Context and resource awareness in opportunistic network data dissemination *

Chiara Boldrini, Marco Conti, and Andrea Passarella
CNR-IIT, Via G. Moruzzi, 1 - 56124 Pisa, Italy
{c.boldrini,m.conti,a.passarella}@iit.cnr.it

Abstract

Opportunistic networks are challenging mobile ad hoc networks characterised by frequent disconnections and partitioning. In this paper we focus on data dissemination services, i.e. cases in which data should be disseminated in the network without a priori knowledge about the set of intended destinations. We propose a general autonomic data dissemination framework that exploits information about the users' context and social behaviour, to decide how to replicate and replace data on nodes' buffers. Furthermore, our data dissemination scheme explicitly takes into account resource constraints, by jointly considering the expected utility of data replication and the associated costs. The results we present show that our solution is able to improve data availability, provide fairness among nodes, and reduce the network load, with respect to reference proposals available in the literature.

1. Introduction

Opportunistic networks are wireless mobile self-organising networks in which the topology is extremely dynamic and unstable. Disconnections of nodes, high churn rates, dynamic creation and merging of partitions are considered as normal features of the network instead of exceptions, as in the conventional MANET paradigm. Specifically, nodes' movements are exploited to bridge disconnected partitions, allowing end-to-end communication despite prolonged disconnection periods. Research on opportunistic-network protocols has mainly focused on routing issues so far [9]. In this paper we focus instead on *data dissemination* services, to support applications in which the set of users interested in receiving a given data is not known in advance. Data dissemination services are required to support, e.g., user-generated content applications in pervasive networks, in which users dynamically produce content on their mobile devices, and share it with a dynamic and possibly unknown set of peers interested in the same type of data. This data generation and access model is one of the key features of the Web 2.0 paradigm.

If nodes' and network's resources were unlimited, data dissemination would be trivially achieved by flooding-based schemes. However, such solutions become unsuitable as soon as resource consumption is considered. Therefore, the first contribution of this paper is proposing a general framework for *resource-aware* data dissemination services. Specifically, as described in detail in Section 2, in our framework data dissemination is driven by the trade-off between the expected *utility* of data for users, and the associated *cost* of in terms of resource consumption. Besides being completely distributed, the proposed framework is general enough to be customisable to any definition of utility, and to consider any set of resources. Providing such a general framework differentiates our work from the PodNet project [7], which proposes heuristics for data dissemination services. A utility-based system is also proposed in [1], which, however, considers routing protocols instead of data dissemination services, and assumes global knowledge about the network status, which might be impractical to obtain. Energy-delay and storage-delay tradeoffs are investigated in [10], which focuses again on routing issues only. Our proposal is completely structure-less, and thus differentiate itself also from data dissemination schemes requiring some type of network structure, such as multicast trees [13], or broker overlay networks [12].

The second contribution of this paper is to customise the framework by proposing an autonomic, *context-based* data dissemination scheme (see Sections 2 and 3). Exploiting context information about the users has proved to result in particularly efficient forwarding schemes for opportunistic networks [2]. Our scheme dynamically learns context information about the users, their habits, the social communities to which they belong, and exploit this knowledge to compute the utility of data objects. Furthermore, in our scheme nodes not only look for data their users are interested into, but also help disseminating data of interest to users they have acquaintance with. To the best of our knowledge, including the social dimension to data dissemination protocols is an original contribution of this paper.

*This work was partially funded by the IST program of the European Commission under the HAGGLE (027918) FET-SAC project, and under the SOCIALNETS (217141) FET-PERADA project.

978-1-4244-2100-8/08/$25.00 ©2008 IEEE

In Sections 4 and 5 we compare our solution with the best heuristics proposed in [7], showing that our utility-based approach provides a more flexible solution, able to optimise given target performance figures, such as, for example, the hit rate or the fairness among users.

