ARTICLE IN PRESS

Computer Networks xxx (2009) xxx–xxx

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks

Chiara Boldrini *, Marco Conti, Andrea Passarella

Institute of Informatics and Telematics, National Research Council, Via G. Moruzzi 1, 56124 Pisa, Italy

ARTICLE INFO

Article history:
Available online xxxx

Keywords:
Opportunistic networks
Data dissemination
Social-aware networking
Data-centric networks
Performance evaluation

ABSTRACT

In this paper we present and evaluate ContentPlace, a data dissemination system for opportunistic networks, i.e., mobile networks in which stable simultaneous multi-hop paths between communication endpoints cannot be provided. We consider a scenario in which users both produce and consume data objects. ContentPlace takes care of moving and replicating data objects in the network such that interested users receive them despite possible long disconnections, partitions, etc. Thanks to ContentPlace, data producers and consumers are completely decoupled, and might be never connected to the network at the same point in time. The key feature of ContentPlace is learning and exploiting information about the social behaviour of the users to drive the data dissemination process. This allows ContentPlace to be more efficient both in terms of data delivery and in terms of resource usage with respect to reference alternative solutions. The performance of ContentPlace is thoroughly investigated both through simulation and analytical models.

1. Introduction

This paper is focused on the problem of data dissemination in opportunistic networks. Opportunistic networks [18] are a recent mobile networking paradigm stemming from the research on conventional Mobile Ad Hoc NETworks (MANET). In this paradigm nodes are assumed to be mobile, and forwarding of messages occurs based on the store-carry-and forward concept [9]. Specifically, no simultaneous end-to-end multi-hop path is required to enable communication between any two nodes, unlike conventional MANET. Instead, each node carrying a message for an intended destination evaluates the suitability of any other node it makes contact with as the next hop. Messages are thus opportunistically forwarded by exploiting nodes encounters, until they reach the intended destination. This paradigm enables end-to-end communication despite possible long disconnections of the communication endpoints and severe network partitions, which is usually cumbersome in traditional MANET architectures.

A significant share of research on opportunistic networks has focused on routing issues (see [18] for a survey). Instead, in this paper, we consider the problem of data dissemination. This is a key research problem, particularly in opportunistic networks. In this environment, according to the user-generated content wave, users are expected to generate large amounts of content by exploiting capability-rich mobile devices (such as PDAs, smartphones, etc.), and to share them with people around them or people they have social relationships with. In the following, we refer to any piece of content (e.g., a picture or an mp3 file) as a *data object*. The problem of efficiently disseminating data objects in opportunistic networks is thus very relevant, and not widely explored in the literature yet, as discussed in Section 2.

Data dissemination in opportunistic networks is a difficult problem. As the topology is very unstable, and users appear in and disappear from the network dynamically, content providers and content consumers might be completely unaware of each other, and never connected at

* Corresponding author.
E-mail addresses: chiara.boldrini@iit.cnr.it (C. Boldrini), marco.conti@iit.cnr.it (M. Conti), andrea.passarella@iit.cnr.it (A. Passarella).

1389-1286/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.comnet.2009.09.001

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001

the same time to the same part of the network. Therefore, data objects should be moved and replicated in the network in order to carry them to interested users despite disconnections and partitions. On the other hand, data dissemination systems should take care of both network and device resource constraints. For example, a trivial solution would be to flood the whole network with any generated data object, but this would clearly saturate both network resources (in terms of available bandwidth) and device resources (e.g., in terms of energy, storage, etc.).

In this paper, we propose and evaluate ContentPlace, which is a data dissemination system for opportunistic networks that exploits social information about users behaviour in order to drive the dissemination process. Exploiting social information is a very promising research direction for opportunistic networks. In this environment the nodes are mobile devices users carry with them all the time. Therefore, the users social behaviour, being a key driver for their movement patterns, is also a key piece of context information to predict nodes' co-location and future encounters. ContentPlace assumes that users belong to social communities, and autonomically learns the time spent by them in each community, which types of data objects users of each community are interested in, and how spread in the communities the data objects are. This information is used to evaluate the *utility* of each encountered data object. Specifically, each node, upon making contact with another peer, evaluates the utility of the data objects the peer is carrying. Assuming that the buffer space devoted to the dissemination process is limited, the node selects which data objects to fetch from the peer, in order to maximise the total utility of the data objects in its own buffer. Therefore, the data dissemination process is driven by the interests and social behaviour of the users, and just requires local interactions between nodes that happen to come in contact.

We assume that users' mobility is driven by the social relationships among users, i.e., that users spend their time with their friends. This assumption may not be always satisfied. This is the case, e.g., of friends living in different part of the world or of virtual friends (e.g., Facebook or chatroom friends). However, also these relationships can be exploited to disseminate messages. For example, in the case of long distance friendship, the distance can be overcome by going through the traditional infrastructure and then switching again to the ad hoc mode when this distance has been covered. The inclusion of such hybrid communications into our data dissemination system is currently under study.

After presenting ContentPlace in Sections 3 and 4, we provide a detailed simulation analysis in Section 5. Specifically, we compare different data dissemination policies that could be plugged in the general ContentPlace design, either considering or not considering social information. We show that social-aware policies outperform naïve ones in which social information is not taken into account. Among the social-aware policies, we identify the one performing best (named Future), and highlight the reasons why it is the most efficient one. Finally, in order to shed additional light on the performance of the Future policy, in Section 6 we provide an analytical model describing its behaviour.

2. Related work

Content dissemination systems have been proposed with regard to legacy Internet networks [1], and also with respect to conventional MANETs [22]. In general, these systems assume that network paths are rather stable, and in some cases generate a significant amount of traffic to maintain knowledge of other nodes' caches. Therefore, they are not suitable to opportunistic networks.

The idea behind ContentPlace is to exploit social information on the environment the nodes operate in, in order to enable the communication. In the framework of opportunistic networks, this approach has already been successfully applied to message forwarding (e.g., [6,12]). The idea is to move messages closer and closer to their destinations following a path based on the social interactions between nodes (as in the famous "six degrees of separation" experiment [17]). In the case of forwarding protocols, however, messages have a specific destination node, while in ContentPlace, following the User Generated Content approach, content generators might be unaware of the nodes interested in their data, and so might be the content consumers about the nodes that generate the content they are interested in. In addition, even if pursuing a similar goal, ContentPlace does not rely on any underlying forwarding protocols and autonomously takes care of the data delivery process.

In principle, ContentPlace shares similarities with pub/sub systems proposed for mobile networks (e.g., [8,23]). Among them, just the pub/sub system designed within the Haggle project [23] explicitly considers the social behaviour of users in the system's design. Furthermore, in pub/sub systems, support for intermittent connectivity is seldom provided. The work in [23] identifies social communities, and "hubs" within communities (i.e., nodes with the highest number of social links inside the community). An overlay network is then built between hubs, that act as the brokers of a standard pub/sub topic-based system. ContentPlace assumes the same community detection mechanisms of [23]. However, ContentPlace does not provide any standard pub/sub system, and does not need any overlay infrastructure, which might be costly to maintain and rather unstable in opportunistic networks. With respect to SocialCast [8], ContentPlace uses a more complete utility function to drive the dissemination process. Specifically, ContentPlace takes into account the estimated utility for all social communities any given user is in touch with, and, within each community, considers the interest for, and availability of, the data objects. Furthermore, the SocialCast dissemination mechanism is more oriented towards traditional forwarding than to actual dissemination with respect to ContentPlace. A fallback of this is that SocialCast works well when all members of each community are interested in the same type of content, but it is not clear how it works in the more general settings considered in this paper.

