

Embedding Evolution in Epidemic-Style Forwarding

Sara Alouf^a, Iacopo Carreras^b, Daniele Miorandi^b, Giovanni Neglia^{a,c}

^a INRIA

^b CREATE-NET

^c University of Palermo

2004 Route des Lucioles, B.P. 93
06902 Sophia Antipolis, France

Via Solteri 38
38100 Trento, Italy

Viale delle Scienze, ed. 9
90128 Palermo, Italy

Sara.Alouf@sophia.inria.fr

Name.Surname@create-net.org

Giovanni.Neglia@ieee.org

Abstract

In this work, we introduce a framework to let forwarding schemes evolve in order to adapt to changing and a priori unknown environments. The framework is inspired by genetic algorithms: at each node a genotype describes the forwarding scheme used, a selection process fosters the diffusion of the fittest genotypes in the system and new genotypes are created by combining existing ones or applying random changes. A case study implementation is presented and its performance evaluated via numerical simulations.

1. Introduction

Epidemic-style forwarding [11] has been proposed as an approach for achieving system-wide dissemination of messages in Delay-Tolerant Networks (DTNs) [3] in face of frequent disconnections [9]. DTNs are sparse and/or highly mobile wireless ad hoc networks where no continuous connectivity guarantees can be assumed. Epidemic-style forwarding in DTNs is based on a “store-carry-forward” paradigm: a node receiving a message buffers and carries that message as it moves, passing it on to new nodes upon encounter. Alike the spread of infectious diseases, each time a message-carrying node encounters a new node not having a copy thereof, the carrier may decide to *infect* this new node by passing on a message copy; newly infected nodes, in turn, behave similarly. The destination receives the message when it first meets an infected node.

An unconstrained epidemic forwarding scheme (in which an infected node spreads the epidemic to all nodes it encounters) is able to achieve minimum delivery delay at the expense of an increased use of resources such as buffer space, bandwidth, and transmission power. Variations of epidemic forwarding have been recently proposed in order to exploit the trade-off between delivery delay and resource consumption. This family includes, among the others, K -hop schemes [4], K -copy techniques [2], probabilistic for-

warding [6], and spray-and-wait [10]. These schemes differ in their “infection process”, i.e., the spreading of a message in network. They need to be combined with a “recovery process” that deletes copies of a message at infected nodes, following the successful delivery of the message to the destination. Various recovery schemes have been proposed: some are simply based on timers, others actively spread in the network the information that a copy has been delivered to the destination, using so-called antipackets [5].

Depending on the specific application scenario, different performance metrics could be envisaged, such as the probability to successfully deliver a message to the destination, the delivery time, the total energy consumption in the system or a combination of the previous ones. For a given optimization goal, the choice of a specific forwarding scheme and its parameters configuration depend in general on the number of nodes in the system, on their mobility patterns and on the traffic generated in the networks [7]. In many scenarios, these characteristics cannot be known at system design and deployment time and may drastically change across time and space. Consider for instance a personal digital assistant (PDA) carried by a user in its daily activities. During the day, the PDA may travel at different speeds (e.g. from zero up to car speed), moving from highly crowded areas (supermarkets, classrooms,...) to sparse ones, with very different trajectories (straight along a highway or following a random walk from shop to shop) and different levels of power availability.

In order to deal with these issues, various adaptive techniques for message forwarding can be envisaged. This approach is limited in that it requires an *a priori* definition of the actions to be taken to optimize the mechanism for some specific situation. The approach we propose is different. We want to embed the ability to evolve *autonomously* in the forwarding service itself. This is achieved by using concepts and tools from the Genetic Algorithms (GAs) field. Each node employs a (potentially different) forwarding policy, which prescribes the operations to be undertaken when receiving a message destined to another node. Such

a policy is described by an array of parameters called the genotype. Genotypes are associated with a fitness measure which, roughly speaking, indicates the ability of the current set of parameters to achieve good performance in the local environment. Fitness is evaluated using local information and feedback which is sent from the destination backwards within ACK messages, which act also as antipackets. When two nodes meet, they may exchange genotypes (and associated fitness levels), updating the pool they maintain. Each node periodically generates a new genotype judiciously using those in its pool. The whole system is engineered in such a way to present a drift towards higher fitness levels.