2. Data dissemination framework

2.1. Reference scenario

There are numerous applications for opportunistic networking environments that can benefit from a data dissemination service, particularly in opportunistic networks. Examples include sharing of user-generated content (clips, photos, texts, ...), feeds updates, customised advertisements, etc. As one concrete example to test our data dissemination framework, we focus on the same podcasting application considered in [7]. Data objects (e.g., files, news updates, etc.) are organised in different *channels* (or *feeds*) to which users can subscribe. Data objects can be generated wherever in the network. The node that generates an object also decides the channel(s) the object belongs to. The data dissemination scheme is responsible for bringing to each interested user the data objects of the channels they are subscribed to. At the high level, with respect to this particular application, our data dissemination scheme provides a topic-based pub/sub kind of service.

We assume that users can be grouped in communities, each community representing a social group (e.g., a working environment, a sport team, etc.). We assume that, at any point in time, each user is associated with one "home" community (e.g., the office mates during working hours) in which they roam. Users can also have social relationships outside their home communities, and therefore occasionally move to "visit" other communities. Finally, we assume that each user subscribes to each given channel with some probability. Since communities represent homogeneous social group, we assume that the subscription probabilities are the same for all users of the same community.

2.2. Data dissemination scheme

The data dissemination scheme we propose is completely structure-less. In our scheme each node, upon meeting a peer, opportunistically evaluates which data objects available on the peer should be pulled and stored in its local buffer. Data dissemination occurs because nodes pull data objects upon meeting with each other.

To enable nodes to decide which objects to pull, we use a *utility-based* approach, under the assumption that nodes can dynamically compute a utility value for each data object. The main idea of the proposed framework is that each node constantly aims at *maximising* the total utility of the data objects it stores in the *local* buffer. Thus, upon each contact with a peer, the node computes the utility of the data objects stored by the peer (i.e., the *peer's objects*) and of the data objects it currently stores (i.e., the *local objects*). For each object, it also computes the cost in terms of resources' consumption related to storing it (e.g., the buffer occupation for a local objects, the buffer occupation and the fetching cost for peer's objects). Then, the node selects the set of data objects (either local or peer's objects) that maximises its total utility, subject to the constraints imposed by the resource limitations. Finally, it fetches those data objects in the identified set, that are stored on the peer (as well as any additional data objects stored on the peer the local user has explicitly requested). Our framework aims at a *global* goal (efficiently disseminating data in the whole network), by using *local* rules and knowledge only (as will be clear from the following, we define utility functions such that nodes do not require any global knowledge to compute them). Such an approach is suitable to an opportunistic networking environment, where global knowledge and control is hard (or even impossible) to achieve. The definition of the utility function, and the related results we present hereafter, are an initial step to quantitatively understand how such an approach works in practice.

Formally, upon each contact with a peer, the node identifies the set of objects to store by solving the following problem:

$$\begin{cases} \max & \sum_k U_k^{(p)} x_k^{(p)} + \sum_k U_k^{(l)} x_k^{(l)} \\ \text{s.t.} & \text{resource constraints are met} \end{cases} \quad (1)$$

where $U_k^{(p)}$ and $U_k^{(l)}$ are the utility of the k-th peer's and local object, respectively, and $x_k \in \{0,1\}$ are the variables of the problem. By assuming that m resources have to be managed at each node, and by denoting with $c_{jk}, j = 1, \ldots, m$ the percentage of consumption of resource j related to the k-th data (i.e., $0 \leq c_{jk} \leq 1$), the problem in Equation 1 can be cast to a Multi-costrained 0-1 Knapsack Problem (MKP) ([6]):

$$\begin{cases} \max & \sum_k U_k x_k \\ \text{s.t.} & \sum_k c_{jk} x_k \leq 1 \quad j = 1, \ldots, m \\ & x_k \in \{0,1\} \quad \forall k \end{cases} \quad (2)$$

Note that, based on the above formulation, when the same data object is stored both locally and on the peer's buffer, the local object will be preferred as long as fetching has a non-negligible cost. Selecting the peer's copy will result in a higher cost (due to fetching costs), and will thus result in a solution of the problem with lower utility with respect to the solution that keeps the local copy. Also note that, as the number of constraints is equal to the number of resources, which is not expected to be high, computing the optimal solution or accurate approximations of the optimal solution is fast from a computational standpoint [6]. The formulation in Equation 2 is quite flexible, as it can be used for any set or managed resources, and for any definition of the utility function. In the following of this section we propose a social-aware definition of the utility. Section 3 presents the protocol that nodes execute during meetings to gather the information required to compute this function.