To the best of our knowledge, the only other works looking at pure content dissemination for opportunistic networks are the PodNet project [16], and the social-aware pub/sub system designed in [23]. As described in detail in Section 5, we use an application and evaluation scenario similar to that defined for PodNet. In PodNet,

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001

users subscribe to channels they are interested in. Upon pair-wise contacts, users exchange their interests and select which data objects to exchange. Different policies have been compared in [2]. With respect to PodNet, ContentPlace takes into consideration social relationships between users to select the data object to exchange, and provides a more general utility based framework for designing content dissemination policies. We actually compare ContentPlace with the best heuristics identified in [16], showing the advantage of the social-aware dimension.

This work is an extended version of our previous paper in [4]. This work provides an extensive analytical model that deals with the average behaviour of the best dissemination policy, the Future policy. Furthermore, we perform an analysis of the impact of inaccurate social information on the performance of the best social-aware policy. This work is also related to [5], where a preliminary design and evaluation of ContentPlace has been presented.

3. ContentPlace general design

This section provides necessary background information required to present the main contribution of this paper by recalling the target application scenario and the main design features of the ContentPlace system.

3.1. Application scenario

The application scenario we target is similar to the one used in PodNet [16], named "podcasting for ad hoc networks". As in the typical opportunistic networking paradigm, we consider a number of mobile users whose devices cannot be encompassed by a conventional MANET. Instead, communication is achieved by opportunistically exploiting pair-wise contacts between users to exchange data objects, and bringing them towards eventual destinations. Sporadic contacts of users with point of access to the Internet (e.g., WiFi hotspots) are possible although not necessary. In podcasting applications, data objects (e.g., MP3 files, advertisements, software updates, ...) are organised in different *channels* to which users can subscribe. We assume that the channel(s) of a data object is decided by the source of the object at the generation time. Data objects might be generated from within the Internet, and "enter" the opportunistic network upon sporadic contacts of users with Internet Access Points. Or, data objects may be generated dynamically by the users of the opportunistic network according to the Web 2.0 model (e.g., users may wish to share pictures taken with their mobile phones). ContentPlace is responsible for the two main tasks of content dissemination, i.e.: (i) managing subscriptions to channels and (ii) bringing data objects to subscribed users (content distribution).

3.2. ContentPlace framework

Since stable network structures cannot be assumed, ContentPlace only exploits direct interactions between nodes (contacts) to gather information about the users' subscriptions and current data objects availability. Each node uses this knowledge to decide which data objects "seen" on other nodes should be locally replicated, according to a *replication* policy. The main challenge of ContentPlace is to define a *local* replication policy (i.e., a policy that does not require precise information about the global state of the network) that achieves a *global* performance target (such as, for example, maximising the hit rate or the per-user fairness).

More in detail, ContentPlace subscription management works as follows. Nodes just advertise the set of channels the local user is subscribed to upon encountering another node. As will be clear in the following, no per-node state is necessary, and thus unsubscription data objects are not required. As far as content distribution is concerned, the core of ContentPlace is the definition of the replication policy, which can be summarised as follows. ContentPlace defines a utility function by means of which each node can associate a utility value to any data object. When a node encounters a peer, it computes the utility values of all the data objects stored in the local and in the peer's cache.[1] Then, it selects the set of data objects that maximises the *local* utility of its cache, without violating the considered resource constraints (e.g., max cache size, available bandwidth, available energy, ...). The node fetches the selected objects that are in the peer's cache, and discards the locally stored objects that are not in the selected set anymore. Finally, a user receives a data object it is subscribed to when it is found in an encountered node's cache. As discussed in more detail in [5], the set of objects to store in the local cache upon each contact can be found by solving the following multi-constrained 0–1 knapsack problem:

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

where k denotes the kth object that the node can select, U_k its utility, c_{jk} the percentage consumption of resource j related to fetching and storing object k, m the number of considered resources, and x_k the problem's variables. When the number of managed resource (m) is not big (which is quite reasonable), solving such problems is very fast from a computational standpoint [15]. Such a solution is therefore suitable to be implemented in resource constrained mobile devices.

It is clear that the core of the content distribution mechanism is the definition of the utility function. In the following sections, we discuss how information about the social behaviour of users can be leveraged to this end.

3.2.1. Utility function

To have a suitable representation of the users' social behaviour, we take inspiration from the caveman model proposed by Watts [21], which is a reference point in the field of social behaviour modelling. We assume that users

[1] Hereafter, we use the term cache to denote a memory buffer that a node contributes to the ContentPlace system.

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001

can be grouped in communities. Users belonging to the same community have strong social relationships with each other. In general, users can belong to more than one community (a working community, a family community, etc.), each of which is a "home" community for that user. Users can also have relationships outside their home communities ("acquainted" communities). We assume that people movements are governed by their social relationships, and by the fact that communities are also bound to particular places (i.e., the community of office colleagues is bound to the office location). Therefore, users will spend their time in the places their home communities are bound to, and will also visit places of acquainted communities.

Different communities will have, in general, different interests. Therefore, the utility of the same data object will be different for different communities. Based on this remark, the utility of a data object computed by a node is made up of one *component* for each community its user has relationships with, be it either a home or an acquainted community. Formally, the utility function is defined as follows[2]:

$$U = \omega_l u_l + \sum_{i \neq l} \omega_i u_i = \sum_i \omega_i u_i, \quad (2)$$

where u_i is the ith component and ω_i measures the social strength of the relationship between the user and the ith community. Finally, in Eq. (2) we stretch a bit the concept of community, and represent the *local* user just as another community the user is in touch with (i.e., the utility for the local user is represented by u_l). Note that the definition of the weights ω_i defines the social-oriented behaviour of ContentPlace. As the main focus of this paper is on this aspect, we now briefly describe the definition of the utility components, and discuss in detail the definition of weights in Section 4.

3.2.2. Utility components

ContentPlace uses the same definition for all utility components. In this paper we consider a simplified version of the general function defined by ContentPlace, and we also assume that (i) the cache space is the only considered resource and (ii) data objects never expire. See [5] and the associated report for the discussion of more general cases. Inspired by the Web-caching literature [1], the utility of a data object for a community is the product of the object's access probability from the community members (p_{ac}) by its cost c, divided by the object's size. The cost is measured as a monotonically decreasing function of the object's availability in the community (denoted as p_{av}), as the more the object is spread, the less it is costly to find it, the less the utility of further replicating it. Dividing by the object's size is also common in the Web-caching literature, as it also allows very simple approximations of the multi-constrained knapsack problem defined by Eq. (1). Specifically, we use the following definition:

$$u = \frac{p_{ac} \cdot f_c(p_{av})}{s} = \frac{p_{ac} \cdot e^{-\lambda p_{av}}}{s}. \quad (3)$$

In Eq. (3) we use an exponential function as cost function, which achieves a fairer behaviour with respect to a (more intuitive) linear decay, as shown in [5].

3.3. Architecture overview

In this section we give an overview of the communication architecture of ContentPlace. Following the opportunistic networking paradigm, ContentPlace does not assume any specific technology (e.g., 802.11, Bluetooth) implementing the single hop wireless communication on which it is based. It only relies on the existence of an underlying abstraction level that manages the available interfaces and provides the correct mapping between them [19]. ContentPlace does not rely on any routing protocols either, as it autonomously uses application layer information (in this case, information on the sociality of nodes) to disseminate objects across the network.

The dissemination of messages in ContentPlace exploits pair-wise contacts between nodes, during which an association is established (Fig. 1). At the beginning of a contact, a new association is established if the two nodes are both free, i.e., not already engaged in an ongoing association. Otherwise, the association request is queued until the busy node becomes free again. During an association, each node acts as both the content uploader and the content downloader, for itself or on behalf of the other nodes of its communities. An association is completed when both peers have finished downloading data objects deemed useful (for themselves or for the nodes of their communities, according to Eq. (2). An association can terminate unexpectedly before the downloading is completed, e.g., because the two nodes have moved out of range or because one of them has run out of battery. In this case, the association is closed, data objects that were successfully transferred are stored in the buffer, the others are discarded.