The rest of the paper is organized as follows. Section 2 overviews the evolutionary delay-tolerant forwarding service engineered in this paper. Its multiple components are detailed in the following sections: Section 3 presents the forwarding policy followed by each node and its unified representation; Section 4 presents the selection process of good forwarding policies and the generation of new policies; and Section 5 presents the fitness estimation process. Then Section 6 describes the specifications of the protocol. The outcomes of a simulative study, performed using a freely available software tool, are reported in Section 7. Section 8 concludes the paper. An extended version is available as INRIA technical report [1].

2. An evolutionary forwarding service

In this paper, we aim at introducing evolution into DTNs forwarding service. Our approach is based on the fundamental observation that forwarding schemes simply perform a decision whether to relay a copy of a given message to an encounter node or not. Thus, multiple forwarding schemes can co-exist and interact within the same network. This flexibility comes from the completely distributed nature of the forwarding process in epidemic-style relaying, which allows node to use different policies in an uncoordinated fashion.

We represent the forwarding policy used at a given node as a *genotype*. The genotype must capture the key features of the used policy. By changing its genotype, a node can make its policy evolve. A *selection process* will drive the evolution towards a predefined objective. This selection process should favor the selection of genotypes achieving good performance with respect to the predefined objective.

For instance, assume that the objective is to minimize the message delivery time. Obviously, the best will be to flood the network with message copies, as one cannot improve over using, for each message, all meetings between nodes. Thus, unconstrained epidemic routing is the *fittest* policy as regards delivery time minimization; genotypes representing this policy should be favored in the selection process.

From this example, it becomes clear that the different

forwarding policies have to be evaluated with respect to the predefined objective. We shall devise a fitness function that returns a high value should the policy be considered to perform well and a low value otherwise. We note that a node cannot evaluate by itself whether its current policy fits the current scenario, because it is in general not aware of the consequences of its actions. For example a given node can never know by itself whether its decisions – according to its forwarding policy – to relay or not to relay a message were the right ones or not. Thus, a node may be relaying a message when the latter has already been delivered to its destination, hence wasting resources. On the other hand, a node may refrain from relaying a message when it happens to be the key node in the message delivery process, e.g., if it is the only node traveling between two disconnected clusters of nodes in the network. We also observe that the fitness of a node’s policy depends on the policies implemented by the other nodes as well. Message delivery is in fact a collaborative process involving many nodes each applying possibly a different policy. A specific policy can be beneficial or detrimental depending on other nodes actions, hence it is difficult to isolate the individual performance of a single policy. The previous considerations imply the need of an online distributed fitness evaluation process. Last, observe that evaluating the fitness of genotypes is a noisy process because of randomness of the mobility process. Therefore, the fitness should be *estimated*. This estimation is affected by a delay between the moment nodes perform actions and the moment the consequences of these actions are known.

In this paper, we report on a case study implementation of the proposed approach to epidemic-style forwarding in delay-tolerant networks. Our main objective with this implementation is to gain insight into the applicability of the proposed approach. In particular, our purpose is to provide a first answer to the following questions:

- (i) Does the distributed genetic algorithm “converge”?
- (ii) If so, what does the convergence point look like and how much time is required for the convergence?
- (iii) What are the performance with respect to an optimally configured static forwarding scheme?

These questions will be tackled by implementing a (reduced) version of the proposed framework, and running numerical simulations to evaluate, in a realistic scenario, the behavior of the system.

