next value of the time series $\widehat{\mathbf{Y}}_{t+1}$.

$$\widehat{\mathbf{Y}}_{t+1} = f(\mathbf{X}_t, \mathbf{Y}_t) \tag{4}$$

We assume that the lag between two subsequent samples $\mathbf{Y}_t$ and $\mathbf{Y}_{t+1}$ of the time series is equal to T. Trend and seasonal components [2] could be added as well. The prediction is re-evaluated periodically according to the (configurable) value of T. We use a Kalman filter predictor for each context attribute. The filter takes as input the current value at time t of the time series representing a particular attribute and returns the estimated value of the time series at time $t + T$ as output.

The main advantage of Kalman filters is that they do not require the storage of the entire past history of the system, making them suitable for a mobile setting in which resources may potentially be very limited. This technique is also very lightweight from a computational point of view, since the forecasting model only requires the update of the values representing the state using a system composed of linear equations (without any integration or differentiation required).

Predicting context values. Hereafter, we focus only on the first two contextual attributes mentioned above, i.e., the change degree of connectivity and the probability of subscriber co-location. These are indeed the most important attributes for our purposes, and by limiting ourselves to their description we keep our treatment simpler. However, the framework is general and open to inclusion of any other context attribute. Formally, we define, for a given host h and a given interest i, the utility functions U_{cdc_h} and $U_{col_{h,i}}$, respectively. Equation (3) becomes then

$$U_{h,i} = w_{cdc_h} \widehat{U}_{cdc_h} + w_{col_{h,i}} \widehat{U}_{col_{h,i}} \tag{5}$$

$\widehat{U}_{cdc_h}$ represents the estimated future value of the change degree of connectivity of host h, i.e., the number of connections and disconnections that the host has experienced over the last T seconds. A high value of this estimate means that host h is likely to be in reach of a large number of different hosts. Instead, $\widehat{U}_{col_{h,i}}$ summarizes the history of colocation of h with subscribers to interest i. A high value of $\widehat{U}_{col_{h,i}}$ means that h is likely to become co-located with one or more subscribers to i in the near future.

These predicted values are computed using the Kalman filter as per Equation (4). We cannot repeat here the mathematical details of Kalman filter based predictors: a comprehensive presentation of these techniques is found in [2] and [10]. However, it is fundamental to say how do we compute the *input values* to the Kalman filter, i.e., the value of the utility at time t, for which Equation (4) computes the predicted value at time $t + 1$ (i.e., after T seconds). In the case of $\widehat{U}_{cdc_h}$, let $n(t)$ be the set of the neighbors of h at time t. U_{cdc_h} at time t is computed by comparing the members of $n(t)$ against those at the beginning of the previous period, i.e., at $t - T$:

$$U_{cdc_h}(t) = \frac{|n(t-T) \cup n(t)| - |n(t-T) \cap n(t)|}{|n(t-T) \cup n(t)|} \tag{6}$$

Intuitively, the formula above yields the number of hosts who became neighbors or disappeared in the time interval $[t - T, t]$, normalized by the total number of hosts met in the same time interval. On the other hand, the computation of the input parameter $U_{col_{h,i}}(t)$, is much simpler: $U_{col_{h,i}}(t) = 1$ if h is co-located with a subscriber to interest i, and $U_{col_{h,i}}(t) = 0$, otherwise.

4. ADDITIONAL PROTOCOL DETAILS

In this section we provide additional details about our approach. We begin by describing two optimizations concerning the content and dissemination of utility information, whose impact is analyzed in Section 5. Then, we discuss how to take into account information on the confidence level about context predictions.

Looking beyond the horizon: Neighborhood utilities. The utility dissemination process we described in Section 2.2 enables a host h to gain knowledge about the utilities of hosts in its neighborhood at ϕ hops. However, a simple modification to the protocol we outlined enables h to "look" beyond the horizon at ϕ hops. This additional context knowledge, albeit less precise, enables significant performance improvements, as shown in Section 5.

We achieve this by defining an additional *neighborhood utility*. The neighborhood utility $N_{x,i}$ of a host x, w.r.t. a given interest i, is defined as

$$N_{x,i} = \max(F_{x,i,l}) \tag{7}$$

i.e., it is the maximum among the forwarding utilities of the outgoing links of x. The neighborhood utility provides aggregate information about a host's ϕ-neighborhood. This information is piggybacked on the message communicating the host utility, and therefore does not require additional messages.

The host h receiving this information only uses it when it is coming from a host that is at an exact distance of ϕ hops, i.e., at the border of its neighborhood. Indeed, the purpose of the neighborhood utility is to increase the context knowledge beyond ϕ hops: therefore, only the hosts on the fringe of the neighborhood matter. This is accomplished straightforwardly by taking into account neighborhood utilities in the computation of the forwarding utility. This requires substituting $U_{x,i}$ in Equation (2) with $U'_{x,i}$, where

$$U'_{x,i} = \begin{cases} \max(U_{x,i}, N_{x,i}) & \text{if } \mathrm{dist}(h,x) = \phi \\ U_{x,i} & \text{otherwise} \end{cases} \quad (8)$$

In other words, forwarding utilities are computed by considering, for the hosts at a distance ϕ, the maximum between their host utilities and their neighborhood utilities. The computation of forwarding utilities for the other hosts is unaltered.

