Structure-less Content-Based Routing in Mobile Ad Hoc Networks *

Roberto BALDONI* Roberto BERALDI* Gianpaolo CUGOLA+ Matteo MIGLIAVACCA+ Leonardo QUERZONI*

* Università di Roma "La Sapienza"
Via Salaria 113, I-00198 Roma - Italy

+Politecnico di Milano
P.zza L. Da Vinci, I-20133 Milano - Italy

Abstract

The decoupling and asynchrony properties of the content-based publish-subscribe paradigm makes it very appealing for dynamic wireless networks, like those that often occur in pervasive computing scenarios. Unfortunately, none of the currently available content-based publish-subscribe middleware fit the requirements of such extreme scenarios in which the network is subject to very frequent topological reconfigurations due to the mobility of nodes.

In this paper we propose a protocol for content-based message dissemination tailored to Mobile Ad Hoc Networks (MANETs) with frequent topological changes. Message routing occurs without the support of any network-wide dispatching infrastructure thus eliminating the issue of maintaining such logical topology on top of a time varying physical topology. The paper reports an extensive simulation study, which provides numerical evidence of the effectiveness of the approach.

Key words: MANET, Publish-Subscribe, middleware.

1 Introduction

A Mobile Ad Hoc Network (MANET) is a self-organizing adaptive network composed of a dynamic collection of wireless mobile devices that can communicate and move at the same time. MANETs can be formed and de-formed on-the-fly without neither the support of a centralized administration function [13], nor fixed wired infrastructures. These exclusive characteristics classify them as a natural support to pervasive computing.

One of the main issue in such a class of networks is to provide the application layer with suitable communication abstractions that can fit the very dynamic nature of the underlying network. Content-based publish-subscribe (cb-ps) is a communication paradigm that decouples components of a distributed application in time, space, and flow [9] and thus is very appealing for such dynamic contexts.

A component of a cb-ps system can act as a *publisher* of anonymous information, called event notifications or simply *messages*, or as a *subscriber* of messages whose content matches a given *predicate*. The decoupling mentioned above is obtained through the fact that publishers and subscribers do not know each other: cb-ps operations, and in particular the delivery of a message to all the interested subscribers, are realized by a *dispatching service*.

The implementation of an efficient dispatching service for a MANET is very challenging. In fixed networks, the dispatching service is often realized by a single, centralized server, which stores predicates that express the interests of subscribers and use them to forward messages coming from publishers. Clearly this approach cannot be adopted in MANETs, in which nodes need to communicate without the support of any stable infrastructure.

More recently, cb-ps middlewares which adopt a distributed implementation of the dispatching service have been developed. In this case several distributed components, called *brokers*, are connected according to a convenient *overlay dispatching network*, e.g. a spanning tree, and collaborate to route messages from publishers to subscribers. In principle this case is more suitable to MANETs, since a broker could run on each mobile node, but the overhead required to maintain paths between the brokers makes this approach unsuitable for settings that exhibit even a discrete degree of mobility.

In this paper we explore a different approach, whose key aspect is the lack of any predefined logical network-wide structure as a support to message diffusion. We realize a distributed implementation of the dispatching service by running a broker on each mobile node of the MANET but, differently from the traditional case, we do not try to keep a stable overlay dispatching network connecting them. Conversely, we leverage off the broadcast communications available in a MANET to forward messages to multiple destinations and let each receiving broker to autonomously de-

*The work described in this paper was partially supported by the Italian Ministry of Education, University, and Research (MIUR) under the IS-MANET and VICOM projects, and by the European Community under the IST-004536 RUNES project.

cide if and when re-forwarding the message on the basis of an estimation of its proximity to potential subscribers for that message. In particular, we use the time elapsed since two nodes have lost direct connection, i.e., they went out from each other's transmission range, as an estimate of their proximity.

The rest of this paper is organized as follows: Section 2 discusses background and related work. Section 3 briefly motivates our work and gives a general description of the routing protocol we propose, while Section 4 provides the details of the protocol. Finally, Section 5 presents the results of an extensive campaign of simulation, which validates our approach, while Section 6 provides some concluding remarks and describes future work.

2 Background and Related Work

This section gives first a brief general description of the cb-pb model of communication, followed by the main contributions related to MANETs appeared in the literature.

2.1 Content-Based routing

Applications exploiting a publish-subscribe middleware are organized as a collection of components, which interact by *publishing* messages and by *subscribing* to the classes of messages they are interested in. The core component of the middleware, the *dispatcher*, is responsible for collecting subscriptions and forwarding messages from publishers to subscribers.

Currently available publish-subscribe middleware differ along several dimensions among which the most relevant are the expressiveness of the subscription language, the architecture of the dispatcher, and the forwarding strategy [2, 12, 3].

The expressiveness of the subscription language draws a line between *subject-based* middleware, where subscriptions identify only classes of messages belonging to a given channel or subject, and *content-based* middleware, where subscriptions contain expressions (called *predicates*) that allow sophisticated matching on the message content.

In general, the architecture of the dispatcher can be either centralized or distributed. In the former case a single component of the middleware, running on a given machine, is in charge of collecting subscriptions and dispatching messages. Both publishers and subscribers distributed on the network are attached to this component through some kind of network link (e.g., a TCP channel).