3. The forwarding policy

A forwarding policy consists of a set of actions to undertake upon message reception. It defines what nodes do when they get within mutual transmission range. The actions can be specified using parameters and may rely on information contained in message headers (like message generation time or message hop count). For instance, a node can transmit a

Table 1. Mapping example

Genotype	1 0 0 1 0 : 0 1 1 0 1
Interpretation	$P = (18/31)^{1.5} \approx 0.44, H = 13$
Policy	<pre> if (msgHops < H) then if (uniform(0,1) < P) then ForwardMessage(); endif endif </pre>

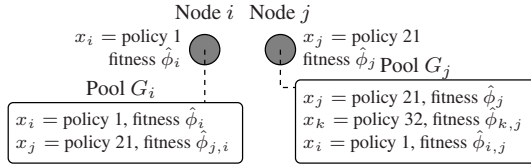


Figure 1. System architecture.

message with probability P as long as the message has not been forwarded more than H times. The values of these parameters uniquely specify one policy.

A simple binary string can be used to represent the policy, as illustrated in Table 1 where the genotype spans 10 bits. The 5 rightmost bits represent the maximum number of hops H , whereas the 5 leftmost bits provide the value of a such that $P := (a/31)^{1.5}$ returns the forwarding probability. Each representation yields a unique forwarding policy; see e.g. Table 1. Many other parameters can be added as *genes* in the genotype, like the maximum number of copies that a node can do for a given message or a timer restricting message lifetime. The behavior of a generic forwarding scheme can be changed by tuning the values of the parameters that specify the actions to be undertaken.

So as to explore the applicability of the proposed approach, we consider a fixed-length genotype comprising only one parameter, which is the probability P of relaying a copy of the message upon encountering a new node.

4. Generation of new forwarding policies

We adopt the genetic algorithms approach to generate new forwarding policies. At this point we assume that each node has an estimation of the fitness of the genotype it is using, leaving the estimation details to the following section.

In our implementation, as nodes meet, they transmit their own genotype and its current fitness estimation. Each node i maintains a pool, denoted G_i , of available genotypes (including the one currently in use) and their fitness estimations, as illustrated in Fig. 1. Let $\hat{\phi}_i$ denote the fitness estimation of node i genotype and $\hat{\phi}_{i,j}$ denote the fitness value of node i genotype as known by node j . $\hat{\phi}_{i,j}$ is the value of

$\hat{\phi}_i$ at the last meeting time between nodes i and j , so these two values may at times be different.

At generation times, nodes apply GA-like operators to the genotypes maintained in their pools. First, genotypes are selected for reproduction based on their weighted fitness. In other words, node j selects genotype x_i with probability $p_{i,j} := \hat{\phi}_{i,j} / (\sum_{k \in G_j} \hat{\phi}_{k,j})$. Then the GA operators are used to create new genotypes from existing ones. These operators are crossing-over and mutation. More details are provided in [1].

Crossing-over is performed with probability p_c and requires to select two genotypes from the pool. It consists in breaking two genotypes at a uniformly-randomly chosen bit position and exchanging the tails of the genotypes; from the two offsprings produced, called *crossovers*, one is equiprobably selected. *Mutation* consists in a random change occurring in the genotype. As an example, mutation can be implemented by swapping, with probability p_m , each bit of a binary representation of the genotype.

After having generated the genotype that will be used, the genotype fitness is set to zero and the pool of genotypes is emptied. For a start, we consider a *synchronized* reproduction phase, leaving the non-synchronous case for future work. Every T_g seconds, the *generation lifetime*, nodes synchronously create a new offspring each, i.e., each updates its own genotype. This synchronism allows to clearly identify different generations during the evolution.

5. Fitness estimation

We have seen in the previous section that the generation of new genotypes relies heavily on the fitness figures of the actual ones. It is then crucial for the proper evolution of the forwarding service to devise a fitness estimator that will closely reflect the performance of the genotype with respect to the targetted optimization.