When a new association is established, the two nodes exchange the state of their respective buffers, i.e., they tell each other what data objects they are currently storing. The algorithm described in Eq. (1) is then run on the joint set of the locally stored objects and the objects advertised by the peer, and, if some of the peer's object are selected to be retrieved, a request is issued to the peer. When the peer has provided the requested objects, the requesting node stores the new objects in its buffer and closes the handshake on its side. The communication exchange is sketched in Fig. 2.

Besides the data object exchange process, in ContentPlace nodes have also to keep up-to-date the estimates of the utility component's parameters (Eq. (3)). To this aim, nodes periodically broadcast a summary message containing the necessary information to compute these estimates. More details on this will be given in Section 4.2.

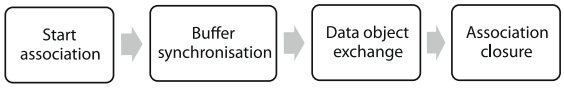


Fig. 1. ContentPlace architecture – setting up an association.

[2] For ease of reading, with respect to Eq. (1) we drop here the k index, as this does not impact the clarity of the discussion.

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001

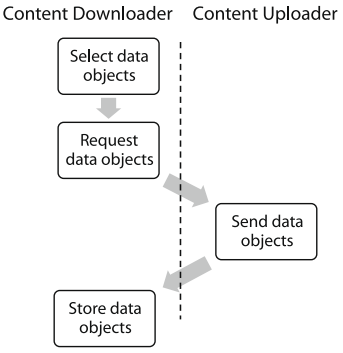


Fig. 2. ContentPlace architecture – data object exchange.

4. Contentplace social design

The use of the social weights in Eq. (2) permits full flexibility in the ContentPlace design. Specifically, it allows us to define and compare different social-oriented policies, as well as, in the limiting case, *non*-social policies such as a greedy behaviour. Although clearly not exhaustive, the set of social-oriented policies we compare covers a fairly large spectrum of possible (reasonable) definitions. Overall, the global goal we wish to achieve is to optimise the hit rate for *all* users in *all* communities. Different policies clearly give different importance to the utility components. The comparison we carry out shows which are the components that provide the best behaviour with respect to the desired global target.

4.1. Policies definition

We consider the following policies:

Most Frequently Visited (MFV): Each community is given a weight proportional to the time spent by the user in that community. Specifically, if t_i is the time spent by the user in the ith community since the system start-up, then w_i is equal to $t_i/\sum_i t_i$. With this policy, dissemination decisions of a user favour the communities the user is more likely to get in touch with.
Most Likely Next (MLN): The weight of the ith community is equal to the probability of visiting the ith community, conditioned by the fact that the user is in the current community. Hereafter, this probability is referred to as $P(c_i|CC)$, where C denotes the current community. The weight of the current community is set to 0. With this strategy, users favour the communities they will be visiting next, but do not contribute to data dissemination within their current community.
Future (F): As in MLN, the weight of the current community is set to 0. However, the weight of all other communities is set as in MFV, i.e., is proportional to the average time spent by the node in that community. With respect to MLN, with F we consider not only the most likely community for the next visit, but all the communities the user is in touch with.
Present (P): The weight of the current community is set to 1, and the weights of all other communities are set to 0. With this strategy, users always behave as members of the community they are currently in, and do not favour any other community.
Uniform Social (US): All the communities the user gets in touch with are given equal weight. Therefore, w_i is equal to 1 for the home and the acquainted communities.

Clearly, it would be possible to consider a number of additional strategies, by modifying the definition of the weights. The strategies we have selected are representative of several *reference* behaviours. In the US policy all communities a user is in touch with are given the same importance. In MFV the importance of a community is proportional to the time spent by the user in the community. P and MLN can be seen as opposite extreme customisations of MFV. In P only the current community is given importance. This is a MFV customisation which looks exclusively at the current "role" of the user, as a member of the current community, without any "look-ahead" behaviour. In MLN the current community is not given any importance, and only future communities are considered. MLN is thus a customisation of MFV with an extreme "look-akead" behaviour. Finally, F is an intermediate policy between MFV and MLN. It keeps the "look-ahead" behaviour of MLN, because it does not consider the current community. However, as in MLN, it considers *all* the other communities the user is in touch with, and not only the most likely one for the next visit.

Finally, for comparison purposes, we consider two non-social policies as well, and specifically a Greedy and a Uniform policy. In the Greedy policy all weights but the one of the local user are set to 0. In the Uniform policy all the weights are equal. As a minor, but important, remark, note that in this case a lot of data objects ends up having the same utility. In this case, a node breaks ties by choosing data objects according to a uniform distribution.

4.2. Social parameter estimation

The ContentPlace social-oriented policies require online, dynamic estimation of the utility components' parameters, as well as an estimation of the parameters required to compute the components' weights. In this section we describe how this can be achieved by avoiding any form of (controlled) flooding, such as that implemented by Epidemic Routing [20], which easily saturates networking resources [14]. Clearly, this choice calls for a trade-off between estimation accuracy and network overhead.

A necessary pre-requisite to estimate the parameters is to detect the communities in the network, and to enable nodes to understand in which community they are currently roaming. Fortunately, there are promising results about autonomic community detection in opportunistic networks ContentPlace can rely on, such as [11,13]. These mechanisms allow nodes of an opportunistic network to (i) be aware of the communities they belong to and have acquaintance with and (ii) be aware of the community in which they are currently in at any given point in time. We assume that one of these mechanisms is in operation.

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001

4.2.1. Estimation of the social weights

The communities' weights defined in Section 4 can be easily computed thanks to the community detection features and, in some cases, by measuring the time spent by nodes in the different communities and monitoring transitions between communities. Specifically, the strategies P, US and MLN just require community detection. This trivially holds true for P and US. MLN requires an estimate of the conditional probability of moving to future communities, starting from a given current community, i.e., $P(c_i|CC)$, where CC is the current community. This is equivalent to estimating the transition probabilities of a Markov process whose states are the different communities the user can visit. These probabilities can be estimated online, by monitoring the transitions between communities during the user's movements. Finally, MFV and F can be implemented by measuring the time spent by the user in each community. In MFV, the weight ω_i is equal to $t_i/\sum_i t_i$, where t_i is the time spent by the user in community i. In F, the weights are defined as in MFV, except for the weight of the current community, which is set to 0.

4.2.2. Estimation of the utility parameters

Gathering context information to compute utility components requires some more steps. According to Eqs. (2) and (3) computing utility components for a given data object a node requires the size of the object (s), and estimates of the access probability (p_{ac}) and the availability of that object (p_{av}) in all the individual communities. Theoretically, to achieve the maximum precision of parameter estimation, nodes should advertise all information for all data objects they become aware of, and for all communities they happen to visit. Clearly, this would result in a very detailed computation of utility values, but in a huge networking overhead. Instead, we choose to exploit nodes' movements to save on network overhead, which is one of the basic principles of opportunistic networks. Basically, when two nodes meet they exchange a summary of data objects in their caches. From these snapshots, each node is able to compute, for each object, a fresh sample of the local availability $\hat{p}_{av,l}$, as the fraction of time during which the object has been seen on neighbours caches. This newly computed value is then used to update $p_{av,l}$ according to a standard smoothed average $p_{av,l} \leftarrow \alpha p_{av,l} + (1-\alpha)\hat{p}_{av,l}$. For what concerns $p_{ac,l}$, we assume that $p_{ac,l}$ is equal to 1 for those objects the user is interested in, and equal to 0 for the others.[3]

The estimation of the utility parameters for a generic community i, different from the local one, stems from the fact that,in ContentPlace, together with the summary vector describing its cache, each node also advertises the set of data objects *its local user* is interested in, and an estimate of the availability of the object for *its* local user, i.e., $p_{av,l}$. Then, every time period T, each node computes a fresh sample for $p_{ac,i}$ and $p_{av,i}$ as the arithmetic mean of the values advertised by the neighbours during T. Then these values are used to update $p_{ac,i}$ and $p_{av,i}$, using the standard smoothed average as seen above. For a more detailed explanation see [5].