When a distributed dispatcher is used, a set of *brokers* are interconnected in an *overlay dispatching network* and cooperatively route subscriptions and messages sent by components attached to them. This strategy increases the scalability of the system and is usually adopted by middleware tailored to large scale networks.

Middleware that exploit a distributed dispatcher can be further classified according to the interconnection topology of brokers and the strategy exploited for message dissemination. We do not consider here solutions based on multicast routing protocols as they are conceptually simple.

The simplest approach is *message forwarding* in which brokers are connected to form an unrooted tree. Publishers send messages to their associated broker, which forwards them to all other brokers by following the tree topology. Moreover, each broker keeps track of the subscriptions coming from the components directly connected to it into a local *subscription table*, which is used to determine the components, if any, that should receive incoming messages.

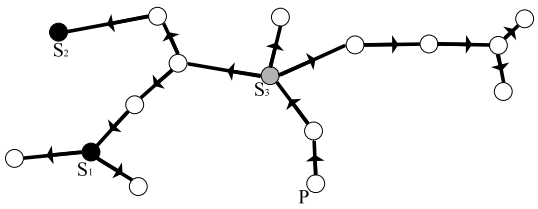
This solution inevitably results in high overhead as all messages are sent to all brokers, regardless if an attached component has subscribed. An alternative and more widely used strategy is *subscription forwarding*, which limits this overhead by spreading knowledge about subscriptions beyond the first broker along the unrooted tree connecting brokers. Specifically, when a broker receives a subscription from one of its peers, not only it stores the associated predicate into its subscription table as in message forwarding, but also it forwards the predicates to the neighboring brokers[1].

In figure 1 the above strategies are compared by showing the same situation, characterized by a distributed dispatcher composed of 16 brokers. Two of them, namely S_1 and S_2, have components connected (not shown to avoid cluttering the figure) that subscribed to the same predicate, represented as a black color, while broker S_3 received a "gray" subscription. Finally, broker P received a message matching the black predicate but not the gray one. The path followed by this message is shown through thick, directed lines, while black and gray arrows represent the content of subscription tables. More specifically, each broker has a colored arrow oriented towards another broker if it received the corresponding subscription from that broker. Figure 1(a) shows how message forwarding incurs in the highest overhead at publishing time, while it does not require subscriptions to be propagated. Subscription forwarding (Figure 1(b)) fills the subscription tables of each broker but offers the best performance at publishing time.

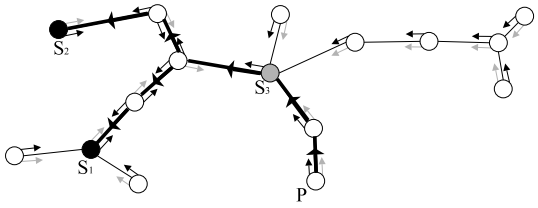
2.2 Content-Based Routing in MANET

The solutions described above are characterized by a permanent network-wide structure that supports message and, optionally, subscription forwarding. It is easy to argue that a naive application of such a structure-based approach to

[1] This basic scheme can be optimized, e.g., by exploiting the notion of "coverage" among predicates, or by aggregating them, as described in [2].



(a) Message forwarding



(b) Subscription forwarding

Figure 1. Publish/subscribe routing strategies.

mobile networks is inefficient, since this requires to maintain a set of logical connections between mobile brokers. Moreover, due to mobility, it may be often the case that the topology of the overlay network of brokers doesn't reflect the actual position of the nodes, and consequently the topology of the physical network.

Some of the authors of this paper already addressed this problem by introducing mechanisms that allow brokers to react to changes occurring at the networking layer by adapting the topology of the overlay dispatching network to the actual networking topology [4, 11]. Unfortunately, none of these approaches fit efficiently enough the case when topological changes become frequent.

Yoneki and Bacon proposed to use the On Demand Multicast Routing Protocol (ODMRP) for constructing an optimized dissemination mesh by applying the context from a cb-ps system to the multicast protocol [14]. Bloom filters are used to summarize subscriptions. In this case, however, the cb-ps scheme is actually approximated to a topic based one, and the cost of this approximation is clearly an intrinsic limitation to such a solution. No performance evaluation is indeed provided in the paper.

Datta et al. introduce a generic epidemic algorithm for selective dissemination of information, dubbed autonomous gossiping (A/G). The algorithm can also be applied to content-based dissemination in a MANET. It associates an utility to each data item. Depending on the hospitality received at the present host, data items decide to either continue to reside, migrate or replicate to another host with a more suitable profile and/or goal zone, and the data items associated utility is used in the decision process. The paper however doesn't report a detailed description of a the algorithm and show only some generic performance result [5].

3 Motivation and General Idea

The idea of a centralized server acting as the dispatcher is clearly totally in contrast with the requirements of MANET. On the other hand, event routing based on a distributed set of brokers interconnected in an overlay dispatching network is hard to implement efficiently in a MANET due to the cost required to cope with the frequent changes in the topology of the physical network.

Consequently, our idea was to develop a cb-ps routing protocol that does not require any predefined logical network-wide structure as a support to message dissemination. In this section we provide an informal description of the main ideas behind this proposal. Details are given in the next section.