Adaptive utility dissemination. Thus far, we assumed that each host broadcasts utility values periodically. In a highly dynamic scenario, this periodic refresh of soft state is usually very efficient. Nevertheless, during periods of relative stability it can cause significant overhead, unnecessarily refreshing unmodified state.

To obviate to this problem and reduce the overhead caused by utility dissemination we designed an adaptive mechanism that dynamically changes the interval T_d between two utility broadcasts. The mechanism exploits a property of the Kalman filter, that is, the possibility of generating a prediction even in absence of a new input value, by relying on past predicted values. A detailed mathematical description of this property can be found for example in [2]. Clearly, this characteristic is a precious asset during temporary disconnections.

To dynamically adapt the value of T_d we rely on the change degree of connectivity of the host, U_{cdc_h}. If U_{cdc_h} is high, the context of host is highly dynamic, and therefore its utility values should be refreshed frequently. Therefore, we set T_d as inversely proportional to U_{cdc_h}:

$$T_d = \begin{cases} T_{d_{max}} & \text{if } U_{cdc_h} = 0 \\ \dfrac{T_{d_{min}}}{U_{cdc_h}} & \text{otherwise} \end{cases} \quad (9)$$

The value of T_d is set to a maximum when the system is stable, and reaches the minimum value $T_{d_{min}}$ when a host changes all of its neighbors, i.e., $U_{cdc_h} = 1$.

Dealing with Unreliable Predictions. Our protocol relies on predictions about the future values of context attributes. However, in some conditions predictions are not reliable, e.g., because the time series describing a particular context attribute is random or exhibit a behavior that cannot be forecasted with accuracy (i.e., within a given prediction error) using the model used. Forwarding decisions based on unreliable predictions can actually be worse than blind, random decisions. Therefore, it is important to assess the confidence level of context predictions, and modify forwarding decisions accordingly.

To assess the quality of context predictions we use the technique presented in [30], based on the analysis of the prediction error [10]. A *predictability component* receives in input both the observed value (at time t) of a context attribute and the predicted value (computed at $t-1$). The analysis over time of the difference between these two values (called the *residual* value) enables to determine whether the prediction model (the Kalman filter in our case) has enough information to predict the next value of the time series with the required accuracy. In essence, this is true when the residuals are randomly distributed and their value is close to zero.

When the predictability component determines that predictions are unreliable, we simply do not use predictions. This essentially means short-circuiting Step 2 of the forwarding rule we presented in Section *2.3.1*: in absence of deterministic information, a message is forwarded along randomly selected links, without leveraging off predictions. Our approach therefore defaults on a semi-probabilistic protocol, albeit enhanced by the store-and-forward mechanism discussed in Section *2.3.2*.

5. EVALUATION

In this section we compare our approach against others, and assess the impact of protocol parameters.

5.1 Simulation Setting

We evaluated the performance of our protocol using the OMNeT++ [39] discrete event simulator. Our simulation code is publicly available to the research community at *[URL withheld for blind review]*. We now illustrate the chosen mobility model and other aspects of the simulation setting.

5.1.1 Mobility Model

Given that our protocol is based on prediction of possible colocation and movement, it is important to be able to evaluate it in the context of a mobility model which is not random. To this end, we used the Community based mobility model presented in [29], where the variability of the host colocation and mobility is based on deterministic values given in input and follows precise patterns, validated against real traces provided by Intel Research [9].

The model is based on the following observation: in mobile networks, devices are usually carried by humans, so their movement is necessarily based on human decisions and social behavior. To capture this type of behavior, the model is heavily dependent on the structure of the relationships among the people carrying the devices, e.g., the social network that links the individuals carrying the mobile devices. The movements of both groups as well as single hosts is driven by social relationships. The simulation area is divided into a grid—8×8 in our experiments. Each group 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 exert the highest attraction towards it (including the current one). This attraction is calculated by evaluating a matrix describing the social network. Finally, in the original model the movements of hosts between groups is an emergent characteristic of the model that cannot be controlled directly. Therefore, to retain more control over the simulation scenario, we modified it by adding hosts (called *travelers*) that always choose to move to another group, even if the current one exerts the highest attraction.

5.1.2 Default Simulation Parameters

We consider a simulation scenario composed by 200 hosts in an area of 2000 $m \times 2000$ m. We assume that every device is equipped with an omnidirectional antenna with a transmission range of 250 m. The host speed is generated using a uniform distribution with values in the range $[1, 10]$ m/s. The speed of traveler hosts is set to 20 m/s to differentiate their roles in the simulation and to evaluate the correctness and the performance of the mechanism of choice of the best carriers, which is partially based on the change degree of connectivity, a function of the relative speeds between the hosts.