Finally, note that the size s of a data object (for which a utility value is required) is easily derived from the summary vectors advertised by the neighbours.

[3] A more precise estimation of the access probability of the local user to a data object would require a refined model of the user behaviour as far as data access is concerned. However, this is an orthogonal problem with respect to the ContentPlace algorithms, and therefore we choose this simplified representation of users' access pattern.

5. Performance evaluation

In this section, we evaluate the performance of the social-oriented policies described in Section 4. To this aim, we developed a custom simulator that is extended from the one in [6] and uses the same assumptions for lower communication layers. As we discuss in the following, the simulation scenario we consider has been chosen because it is able to highlight *general* features of the social-oriented policies. Note that in [5] we have already presented results showing the impact of system parameters such as the cache size and the number of nodes, which are not evaluated here. The main focus of this analysis is to understand the impact of the different social-oriented policies. We also include in the comparison the two non-social policies, i.e., the Greedy and the Uniform policies. Uniform has been identified as the best heuristic (from a number of standpoints) in [16]. Furthermore, Greedy and Uniform have been shown to achieve boundary performance result in [5]. Specifically, Uniform is the best possible policy in terms of fairness, while Greedy the best in terms of hit rate, *in a scenario with a single homogeneous community*.

We hereafter describe the default scenario for our simulations. The default scenario is composed of 45 nodes, divided into three communities, moving according to the HCMM model [3] in a 4×4 grid (1000 m wide). HCMM is a mobility model inspired by the Watt's caveman model, that has shown to reproduce statistical figures of real user movement patterns, such as inter-contact times and contact duration [3]. In HCMM, each group represents a social community, and nodes within the same group have social relationships among themselves. Also nodes belonging to different communities can have social relations: according to the rewiring probability (p_r), each link towards a friend is rewired to a node belonging to a different community. Social links in HCMM are used to drive movements: each node moves towards a given community with a probability proportional to the number of links he has towards the community. Thus, the rewiring parameter allows us to control the degree of interactions between nodes of different communities. In our simulations, each group is initially assigned to a cell (its home-cell) avoiding that two groups are physically adjacent (no edge contacts between groups) or in the same cell. This allows us to eliminate physical shortcuts between groups, which would bias the evaluation of the ability of policies to bring data objects from one community to the others. Therefore, nodes can exchange data only due to social mobility of nodes and not due to random colocation. The rest of HCMM parameters are set acconrding to Table 1. We consider as many channels as the number of groups (n_{gr}). Each group is the source for $1/n_{gr}$ of the objects belonging to each channel, i.e., $1/n_{gr}$

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001

of the objects of each channel are generated (at the beginning of the simulation) in each group. Note that a data object is *always* available from the node that generated it. To not interfere with the data dissemination performance figures, nodes generating data objects make them available through a separate buffer with respect to the cache. Therefore, for any node, the only way to obtain objects not generated in the local group is to get in touch directly (i.e., the node itself moves in a different group) or indirectly (i.e., one of the nodes of the local community goes out and then comes back, with the desired data object in its cache) with an external community. To have an integer number of objects generated in each group, we consider 99 data objects per channel. Each node can subscribe just to one channel. When not otherwise stated, nodes' interests are distributed according to a Zipf's law (with parameter 1) within each group. Unless otherwise stated, we consider three channels. The popularity of channels is rotated in each group, such that each channel is the most popular in one group, the second-most popular in another group, and the least popular in the last group. Cache space on a node can accommodate exactly all data objects belonging to an individual channel. An analysis with varying cache sizes has been presented in [5]. The cache size we choose is a good trade off between seeing the impact of dissemination policies, and avoiding data object disappearance due to low utility on too many caches.

Nodes' requests for data objects follow a Poisson process with parameter $\lambda = 200$ (on average three requests every 10 min for each node). Nodes can request data only for the channel they subscribe. Within the channel, the data object they request is select according to a uniform distribution. Requests are valid until the simulation ends. As we show in the following, the delay distribution highlights that our simulation length is long enough to reasonably approximate an infinite request validity timeout. As requests are buffered at issuing nodes only, and do not occupy cache space, having infinite validity does not impact the system's performance. Instead, it allows us to derive a complete analysis of the system's performance for increasing validity timeouts.

All policies are evaluated in terms of the quality of service (QoS) perceived by the users and the resource consumption. The QoS is measured in terms of the hit rate experienced by each channel (the evaluation of other QoS metrics can be found in [7]). The hit rate is given by the number of successful requests divided by the number of overall requests. Note that, unless otherwise stated, we show the hit rate with infinite timeout values. As not all policies reach 100% hit rate even in this case, this index allows us to show the fraction of requests that *cannot* be served by the policies. We then separately investigate the hit rate index as a function of the validity timeout. The latency in satisfying the requests has been computed as the difference between the time at which the request is satisfied and the time at which the request was generated. 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 data objects, request data objects and data objects themselves. Simulations run for 50,000 s. Exchanges of data objects upon nodes' contacts start after an initial transitory required by parameter estimators to reach the steady state. Results shown in the following section have a 95% confidence interval, obtained through standard independent replication techniques.

Table 1
Configuration parameters.

Node speed	Uniform in $[1, 1.86]$ m/s
Transmission range	20 m
Sampling period	5 s
Cost function parameter (λ)	15
Smoothed average parameter (α)	0.9

5.1. Analysis with uniform rewiring

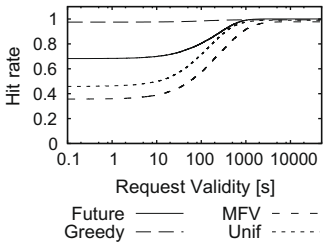
Our first experiment is based on a configuration in which the rewiring probability is the same for all nodes, and equal to 5%. This means that the average number of social links across communities is the same for all the three communities. Although the probability of rewiring is not particularly high, it is already sufficient to mix communities enough to let data objects circulate across all communities, independently of the data dissemination policy. Indeed results in Fig. 3a and b show that all policies (Future and MLN are overlapped, just as MFV, Present and Uniform Social) achieve 100% hit rate with an infinite timeout. Furthermore, results do not change depending on the channel to which the travellers are subscribed.

If we reduce the timeout of the request, we realize that the QoS provided by different policies may vary with the specific configuration. In fact, Fig. 3a and b shows that data objects are delivered faster by the Greedy policy, and more slowly by the Uniform and MFV policies. Thus, when reducing their timeout, many requests will not be fulfilled under these policy, resulting in a decrease in the hit rate. We refer to Section 5.3 for a more detailed explanation of the behaviour of each policy.

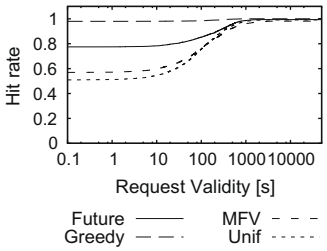
Table 2 shows that policies are not equivalent, even when they provide the same hit rate. In fact, with respect to the bandwidth overhead, the Uniform policy uses triple the bandwidth that MFV, Present and Uniform Social need, and around 20 times more than Future and MLN. The most significant difference is between Greedy and Uniform, where the increase in the data transmitted is around 3500%. In the next section we will see in more details the reason for this behaviour. Here we only anticipate the fact that when not exploiting context information, the bandwidth consumption may become huge. Another implication of Table 2 is that a selfish policy (Greedy) can pay when the contacts between nodes are numerous and uniformly distributed. In fact, when nodes can freely move from one community to another and meet many other nodes, the likelihood of finding another node with the same interests is high, and therefore the data objects can easily be found by their interested nodes.