3.1 Assumptions

We assume that the cb-ps system is composed of a fixed set of N brokers, each running on a different mobile node, i.e., device. When necessary to stress the difference we will use the notation n_i to indicate the i-th mobile node of the network, and b_i to refer to the broker running on that node.

When an application component running on a node n_i wants to receive some message, it subscribes to b_i, which then stores the predicate associated with the subscription into its subscription table. Similarly, to publish a message, a component running on a node n_i send it to the broker b_i, which acts as an entry point to the cb-ps dispatching service for every component running on node n_i.

For efficient transmission to other nodes, we assume that the interests of all the application components connected with a broker b_i can be condensed in a single *predicate*, which reflects the content of b_i's subscription table[2].

3.2 The Idea

To develop our protocol we started from the observation that in a MANET a broker b_n can be efficiently reached starting from a broker b_0 if we can find a sequence of brokers, say $b_1, b_2, .., b_{n-1}$, such that their euclidian distance from b_n are strictly decreasing and such that b_i and b_{i+1} are

[2]Note that this assumption is realistic for content-based publish-subscribe systems whose subscription language is usually powerful enough to allow it.

adjacent for each $i \in [0, n-1]$. We say that two brokers are adjacent if the corresponding nodes are one-hop neighbors, i.e. they can directly communicate with each other.

Since we do not want to rely on any positioning device, e.g., a GPS, we decided to estimate the distance between two brokers by measuring the time elapsed since they were most recently adjacent to each other. This estimation technique is very simple (a beacon signal is sufficient for this purpose) and reasonably accurate, provided that the elapsed time is not too long. Positive results are reported in [7] where it was originally defined and applied for reducing the cost of a network-wide path search and in [1], where it was exploited for unicast routing.

The second goal that guided the development of our protocol was that of keeping any routing decision as simple and "distributed" as possible. In particular this is obtained by letting each broker to autonomously decide if it has to act as a forwarder for a message or not. When a broker sends a message it doesn't provide any explicit indication (e.g. the address) about which of its own adjacent brokers should actually forward the message again. Rather, it simply broadcasts the message and let the adjacent brokers autonomously determine whenever re-sending the message or not. Although a broker has to process each message it receives, we argue that this is an efficient technique: it can exploit the broadcast nature of the wireless transmissions to send multiple copies of the same message via a single transmission; it avoids the burden of link breakage detection and, even more important, it provides an intrinsic resilience to the topological changes caused by the mobility of the nodes.

Let now consider how the basic message forwarding scheme works. Each broker b_i periodically broadcasts a beacon message containing the predicate that summarize its own subscription table. A broker b_j, which is adjacent with b_i, receives this message and stores the predicate together with the time it received the beacon into its *hint table*. This mechanism allows each broker to determine the number of beacons missed from any other broker. This value, which is infinite if the two brokers never come in contact and zero if they are still adjacent, will be called the *hint* h_{ji} of b_j with respect to b_i and, as mentioned above, it will be used as an estimate of the distance of b_j from b_i.

Moreover, to implement the "decreasing distance" routing mechanism described above, each message m carries a *destination list*: the (estimated) list of brokers interested in receiving the message, each coupled with the lowest hint computed by the brokers that forwarded the message so far[3]. As an example, the destination list of a message m includes a couple $< i, h >$ if broker b_i is known to be interested in receiving the message (i.e. m matches a subscription issued by some subscriber attached to b_i) and the lowest number of b_i's beacons missed by all the brokers that forwarded m. The message has also a unique network-wide identifier provided by the source broker, we will refer to it with the notation $m.id$.

[3]Please note that we are considering application level messages and thus the size of a message is virtually unbounded.

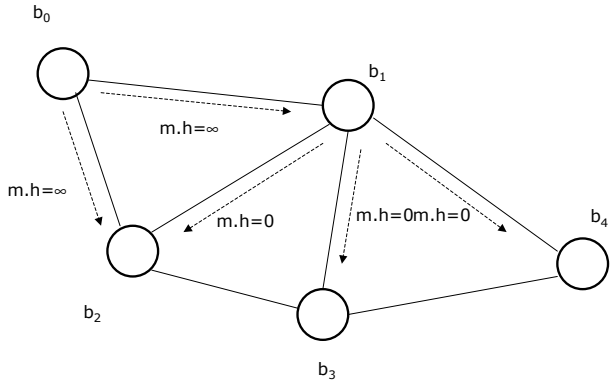


Figure 2. The basic coordination mechanism.

Suppose now that at time t the broker b_i receives a message m for the first time. It will resend the message if (i) it is aware of some new broker not mentioned in the destination list carried by m or (ii) its hint table holds for some broker b_k a hint lower than that associated to the same broker b_k into m's destination list.

Such a condition is in general not sufficient to trigger the actual transmission of the message. The broker b_i, in fact, schedules the transmission of the message after a delay proportional to h_{ik} (the lowest hint is considered if such a condition holds for more than one broker, see later). If during such a time interval it doesn't hear the same message again (i.e. a message with the same identifier) then the transmission will take place. Otherwise b_i silently drops the message. The rational behind this decision is to avoid that two adjacent brokers will send the same message and to let brokers closest to some destination to "suppress" transmission of adjacent brokers less close.