The percentage of publishers and subscribers is set to 50%. The number of possible interests in the network is set to 100. A characteristic of content-based addressing is that these interests can be overlapping, and therefore a message can match multiple subscriptions. Therefore, for each message a 10% of the subscribers are chosen randomly as message recipients. The number of subscriptions per node is chosen equal to 2. The publishing interval is set to 60 s. The simulation time is set to 600 s. Protocol performance is evaluated by considering only the messages sent during the interval $[100\ s, 300\ s]$, to reserve at least 300 s for the delivery of the messages.

As for the parameters specific of our approach, the default values are $\tau = 0.4$, $\sigma = 0.4$, $\phi = 1$. Utility weights are $w_{cdc} = w_{col} = 0.5$. Message buffers used for store-and-forward are of size $\beta = 100$. The period of the Kalman filter predictors is $T = 10$ s. The retransmission interval of the messages stored temporarily in the buffer of the hosts is 60 s. The interval between two subsequent transmissions of the utilities ranges from $T_{d_{min}} = 10$ s to $T_{d_{max}} = 60$ s when the adaptive technique described in Section 4 is used; $T_d = T_{d_{min}} = 10$ s otherwise.

Finally, we averaged results over 20 runs, using different seeds for each scenario.

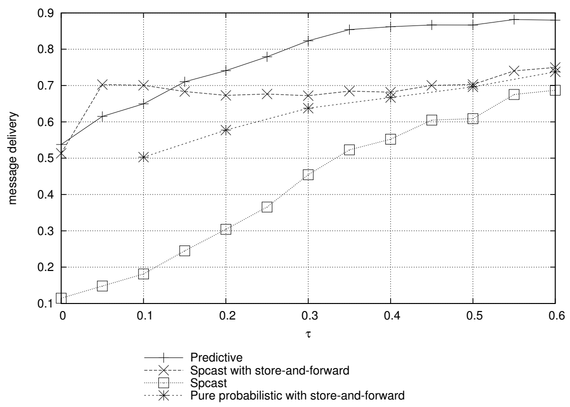
5.1.3 Compared Protocols

In our simulations, we compared the performance of our protocol against the semi-probabilistic approach described in [12]. This protocol, hereafter referred to simply as *spcast*, is probably the closest to ours among all the existing approaches, although inadequate in the context of DTN: comparing with it allows us to appreciate directly the impact of prediction in selecting the most appropriate message carriers and support disconnections. In addition, we use as a baseline a purely probabilistic forwarding, i.e., *spcast* with no deterministic information available (ϕ=0).

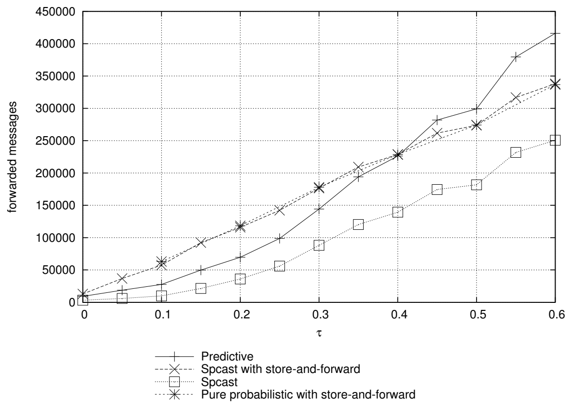
Neither of these protocols has the ability to perform store-and-forward and therefore bypass partitions. Therefore, to be fair we decided to enhance them with this capability, by enabling each host to store a copy of the message and periodically attempt its retransmission for a given number of times, 10 in our simulations.

5.2 Simulation Results

In this section we present the results of our simulations. We mainly concentrate our analysis on message delivery and network traffic due to forwarded messages, since the traffic due to the control messages (subscriptions in *spcast* and utilities in our protocol) is the same.



(a) Message delivery



(b) Forwarded messages

Figure 3: Message delivery and network traffic vs. message propagation threshold τ.

Message propagation threshold. The most critical parameter related to message forwarding is clearly the message propagation threshold τ, as it directly influences the performance of our protocol by determining the number of message copies forwarded. In Figure 3, we plot message delivery and overhead against different values of τ. For values of $\tau < 0.2$ the behavior of *spcast* and of our protocol are very similar: given the small number of links $\bar{f}$ used through τ, the main contribution to forwarding decisions is given by the deterministic component of the two protocols, which is handled identically. This also explains why the performance of the purely probabilistic protocol is poor, since a low value of τ is not sufficient to "infect" the network. This effect is exacerbated in the original version of *spcast* without store-and-forward, also shown in Figure 3(a): with a low value of τ, the role played by the message buffering is fundamental to achieve a good delivery.

