Self-adaptive and stateless broadcast in delay and disruption tolerant networks

Francesco Giudici, Elena Pagani, Gian Paolo Rossi
Information Science and Communication Department, Università degli Studi di Milano
v.Comelico 39, Milano, Italy
Email: {fgiudici,pagani,rossi}@dico.unimi.it ; phone: +39 02 5031 6271

***Abstract*—In delay and disruption tolerant networks, DTNs, the broadcast communications have been so far disregarded under the conviction that their cost is unaffordable in the presence of highly sparse and mobile nodes. This paper defines the problem of a topology-independent broadcast in DTNs in terms of *effectiveness* and *efficiency*, and provides some interesting contribution to understand broadcast in the DTN scenario. Firstly, the paper shows that it is possible to design simple self-adaptive control mechanisms that keep the broadcast overhead surprisingly low, while ensuring high node coverage. Secondly, the paper throws the attention onto the fact that, despite the effective control mechanism, a sender-based broadcast has a cost to reach the last 10% of nodes much higher than the cost of reaching the first 90%. Finally, it shows that, under certain simplified constraints, a weak reliable broadcast service can be achieved without relevant extra costs over the best effort service. All the above points are discussed with the support of simulation results.**

Keywords: mobile networking; opportunistic networks; epidemic algorithms; performance evaluation

I. Introduction

In a delay and disruption tolerant network, or DTN [6], the network connectivity is intermittent, the radio links are poor and unstable, the nodes dynamically move and are sparsely distributed. These critical conditions have imposed a departure from the routing mechanisms viable in ad hoc networks and the adoption of a topology independent, *store-carry and forward* paradigm that allows the creation of an improvised path towards the destination(s) by exploiting the relay opportunity of nodes happening to be in the radio range as the effect of mobility. So far, in such a critical scenario, the research mainly focused on the problem of providing unicast communications (e.g., [5], [9], [10], [1], [14]). By contrast, the one-to-all communication scheme has not received the same attention despite the fact that its service is strategic to support protocols at both application and routing levels. For instance, a broadcast service is required to diffuse scoped advertisements – e.g. about available services or events – and summaries [11], to support podcasting [12], to upload software patches or new parameter settings – e.g. in environmental observation systems – or to diffuse acknowledgements, or cure, packets [8]. In a DTN, the broadcast of a message to the entire set of destinations still has to satisfy the general requirements of effective delivery and efficient use of resources. Effectiveness is satisfied when the protocol achieves a node coverage arbitrarily close to 1 and it is relatively simple to reach because mobility helps to quickly deliver the message. By contrast, efficiency is the main source of worries in the design of a broadcast protocol. In fact, an effective, best effort, broadcast can be designed by exploiting one of the gossip-based mechanisms that have been proposed in the literature in a few slightly different alternatives and even purposes, e.g. [16], but always starting from the following basic scheme: when a node has in its cache a message m to diffuse, it forwards m to one or more (and possibly all) encountered neighbors. The forwarding, elsewhere called infection or epidemic, can be either performed periodically [13] or whenever the contact occurs [16]. Infection can continue up to the message life time or up to a given hop/copy count [8]. This PUSH-based algorithm can be combined with a PULL-based policy that enables the encounters to align one another state, or message list, by pulling missing messages after pushing summaries [16]. We will show that, in the considered scenario, the described basic mechanism tolerates intermittent connectivity, nodes dynamics and sparsity, and guarantees high node coverage with low latency. Unfortunately, this result is paid with an excessive overhead that seriously limits the practical use of the protocol. This is mainly motivated by the fact that nodes perform epidemic forwarding with a very limited knowledge about the state of the encountered nodes and, as a consequence, they often happen to forward the message to already infected nodes. Similarly, the delivery of a summary to an infected node is still a useless waste of resources. The primary focus in the design of a deployable, i.e. both efficient and effective, broadcast protocol is to increase the node likelihood of delivering the message only to uninfected nodes. This critical design point deserves the attention that has not attracted so far. There are several, growing levels of knowledge a node can achieve about the neighbors state to approximate the global system state. They range from zero knowledge, as for the stateless PUSH-based algorithm, to full knowledge, that can be approximated by maintaining some log of encounters and by enabling the log exchange among encountering nodes. Of course, the more information a node obtains, the higher the amount of resources (both local and system) it uses and the likelihood of packet dropping when the protocol is requested to scale together

This work has been partially funded by the Italian Ministry of University and Research in the framework of the "Context-Aware RouTing Over Opportunistic Networks (CARTOON)" PRIN Project.