In order to clarify this basic mechanism, let us consider the Figure 2. The broker b_0 publishes a message matching the broker b_4's subscriptions. The message is sent via broadcast and received both by b_1 and b_2 (an arrow represents the transmission of the message). Assume that b_0 and b_4 have never came in contact so that the destination table carried by m is initially empty. Assume that b_2 missed $h_{24} = 5$ beacons from b_4. The broker b_2 schedules the transmission with some delay proportional to 5. However, b_1 is adjacent to b_4 (i.e., $h_{14} = 0$) and immediately sends the message. Broker b_2, on receiving the message from b_1 aborts the scheduled transmission and silently drops m. Moreover,since the hint carried by the message sent by b_1 is zero, the broker b_3 ignores the message (by definition zero

```
Node state
  st: Array of ⟨pred, id⟩
  ht: Array of ⟨id, pred, last⟩
Every Δt seconds do
begin
    p ← summarize(st)
    broadcast(p)
    cleanUp(ht)
end
predicateReceived(p, n)
begin
    if ∃k such that ht[k].id = n then
        ht[k].pred ← p
        ht[k].last ← currentTime
    else
        append(ht,⟨n, p, currentTime⟩)
    end
end
forward(m)
begin
    if message m' s.t. m'.id = m.id was already received then
        de-schedule transmission of m'
        return
    end
    foreach ⟨pred, id⟩ in st do
        if pred.matches(m) then
            m.setHint(myId,0)
            forwardToClient(m, id)
        end
    end
    minHint ← 1.0
    matched ← false
    foreach ⟨id, pred, last⟩ in ht do
        h ← hintFor(id)
        if pred.matches(m) and (id ∉ destinationOf(m) or
        h < m.getHint(id)) then
            matched ← true
            m.setHint(id,h)
            if minHint > h then minHint ← h
        end
    end
    if not matched and m.credit⟩0 then
        m.credit ← m.credit − 1
        matched ← true
    end
    if matched then
        schedule m for transmission at
        currenTime + minHint/3 + randomDelay
    end
end
```

Figure 3. The Hint-Driven Routing Protocol

is the lowest possible hint).

4 Protocol Details

The pseudo-code of our protocol, called Hint Driven Routing protocol, is reported in Figure 3.

Each broker maintains the following data structures:

- A subscription table organized as an array st of pairs $\langle pred, id\rangle$, where $pred$ is the predicate carried by a subscription and id is the identifier of the subscriber that issued the subscription.

- A hint table organized as an array ht of triples $\langle id, pred, last\rangle$, where id is a node identifier, $pred$ is the predicate received from that node, which summarizes its subscription table, and $last$ the time when the predicate was received.

Each broker b_i beacons a summary of the predicates stored into its subscription table every ΔT seconds, using a broadcast packet. A broker b_j that is within the transmission range of b_i receives such a beacon and executes the procedure `predicateReceived` of Figure 3 to update its hint table. If the same predicate was already received from the same node, then the entry is refreshed, i.e. the time associated to the entry is set to the current time. Otherwise a new element is appended to the table. An entry is deleted from the table if it was not refreshed for more than a timeout value experimentally set to $10\Delta T$, i.e., if more than 10 beacons where missed.

The information stored in the hint table, together with the fact that the beacon interval ΔT is known globally, allow each broker b_i to calculate the hint h_{ij} at time t with respect to any other broker b_j as follows: h_{ij} is infinite if b_j is not present into b_i's hint table; otherwise it is a value in the range $[0..1]$ calculated as the number of b_j's beacons missed by b_i divided by 10.

Remembering from previous section that each message carries a destination list composed of couples $\langle id, hint\rangle$, it is now time to describe how message forwarding proceeds. On receiving a message m a broker checks if the same message, i.e., a message with the same identifier, has been received before. If this is the case, the message is removed from the list of messages scheduled for trasmission (if present) and it is dropped without any further processing.

If m was never received before then the broker checks if it matches some predicate into its subscription table. If this is the case, the broker delivers m to the corresponding subscriber and set the hint for itself into the m's destination list to 0 (this will avoid to trigger further transmissions aiming at hitting the broker, as clarified next). Furtherly, the broker determines if it has to re-forward the message. This happens when m matches at least a predicate advised by a broker b_i such that: (1) b_i doesn't belong to the destination list of the message or (2) the hint for b_i computed by the receiving broker according to its hint table is less than the one carried into the message.

In both cases the retransmission of the message m is scheduled after a delay proportional to the hint for b_i owned by the receiving broker. When more than one broker exits that satisfies the conditions above, the delay is determined by the lowest hint.

If none of the above cases hold, message should be dropped, but in order to increase delivery at the price of some more traffic, a new chance is given to the message for being forwarded. To this end, a message also carries an integer value, called the $credit$ of the message, which represents the number of times a broker can force the retrans-

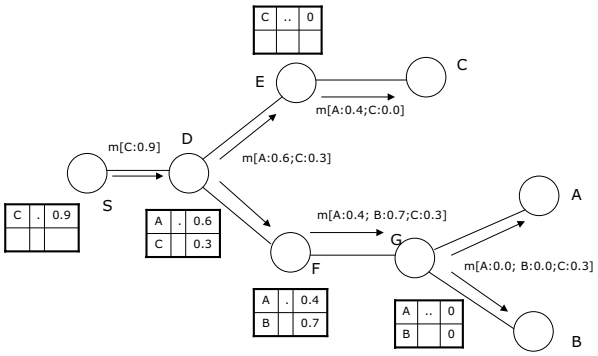


Figure 4. An example of message routing.

mission of the message despite no such a condition holds. As shown in Figure 3, if such a case occurs, the message is scheduled for transmission with the delay associated to the maximum hint, i.e. one. This way forwarding due to credit tends to be cancelled by forwarding due to hints.