When $\tau > 0.2$ the performance of *spcast* and of the probabilistic protocol become similar, as the probabilistic part assumes higher importance, also due to our choice of $\phi = 1$. Conversely, our protocol is able to exploits context predictions to make informed forwarding decisions and achieve higher message delivery. Interestingly, the performance of *spcast* and of the probabilistic protocol do not vary when τ increases, since the buffers quickly saturate and many messages need to be discarded, hampering delivery in presence of network partitions. Instead, as we discuss later in this section, the storing rule used by our protocol keeps buffers small and takes advantage of the higher τ to improve delivery. The performance of pure *spcast* without store-and-forward improves too, as it is not affected by buffer management issues and therefore exploits the larger fanout to disseminate more messages.

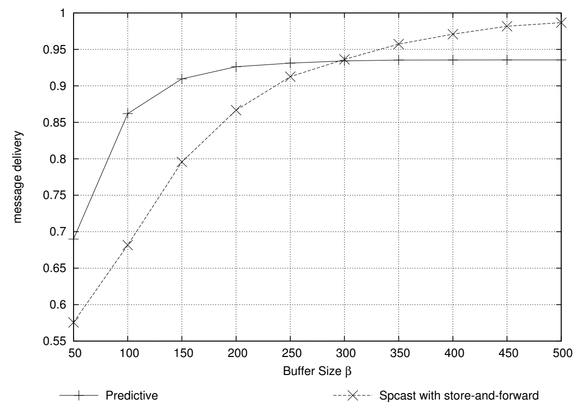
The improvement achieved by our protocol is even more evident in Figure 3(b), where we plot the overall number of forwarded messages. As expected, with the exception of the pure *spcast* protocol, the traffic of the other three is very similar. Indeed, the number of forwarded messages is directly affected by the fanout determined by τ and by the buffer size β, which are equal for all of them. Our protocol generates a comparable amount of traffic but provides higher message delivery, thus confirming its ability to steer messages towards interested hosts by selecting "good" carriers.

Message Buffering. As we pointed out, our predictive mechanism (i) drives the routing decisions by choosing potentially good carriers and (ii) enables a clever buffer management, by allowing only nodes with a high utility function to store messages. The effect of (ii) is in part already evident in Figure 3, however it becomes clearer in Figure 4. If buffers are small (e.g., $\beta < 300$), as usually assumed in resource-constrained devices like PDAs or sensors, our protocol is always better in terms of message delivery (Figure 4(a)) and, notably, generates less traffic (Figure 4(b)).

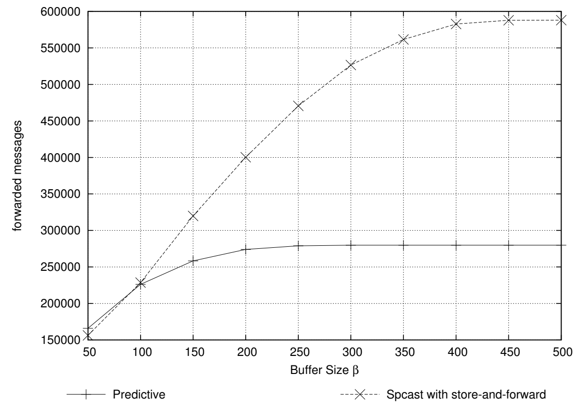
Looking at Figure 4(b), one might ask why our protocol does not reach a message delivery close to 100% with a larger buffer size, although the overhead is significantly lower. The reason is that, in our protocol, hosts store messages only if their utility is greater than σ and hence, even if there is room in the buffer, they do not exploit it.

The impact of σ is shown in Figure 5. We plot two curves for message delivery: one with a fixed buffer size ($\beta = 100$) and the other with an infinite buffer. In the latter case, the behavior is as expected: by letting nodes store more messages, delivery clearly increases up to 100%. Nevertheless, even when $\sigma = 0$ (i.e., messages are always buffered, as in *spcast*) the contribution of prediction to the forwarding is still significant, as the number of forwarded messages (not shown) is sensibly lower: 412,156 against 579,639 messages, corresponding to a gain of 29%.

Interestingly, instead, when a fixed buffer size is used the delivery decreases when σ decreases. This may appear strange as one would expect a behavior similar to the one with infinite buffer. Nevertheless, this phenomenon occurs because lower values of σ imply a less selective message buffering, as nodes store messages even when they have a low utility, i.e., they are not good carriers. However, being the buffers finite, when a buffer is full the oldest messages are discarded, regardless of whether the node is a good carrier for them. Therefore, a node may need to delete a message for which it has a high utility function only to buffer another for which it is not a good carrier, thus decreasing the probability to successfully deliver messages.



(a) Message Delivery



(b) Forwarded Events

Figure 4: Message delivery and network traffic vs. buffer size β.