with the growing of messages and nodes. Scalability problems lead to introduce congestion control mechanisms that push further the use of network and node resources. From the above arguments, there is an interesting space to design a lightweight broadcast protocol that addresses efficiency by keeping very low the amount of information it requires to improve the performances. We use the term *zero-knowledge* to indicate this approach. This paper moves into this research track and provides some original contributions to understand broadcast and multicast delivery over DTNs when resources are limited. Firstly, the paper shows that it is possible to design simple control mechanisms that keep the broadcast overhead surprisingly low. Unlike other approaches that attempt to limit the retransmission overhead by controlling the forwarding through state independent mechanisms, e.g. hop/copy count or probabilistic forwarding [7], this paper presents a context dependent and autonomic forwarding scheme that infers an approximation of the global state from locally observed events. Secondly, the paper throws the attention onto the fact that, despite the effective control mechanism, broadcast has a cost to reach the last 10% of nodes much higher than the cost of reaching the first 90% of them. This, together with termination and scalability arguments, leads to introduce an original hybrid PUSH-PULL policy that enables the switch from one to the other according to context-aware observations. Finally, it shows that, under certain simplified constraints, a weak reliable broadcast service, similar to the service proposed in [3], can be achieved without relevant extra costs on top of the best effort service. All the above points are discussed with the support of simulation results.

II. System Model and Assumptions

The scenario we consider in this paper includes people walking in a limited urban area, such as a *campus area*, and equipped with wireless portable devices. No base stations are assumed and the communication between a source s and a destination d may eventually occur through either direct contact, when, for instance, node d moves into the range of s, or indirect contact, when one or more relaying nodes help to create the multi-hop path towards the destination and the last of them finally enters the range of d. The devices have a unique identifier ID, are not required to have positioning capabilities on board and, to meet resource saving requirements, are supposed to adopt a short radio range to communicate. This latter point, together with the fact that devices can be sparsely distributed over a large area, makes high the probability of network partitions and link disruption. Throughout the paper we only assume that each mobile device, or node, periodically broadcasts a *beacon* message in its radio cell. Beacons are used to discover other devices in the neighborhood and their content is limited to the device identifier. In such a scenario, people mobility might follow either a random way point, RWP [2], model or a more structured motion throughout a set of aggregation points, such as the classrooms, the library or the faculty offices. However, whatever are the mobility traces people follow, our attempt is to design a broadcast mechanism that is as independent as possible of the underlying mobility model.

III. Considered Problem

Given a general DTN sparse scenario as described in the previous section, purpose of this paper is the design of a topology-independent broadcast protocol that eventually satisfies both the *effectiveness* and the *efficiency* requirements. Each broadcast message m is supposed to have a *"scope"* that is defined by the source and specified through either a *lifetime* or a hop/copy count; when this time or count expires, a node deletes the copy of m and stops its diffusion. In the following, we will use a scope defined in terms of lifetime and we assume long lived messages to better understand the broadcast behavior independently of other constraints.

This paper will show that the basic PUSH-based broadcast protocol, or P-BCAST, as sketched in Sec.I, has good capabilities to achieve high coverage, but the cost paid for effectiveness is the high overhead of duplicated messages that keeps the efficiency low. A message is duplicated when it is forwarded to an already infected node; the overhead of useless forwarding grows with the decrease of the knowledge a node has about the state of the encounters. The primary focus of the design of an epidemic is to control the forwarding policy according to the level of information a node has of the neighborhood. The more information it has, the lower the cost paid for duplicated messages. To bound this amount of extra traffic that we are inclined to tolerate to achieve high coverage, we can consider the following. In [4], the authors have shown that, under certain simplifying assumptions – i.e. single encounter node and well known system cardinality n and inter-contact times – an epidemic broadcast obtains a total broadcasts-per-message in the order of $O(n\ ln\ n)$. The aim of this paper is the design of a controlled PUSH-based broadcast that meets the problem's requirements that we can now more precisely define as follows:

- **broadcast effectiveness:** the capability of the protocol to eventually achieve node coverage arbitrarily close to 1;
- **broadcast efficiency:** the capability of the protocol to keep the generated broadcasts-per-message as close as possible to $O(n\ ln\ n)$.
- **broadcast scalability:** the capability of the protocol to scale when the number of sources and of per-source messages grow up.

There is a large set of DTN applications, including underwater, sensors or wearable devices, where node resources are highly constrained and protocols are forced to keep low the amount of neighbors information maintained by a node. In the following, we will focus on such a constrained scenario whose challenge is the control of the epidemic forwarding by exploiting as less state information as possible. Intuitively, any zero-knowledge forwarding mechanism can easily address the effectiveness and the scalability requirements, while the efficiency is the hardest requirement to achieve.