Summarizing, when nodes are highly mixed, data objects spread naturally across the network, regardless of the policy used. However, policies differ in their resource

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001



(a) Requests on ch. 1



(b) Requests on ch. 3

Fig. 3. Hit rate with varying timeouts – rewiring 0.05 – travellers subscribed to channel 1.

consumption and in the speed of the resulting dissemination process. In a setting with high node mobility, the Greedy policy guarantees the lowest bandwidth overhead and the quickest delivery.

5.2. Analysis in the default scenario

Based on the results in Section 5.1, here we consider a less mixed mobility pattern, as follows. Each pair of communities is connected through just one node. Specifically Community 1 (C1) has two nodes ("travellers") with relationships outside C1, one with Community 2 (C2), the other with Community 3 (C3). Hereafter, we show the performance figures related to nodes in C1. The same remarks can be made for nodes in the other communities, as well. Note that, from Eq. (2), it is clear that the channel the traveller nodes are subscribed to (which affects the u_l component) impacts the data dissemination process. For this reason, we replicate the experiments by varying the channel the travellers are subscribed to.

Fig. 4 shows the hit rate for C1, when travellers are subscribed to channel 1. Specifically, the group of boxes related to channel *i* shows the hit rate achieved by nodes whose home community is C1 and are subscribed to channel *i*. Experiments with travellers subscribed to the other channels provide similar results for all policies but the Greedy, that achieves 100% hit rate only for the channel the travellers are subscribed to. Recall that these plots are related to an infinite validity timeout. When shorter timeouts are used, the misbehaviours of Present, MFV and Uniform Social we discuss below become more serious. Also, the performance difference between MLN and Future (which already in this case appear as the best policies) and the other policies increases. We will analyse these aspects in detail in Section 5.3. As a preliminary remark note that, since data objects are always available at generating nodes, a hit rate of 33.33% is always guaranteed. We can identify four different behaviours. With the Greedy policy, travellers store only data objects in which they are directly interested. This results in a 100% hit rate on the subscribed channel and a 33.33% hit rate on the other channels (there is hit only on the 1/3 of data generated locally). As expected, the Greedy policy is able to improve the hit rate of the nodes interested in the same objects the travellers are interested, and is not able to disseminate other objects. With the Uniform policy, the hit rate of all channels improves and reaches about 80%. The set of social policies MFV, Present and Uniform Social all show a similar behaviour: they tend to be slightly unfavourable towards the most popular channel. This is due to the fact that nodes within the same group tend to *synchronise* their behaviour: when nodes realise that certain objects are poorly available, they all fetch them as soon as they become available. This results in the objects becoming highly available, and being dropped simultaneously by all nodes. A detailed analysis of the logs confirms this behaviour. In [4] we have also shown that the effect of this synchronisation get worse the more channels are popular. Finally, MLN and Future policies have a very good hit rate in all cases. The key of these policies is that nodes do not consider the community in which they currently roam, but just future communities. It is easy to see that this results in *static* nodes (i.e., nodes without social relationships outside their home community) behaving greedily, and in travellers working to help communities they will visit in the future. This clearly avoids the problems related to the synchronisation effect of the other policies. The results for the other communities (not shown here) highlight that these policies guarantee the same hit rates shown in Fig. 4. This is quite important. For example it shows that nodes subscribed to channel 3 in C2 are able to get not only the locally generated objects, and not only the objects carried generated

Table 2
Resource consumption – rewiring 0.05 – travellers subscribed to channel 1.

Bandwidth overhead [MB]			
Greedy	175.45 ± 1.77	MFV	21583.80 ± 1112.72
Future	2589.45 ± 533.18	UnifSoc	21124.10 ± 1106.01
MLN	3412.71 ± 352.65	Unif	62306.43 ± 3828.69
Present	22796.61 ± 1447.52		

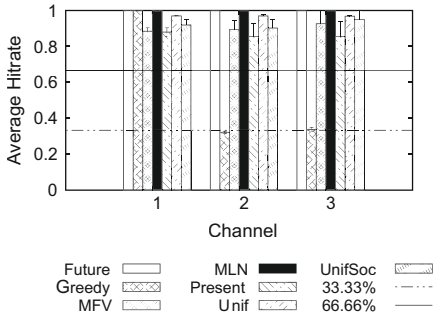


Fig. 4. Hit rate – travellers subscribed to channel 1.

in C1 (directly carried by the traveller of C1 visiting C2), but even those objects generated in C3 that are firstly brought in C1 by the traveller of C1 visiting C3. The identification of these "social" paths is something that is peculiar to the ContentPlace MLN and Future policies. We can anticipate that this type of "collective" social behaviour, in which just travellers adopt a social-oriented strategy turns out to be the best solution. More plots confirming the above remarks can be found in [4].

Table 3 shows the bandwidth overhead for each protocol. The Uniform policy, although closely approximating MLN and Future in terms of hit rate and fairness, significantly overuses network resources. This because nodes continuously exchange data objects as a consequence of the tie breaking policy (see Section 4.1). It is easy to show that this is the only way for Uniform to make objects circulate. Furthermore, MFV, Present and Uniform Social pay for their synchronisation problem also with regard to the bandwidth overhead. Instead, MLN and Future have a very good performance also with respect to network overhead, while the excellent performance of the Greedy policy in terms of overhead is paid from hit rate standpoint.

Finally, Fig. 5 shows the CCDF of the delay for *satisfied* requests on channel 1, when the travellers are subscribed to channel 1. For the sake of readability, we only show the Greedy, Uniform, Future and MFV policies. The Present and Uniform Social policies are basically equivalent to MFV, while Future and MLN are overlapped. We show this plot only, as plots for the other cases are qualitatively similar. Clearly, Greedy achieves the best performance in this case. The social-oriented policies other than MLN and Future suffer quite high delay. This is a side effect of the unwanted synchronisation issue that we have discussed above. Requests for data objects that have disappeared need a long time to be satisfied. MFV and Future clearly outperform Uniform also from this standpoint. Note that the maximum delay is in the order of 20,000 s. This confirms that simulation runs of 50,000 s are reasonably long to consider the request timeouts as infinite.

In this section we have presented the policies' behaviours with respect to all performance figures. In the following, we concentrate on the hit rate index only to highlight the policies' different behaviour in different scenarios. The comparison with respect to the other performance figures is qualitatively similar to the one presented in this section.

5.3. Reduced requests' validity timeouts

In this section we analyse the hit rate index as a function of the requests' timeout. Fig. 6 shows the hit rate with increasing requests' timeout for requests on channel 3, the least popular channel. Again, we only present results related to the Greedy, MFV, Uniform and Future policies. The plot is related to the case when travellers are subscribed to channel 1. Note that the hit rate in the Greedy policy do not change with the validity timeout. Recall that requests start after an initial transient in which the policies reach stability. Thus, the Greedy policy has already moved all the data objects it is able to move across communities. Uniform and the social policies increase the hit rate when the validity timeout increases. The dynamic is slower with respect to the Greedy policy, but they are definitely able to provide higher performance to channels travellers are not subscribed to (see Fig. 6). In that case, as expected, the Greedy policy is not able to bring any new data objects in addition to those already generated in the community. Finally, again the MLN and Future policies are confirmed to be the best policies among the one investigated, for all the validity timeouts.

Table 3
Resource consumption – travellers subscribed to channel 1.