In this paper, we consider that the function to optimize, F , is the expected value of some performance metric, say f , which can be evaluated for a specific infection process I . By infection process, we mean the complete history of events related to the delivery and cancellation of a generic message in the network. Examples of $f(I)$ are the time needed to deliver a message to the intended destination, the time before an infection dies (i.e., when all copies are erased from the network), the number of copies done for a given message and the power required to propagate the message.

The optimization goal, defined as $F = \mathbb{E}[f(I)]$, depends on the entire set of forwarding policies used in the network. A candidate solution is then an N -tuple (x_1, x_2, \dots, x_N) , where N denotes the number of nodes in the network and x_i denotes the genotype used at node i , for $i = 1, 2, \dots, N$. In standard GAs, the fitness ϕ of a candidate solution is simply the function to optimize evaluated at this point, namely $\phi :=$

$F(x_1, x_2, \dots, x_N)$. However, this is not useful in our case as we want to evaluate the fitness of a single policy not that of an N -tuple of policies.

In our implementation, we let $f = T_D + \gamma \sum_{i \in W} C_i$ where T_D is the delivery time of a generic message, C_i is the number of message copies made by node i , W is the set of nodes contributing to the delivery of the first message copy to reach the destination, and γ is a parameter which can be understood as the time-equivalent cost of a copy. In other words, γ represents the minimum decrease of the delivery time we would like to observe should an additional copy be made by a node in W . The optimization goal is to minimize the *cost*

$$F = \mathbb{E} \left[T_D + \gamma \sum_{i \in W} C_i \right] . \quad (1)$$

Introduce h as the number of hops done by the first copy of a message that reaches destination. We have $|W| = h$. The quantity $T_D/h + \gamma C_i$ can be seen as an approximation of the contribution of node i ($i \in W$) in the global performance metric f . Should h be a constant, minimizing $T_D/h + \gamma C_i$ shall also minimize f and thereby the cost F . The fitness of node i genotype is hence defined as

$$\phi_i = \mathbb{E} \left[1 - \frac{T_D/h + \gamma C_i}{R} \right] , \quad (2)$$

where R is set to a high enough value, for instance $2\mathbb{E}[T_D] + \gamma N$, so that $\phi_i > 0$. By collecting samples of $T_D/h + \gamma C_i$, node i can simply estimate ϕ_i using the sample mean. In the following we are going to describe the whole fitness estimation process.

Let the message header contain fields specifying the hop count h and the time at which the payload was generated at the source. Assuming that all nodes are synchronized (this assumption can be removed [1]) the destination can evaluate the delivery time as soon as it receives the first copy of a message. Let each node, before forwarding a copy of a message, add its own identifier to the message header. The set W is nothing but the set of node IDs present in the header of the first copy reaching the destination. The destination node sends to nodes in W a new acknowledgment (ACK) message, which provides feedback specifying the delivery delay T_D and the number of hops $h = |W|$. At this point, the node evaluates the ‘‘reward’’ obtained for having taken part in the infection, as a decreasing function of the quantity $T_D/h + \gamma C_i$. More precisely, we define the reward at node i as

$$r_i = \max \left\{ 1 - \frac{T_D/h + \gamma C_i}{R}, 0 \right\} . \quad (3)$$

Upon receiving the n -th ACK message and computing the reward $r_i(n)$ according to Eq. (3), node i updates its estimation of the genotype fitness as follows (this corresponds

to averaging all rewards received):

$$\hat{\phi}_i(n) = \frac{n-1}{n} \hat{\phi}_i(n-1) + \frac{1}{n} r_i(n) . \quad (4)$$

6. Protocol specifications

Two nodes are able to exchange messages (should they decide to do so according to their policies) when they get within mutual communication range. The evolving protocol makes use of two types of messages to be exchanged over the network: DATA messages and ACK messages. DATA messages are those carrying the payload transmitted by any mobile node to a specific destination, whereas ACK messages are used:

- to acknowledge the successful delivery of the message at its intended destination;
- to feed back the reward to the nodes along the successful path from source to destination (rewarding);
- to serve as anti-DATA, removing already delivered messages from nodes buffer hence blocking their diffusion.