2.3. Context-aware utility

The utility value of a data object represents the utility of storing the object, for the node that computes the utility value. We define the utility function so as to capture the utility of an object for i) the local user of the node, and ii) the communities the local user is in touch with. Therefore, the values computed by our function reflect the social relationships of the local user, as they represent the "social utility", for the node, of storing the data objects. Specifically, we define the utility value for a data object k as:

$$U_k = u_k^{(l)} + \sum_i \omega_i u_{k,i}^{(c)} \quad (3)$$

where $u^{(l)}$ is the utility component related to the local user, $u_{k,i}^{(c)}$ is the component related to the i-th community the user is acquainted with, and ω_i is a cooperation index (a real number between 0 and 1), which defines the willingness of the user to cooperate with the i-th community. It is clear that this utility definition permits a social, non-greedy behaviour.

In general, we define each utility component of Equation 3 as a function of *context* parameters that describe the local user or the considered community. A discussion about these parameters in the most general case can be found in [4]. In this paper, we consider a simplified, yet significant, case, in which each utility component is a function of i) the *access probability* to the object, ii) the object *availability* in the context, and iii) the object *size*. The function we define to compute utility components is the same for all components. The values of the parameters change depending on the considered component (e.g., the access probability to a data object for the local user and for the communities the user is in touch with will, in general, be different). This permits to handle all utility components in Equation 3 in a homogeneous way. Specifically, by denoting by p_{ac} the access probability to the object, by p_{av} the probability that the object is available in the considered context, and by s the size of the object, the utility components are computed as[1]:

$$u = \frac{p_{ac} \cdot e^{-\lambda p_{av}}}{s} \ . \quad (4)$$

Equation 4 has the same general form of utility functions widely adopted in the literature about cache replacement systems (e.g.,[11]), where utility is defined as the product of the access probability by the value of the data object. In our definition, the object's value is a decreasing function of the object's availability (i.e., the more the object is spread, the lesser it is useful), which is inspired from the "rarest first" policy of BitTorrent. Finally, normalising by the object's size is also common in the literature, as it allows for very simple and effective approximations of the multi-constrained knapsack problem defined by Equation 2.

[1] Since there are no distinctions between the definition of the components for the local user and its communities, hereafter we drop the indices used in Equation 3.

3. Context-aware data dissemination protocol

To implement our framework in a real opportunistic network we define HiBOp-D, an extension of the HiBOp context-aware routing framework defined in [2]. Before presenting the HiBOp-D details, it is worth discussing a few assumptions that we consider.

In general, our framework requires that each user can detect the communities they are acquainted with, so as to separately gather context information related to the different communities, and correctly compute the utility components as per Equations 3 and 4. Some preliminary results on autonomic community detection can be found in [5], however, this is still an open research problem. To show that our framework can be used even with imperfect knowledge about the communities' structure, in HiBOp-D nodes consider just two contexts that can always been identified, i.e., i) the context related to the local user, and ii) the context describing the historical information about *all* the users met in the past. The latter context aggregates *all* information related to the different communities a user visits. Therefore, with respect to the general definition in Equation 3, in HiBOp-D nodes compute two utility components, representing the local user, and the aggregate of all communities, respectively.

Furthermore, to compute the utility components according to Equation 4, each node has to estimate the access probability and availability of data objects related to each context. For simplicity, in the following we describe a simplified way to compute these indices that leverages some features of the podcasting scenario described in Section 2.1. Extending the algorithms to more general cases is quite simple. Firstly, we consider that, given a particular context, the access probability is the same for all data objects belonging to the same channel. Furthermore, we approximate the availability of *each* data object of a channel with the *aggregate* availability of all the channel's objects. That is, we see p_{av} as the probability of "seeing" objects of the channel in the context. This approximation allows us to significantly reduce the state required by the protocol. We are currently investigating ways of maintaining information about availability at a finer granularity, while still keeping the protocol state at a reasonable level.