IV. Self-Adapting Broadcast Protocol

A. Protocol Overview

The arguments provided in the previous sections show that, under critical resource constraints, a P-BCAST protocol can be improved by adding some autonomic capability that extracts an approximation of global state from locally observed data. To this purpose, a node continues to observe situational data and attempts to answer the following questions: (1) when is it convenient to activate the algorithm run? (2) what is the locally sensed event helping to tune the node's diffusion probability?

In the basic P-BCAST approach, a node p starts a new diffusion of a message m with probability $Prob_p = 1$ whenever a node enters its radio range. When the node is moving according to a RWP model in a sparse area, this should not affect performances; however, when moving throughout aggregation points (ap), locations where nodes meet according some spatial or functional need, the broadcast of m whenever a new node enters the ap is likely to reduce efficiency by increasing duplicates. The first improvement of P-BCAST is to let p start a new diffusion of m when at least K neighborhood changes have been observed with respect to the previous diffusion. To this purpose, each node maintains a local view of the neighborhood changes by exploiting the underlying beaconing. To avoid broadcast storms, the nodes in range adopt a random mechanism that desynchronizes transmissions and suppresses duplicates. This way, p, and the nodes in range of p, control the duplicate generation and maximize the probability of infecting new nodes by adopting a membership-driven infection. For this reason, we use the term MP-BCAST to indicate this intermediate protocol that is supposed not to be influential when moving in a sparse area and to positively affect performances when moving across ap.

However, this cannot be the only policy to adopt. In fact, the duplicates overhead can be further reduced if, when moving throughout the area, a node p were able to tune the value of $Prob_p$ according to the delivery status of m in the area where p is moving. When most nodes in the neighborhood have delivered m, p should reduce the probability $Prob_p$ of a new diffusion accordingly, and vice versa. In the system model we adopted, the node p is unable to achieve this type of knowledge; however, p can derive the symptoms of the delivery status from sensing the events generated by its encounters. The number of received duplicates of m is the first of these symptoms; it reveals that other encounters have delivered m and is helpful to decrease, according to a given function $\mathcal{F}$, the local value of $Prob_p$. We will show that this simple mechanism works properly to limit the number of duplicates under high coverage conditions, although it is unable to promptly increase $Prob_p$ when p moves from a covered area to another where m has not been widely delivered. The question is: how can p get the encounters' delivery status of m by simply exploiting the controlled diffusion of m, i.e. without assuming the exchange of extra packets? To this purpose, the simple idea of a periodic refresh to the value $Prob_p = 1$ is clearly unsuitable. On the contrary, p would benefit of knowing the infection age of the encounters that delivered m; in fact, if the most part of them have been infected long time ago, then p should lower its $Prob_p$, while it should become more aggressive by setting $Prob_p = 1$ when most part of them have been recently infected. A low infection age is here adopted as a symptom of the fact that there are still several nodes to infect about, and vice versa. Each node, for each message m, locally determines whether its infection age is *recent* or not, according to a given threshold (sec.IV-B) and propagates this boolean information whenever it decides to send m. As a consequence, p decreases $Prob_p$ when it receives a duplicate *and* the duplicate carries the RECENT bit set to 0; it increases $Prob_p$ when the duplicate carries the RECENT bit set to 1. We do not need any other mechanism to trade off between *non-recent* (prudence) and *recent* (aggressiveness) attributes that are diffused within m; in fact, the uniformly distributed random mechanism, at the base of the duplicate suppression, is sufficient to provide a fair opportunity to both *recent* and *non-recent* nodes to correctly move the system towards prudence or aggressiveness. These mechanisms have been included in the self-adaptive broadcast, or SA-BCAST, protocol.

B. Protocol Description

MP-BCAST involves only a *reactive* mechanism, independent of the state of message diffusion, which allows nodes to possibly skip some contact opportunities. To record the encounters, p maintains a bitmap BM_p. The bitmap is updated every time p discovers a new neighbor through beaconing: node IDs are mapped into bitmap entries through a *hash* function; the bit corresponding to a new neighbor is changed. Since, by assumption, nodes do not know the system cardinality, the size of BM_p could be different from the number of nodes. A small BM_p lowers the risk of wasting memory with useless entries, but increases collisions of IDs in the same position. However, as the mechanism is based on the ratio of neighborhood change rather than on the neighbor identities, having a small bitmap does not hamper the algorithms. Whenever the message m is sent, the current bitmap is recorded. Upon every update, the amount of changes to the neighborhood from the last diffusion is evaluated, as the ratio of the Hamming's distance HD between BM_p and the $bitmap_p$ associated to m upon the last diffusion, and the current number of neighbors NV. If the ratio is greater than a threshold Nth, then a new diffusion could be scheduled. Nth tunes the aggressiveness of the algorithm.