Bandwidth overhead [MB]			
Greedy	68.89 ± 0.28	MFV	16497.79 ± 660.57
Future	508.32 ± 21.84	UnifSoc	16749.46 ± 430.16
MLN	509.05 ± 21.61	Unif	71974.40 ± 1316.68
Present	15500.51 ± 881.80		

5.4. Increased number of nodes

In [4] we have shown that an increased number of travellers improves the dissemination of data objects because it results in a more mixed scenario. In this section we want to evaluate the impact of an increased number of nodes that are not travellers. While the travellers have an active role by making possible the spreading of objects from one community to another, the nodes that always roam within the same community are just content generators/consumers, and their contribution to the dissemination process is very limited. In this section we double the number of nodes in each community, going from 15 nodes to 30 nodes per community, and again we assign nodes' interests according to a Zipf's law with parameter $\alpha = 1$.

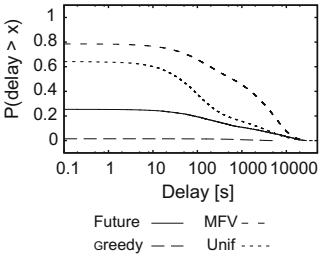


Fig. 5. Delay CCDF for requests on channel 1 – travellers subscribed to channel 1.

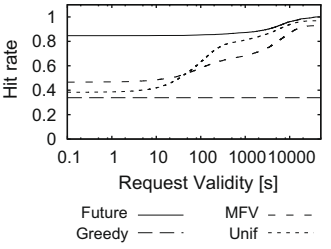


Fig. 6. Hit rate with varying timeouts – requests on channel 3 – travellers subscribed to channel 1.

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001

In terms of the hit rate, Fig. 7 shows that the behaviour of the Greedy, Uniform, Future and MLN policy is approximately the same as in the case of 45 nodes (Fig. 4). With many nodes, however, the performance of the social policies that suffer from synchronisation problems worsens. The reason is that, with more nodes per community, the synchronisation effect is stronger than before, and this results in a decrease in the hit rate. Varying the request timeout, we have found that the hit rate shows the same trend as in Fig. 6. Also, the delay CCDF has the same pattern as in Fig. 5. Please refer to [7] for more details. Therefore, from the QoS standpoint, increasing the number of nodes that are not travellers has only the effect of boosting the synchronisation problems of the MFV, Present and Uniform Social policies.

In order to complete the analysis, in Table 4 we also give the results for the resource consumption. The most interesting finding here is that under the Uniform policy the bandwidth overhead with 90 nodes is about four times the overhead in the case of 45 nodes (Table 3). This means that the Uniform policy does not scale well with the number of nodes in the network, and this can be a problem in all cases in which resource usage is a concern. Instead, for all other policies there is a roughly linear dependence between the increase in the number of nodes and the resource consumption, thus implying a more judicious use of the network resources.

Considering both the QoS and the resource consumption, also in the case of an increased number of nodes per community, the best trade-off is provided by the Future and MLN social policies.

5.5. Multi-hop social paths

The results presented so far show that in the considered scenarios MLN and Future are the best policies, and perform almost the same. However, there are cases in which they behave differently. Specifically, both MLN and Future fetch data objects by estimating social paths of travellers. Within any community, MLN just takes into consideration the *next* hop only, i.e., the next community it will visit. Future takes into consideration *all* the communities it is likely to visit in the near future, weighted by the probability of visiting each. Therefore, MLN might miss exploiting "multi-hop" social paths across multiple communities. To investigate this effect, we consider a scenario with one traveller only, belonging to C1, subscribed to channel 1. It can visit either C2 or C3 with equal probability when in C1, while it always gets back to C1 after having been in C2 or C3. Furthermore, all nodes of C1 (C2, C3) are subscribed to channel 1 (2, 3). In this case, it is expected that MLN is not able to completely serve communities C2 and C3. While in C2 or C3, it will consider only the interests of users in C1, and thus it will only bring back data objects of channel 1. It will bring to C2 (C3) only data objects originated in C1 for channel 2 (3). On the other hand, Future is expected to provide 100% hit rate to all communities. This behaviour is totally confirmed by the simulation results in Figs. 8 and 9, that show the hit rate as a function of the request validity timeout. As expected, both policies are fast in achieving the maximum hit rate. However, MLN can only serve 66.66% of the requests for users in C3, as it can only move the data objects of channel 3 generated in C1. Results similar to those in Fig. 9 hold for community C2, as well.

6. Average analysis of the Future policy

In this Section we perform an average analysis of the dissemination behaviour of the Future policy, that we have shown to be the best in Section 5. The scenario we consider is that of two communities A and B, having N nodes each that roam locally in the community, with a traveller node commuting between A and B. We consider the following situation: (i) an initial set of data has been already spread among the nodes of the two communities and its dissemination process has ended, leaving the system in a stationary state and (ii) at time $t = 0$ a new data object is generated for a generic channel ch_{new} within community B (given the symmetry of the topology, the choice of the generating community does not affect the analysis). The focus of our investigation is the time it takes for the traveller to deliver this new object to the interested nodes in community A (the underlying assumption is that there is at least one node in community A interested in the new

Table 4
Resource consumption with 90 nodes – travellers subscribed to channel 1.

Bandwidth overhead [MB]			
Greedy	181.86 ± 0.88	MFV	36758.21 ± 2937.98
Future	1114.23 ± 46.15	UnifSoc	36574.81 ± 3322.24
MLN	1132.90 ± 65.11	Unif	321973.33 ± 4324.55
Present	31836.81 ± 2457.00		

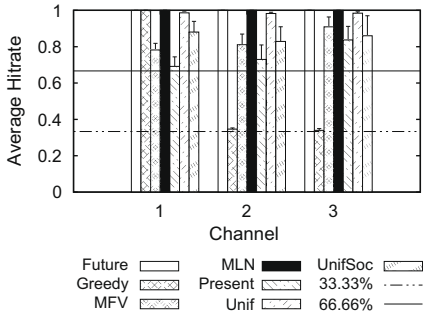


Fig. 7. Hit rate with 90 nodes – travellers subscribed to channel 1.

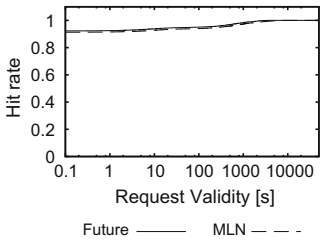


Fig. 8. Hit rate for C1.

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001

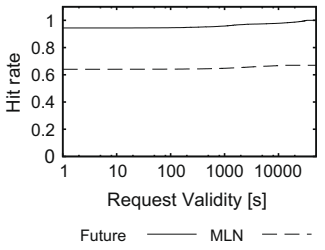


Fig. 9. Hit rate for C3.

data). We have chosen to restrict the analysis to the case of two communities, because this configuration represents the building block of every more complex scenario. In fact, the traveller's movements between multiple communities are nothing but a composition of its movement patterns between pairs of communities.

In this analysis, we assume that nodes' interest in the available channels follows a Zipf's distribution and that the ranking of the popularity of channels is the same in all communities. Then the probability that a generic node is interested in channel j (ranked jth according to its popularity) is given by $Pop(j) = \frac{1/j^{\alpha}}{\sum_{i=1}^{C} 1/i^{\alpha}}$, where α is the parameter of Zipf's distribution and C is the number of available channels. In this paper we consider $\alpha = 1$, thus we have

$$Pop(j) = \frac{1}{jH_C}, \tag{4}$$

where H_C is Cth harmonic number. In addition, each node can be interested in just one channel and this interest implies that the node wants to receive a copy of each new data object of that channel as soon as possible, without the user issuing a specific request. For this reason, the terms "interested nodes" and "destinations" will hereafter be used interchangeably. In order to avoid the superposition of different effects, we assume that the traveller has no direct interest in any available data. We also assume that data objects never disappear. Given that, according to the Future policy, all nodes except for the traveller are greedy, it is reasonable to consider this condition verified when the nodes' buffers are not too small with respect to the number of data objects for their interested channel. We assume that the dissemination process stops when a given percentage of interested nodes in the same community has been reached by the data object. This percentage is referred to as replication factor. For data objects belonging to channel c, the replication factor $f(c)$ might range from $1/N_c$ and 1. In our analysis we consider $f(c) = \frac{1}{N_c}$, i.e., each object is stored by one node only. This is the worst case for the time required by traveller to get in touch with a copy of the data object.