Since, according to these assumptions, the access probability and availability are the same for each channel, and each utility value consists of two components, HiBOp-D must compute four indices for each channel, i.e., the access and the availability probabilities for the local and the historical contexts. Hereafter, the indices related to data objects of channel j will be denoted, respectively, as $p_{ac,j}^{(l)}$, $p_{ac,j}^{(h)}$, $p_{av,j}^{(l)}$, and $p_{av,j}^{(h)}$.

When a node meets a peer, it must know which data objects are stored on the peer's buffer, and the objects' size. To this end, upon a contact, nodes exchange an index of the

data they are storing with a unique object ID and the object size[2]. The rest of HiBOp-D deals with gathering information to compute the four parameters of the utility components. For the generic channel j, the access probability for the local context ($p^{(l)}_{ac,j}$) is 1 if the user is interested into the channel, or 0 if the user is not interested. The access probability for the historical context ($p^{(h)}_{ac,j}$) is seen as the average access probability of nodes met in the past. This index is dynamically computed over time as follows. Periodically, each node broadcasts the set of channels it is subscribed to. Once every period, each node computes a sample of the access probability for the historical context (denoted as $s_{ac,j}$), as $s_{ac,j} = \sum_{i=1}^{N} \delta_{ij}/N$ where N is the number of nodes met in the period, and δ_{ij} is equal to 1 if the i-th met node is interested in channel j, and 0 otherwise. The access probability for the historical context ($p^{(h)}_{ac,j}$) is computed as the smoothed average of the $s_{ac,j}$ samples, i.e., $p^{(h)}_{ac,j} \leftarrow \alpha p^{(h)}_{ac,j} + (1-\alpha)s_{ac,j}$. The rationale of the algorithms to compute the availability indices is similar. The local availability ($p^{(l)}_{av,j}$) is seen as the average availability of channels on nodes met in the past, i.e., it is a measure of how "easy" is for the *local* user to see objects of the channel j on the neighbours' buffers. To compute $p^{(l)}_{av,j}$, each node periodically broadcasts the fraction of the buffer occupied by each objects of each channel (hereafter f_{ij} denotes the fraction of the i-th node's buffer occupied by objects of channel j). At the end of each period, a sample of the local availability ($s_{av,j}$) is computed as $s_{av,j} = \sum_{i=1}^{N} f_{ij}/N$, and $p^{(l)}_{av,j}$ is updated as $p^{(l)}_{av,j} \leftarrow \alpha p^{(l)}_{av,j} + (1-\alpha)s_{av,j}$. Finally, the availability for the historical context ($p^{(h)}_{av,j}$) is seen as a measure of how "easy" is, for the *nodes the local user met in the past*, to access data objects of channel j. To compute it, nodes also broadcast their current value of $p^{(l)}_{av,j}$. At the end of each period, a sample of historical availability is computed as $\hat{s}_{av,j} = \sum_{i=1}^{N} p^{(l)}_{av,j,i}/N$, and the the availability for the historical context is updated as $p^{(h)}_{av,j} \leftarrow \alpha p^{(h)}_{av,j} + (1-\alpha)\hat{s}_{av,j}$. The detailed mechanisms used to implement these computations by exploiting the HiBOp framework are presented in [4].

4. Simulation Setup

To properly simulate users' movements driven by social behaviour, we consider the Home-cell Community based Mobility Model defined in [3]. In HCMM user's movements are driven by the attraction exerted on the user by i) other peers (e.g., users move to meet their friends), and ii) physical locations (e.g., users move to their office building). In this work we consider a very simple scenario, as a first step to analyse the performance of the utility-based framework. Specifically, we consider just a single community, and focus on intracommunity dynamics only. It is easy to show that in HCMM, in the case of a single community, users move according to the corrected Random Waypoint (RWP) mobility model [8]. The single-community scenario is a worst-case scenario for our data dissemination framework, as the advantage of exploiting information about users' social behaviour is likely to be much more evident in the case of several groups of users, each with different subscription distributions.