Figure 4 portraits an example of message forwarding. The hint table of a node is reported close to the node. For the sake of simplicity instead of storing the absolute time when the node received a beacon message, the last column of the table stores the hint computed as explained above.

Suppose the node S generates a message matching subscriptions on nodes A, B, C. The source node is only aware of the subscriptions at node C. The hint for C is 0.9. It then sends the message with destination list $[C : 0.9]$. On receiving the message, node D decides the forward m because it knows another node, node A, which is interested in the message. Moreover, the hint for C is lower than 0.9. Nodes E and F receive the message (they are both neighbor of D). Node E resends the message since it has hint 0 for C, while node F because it is aware of node B. Finally, G broadcasts the message to A and B.

5 Evaluation

To asses the performance of our protocol we have estimated the following performance metrics via simulations

- *delivery*: the average ratio of subscribers that received a message to the total number subscribers interested in the message.
- *overhead*: the average total number of link layer packets generated in the network for each delivered message. The overhead includes beacon packets.

At the best of our knowledge no detailed descriptions of content-based routing protocols for MANETs are given in the literature; thus, we decided to use a gossip protocol as baseline to compare our protocol. This is perhaps the most simple structure-less protocol for event dissemination. In the gossip protocol we considered, brokers send a message via broadcast and when another broker hears a message for the first time it re-sends it with forwarding probability $p \in (0, 1]$.

To evaluate the performance of our protocol we used the open source network simulator J-Sim [6]. Among other interesting features, it provides a full simulation of the 802.11 protocol stack as well as a detailed propagation model.

Parameter	Default Value
Number of nodes	$N = 100$
Field area	$A = 1000 \times 1000\ m^2$
Minimum speed	$S_m = 10\ m/s$
Maximum speed	$S_M = 20\ m/s$
Number of publishers	$N_p = 2$
Publishing rate (for each publisher)	$Pr = 0.5\ msg/sec$
Number of subscribers	$N_s = 10$
Beacon interval	$\Delta t = 5\ sec$
Message credits	$Cr = 0$
Forwarding probability	$p = 0.5$

Figure 5. Default simulation parameters.

5.1 Simulation Settings

The reference scenario we considered is that of a MANET composed of a number of nodes dispersed in a square field, which move around according to a random waypoint mobility model [10]. Each node randomly chooses a destination and starts moving toward it at a random speed. Once the destination has been reached, the node randomly determines another destination, and continues in that direction with a new randomly chosen speed.

The total number N of nodes, the area A of the field, and the minimum S_m and maximum S_M speed nodes can move at are the main physical parameters that characterize the simulated scenario.

A broker runs on each node and it has either a single publisher or a single subscriber attached to it. We assume that N_p publishers produce messages of interest for a N_s subscribers at a publishing rate of Pr msg/s. These parameters characterize the cb-ps application model.

To reflect a realistic open field scenario, we choose a two rays ground propagation model with a random transmission range varying between 100 and 200 meters.

Finally, the main parameters that characterize our protocol are the beaconing interval Δt and the number of credits Cr initially assigned to a message.

Table 5 lists the simulation parameters and their default values.

5.2 Simulation Results

To have a baseline to start evaluating our protocol we first simulate the gossip protocol in our reference scenario (see

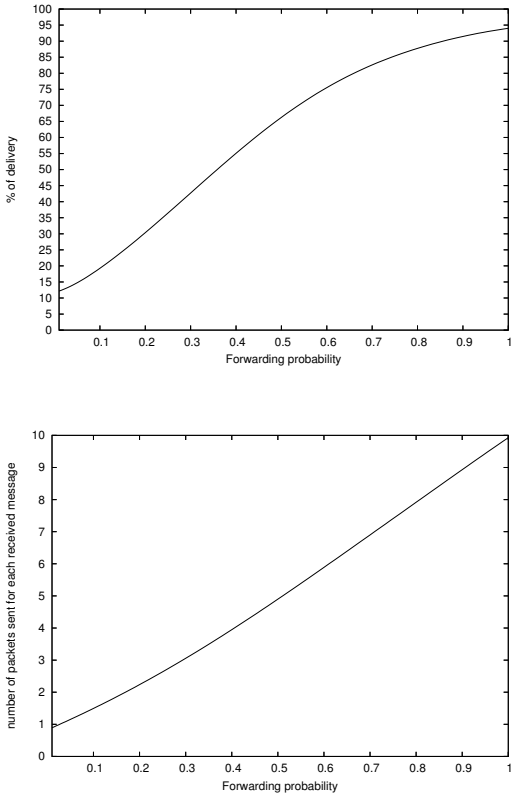


Figure 6. Impact of forwarding probability on delivery (top) and network load (bottom).