Each mobile node maintains two internal data structures dedicated to the storage of DATA and ACK messages respectively. In the structure storing DATA messages, each item additionally stores a counter of the number of copies of that message already disseminated in the network.

Whenever a node receives a DATA message to be relayed, it first adds its own node ID to the header, then increments by one the hop count in the message header. The DATA message is stored until its ACK message is received. In addition to DATA messages, nodes diffuse also ACK messages following the IMMUNE_TX strategy in [5].

Whenever a node receives an ACK message, it checks whether the corresponding DATA message is stored in its internal memory, in which case it is erased. Should the corresponding DATA message be present and should the node be in the set W present in the ACK message header, the node will then apply the proper rewarding scheme to update its genotype fitness, as described in the previous section.

7. Numerical results

In order to evaluate the performance of the presented algorithms, we run extensive simulations using the freely available simulation tool OMNeT++ [8].

We consider N mobile nodes, moving at constant speed v over a $L \times L$ square playground according to the random direction mobility model. Each node selects the angular direction of its next movement uniformly in $[0, \pi]$, moves along this direction with a uniform speed; upon reaching the border, it generates a new angular direction and moves accordingly. Nodes initial locations are sampled from a uniform distribution which is the nodes stationary distribution under this mobility model (perfect simulation).

Table 2. Simulation parameters

$\gamma/100 = 1, 4, 8, 16s$	$T_g = 12000s$	$L = 500m$
$p_c = 0.10$	$T_s = 3000s$	$r = 25m$
$p_m = 0.01$	$T_{step} = 2s$	$v = 1m/s$
Static scenario	$N = 20$	
Dynamic scenario	N varies in $\{5, 10, 20, 30, 40\}$	

Distinct nodes are considered to be in communication range if the mutual distance falls below the communication range r . Each mobile node generates a new DATA message every τ seconds where τ is uniformly distributed between 0 and T_s seconds. The destination of the new message is chosen uniformly among the nodes in the simulation. Each message generated is stored in the out queue of the generating node. The position of every mobile node in the simulation is updated every T_{step} seconds. Each generation lasts for T_g units of time.

The specific values used are in Table 2. GA parameters p_c , crossover probability, and p_m , mutation probability, are taken to be fixed. The investigation of performance sensitivity to these parameters is left to future work.

As regards the genotype, the forwarding probability P has been quantized non-uniformly using 5 bits for the representation. P takes value in $\left\{ (i/31)^{1.5}, i = 0, 1, \dots, 31 \right\}$.

Static scenario We first consider a case where the number of nodes is kept constant ($N = 20$) throughout the simulation run. The time-equivalent cost of one message copy is set to $\gamma = 800$. The N initial forwarding probabilities are chosen independently in the set of possible values according to a uniform distribution.

Figure 2 depicts the forwarding probabilities present in each generation. As it might be seen, after few generations only small values are used across the population, but for some occasional high values due to random mutations. Using different initial random seeds yields similar results.

Figure 3 shows the corresponding evolution over time of the cost (1), expressed in seconds. Observe how the cost rapidly decreases across the first generations and how it “converges” to an almost constant value after 6 generations. Again, running simulations with different initial random seeds yields similar conclusions.

Comparison with probabilistic forwarding We have conducted a series of simulation runs in which message delivery was achieved through pure probabilistic forwarding [6]; that is, a scheme in which all nodes use the same fixed forwarding probability. We ran simulations varying the forwarding probability P from 0 to 1 and computed the cost for γ in $\{100, 400, 800, 1600\}$. For each γ we conducted simulations using our evolutionary forwarding scheme.