Subscription Horizon. In *spcast*, the ϕ parameter determines how far subscription information is propagated and, hence, directly controls the amount of deterministic information disseminated in the network. The importance of this parameter is even greater in our protocol, as it defines the scope of utility dissemination. Therefore, its role is twofold: on one hand it enables to discover how many subscribers are around (i.e., all nodes with a utility function equal to 1), as in *spcast*. On the other hand, it enables building up the limited knowledge of the network upon which predictions are performed.

In terms of message delivery, shown in Figure 6(a), the effect of incrementing ϕ is more sensible in *spcast*. Thanks to the additional deterministic information, routing becomes more efficient and consequently delivery increases. In our protocol, instead, message delivery is quite stable, since pre-

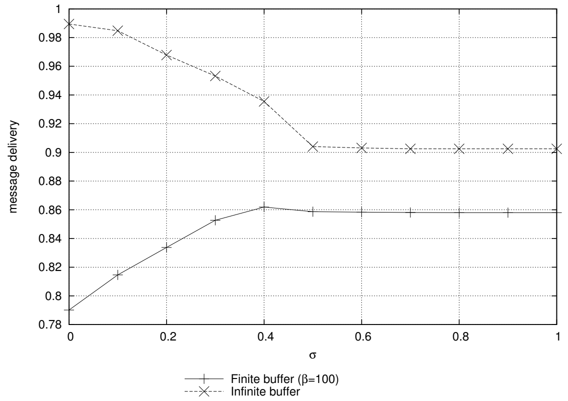


Figure 5: Message delivery vs. storing threshold σ.

dictions compensate the absence of deterministic information.

Nevertheless, the main impact of ϕ is clarified in Figure 6(b): the increased deterministic information available clearly reduces the forwarded events. This occurs both in *spcast*, where nodes have more entries in their routing table, as well as in our protocol, where not only more subscribers are known but also predictions become more reliable as more data are available.

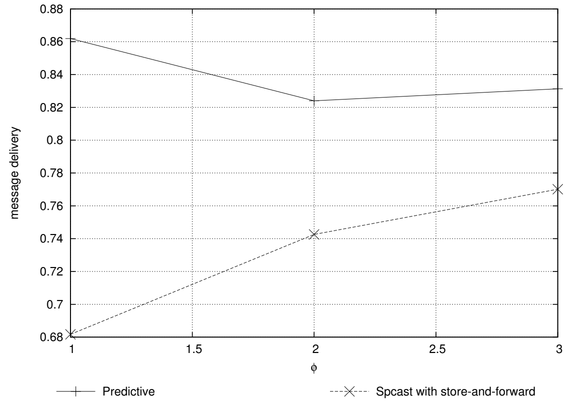
Also, in Figure 7 we report the results showing the impact of neighborhood utilities. Results show that message delivery is unaffected by this mechanism (the difference is less than 4%), whereas the number of forwarded messages is greatly reduced, thanks to the ability of hosts to "look" beyond the subscription horizon.

Mobility Parameters. Thus far we discussed the effect on performance of protocol parameters. Now, instead, we vary some scenario parameters.

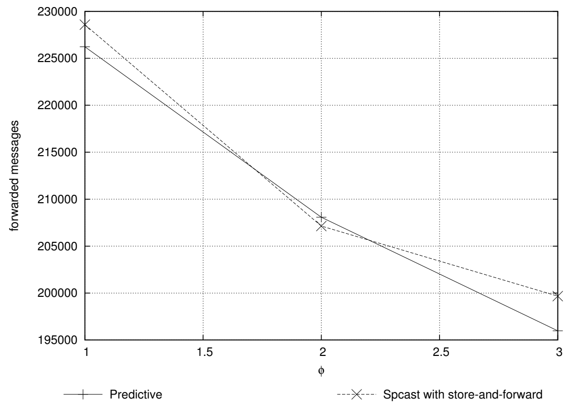
The first one is the speed of hosts, as shown in Figure 8. Our approach relies on carriers that are chosen for their relative mobility and colocation with subscribers. In other words, messages are replicated only on potential carriers. As the speed increases, delivery increases as well, since our protocol is effective in identifying travellers and, clearly, the faster these nodes are, the more efficient message dissemination will be.

Similar considerations hold for the results in Figure 9, where we vary the number of travellers in the network. As already pointed out, travellers are key to disseminate messages in remote portions of the network and a peculiarity of our approach is the ability to identify these special hosts. Naturally, the less travellers are present in the system, the less our prediction is effective, which justifies the delivery decreases with few travellers.

Utility Functions Weights. We also evaluated the influence of the utility weights by varying their values. The results are shown in Figure 10 for a scenario with 16 carriers, and the default values of $\tau = 0.4$ and $\sigma = 0.4$. It



(a) Message Delivery



(b) Forwarded Events

Figure 6: Message delivery and network traffic vs. subscription horizon ϕ.

	Delivery	Forwarded Messages
With	93.54%	279,904
Without	97.25%	378,621

Figure 7: Impact of neighborhood utilities.

is interesting to note that both attributes contribute to the performance of the protocol in terms of delivery ratio: when one of the two weights is set to 0, the delivery decreases.