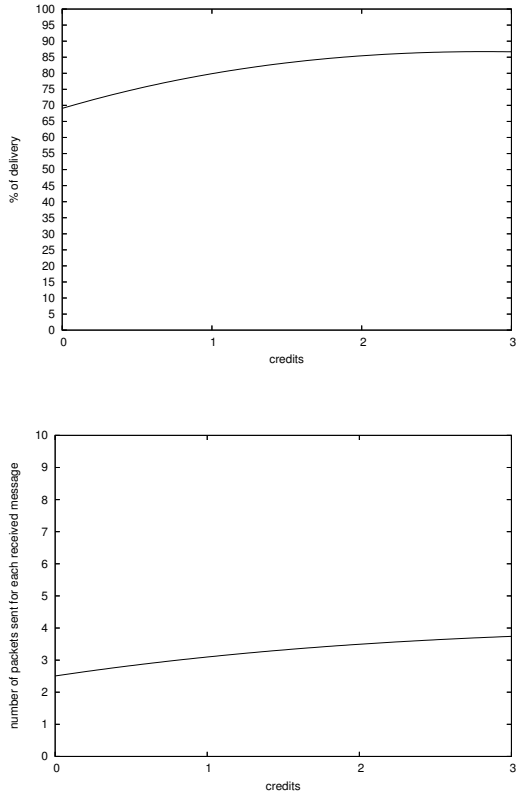


Figure 7. Impact of credits on delivery (top) and network load (bottom).

Figure 5) varying the forwarding probability p. Results are reported in Figure 6. It is worth observing how the delivery exhibits the topical bimodal behavior of gossip protocols [8]. We also note how 100% delivery is never reached due to collisions and network partitioning while a reasonable percentage of delivery, say more than half the number of interested subscribers, can be achieved at the cost of at least 5 packets per delivered message.

Figure 7 shows the performance of our protocol as a function of the number of credits under the same reference scenario. Although the maximum delivery fraction is slightly lower than the one measured under gossip, reasonable high values can be reached at much lower cost. For example, a delivery fraction of 0.7 can be reached with no credit at less than half the cost required under gossip (respectively 2.5 and 5.5 packets per message). By increasing the number of credits the delivery can be increased while still keeping a high convenience.

The next point to evaluate is how the number of subscribers affects the protocol's performance. Figure 8 shows the delivery and cost as a function of the number of subscribers measured under a different number of credits. The performance of the gossip protocol are also reported. It is interesting to note the effectiveness of credits mechanism as a means to increase the delivery, which is particular useful under a low number of subscribers. Our protocol is always able to assure a high delivery fraction (more than 85%) independently of the number of subscribers and at progressively decreasing cost. Clearly the efficiency of the gossip algorithm increases with the number of subscribers since flooding becomes by definition the most appropriate dissemination algorithm.

Another parameter that may influence performance is the rate of published messages. As shown in Figure 9, our protocol is only very marginally influenced by this parameter, while gossip and flooding are more sensible. This can be explained by remembering, from previous simulations, that gossip loads the network much more than our protocol. As a consequence, when the publishing rate increases, gossip suffers from a relevant number of collisions, which do not

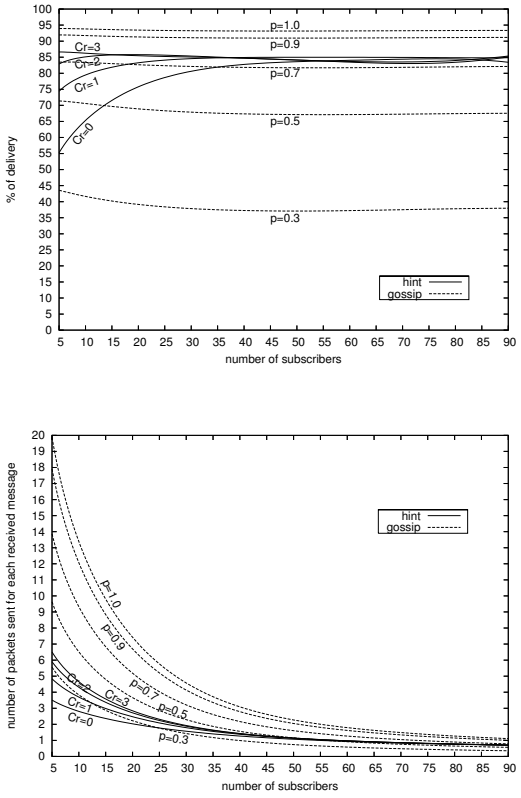


Figure 8. Effect of increasing the number of subscribers on delivery (top) and network load (bottom).

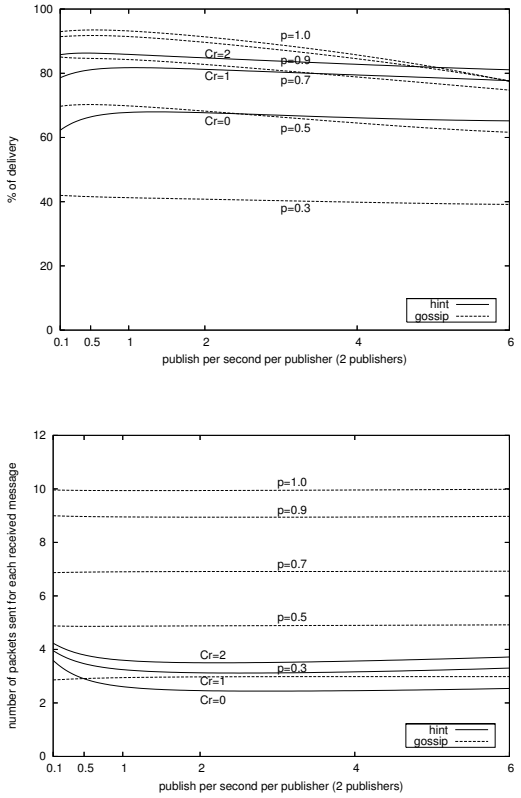


Figure 9. Effect of increasing the publishing rate on delivery (top) and network load (bottom).

occur when our protocol is used. It is worth noticing that an increase in the publishing rate also increases the efficiency of our protocol because it reduces the impact of beaconing traffic.