Besides of the reactive mechanism, SA-BCAST involves an *adaptive* mechanism, which monitors the number of received duplicates and the RECENT bit to act on $Prob_p$ as discussed before. The pseudo-code for the SA-BCAST approach is shown by Algorithm 1. When a node q diffuses m, the message includes the residual lifetime $message\_lf$ and the RECENT bit. Each node starts executing the *Passive Thread* at the bootstrap. When p receives m for the first time (lines 8-12), it becomes a relay for m and starts the Active Thread. It maintains a copy of the message together with the copy of the current BM_p, and records both the local infection time

Algorithm 1 SA-BCAST pseudo-code

```
 1: Passive Thread:
 2: while True do
 3:    when message received do
 4:       if duplicate message then
 5:          Prob_p ← F(Prob_p, RECENT);
 6:          discard message;
 7:       else
 8:          buffer message;
 9:          age_p ← message_lf;
10:          t^p_infect ← current time;
11:          bitmap_p ← BM_p;
12:          start Active Thread;
13:       end if
14:    end do
15:    when infected and (current time - t^p_infect) ≥ age_p do
16:       discard message;
17:       stop Active Thread;
18:       HALT;
19:    end do
20: end while
21:
22: Active Thread:
23: Prob_p ← MAXP;
24: while True do
25:    when new neighbor n do
26:       if relevant neighborhood changes then
27:          if flip(Prob_p) then
28:             if no duplicates in small random wait then
29:                RECENT ← ((current time −t^p_infect) ≤
                   RECENT_threshold);
30:                message_lf ← age_p − (current time −t^p_infect);
31:                broadcast message to neighbors;
32:             end if
33:             bitmap_p ← BM_p;
34:          end if
35:       end if
36:    end do
37: end while
```

(t^p_{infect}) and the residual message lifetime (age_p). When p is infected and detects that the message lifetime has expired (lines 15-18), the algorithm run terminates. When p is infected, the adaptive mechanism is run. When duplicates are received (lines 4-6), p updates statistics used to adapt the behavior of the Active Thread: the RECENT bit of the sender of the duplicate is used to adapt the probability $Prob_p$ of starting new transmissions, by means of the function $\mathcal{F}$, so that it varies in the range [MINP, MAXP]. Initially, the probability of performing a transmission is set to the maximum value (line 23). When the neighborhood changes are relevant, with probability $Prob_p$ p starts a new infection, while with probability $(1 - Prob_p)$ p skips. In the former case, p waits for a random time (much lower than the beacon period); if no duplicate is received in the meantime, then p starts a new diffusion. Whenever p has to send m, it computes the residual lifetime. The RECENT bit is set if (current time $-t^p_{infect}$) $\leq$ RECENT_$threshold$. The management of both infection age and message lifetime does not require synchronization of the clocks. Whether the diffusion is performed or is suppressed, p updates the $bitmap_p$ associated to the message with the current BM_p. This operation is performed also in case of duplicate suppression, because if the current neighbors of the node are already infected, the node must wait for neighborhood changes before scheduling a new transmission.

C. Protocol scalability and termination

The described protocol uses very few bits, i.e. the bitmap and the RECENT bit, to maintain per-message local information. Their size does not affect scalability that, by contrast, can be affected by the need of maintaining a copy of the messages up to their lifetime. In a multi-message scenario, in fact, the longer is the message lifetime the higher is the likelihood of message dropping at a node with limited memory resources. Although the message dropping has the only effect of reducing the amount of relaying nodes and thus increasing the latency to achieve coverage, the problem may have some effect on efficiency because the protocol is unable to totally stop the infection when all nodes have been infected. In fact, in a zero knowledge scenario it is not possible to enable the protocol termination earlier than the message lifetime and the following claim can be easily proved:

Claim 1: If no information is available about the node mobility, cardinality and state, then the diffusion procedure cannot terminate before the message lifetime expires.

The proof of this claim is not included in the paper, but it can be intuitively derived by observing that the algorithm has to ensure a drip feed of message transmissions to manage node joins and temporary partitions. Simulations will show that the claim has severe impact on the efficiency if not mitigated by switching to a PULL-based mechanism. The above and the following (Sec.V) arguments motivate and highlight the opportunity of performing the broadcast forwarding as the combination of two phases: a fast-diffusion phase, where a message $m1$ is quickly pushed at any contact opportunity, and an on-demand phase, where the pull of $m1$ is enabled by piggybacking the $m1$'s summary on a message $m2$ that follows $m1$. Although the detailed discussion will be included in the extended version of the paper, the next Section provides some element to define how long the fast-diffusion phase lasts and when a node can autonomously decide to switch from one phase to the other.