Throughout this section we consider a simplified version of the utility function defined in Section 3.2.2. Instead of including in the availability the notion of redundancy (e.g., the availability for a data object with two copies in a community is higher than that for a data object with one copy), the availability used hereafter only reflects the presence or not of a data object. Therefore the availability is equal to 1 for data objects seen by the node, 0 otherwise.

For what concerns the mobility of nodes, we assume that inter-contact times between nodes roaming in the same community are exponentially distributed with rate λ. This assumption is backed up by the fact that, in HCMM, nodes, when roaming within a community, move according to the Random Waypoint mobility model, whose inter-contact times have been shown to follow an exponential distribution [10]. This consideration holds true also for the traveller because it roams in either community A or B.

The dissemination process for our social-aware policies is heavily influenced by the accuracy of the statistics (access probability and availability) used by the travellers to make retrieval decisions. For example, in Section 5 the dissemination of data objects starts after a transitory during which the traveller has constructed the statistics for the access probability and availability. This is the best case for our social-aware policies, which are as good as the statistics that the traveller nodes are able to collect. Therefore, our analysis is split into two cases. We first consider the case in which the statistics used by the travellers are congruent with the state of the system. The second part of our analysis focuses on the functioning of the system when the social information is not yet available or is not precise. We refer to the latter part as *transient analysis*, because we target the transitory phase of collecting statistics, while the former is denoted as *stationary analysis*.

6.1. Stationary regime of context information

At time $t = 0$, when the available data objects have already been disseminated to the interested nodes and the context information on the traveller is up-to-date, the system is in a situation where each data object has utility equal to zero. In fact, the data objects being available in both communities, their availability (as seen by the traveller) is at the maximum level. If at time $t = 0$ a new data object is generated, this data object will be the only one with utility different from 0. This means that when the traveller gets in touch with the data object, it will surely get a copy of it, this data object being the one with the highest utility. Therefore, when the context information on the traveller is correct, the delay for the new data object to reach its destinations depends only on the physical time for the traveller to get in touch with the data object and to get back to community B: i.e., the delay is only a function of the mobility characteristics of the system.

We define $E[D]$ as the expected delay from the generation of the new data object in community B to its delivery to its destinations in community A. $E[D]$ can be separated in its different components as follows:

$$E[D] = E[D_{fetch}] + E[D_{local}]. \tag{5}$$

$E[D_{fetch}]$ is the average time required for the data object to enter the community A for the first time. $E[D_{local}]$ is the expected time for the interested nodes in community A to be reached by a copy of the data object. As we have assumed that a data object is not further disseminated when its replication level is reached, with $f(c) = \frac{1}{N_c}$ the spreading in the local community stops after the first interested node is reached. Therefore, in this case, $E[D_{local}]$ is equal to the ex-

pected time for the traveller to deliver a copy of the data object to the first interested node in community A.

When the new data object appears in community B at time $t = 0$, the travel can be either in community A or in community B, and its location impacts $E[D_{fetch}]$. Applying the law of total probability we can write $E[D_{fetch}]$ as

$$E[D_{fetch}] = E[D_{fetch}|S_T = C_A] \cdot P(S_T = C_A) + E[D_{fetch}|S_T = C_B] \cdot P(S_T = C_B), \quad (6)$$

where S_T is the community in which the traveller T is roaming when the data object is generated. The probability of finding the traveller in either community depends on the mobility model in use and is equal to the stationary distribution (if any) of the location of nodes. For the HCMM mobility model used in Section 5, these probabilities can easily be found [3]. Intuitively, the fact that the traveller commutes between the two communities suggests that the overall expected delay is a composition of multiples of the expected roaming time in these communities, and that it depends on how many round trips are required on average to retrieve and deliver the data object. The fact that the traveller can be in either community when the new data object is generated impacts the delay for the first round trip. We define the duration of a normal round trip (T_{cycle}) as the time it takes for a traveller to come back to community A, given that it has just entered that community. T_{cycle} is the sum of the expected roaming time in two communities plus the expected time to travel from one community to the other.

$$T_{cycle} = E[T_A] + 2E[T_{tr}] + E[T_B]. \quad (7)$$

When the data object is generated, the traveller can have been in either A or B for an unspecified amount of time and not having just entered that community. Therefore, for the first round trip we have to take into consideration the expected *remaining* roaming time in the community in which the traveller is roaming at $t = 0$, instead of the expected roaming time itself. However, if we assume that the roaming periods follow an exponential distribution, then the expected remaining roaming times are equal to the expected roaming periods, according to the PASTA principle. Thus, we have that $E[T_A|S_T = A] = E[T_A]$ and $E[T_B|S_T = B] = E[T_B]$. The only difference between the traveller starting from community A or community B is that, when the traveller starts from community B, it has a chance of getting in touch with the new data object immediately, without having to wait for the next round trip. This implies that in this case the contribution to the expected delay is shorter than when the traveller is roaming in A at $t = 0$. Finally, in both cases, once the traveller enters community A with the data object, the expected delay to reach the first destination is given by $E[D_{local}]$. After putting all these considerations together, we obtain for the first round trip:

$$T'_{cycle} = E[T_A] + 2E[T_{tr}] + E[T_B], \quad (8)$$

$$T'_{cycle\text{-}short} = E[T_{tr}] + E[T_B]. \quad (9)$$

T_{cycle}, T'_{cycle} and $T'_{cycle\text{-}short}$ only depend on the mobility of the traveller. Again applying the law of total expectation to Eq. (6), we have:

$$E[D_{fetch}|S_T = C_A] = T'_{cycle} + \sum_{i=1}^{\infty} T_{cycle}(i-1) \cdot P(Y = i), \quad (10)$$

$$E[D_{fetch}|S_T = C_B] = T'_{cycle\text{-}short} + \sum_{i=1}^{\infty} T_{cycle}(i-1) \cdot P(Y = i), \quad (11)$$

where Y is a random variable representing the probability that the traveller fetches the data object at the ith round (i.e., that the travel meets one of the nodes currently having a copy of the data object in community B).

If the number of nodes currently having a copy of the data object in community B is j, then the distribution of the delay until the first one is reached is the minimum over a set of j exponentially distributed random variables, all having the same rate λ. The minimum of these variables follows an exponential distribution with rate equal to $j\lambda$:

$$U_j \sim \exp(j\lambda). \quad (12)$$

Then, the probability that the traveller sees the data object during a generic round i is equal to the probability of meeting one of the nodes currently having a copy of the data object before moving back to community A, i.e., within the roaming period T_B. As in our analysis there is just one node in B having a copy of the new data object, the probability that the traveller gets in touch with that node within a roaming period with distribution T_B is $P(U_1 < T_B)$. While we know from Eq. (12) that U_1 follows an exponential distribution with rate λ, the distribution of T_B depends on the mobility model in use. If, for the sake of tractability, we assume that T_B follows an exponential distribution with rate $\frac{1}{E[T_B]}$, using standard probability theory we obtain that $P(U_1 < T_B) = \frac{\lambda E[T_B]}{\lambda E[T_B]+1}$. Each roaming period in community B can be seen as a Bernoulli trial with success probability $p_B = P(U_1 < T_B) = \frac{\lambda E[T_B]}{\lambda E[T_B]+1}$. Thus, the probability that the traveller gets the data object at exactly the ith roaming period follows a geometric distribution:

$$P(Y = i) = \left(\frac{1}{\lambda E[T_B] + 1}\right)^{i-1} \frac{\lambda E[T_B]}{\lambda E[T_B] + 1}. \quad (13)$$