The choice of these weights can be tuned by experimentation to achieve the best performance. In the scenario taken into consideration, the best choice among the combinations that we have simulated is $w_{cdc_h} = 0.5$ and $w_{col_{h,i}} = 0.5$, as shown in Figure 10. These values are obtained in the case of an infinite buffer to avoid the influence of other factors.

Adaptive Update of Utility Values. In the previous charts, we never considered the traffic generated by our approach to

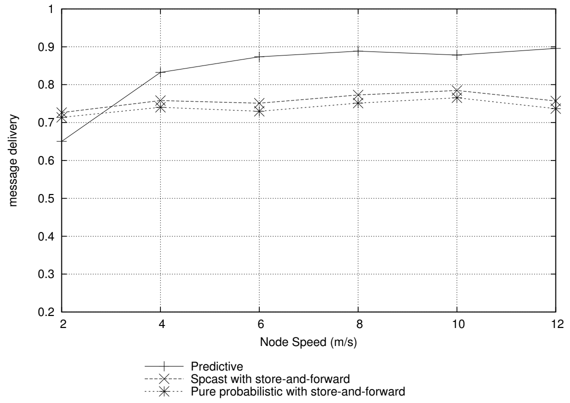


Figure 8: Message delivery and network traffic vs. host speed.

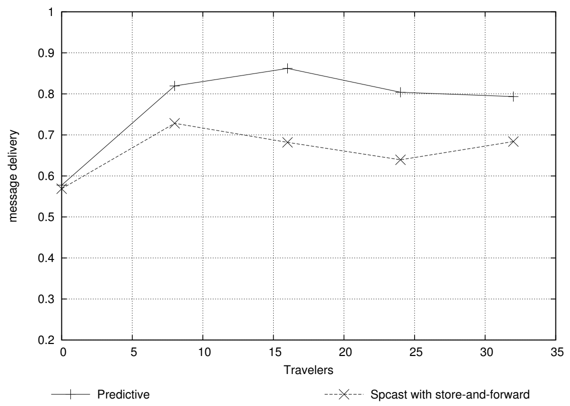


Figure 9: Message delivery against different numbers of travellers.

w_{cdc_h}	$w_{col_{h,i}}$	Delivery	Forwarded Messages
0.0	1.0	86.85%	270,992
0.25	0.75	92.24%	281,120
0.5	0.5	92.24%	284,630
0.75	0.25	88.17%	260,482
1.0	0.0	87.47%	244,077

Figure 10: Impact of utility function weights.

update the values of utility functions. Indeed, this traffic is an order of magnitude less than the traffic due to forwarded events (36,244 update messages against 279,904 event messages). Moreover, the number of updates disseminated in the network by our protocol is essentially the same as the number of subscription messages generated by *spcast*. The difference is only in the content of the messages, but not in their number.

	Delivery	Forwarded Messages	Updates
Adaptive	90.75%	243,034	10,771
Non-adaptive	93.54%	279,904	36,244

Figure 11: Impact of adaptive dissemination of utility functions.

However, here we show the impact of the adaptive technique we illustrated in Section 4 to further reduce this traffic, based on the possibility to dynamically adjust the interval between two subsequent updates to save message if network conditions do not change. The results are shown in Figure 11. As expected, the adaptive setting of the update interval drastically reduces the number of updates spread across the network, without significantly affecting delivery and overhead. Since this optimization is characteristic of our use of Kalman filters, subscription propagation in *spcast* cannot be easily extended along the same lines. Indeed, a host would not know whether it missed a subscription because the corresponding subscriber has disappeared or as a consequence of adaptivity. Instead, in our protocol the Kalman filter yields a valid prediction based on input values obtained in the recent past.

6. RELATED WORK

In the last decade, the research community has carried out a large effort in the area of content-based routing on fixed networks. Unfortunately, the vast majority of available approaches do not deal with topological reconfiguration, thus hampering their applicability in highly dynamic scenarios. Some researchers addressed the problem of supporting client mobility [5, 13, 31] but their approaches cannot be extended to the more general case of dispatcher mobility, as demanded by MANET scenarios.

In the area of MANETs, there has been a consistent body of work concerning multicast communication [24,26,33]. However, results are not directly reusable given the peculiarity posed by content-based routing, which instead has been addressed by very few works in literature. Content Based Multicast (CBM) [41] and STEAM [27] provide a notion of *spatial scope* which defines the area messages are propagated within. In particular, CBM allows publishers to specify the direction and the distance an message is spread. Similarly, STEAM limits the message propagation to a proximity area, inside which messages are broadcast and locally matched against subscriptions. With respect to these approaches, our protocol enjoys wider applicability, as messages are delivered throughout the network, based on node interests, regardless their locations. Autonomous Gossip [14] shares an idea similar to ours, by pushing message towards potential receivers in a content-based fashion, according to node "similarities". However, the authors neither give details on how this notion of similarity is actually computed and disseminated, nor provide quantitative analysis on the performance of their protocol. Another closely related approach is discussed in [1], where nodes keep track of the last time they have been in range of others with the same interests. Forwarding occurs by selecting the nodes with matching interest, and waiting for an interval proportional to the smallest time since when one of these nodes was seen. If, in the meanwhile, the forwarding node overhears the same message being broadcast by one of its neighbors (i.e., meaning that the neighbor has seen an interested node more recently), the message is discarded. This prediction mechanism is more primitive and coarse-grained than ours, therefore enabling more inaccurate message forwarding. Moreover, the approach offers no support for disconnected operation.