V. Performance Evaluation

A. Simulated Conditions and Performance Indexes

We implemented P-BCAST, MP-BCAST and SA-BCAST algorithms in the framework of the GloMoSim [15] simulation environment. The simulation setting considers a system of 50 nodes sparsely distributed over a 1000 $\times$ 1000 mt. area. Nodes move according to a RWP model at a speed in $[1, 2]$ m/s, thus reproducing a pedestrian environment. They are equipped with a low power 802.11 radio device with 30 mt. communication range and DCF at the MAC layer. Beaconing is performed every 1 sec.; after 3 missing beacons, the entry is removed from the neighbor list. The BM uses 32 bits.

The simulations run different values of Nth and two different functions $\mathcal{F}$: a linearly decreasing function (or Lin10)

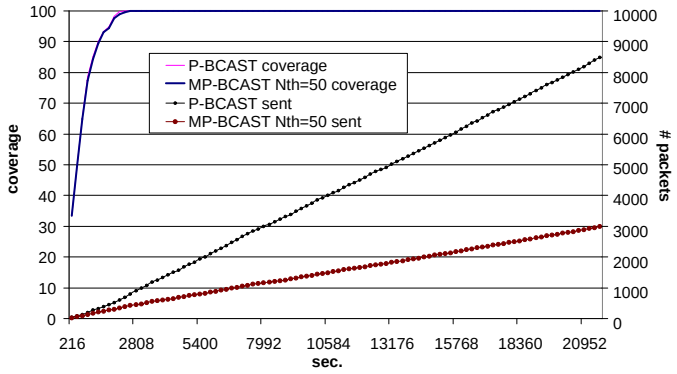


Fig. 1. Coverage and cumulative number of generated packets vs. time

and an inverse exponential function (or InvExp). If the node's infection time is < 3 min. then all the packets it sends have the RECENT bit set, and all receiving nodes will set $Prob_p$ to MAXP$= 1$. Otherwise, packets have RECENT bit set to 0 and $Prob_p$ will be either decremented of 0.1 or halved in line with functions Lin10 or InvExp, respectively. $Prob_p$ has a lower bound defined by MINP. We consider long lived broadcasts, with simulations lasting up to 6 hours, and nodes moving according to RWP or through aggregation points. All simulation results are averaged over 50 simulations performed with variable random seed.

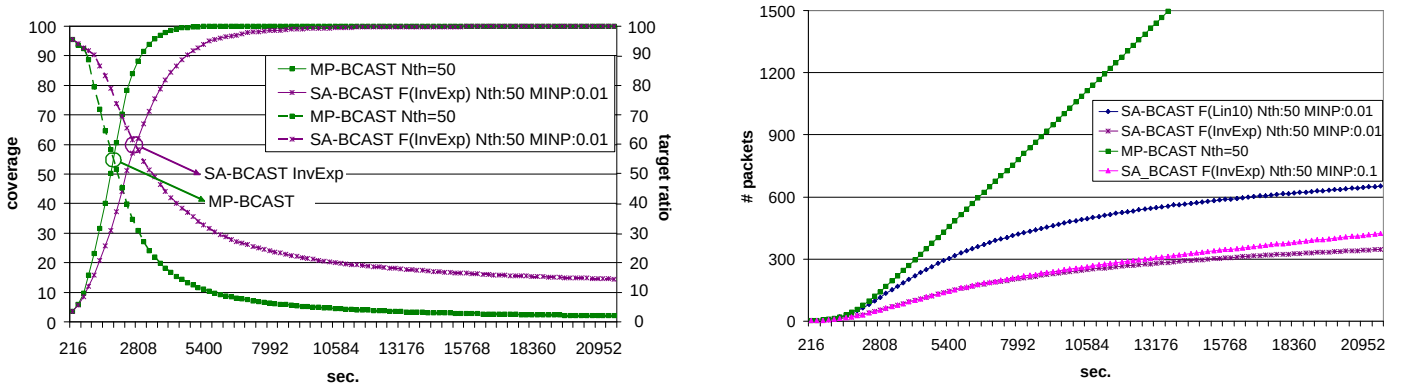
B. Simulation Results

In the following, we use the *node coverage*, as the index to evaluate the *effectiveness* and the *target ratio*, T, to estimate the *efficiency* in terms of the success ratio of the infection activity: $T = (msgrec - dups)/msgrec$, with $msgrec$ the total number of received messages and $dups$ the total number of duplicates among them. Of course, T is optimized by $dups = 0$ and is affected by the number of the encounter nodes and by the progress of the infection in the neighborhood. In fact, packets are broadcast to the nodes in range, let us say k; so that, for any message sent, we count $msgrec = k$ and the $dups$ value depends on the level of infection among the k nodes.