Figure 4 reports the cost, as expressed in (1), achieved by

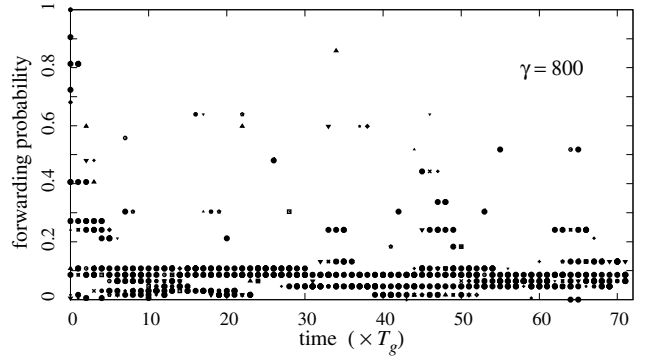


Figure 2. Genotype evolution over time.

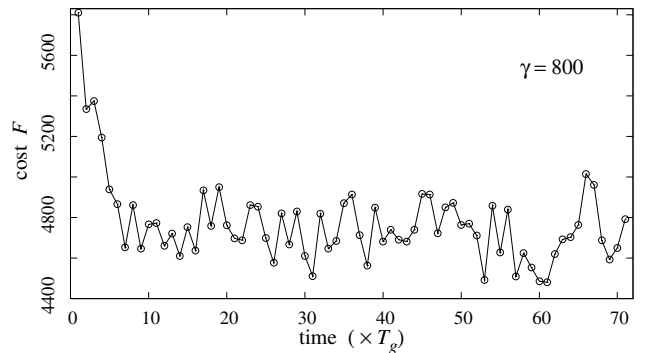


Figure 3. Cost averaged over all messages delivered in a generation vs. time.

the probabilistic forwarding scheme against the forwarding probability. The cost achieved by our scheme is also reported for the sake of comparison.

As one could expect, for low values of γ , like $\gamma = 100$ (corresponding, roughly speaking, to a scenario where resources are not an issue but low delays are required), the cost function is monotonically decreasing in P and flooding ($P = 1$) is the best forwarding policy. On the other hand, if the value of γ is high (resource-constrained scenario, e.g. $\gamma = 1600$), the minimum is for $P = 0$ and nodes should make no copies letting the source deliver its message to the destination. For intermediate values of γ (in our case $\gamma = 800$), a minimum exists and a tradeoff between low delay and low resource consumption can be found. The graph reports also the performance achieved by our evolutionary scheme, after the initial convergence transient. The scheme is able to achieve almost optimal performance for all γ .

Dynamic scenario Last, we consider a simulation case in which N , the number of nodes in the system, varies with time. In particular, it is increased every 20 generations, following the sequence 5, 10, 20, 30, 40. This dynamic scenario challenges the ability of the proposed framework to track the variations in the network and adapt its parameters

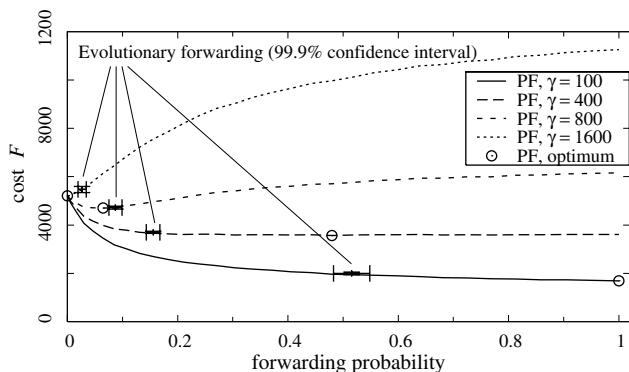


Figure 4. Cost for probabilistic forwarding (PF) and evolutionary forwarding.