All the approaches above are unable to deal with intermittent connectivity and network partitions. On the other hand, a number of approaches have been proposed in the area of delay tolerant routing [37] to deal specifically with these problems. The most basic approaches are variants of epidemic approaches [38] where messages are flooded and stored and retransmitted periodically until they expire. Chen and Murphy [11], in their Disconnected Transitive Communication paradigm, were among the first to argue for the use of utility functions. Nevertheless, their paper provides a general framework rather than a detailed instantiation, therefore aspects related to the composition of calculated delivery probabilities are almost entirely missing. In [35] an application of epidemic routing protocols to a problem of cost-effective data collection is presented, using whales as message carriers. In [25], Lindgren et al. propose a probabilistic routing approach to enable asynchronous communication among intermittently connected groups of hosts. The calculation of the delivery probabilities is based, somewhat simplistically, on the period of time of colocation of two hosts and not on a forecasted colocation probability.

Zhao et al. in [40] 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. Our approach does not assume the knowledge a priori of the movement of the potential carriers, but it is able to infer it by evaluating the history of colocation with the other hosts.

Finally, as mentioned earlier, [28] presents a Context-aware Adaptive Routing (CAR) protocol based on Kalman filters. The approach has quite a refined model of prediction over time series which we partly adopted for this work. However, this protocol only deals with unicasting and not with the routing of multiple messages through content filtering.

7. CONCLUSIONS AND FUTURE WORK

This paper presents a novel approach to content-based routing in delay-tolerant mobile ad hoc networks. It is based on an informed selection of the best carriers for messages matching content-based interests. This selection is made by taking into account predictions about contextual parameters (e.g., mobility patterns and connectivity), based on previous observations. We evaluated our approach against some of the existing content-based routing approaches for MANETs, using a realistic mobility model inspired by social interaction patterns. Results confirm the superiority of our technique under many dimensions. Future work will address the inclusion of additional contextual information—most notably residual energy—in our predictions, as encompassed by the general framework we described in Section 3. This will enable further improvements, as well as adaptation of our technique to related fields, e.g., wireless sensor networks. We also plan to port our work onto the DTN architecture defined by the DTN Research Group [16] and to perform more evaluation on a real test-bed.

¡¡¡¡¡¡ mobicom.tex

======= ¿¿¿¿¿¿¿ 1.25

8. REFERENCES

[1] R. Baldoni, R. Beraldi, G. Cugola, M. Migliavacca, and L. Querzoni. Content-based routing in highly dynamic mobile ad hoc networks. *Journal of Pervasive Computing and Communication*, 1(4), 2005.

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

[3] A. Bulut, A. K. Singh, and R. Vitenberg. Distributed data streams indexing using content-based routing paradigm. In *Proceedings of 19th IEEE Int. Parallel and Distributed Processing Symposium (IPDPS'05)*, 2005.

[4] F. Cao and J. P. Singh. Efficient event routing in content-based publish/subscribe service network. In *Proceeding of the* 23^{th} *IEEE Conference on Computer Communications (INFOCOM 2004)*, 2004.

[5] M. Caporuscio, A. Carzaniga, and A. Wolf. Design and Evaluation of a Support Service for Mobile, Wireless Publish/Subscribe Applications. *IEEE Trans. on Software Engineering*, 29(12):1059–1071, Dec. 2003.

[6] A. Carzaniga, D. Rosenblum, and A. Wolf. Design and evaluation of a wide-area event notification service. *ACM Trans. on Computer Systems*, 19(3):332–383, 2001.

[7] A. Carzaniga, M. Rutherford, and A. Wolf. A routing scheme for content-based networking. In *Proc. of INFOCOM*, Mar. 2004.

[8] A. Carzaniga and A. L. Wolf. Forwarding in a content-based network. In *Proceedings of ACM SIGCOMM 2003*, pages 163–174, Karlsruhe, Germany, Aug. 2003.

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

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

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

[12] P. Costa and G. Picco. Semi-probabilistic content-based publish-subscribe. In *Proc. of the* 25^{th} *IEEE Int. Conf. on Distributed Computing Systems (ICDCS05)*, Columbus (Ohio, USA), June 2005.

[13] G. Cugola, E. Di Nitto, and A. Fuggetta. The JEDI event-based infrastructure and its application to the development of the OPSS WFMS. *IEEE Trans. on Software Engineering*, 27(9):827–850, Sept. 2001.