We have observed in sec.IV that the membership-driven approach of MP-BCAST and the encounter-driven approach of P-BCAST have similar performances when nodes move within a single-encounter scenario. In this case, in fact, the Nth threshold adopted in MP-BCAST is exceeded at every encounter because of the low (1-2) number of neighbors. By contrast, performances can be improved when moving within an area where nodes meet in aggregation points. Simulations have confirmed this hypothesis, as it can be observed in fig.1 that reports the coverage and the cumulative amount of packets in the following aggregation scenario: the source node is the first node in an aggregation point and the other nodes enter each aggregation point with an inter-arrival time of 5 minutes. It is easy to observe that MP-BCAST and P-BCAST have similar coverage and latency behaviors, but the former increases the efficiency because it is able to optimize the infection rate by waiting for a relevant number of new neighbors (50%) before forwarding. This shows that MP-BCAST is able to capture the membership changes in the neighborhood and to adapt accordingly by tuning the forwarding rate. However, being unaware of the infection status, MP-BCAST is unable of smoothing the packet generation rate with the growing of the infection (fig.1). This is the capability we expect from SA-BCAST.

In fig.2(a), we show the trade-off between coverage and T for SA-BCAST and MP-BCAST in the RWP model; the higher the value of the y-axis corresponding to the point where the two curves cross, the better. Of course, by properly choosing $\mathcal{F}$ and Nth it is possible to tune the aggressiveness of the algorithm and define the proper trade-off between coverage, latency and efficiency. For instance, with $\mathcal{F} =$Lin10, a behavior intermediate between MP-BCAST and $\mathcal{F} =$InvExp is obtained. The capability of the adaptive mechanism of (almost) stopping transmissions when nodes are covered is shown in fig.2(b): unlike MP-BCAST, the plot of the cumulative number of messages tends to become flat with SA-BCAST. The parameter MINP can be used to tune the minimum rate of diffusion when all nodes are infected and their $Prob_p$ converged to MINP, that is, the promptness with which a node newly entered in the system becomes infected. We observed that the value of MINP does not affect the coverage latency: rather, MINP influences the amount of traffic generated, because of Claim 1, during the infection tail (Fig.2(b)) and the latency time to infect the last node. This protocol's tail can analytically be described as follows. Let F be the contact inter-arrival rate in the system. If a node p has F_p contacts with other nodes in a time unit, then $F = \frac{1}{2}\sum_p F_p$. If all processes have the same F_p – which seems a reasonable assumption in the RWP model – then $F = \frac{N}{2}F_p$, with N the number of nodes in the system. We say that the system is in *quiescent* state when all nodes have been infected and $\forall p\ Prob_p =$MINP.[1] Under the assumption of single encounters – satisfied in the sparse environment reproduced in simulations – a reasonable value for Nth is $Nth \leq 100$ that, as NV is usually 1, makes a single encounter sufficient to schedule a new diffusion. In these conditions, when two nodes p and q enter in contact, the probability that no message is generated is $(1-\text{MINP})^2$, and the probability of at least one message generated is $1 - (1-\text{MINP})^2 = 2\text{MINP}-\text{MINP}^2$. Hence, the traffic globally generated per time unit is $F(2\text{MINP}-\text{MINP}^2)$. Let p be a node joining the system once all nodes have been infected; p will be infected as soon as it encounters a node q that starts a new diffusion. Node p has a contact every $1/F_p$ time units. The expected number of nodes it encounters before being infected is $1/\text{MINP}$. Hence, the expected time before p is infected is $(1/F_p) \times (1/\text{MINP})$. This is also the expected time needed to infect the last node in the system. Simulation results are compliant with these estimates: from

[1]It can be proved that, when all nodes have been infected, their $Prob_p$ converges to MINP and is no longer increased.