In the next figures we report how mobility affects the performance of the protocol. Figure 10 shows the performance as a function of the speed under different beacon interval.

Recall that a broker uses the number of missed beacons as an estimation of its distance from a broker. Hence, it is important to assure that such a missed-beacon distance correlation is valid for the entries stored in the hint table. Entries should not be removed too early (i.e. when the correlation still holds) or too late (when the correlation is too weak). Our protocol deletes an entry from the table after 10 missed beacons. Under low mobility a short beacon interval thus results in removing valid entries from the table (i.e. those for which the correlation is still valid). Similarly, under a high mobility degree a long beacon interval causes stale entries in the table. This explains the behavior of the delivery.

The graphic at the bottom of Figure 10 shows how the cost is only slightly influenced by the beacon interval and increases smoothly with the speed.

We will now analyze the scalability of the protocols: the first graphic, see Figure 11, is obtained by increasing both A and N at the same time, thus keeping the density of nodes at a constant value. Given the high variation in the protocol performance when changing the subscriber density, as demonstrated in Figure 8, we fixed the percentage of subscribers with respect to the total number of nodes N to 10%. We also kept a fixed percentage of publisher to 2% of N and the publishing rate (per publisher) constant. All the protocols maintain their performance as the network size is increased, with gossip decreasing slightly its delivery and our protocol marginally increasing it.

The second scalability test, see Figures 12 and 13, consists in increasing the number of nodes N while keeping the area A constant, hence producing an increase in the node

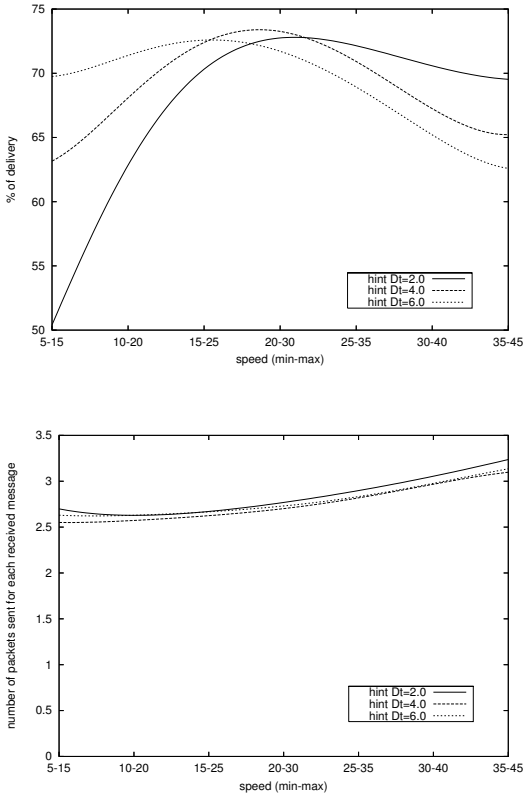


Figure 10. Delivery and load versus speed at different beaconing interval.

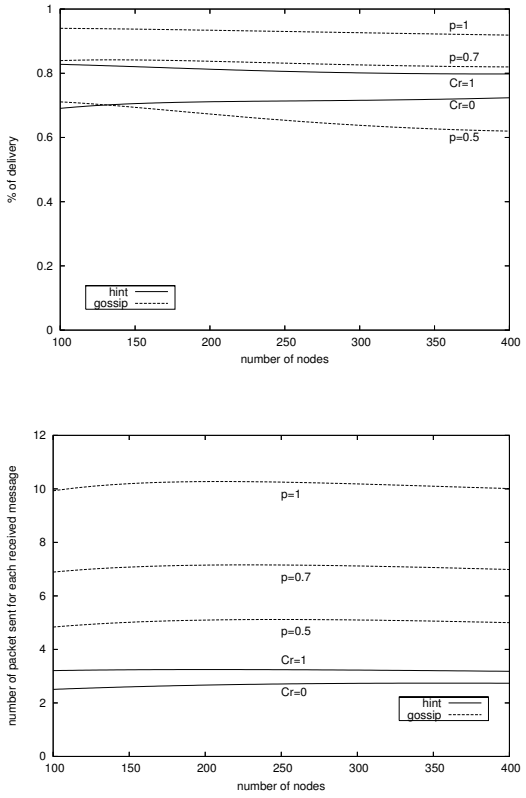


Figure 11. Delivery and load as network size increases.

density. We observe an interesting phenomenon here which is due to the increasing number of collisions: a low gossiping probability provides better performance as the density increases, while by using a higher probability performance starts decreasing after a given value of nodes. Our protocol seems to be more resistant to collisions because of the suppression mechanism it uses, which can be considered a form of auto-adaptation to the density of the network. Here, as usual, the efficiency of our protocol is far better than gossip and is rather constant with respect to the increasing density.