Node Speed	uniform in [1,1.86] m/s
Buffer size	10 objects
Transmission Range	20m
Sampling period	5s
λ	15

Table 1: Configuration Parameters

In our setup data objects can belong to three different channels, and we assume that 100 distinct data objects exist for each channel. These objects are generated before the simulation begins and never expire. All the data objects have the same size, equal to 50 KB. Each node is interested into one channel, and interests are distributed among nodes according to a Zipf's distribution (with parameter equal to 1 in our simulations). The nodes move, with a typical pedestrian speed, in a 250x250m cell. In a first set of experiments, we assume that an Access Point (AP) is placed in the middle of the cell, which constantly stores all the 300 data objects. In such configuration, nodes can get whatever data object they want when passing by the AP. In this case the data dissemination system helps distributing data without relying on a unique AP only. This is particularly relevant in sparse scenarios, in which contacts with the AP might be quite sporadic. In a second set of experiments, we assume that the AP is not available. In this case data objects are available in the network just because they are stored by nodes. In all experiments nodes' buffer size is set to 10 objects, so as to test the framework in a resource constrained environment as far as buffer limitations. Furthermore, we consider only the buffer as the constrained resources to be managed according to the scheme in Equation 2. Reference values for other configuration parameters are summarised in Table 1.

In the following, we compare the following policies, used to choose the data objects to store upon a contact:

Context-aware (CA) Objects are selected according to our framework, by using the context-aware function as per Equations 3 and 4.

Access Probability (aP) Objects are selected according to our framework, but by considering the access probability only in Equation 4.

Uniform (U) Objects are selected randomly among all available objects in the local and peer's buffers (regardless the channel they belong to). This policy has been proved to be one of the best in [7].

[2]Since in opportunistic networks data are carried in form of *large* bundles, the index is typically much shorter than the data itself.

Uniform mono-channel (UM) Nodes randomly select a channel, and store objects of that channel only. We consider this version of the uniform policy as we found that nodes under the aP and CA policies tend to store objects of a unique channel only.

The performance of these policies is evaluated in terms of the quality of service (QoS) perceived by the users and the resource consumption. The QoS is measured in terms of hit rate, system utility and fairness (both per user and per channel). To measure the hit rate, we simulate a user periodically requesting (with a period equal to 300s) a data object for each channel (the object of a given channel is randomly selected according to a uniform distribution). We consider a hit if the object is available on at least one node's buffer, a miss otherwise. The system utility is computed as the sum of the channel hit rate weighted with the access probability of each channel, i.e., $SU = \sum_i p_{ac,i} hr_i$. The fairness (with respect to the hit rate) of each policy has been computed according to the Jain's fairness index. Resource consumption has been measured in terms of the traffic generated in the network, i.e., the average number of data transmitted by all nodes during the simulations. This includes data exchanged for context creation, buffer state messages, request messages and data objects themselves. Simulations run for 90000s. Exchanges of data objects upon nodes' contacts start after an initial transitory required to build nodes context. Results shown in the following section have a 95% confidence interval, obtained through the independent replication technique. For space reasons, in the following we show the system utility, fairness, and traffic overhead results only. Non-aggregated results showing the hit rate of each channel for each policy can be found in [4].

5. Simulation Results

5.1. Impact of network size

In this first set of experiments we evaluate the performance of the policies under a variable number of nodes, ranging from 8 to 72. We assume the Access Point is available. In this configuration, we have replicated experiments by initially filling the nodes' buffers with different policies (including empty buffers, random uniform selection over all the 300 messages with and without replication on different nodes). However, when the AP is available, the initial filling of nodes' buffers does not have a significant impact on the results [4]. Figure 1 shows the system utility index. Being SU a measure of user satisfaction, it is quite expected that the aP policy, by favouring the most popular channel, satisfies the majority of the users. However, Figure 2 shows that this is paid in terms of per-user fairness (the per-channel fairness provides similar results). Uniform policies achieve, by definition, the highest fairness, at the cost of lowest system utility. Figure 1 highlights that a better trade off can be reached by the CA policy. Its fairness level approaches that of the uniform policies while still maintaining an higher