(a) *(b)*

Fig. 2. (a) Coverage and target ratio vs. time. (b) Cumulative number of generated packets vs. time

the observed data, a node has $F_p = 3.426 \times 10^{-3}$ contact opportunities per sec., yielding $F = 3.426 \times 10^{-3} \times 50/2 = 0.0856$. When MINP$= 0.1$, the traffic generated per sec. is $0.0856 \times (2 \times 0.1 - (0.1)^2) = 0.01627$, that is, around 58 packets per hour. Results yield 48 packets generated in the last hour of simulation. The time spent to infect the last node is $(1/(3.426 \times 10^{-3})) \times (1/0.1) \simeq 2919$ sec. In simulation, 49 nodes have been infected in roughly 7100 sec. The last node has been infected on average after 9500 sec., that is 2400 sec. after the last but one. The time spent is slightly lower than the estimated one, because simulations do not guarantee that all nodes have $Prob_p =$MINP.

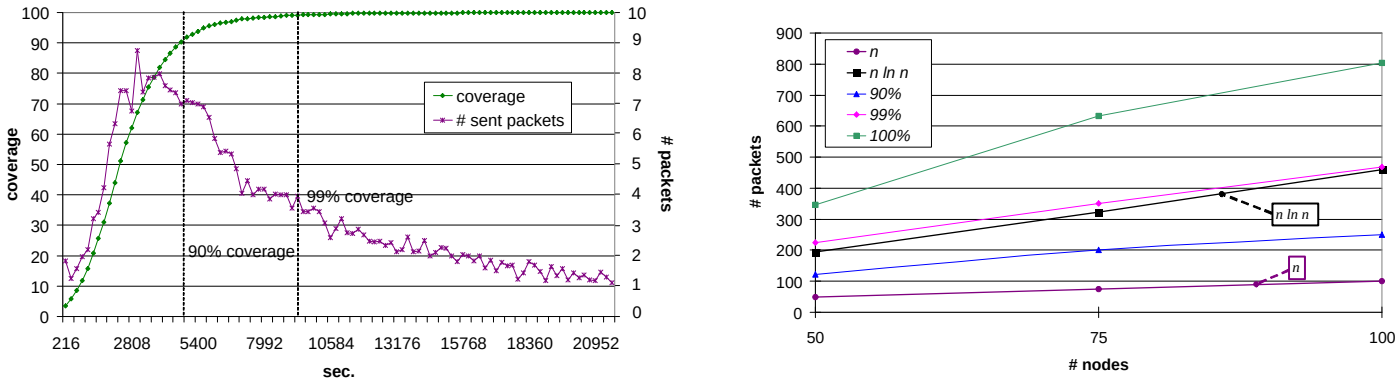
According to Claim 1, it is evident, fig.3(a), that nodes generate a minimum of packets also when coverage approximates 1. It is interesting to observe that nearly the same time interval is required to deliver the message to the first 90% of nodes and to infect a further 9% of them. A long time is still required to discover the last uninfected node about. In the meanwhile, T drastically decreases (fig.2(a)), thus showing that most of the packets are duplicates. This leads to consider the following: the broadcast may benefit of a fast-diffusion phase in which SA-BCAST forwards the message $m1$ within a slightly infected environment; this outperforms the approach of diffusing summaries because it eliminates the handshake phase among encounters and the requests implosion problem. However, to control the growing of duplicates, the algorithm should let a node to timely switch from the PUSH-based approach to a PULL-based one by exploiting context information. The switch will enable the pull of $m1$, whose summary is piggybacked on a message $m2$, that follows $m1$, and eliminate the described inefficiencies. To answer the question "when may a node autonomously decide the switch?" we observed that, whenever the 90% of nodes is infected (also with Lin10 and other values of Nth), the packet generation rate starts decreasing and the probability $Prob_p$ decreases accordingly. From our preliminary analysis it emerges the chance of using the continuous decrease of $Prob_p$ as the *local* symptom of high coverage. This would lead to the design of a stateless, autonomic switching mechanism that does not generate memory and communication overheads.

In Fig.3(b), the number of packets needed to reach different values of coverage, for different group cardinalities, is reported for InvExp function, $Nth = 50$ and MINP$= 0.01$, and compared with the order of magnitude derived in [4] for an ideal system setting. Although SA-BCAST is assumed to run in a zero-knowledge scenario, the adopted RWP model is totally comparable to the one used in [4]; in fact, the inter-contact times have been evaluated to satisfy an exponential distribution of type $f(x) = \lambda(e + \delta)^{-\lambda x}$ and the encounter rate is very low (each node has on average less than 0.3 neighbors at a time with a system of 100 nodes). As a consequence, the ideal setting provides reference results for the practical setting. From the analysis of Fig.3(b), it can be firstly observed that the adaptive mechanism keeps the efficiency surprisingly close to the ideal bound of $O(n\ ln\ n)$, but it is also confirming the protocol's problems to achieve a coverage close to 1, or exactly 1.