[14] A. Datta, S. Quarteroni, and K. Aberer. Autonomous Gossiping: A self-organizing epidemic algorithm for selective information dissemination in mobile ad-hoc networks. In *Proc. of , Int. Conf. on Semantics of a Networked World*, June 2004.

[15] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec. The many faces of publish/subscribe.

ACM Comput. Surv., 35(2):114–131, 2003.

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

[17] S. Ganguly, S. Bhatnagar, A. Saxena, R. Izmailov, and S. Banerjee. A fast content-based data distribution infrastructure. In *Proceeding of the 25^{th} IEEE Conference on Computer Communications (INFOCOM 2006)*, Barcelona, April 2006.

[18] K. Harras, K. Almeroth, and E. Belding-Royer. Delay Tolerant Mobile Networks (DTMNs): Controlled Flooding in Sparse Mobile Networks. In *Proc of IFIP Networking*, May 2005.

[19] D. Heimbigner. Adapting publish/subscribe middleware to achieve gnutella-like functionality. In *Selected Areas in Cryptography*, pages 176–181, 2001.

[20] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In *Proc. of MobiCom*, 2000.

[21] Internet Society. Interplanetary Internet Project. http://www.ipnsig.org.

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

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

[24] S.-J. Lee, W. Su, and M. Gerla. On-demand multicast routing protocol in multihop wireless mobile networks. *MONET*, 7(6):441–453, 2002.

[25] A. Lindgren, A. Doria, and O. Schelen. Probabilistic routing in intermittently connected networks. *Mobile Computing and Communications Review*, 7(3), July 03.

[26] J. Luo, P. T. Eugster, and J.-P. Hubaux. Pilot: Probabilistic lightweight group communication system for ad hoc networks. *IEEE Trans. Mob. Comput.*, 3(2):164–179, 2004.

[27] R. Meier and V. Cahill. STEAM: Event-Based Middleware for Wireless Ad Hoc Networks. In *Proc. of the 1^{st} Int. Wkshp. on Distributed Event-Based Systems*, July 2002.

[28] M. Musolesi, S. Hailes, and C. Mascolo. Adaptive routing for intermittently connected mobile ad hoc networks. In *Proceedings of the 6th International Symposium on a World of Wireless, Mobile, and Multimedia Networks (WoWMoM 2005). Taormina, Italy.* IEEE press, June 2005.

[29] M. Musolesi and C. Mascolo. A Community based Mobility Model for Ad Hoc Network Research. CS-UCL Research Note RN/05/31, Department of Computer Science, University College London, December 2005.

[30] M. Musolesi and C. Mascolo. Evaluating context information predictability for autonomic communication. In *Proceedings of 2nd IEEE Workshop on Autonomic Communications and Computing (ACC'06). Co-located with 7th IEEE Int. Symp. WoWMoM'06*, Niagara Falls, NY, June 2006. IEEE Computer Society Press.

[31] V. Muthusamy, M. Petrovic, and H.-A. Jacobsen. Effects of routing computations in content-based routing networks with mobile data sources. In *Proceedings of the 11th Annual International Conference on Mobile Computing and Networking, MOBICOM 2005*, pages 103–116, Cologne, Germany, August 2005.

[32] A. S. Pentland, R. Fletcher, and H. Hasson. Daknet: Rethinking connectivity in developing nations. *IEEE Computer*, 37(1):78–83, January 2004.

[33] E. Royer and C. Perkins. Multicast operation of the ad-hoc on-demand distance vector routing protocol. In *Mobile Computing and Networking*, pages 207–218, 1999.

[34] R. C. Shah, S. Roy, S. Jain, and W. Brunette. Data MULEs: Modelling a Three-Tier Architecture for Sparse Sensor Networks. Technical Report IRS-TR-03-001, Intel Corporation, January 2003.

[35] 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 *Procs. of MobiHoc'03*, June 2003.

[36] R. Strom, G. Banavar, T. Chandra, M. Kaplan, K. Miller, B. Mukherjee, D. Sturman, and M. Ward. Gryphon: An information flow based approach to message brokering. In *International Symposium on Software Reliability Engineering (ISSRE '98)*, 1998.

[37] J. Sushant, K. Fall, and R. Patra. Routing in a delay tolerant network. In *Procs. of SIGCOMM'04*, August 2004.

[38] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical Report CS-2000-06, Department of Computer Science, Duke University, 2000.

[39] A. Varga. The OMNeT++ discrete event simulation system. In *Proceedings of ESM'2001*, Prague, 2001.

[40] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Procs. 5th ACM International Symposiunm on Mobile Ad Hoc Networking and Computing (MobiHoc'04)*, May 2004.

[41] H. Zhou and S. Singh. Content based multicast (CBM) in ad hoc networks. In *Proc. of the Workshop on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, Aug. 2000.