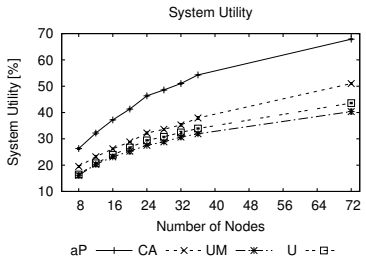

Figure 1: System Utility

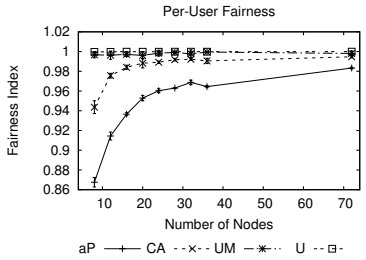

Figure 2: Per-user fairness

level (between 10% and 20%) of user satisfaction. Also note that there is basically no difference, in our setup, between the U and UM policies. Figure 1 also highlights that no policy is able to reach 100% hit rate, not even when the sum of nodes' buffers would be enough for storing all the data objects for all channels (this is the case for nodes greater than 30). This is because no policy is able to control the replication level of data objects across the network. Improving the policies to achieve higher diversity of stored objects is one of the most compelling extensions of our framework. Beside providing flexible trade-offs between user satisfaction and fairness, the utility-based framework is resource-efficient, as well. Figure 3 shows the traffic overhead figures. The most efficient policy is aP because, once node interests have spread all over the network, values for access probabilities don't change anymore. This results in fewer exchanges of data objects (basically, only nodes sharing the same interest exchange objects). In the CA policy, the utility ranking changes dynamically based on the availability measure, thus resulting in a slightly higher traffic overhead. The uniform policies waste more resources because they don't take into account the state of the network, and thus objects are exchanged at every association between nodes.

5.2. Turning the AP off

The AP is fundamental to the regeneration of data objects in the network: in fact, even when all nodes have deleted a given object, the AP guarantees that this object can be injected into the network again. We consider two filling policies of nodes' buffers at the simulation start: uniform filling, where data to be copied on nodes' buffers are selected uniformly among all available objects, and maximum diversity filling, where objects are copied on nodes as to minimise replications. We consider the case of 16 nodes,

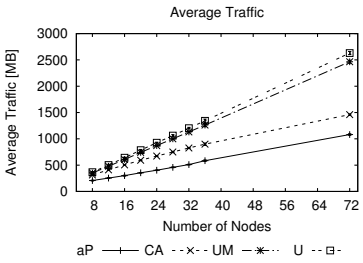

Figure 3: Traffic overhead

	uniform		max div	
	AP on	AP off	AP on	AP off
aP	35.38 ± 0.16	9.33 ± 0.32	35.38 ± 0.14	9.43 ± 0.37
CA	26.23 ± 0.30	6.09 ± 0.39	26.10 ± 0.26	6.06 ± 0.44
U	23.93 ± 0.17	5.45 ± 0.23	23.88 ± 0.20	5.65 ± 0.25
UM	22.92 ± 0.39	4.2 ± 1.06	22.93 ± 0.35	6.34 ± 1.34

Table 2: System utility with and without the AP.

in which not all objects can fit into nodes buffers to highlight the importance of the AP role in this challenged setting. Table 2 shows that when the AP is off a certain level of service is maintained, though highly downgraded. This shows that, when the buffer size is very small, the AP is key to provide a reasonable service level. These results also show that, in these settings, the different initial filling policies seem to have no, or very little effect, on system performance.

5.3. Dense scenario

The last set of results we show highlight a mis-behaviour of the CA policy that arises in denser scenarios without AP, and a simple modification to fix it. Specifically, we consider a scenario with 16 nodes, but with transmission range equal to 80m (instead of 20m). With this setting nodes have a higher number of neighbours, and quickly achieve a complete view of what is available in the buffers of all the other nodes. Since the CA policy takes into consideration availability to compute utility, the most useful channel at the beginning of the simulation tends to be the only one that survives. This is because nodes' choices tend to be synchronised as a side effect of the complete knowledge about availability that each node quickly achieves. Thus, the initial objects' allocation on buffers determines the channel that will survive, and different simulation runs provide completely different performance results. Table 3 shows the percentage of total buffer space (considering all the nodes' buffers as a single shared buffer) occupied by each channel for the utility-based policies (CA and aP only).