VI. Conclusions and Future Work

The paper has shown that it is possible to design a broadcast protocol that satisfies the problem requirements by adopting a situational and autonomic forwarding mechanism. We considered a scenario where nodes were supposed to have no state information and resource constraints. The zero-knowledge protocol we have proposed greatly improves the performances of existing approaches. It is worth to notice that a further performances improvement can be granted by moving from zero-knowledge protocols towards protocols capable to exploit the knowledge they achieve from maintaining and exchanging a history of encounters. This is one of the topics of the ongoing research.

The arguments considered throughout this paper have another hidden implication that deserves to be investigated. In fact, at least under the mobility conditions we have considered so far and the assumption of no perpetual partitions, it is easy to see that, in practice, Claim 1 leads to eventually



(a) *(b)*

Fig. 3. (a) Coverage and number of generated packets vs. time for SA-BCAST, with $\mathcal{F}$ =InvExp, $Nth = 50$ and MINP= 0.01. (b) Cardinality of sent packets for different number of nodes in the system

achieve a weak *reliability*. In fact, if the algorithm keeps on forwarding, either through a PUSH or a PULL approach, then the *eventually reliable broadcast* service over DTNs can be provided without relevant extra cost.

REFERENCES

[1] J. Burgess, B. Gallagher, D. Jensen, and B.N. Levine, *MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks*. Proc. IEEE INFOCOM 2006.

[2] Camp T., Boleng J., Davies V., *A Survey of Mobility Models for Ad Hic Network Research*. Wireless Communication and Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications, 2(5), pp. 483–502, 2002.

[3] R. Chandra, V. Ramasubramanian, and K. Birman, *Anonymous gossip: Improving multicast reliability in mobile ad-hoc networks*. Proc. 21st International Conference on Distributed Computing Systems (ICDCS), pp. 275–283, 2001.

[4] D.E. Cooper, P. Ezhilchelvan, and I. Mitrani, *High Coverage Broadcasting for Mobile Ad-hoc Networks*. Proc. NETWORKING 2004, Lecture Notes in Computer Science, vol. 3042, pp. 100–111, 2004.

[5] J.A. Davis, A.H. Fagg, and B.N. Levine, *Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks*. Proc. 5th IEEE Intl. Symp. on Wearable Computers, 2001.

[6] *Delay Tolerant Networking Research Group*. http://www.dtnrg.org/wiki

[7] DrabkinV., Friedman R., Kliot G., Segal M., *RAPID: Reliable Probabilistic Dissemination in Wireless Ad-Hoc Networks*. Proc. 26th IEEE Intl Symposium on Reliable Distributed Systems (SRDS), Oct. 2007, pp.13–22.

[8] K.A. Harras, K.C. Almeroth, E.M. Belding-Royer, *Delay Tolerant Mobile Networks (DTMNs): Controlled Flooding in Sparse Mobile Networks*. IFIP Networking 2005.

[9] E.P.C. Jones, L. Li, and P.A.S. Ward, *Practical routing in delay-tolerant networks*. Proc. ACM SIGCOMM Workshop on Delay-tolerant networking (WDTN), 2005, pp. 237–243.

[10] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein, *Energy-effi cient computing for wildlife tracking: Design tradeoffs and early experiences with Zebranet*. Proc. ASPLOS, October 2002.

[11] U. Lee, E. Magistretti, B. Zhou, M. Gerla, P. Bellavista, and A. Corradi, *MobEyes: Smart Mobs for Urban Monitoring with a Vehicular Sensor Network*. IEEE Wireless Communications, 13(5), Sep. 2006.

[12] V. Lenders, G. Karlsson, and M. May, *Wireless Ad Hoc Podcasting*. Proc. 4th IEEE Conf. SECON, June 2007, pp. 273–283.

[13] A. Montresor, M. Jelasity, and O. Babaoglu, *Gossip-based Aggregation in Large Dynamic Networks*. ACM Transactions on Computer Systems, 23(3), Aug. 2005, pp.219–252.

[14] T. Spyropoulos, K. Psounis, and C.S. Raghavendra, *Single-copy routing in intermittently connected mobile networks*. Proc. 1st IEEE SECON, pp. 235–244, Oct. 2004.

[15] UCLA Parallel Computing Laboratory, *GloMoSim – Global Mobile Information Systems Simulation Library*. University of California at Los Angeles. http://pcl.cs.ucla.edu/projects/glomosim/

[16] A. Vahdat, and D. Becker, *Epidemic Routing for Partially Connected Ad Hoc Networks*. Technical Report CS-200006, Duke University, April 2000.