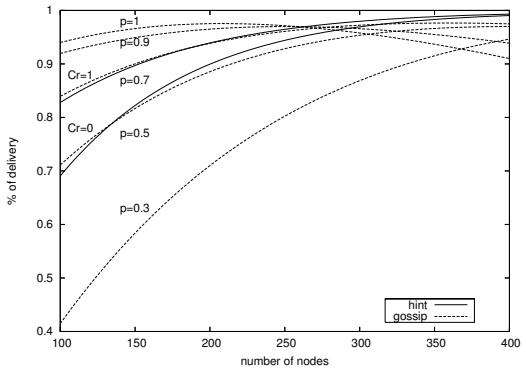


Figure 12. Delivery as node density increases.

6 Conclusions

In this paper we have explored a new approach to content-based routing in Mobile Ad Hoc Networks. The protocol doesn't require any network-wide structure to support routing decisions. Rather, it uses broadcast to efficiently send a message to all neighbor nodes and defers to them the decision to forward the message based on an estimation of their distance from a potential subscriber of the message.

The protocol is very resilient to topological changes and can thus be best used in settings characterized by a high

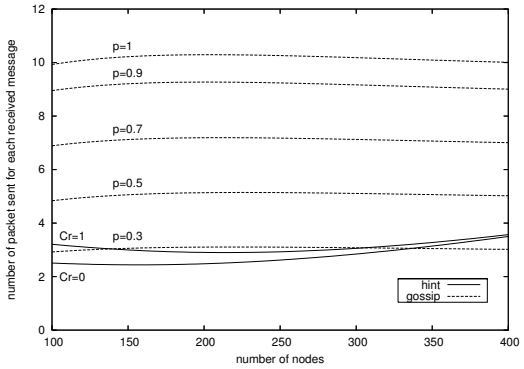


Figure 13. Network load as node density increases.

mobility degree. We have shown through simulations that messages can be delivered with high probability to the interested subscribers at a low cost. We are currently investigating how to improve the performance by increasing the accuracy of the estimations taking other information, e.g. the permanence of a node close to another, into account.

Acknowledgments - The authors would like to thank Gian Pietro Picco and Paolo Costa for the interesting discussions about content-based publish/subscribe systems in mobile ad-hoc networks.

References

[1] R. Beraldi. On message delivery through approximate information in highly dynamic mobile ad hoc networks. In *The Seventh International Symposium on Wireless Personal Multimedia Commuications (WPMC2004)*, 2004.

[2] 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, Aug. 2001.

[3] 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.

[4] G. Cugola, D. Frey, A. Murphy, and G. Picco. Minimizing the Reconfiguration Overhead in Content-Based Publ ish-Subscribe. In *Proceeding of the 2004 Symposium on Applied Computing*, pages 1134 – 1140. ACM Press, 2004. Available at `http://portal.acm.org/citation.cfm?doid=968130`.

[5] A. Datta, S. Quarteroni, and K. Aberer. Autonomous gossiping: A self-organizing epidemic algorithm for selective information dissemination in wireless mobile ad-hoc networks. In *In Proceedings of the International Conference on Semantics of a Networked World (IC-SNW04)*. LNCS, Springer Verlag, 2004.

[6] J-Sim Web Page. `http://www.j-sim.org`.

[7] H. Dubois-Ferriere, M. Grossglauser, and M. Vetterli. Age matters: efficient route discovery in mobile ad hoc networks using encounter ages. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 257–266, Annapolis, Maryland, USA, 2003. ACM Press.

[8] Z. H. et al. Gossip-based ad-hoc routing. In *Proceedings of IEEE INFOCOM 2002*, New York, USA, 2002. IEEE.

[9] P. Eugster, P. Felber, R. Guerraoui, and A.-M. Kermarrec. The many faces of publish/subscribe. Technical Report TR-DSC-2201-4, EPFL, 2001.

[10] D. Johnson. Routing in Ad Hoc Networks of Mobile Hosts. In *Proc. of the Workshop on Mobile Computing Systems and Applications*, pages 158–163, 1994.

[11] G. Picco, G. Cugola, and A. Murphy. Efficient Content-Based Event Dispatching in Presence of Topological Reconfiguration. In *Proc. of the 23^{rd} Int. Conf. on Distributed Computing Systems (ICDCS03)*, pages 234–243. ACM Press, May 2003.

[12] D. Rosenblum and A. Wolf. A Design Framework for Internet-Scale Event Observation and Notification. In *Proc. of the 6^{th} European Software Engineering Conf. held jointly with the 5^{th} Symp. on the Foundations of Software Engineering (ESEC/FSE97)*, LNCS 1301, Zurich (Switzerland), Sept. 1997. Springer.

[13] C.-K. Toh. *Ad hoc Mobile Wireless Networks*. Prentice Hall PTR, Upper Saddle River, 2002.

[14] E. Yoneki and J. Bacon. An adaptive approach to content-based subscription in mobile ad hoc networks. In *Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW04)*. IEEE, 2004.