A simple fix to this problem for the CA policy is as follows. When nodes meet, they compute the utility of the channel as usual, but then they use the utility values to define the *share* of local buffer to be occupied by each channel's objects. The last row in Table 3 shows that this simple modification is able to fix the CA misbehaviour, as less useful channels never disappear. We are currently investigating the applicability of this modified CA policy in more general settings.

	ch1	ch2	ch3
aP	56.04 ± 0.00	24.81 ± 0.00	18.54 ± 0.00
CA (orig)	0.61 ± 0.60	0.56 ± 0.62	98.10 ± 1.54
	89.66 ± 0.66	9.92 ± 0.10	0.00 ± 0.00
	0.00 ± 0.00	99.25 ± 1.04	0.00 ± 0.00
CA (fix)	42.04 ± 8.59	34.33 ± 7.46	22.95 ± 15.94

Table 3: Percentage of nodes' buffer occupied by each channel.

6 Conclusions

In this paper we have introduced a new autonomic utility-based data dissemination framework for opportunistic networks which exploits context information about the users' social behaviour. We have compared the performance of the framework with other solutions considered among the best in the literature. Results have shown that our approach is flexible enough to achieve different targets, and specifically can be customised either to achieve the maximum hit rate for most popular data, or to achieve a fairer behaviour with acceptable reduction of the hit rate. We have also shown preliminary yet interesting results showing that, in particularly challenged scenarios, network elements, such as Access Points, that can store *all* available data, are key. The preliminary results we have presented highlight that a utility-based framework is an interesting direction to be further investigated to provide data dissemination services in opportunistic networks.

References

[1] A. Balasubramanian, B. N. Levine, and A. Venkataramani. Dtn routing as a resource allocation problem. *Proc. of SIGCOMM*, 2007.

[2] C. Boldrini, M. Conti, I. Iacopini, and A. Passarella. HiBOp: a History Based Routing Protocol for Opportunistic Networks. In *Proc. of IEEE WoWMoM*, June 2007.

[3] C. Boldrini, M. Conti, and A. Passarella. Users mobility models for opportunistic networks: the role of physical locations. In *Proc. of IEEE WRECOM*, 2007.

[4] C. Boldrini, M. Conti, and A. Passarella. Context and resource awareness in opportunistic network data dissemination. Technical Report, IIT-CNR, 2008. http://bruno1.iit.cnr.it/∼chiara/dataDisseminationTR.pdf.

[5] P. Hui, E. Yoneki, S. Chan, and J. Crowcroft. Distributed community detection in delay tolerant networks. *Proc. of ACM MobiArch*, 2007.

[6] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack Problems*. Springer, 2004.

[7] V. Lenders, G. Karlsson, and M. May. Wireless Ad hoc Podcasting. In *Proc. of IEEE SECON*, 2007.

[8] W. Navidi and T. Camp. Stationary distributions for the random waypoint mobility model. *IEEE Transactions on Mobile Computing*, 3(1):99–108, 2004.

[9] L. Pelusi, A. Passarella, and M. Conti. Opportunistic Networking: data forwarding in disconnected mobile ad hoc networks. *IEEE Communications Magazine*, 44(11), Nov. 2006.

[10] T. Small and Z. J. Haas. Resource and performance tradeoffs in delay-tolerant wireless networks. *Proc. of ACM WDTN*, 2005.

[11] L. Yin, G. Cao, and Y. Cai. A generalized target-driven cache replacement policy for mobile environments. *J. Parallel Distrib. Comput.*, 65(5):583–594, 2005.

[12] E. Yoneki, P. Hui, S. Chan, and J. Crowcroft. A socio-aware overlay for publish/subscribe communication in delay tolerant networks. In *Proc. of ACM MSWiM*, 2007.

[13] W. Zhao, M. Ammar, and E. Zegura. Multicasting in delay tolerant networks: semantic models and routing algorithms. In *Proc. of the ACM WDTN*, 2005.