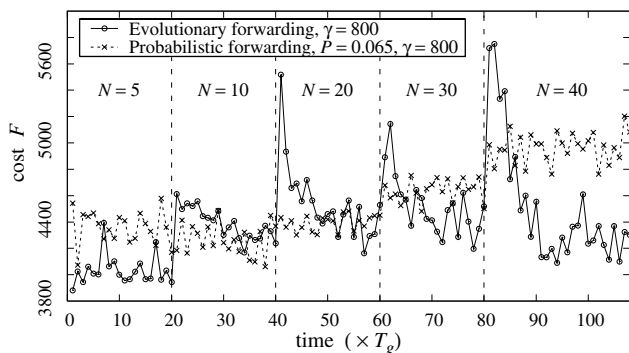


Figure 5. Dynamic scenario: cost vs. time.

accordingly. The time-equivalent cost of a copy is set to $\gamma = 800$. We compute the cost (in seconds) achieved by our evolutionary forwarding scheme in each generation. The results are reported in Fig. 5 which depicts as well the performance of the probabilistic forwarding scheme with the forwarding probability set to its optimal value when $N = 20$ ($P \approx 0.065$; see Fig. 4).

As expected, probabilistic forwarding exhibits steady performance as long as the number of nodes does not change. Instead the cost plot of our evolutionary forwarding scheme presents spikes whenever N increases. The abrupt change is mainly due to the arrival of new nodes, whose initial forwarding probability is set uniformly at random. During the transient following the spike, genotypes fitter to the new scenario are identified and the cost reduces.

Given a network scenario, our evolutionary solution exhibits performance similar to a pure probabilistic forwarding scheme tuned for the specific scenario. But it outperforms it whenever the scenario changes.

8. Conclusion

In this paper, we have presented a framework for embedding autonomous evolution in epidemic-style forwarding schemes. The proposed approach is based on the application of a GA-like mechanism to parameters arrays describing the policy employed by the nodes in the system. The simulation results presented indicate that the proposed case-study implementation is able to track changes in the system conditions (e.g., number of nodes in the scenario considered), and achieves similar if not better performance than solutions statically optimized for a given operating point.

9. Acknowledgments

The authors would like to thank L. Yamamoto, E. Altman, K. Avrachenkov and P. Nain for helpful discussions. This work has been partially supported by the European Commission within the framework of the BIONETS project, IST-FET-SAC-FP6-027748, www.bionets.eu.

References

- [1] S. Alouf, I. Carreras, D. Miorandi, and G. Neglia. Evolutionary epidemic routing. Research Rep. RR-6140, INRIA, 2007. <http://hal.inria.fr/inria-00130803>.
- [2] A. Chaintreau, P. Jui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on the design of opportunistic forwarding algorithms. In *Proc. of IEEE INFOCOM*, 2006.
- [3] K. Fall. A delay-tolerant network architecture for challenged Internets. In *Proc. of ACM SIGCOMM*, pages 27–34, 2003.
- [4] R. Groenevelt, P. Nain, and G. Koole. Message delay in mobile ad hoc networks. *Perf. Eval.*, 62(1-4):210–228, October 5-7 2005. Proc. of Performance 2005, Juan-les-Pins, France.
- [5] Z. J. Haas and T. Small. A new networking model for biological applications of ad hoc sensor networks. *IEEE/ACM Trans. on Networking*, 14(1):27–40, 2006.
- [6] A. Lindgren, A. Doria, and O. Schelen. Probabilistic routing in intermittently connected networks. In *Proc. of SAPIR Workshop*, volume 3126 of LNCS, pages 239–254, 2004.
- [7] G. Neglia and X. Zhang. Optimal delay-power tradeoff in sparse delay tolerant networks: a preliminary study. In *Proc. of ACM SIGCOMM CHANTS*, pages 237–244, 2006.
- [8] OMNeT++ Discrete Event Simulation System. <http://www.omnetpp.org>, 2007.
- [9] L. Pelusi, A. Passarella, and M. Conti. Opportunistic networking: data forwarding in disconnected mobile ad hoc networks. *IEEE Comm. Mag.*, 44(11):134–141, 2006.
- [10] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proc. of ACM WDTN*, 2005.
- [11] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical Report CS-200006, Duke Univ., 2000.