If we rewrite Eq. (10) using Eq. (13), after routine manipulation we obtain:

$$E[D_{fetch}|S_T = C_A] = T_{cycle} \frac{1}{p_B}, \quad (14)$$

where the last equality follows from the exponential assumption for the roaming times. Following the same line of reasoning, we also have:

$$E[D_{fetch}|S_T = C_B] = T'_{cycle\text{-}short} + T_{cycle} \frac{1 - p_B}{p_B}. \quad (15)$$

Now we move to computing $E[D_{local}]$. Once the traveller has fetched the data object, the delay of the delivery to the first destination met depends on how long the traveller roams in community A. If during this roaming interval no destination is met, then we have to wait an entire cycle for the traveller to be back in that community again, and so on. If the data object is delivered at the ith round, then $E[D_{local}]$ is given by the delay up to the $(i-1)$th round, plus the time required to reach the destination during the ith round. Therefore, $E[D_{local}]$ can be written as $E[D_{local}] = E[U_{N_{ch_{new}}}] + \sum_{i=1}^{\infty}(i-1)T_{cycle} \cdot (1-p_A)^{i-1}p_A$, where p_A is the

probability of meeting one of the interested nodes during the roaming period. Assuming that the number of nodes of community A interested in channel ch_{new} is $N_{ch_{new}}$, $p_A = P(U_{N_{ch_{new}}} < T_A)$. From Eq. (12) we know that $U_{N_{ch_{new}}} \sim Exp(N_{ch_{new}}\lambda)$. Thus, $E[U_{N_{ch_{new}}}]$ is equal to $\frac{1}{N_{ch_{new}}\lambda}$ and $p_A = P(U_{N_{ch_{new}}} < T_A)$ is equal to $\frac{N_{ch_{new}}\lambda E[T_A]}{N_{ch_{new}}\lambda E[T_A]+1}$. Assuming independent and identically distributed node encounters, the average number of nodes interested in channel ch_{new} is given by $N_j^n = Pop(j) * N^n$. After standard manipulation we get:

$$E[D_{local}] = \frac{T_{cycle} + E[T_A]}{E[T_A]NPop(ch_{new})}. \quad (16)$$

Now that we have the expressions for $E[D_{fetch}]$ and $E[D_{local}]$, the formula for $E[D]$ follows from simple substitutions in Eq. (5).

In the remainder of this section, we give a brief overview of the dependency of the delay on the characteristics of the mobility and on the popularity of the channel. Fig. 10 shows $E[D]$ against the popularity of the channel to which the new data object belongs. The channel for the new data object is selected among 100 channels. Small, medium and large values of λ correspond, respectively, to 0.0001, 0.001 and 0.01. These settings will be used throughout Section 6. In Fig. 10 the delay experienced by the new data object decreases linearly with increasing popularity. The intuitive reason behind this behaviour is that the less popular a data object is, the more difficult is for the traveller to find an interested node to which deliver the object. The second result from Fig. 10 is that this trend worsens when the frequency of contacts is smaller. Figs. 11 and 12 show the separated contributions to the overall delay with an intermediate value of λ and varying average roaming periods. In Fig. 11, $E[T_A]$ is 10 times smaller than $E[T_B]$ and $E[T_{tr}]$, while the opposite is true for Fig. 12. When the roaming time in community B is long, it is very easy for the traveller to get the data object, and the overall delay is strictly dependent on $E[D_{local}]$. This effect is mitigated in Fig. 12.

6.2. Transient regime of context information

The case in which at the time the new data object is generated, the traveller has not yet statistics on the popularity and availability of the data is the worst case for our social-aware dissemination policies. In fact, during the transitory period when the traveller builds these statistics from scratch, the precision of the collected social information might be unreliable and this may lead to wrong retrieval decisions by the traveller, resulting in an increased delay experienced by the data objects. To characterize the delay in the transient phase, during which the traveller collects the information, we focus separately on two aspects: (i) how the traveller gets in touch with the other nodes and (ii) how the traveller discovers the data objects.

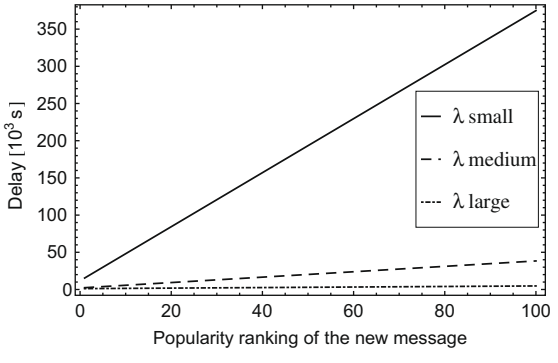


Fig. 10. Expected delay.

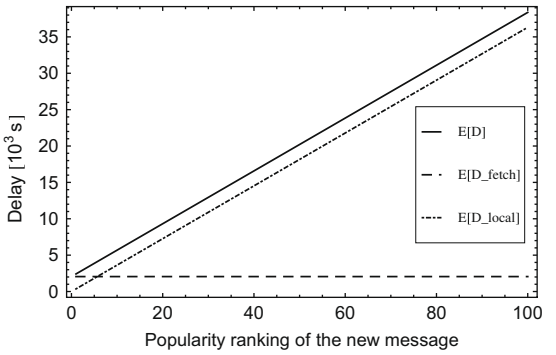


Fig. 11. Expected delay with $E[T_B] = E[T_{tr}] = 1/10E[T_A]$.

6.2.1. Meetings between the traveller and the other nodes

The statistics on the interests and on the distribution of data objects for the nodes of a community are collected by the traveller while it roams in that community. If the expected roaming time in a community A is $E[T_A]$ and the inter-contact time between the traveller and each other node is exponentially distributed with rate λ, the expected number of nodes met after the first roaming period (N^1) is equal to $N(1 - e^{-\lambda E[T_A]})$, being $(1 - e^{-\lambda E[T_A]})$ the probability of seeing a node during the roaming period. In general, after the nth round, the number of nodes met is $N^n = N(1 - e^{-\lambda nE[T_A]})$. Usually the traveller does not meet all nodes in the first roaming period, but multiple rounds are required. Fig. 13 shows the time required to meet all nodes[4] under different mobility conditions. When the nodes get in touch rarely (small λ), it takes some time for the traveller to meet all the nodes of a community, while, when the frequency of meetings is higher, the time required is much lower. It is also interesting to note that this time does not depend significantly on the number of nodes in the community.

Among the N^n nodes met after the nth round, the average number of nodes interested in channel j is $N_j^n = Pop(j) * N^n$, assuming independent and identically distributed node encounters. As the access probability p_{ac} to channel j for a generic community is equal to the proportion of nodes of that community interested in channel j, the access probability for channel j as seen by the traveller after round n is

[4] Given that the exponential function has an asymptote towards infinity, we consider the process completed when $N - \epsilon$ nodes are reached, with ϵ very small.

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001

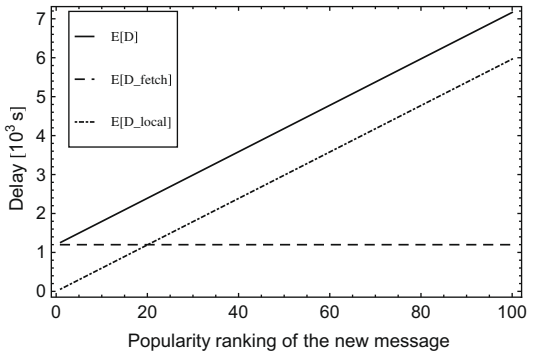


Fig. 12. Expected delay with $E[T_B] = E[T_{tr}] = 10E[T_A]$.

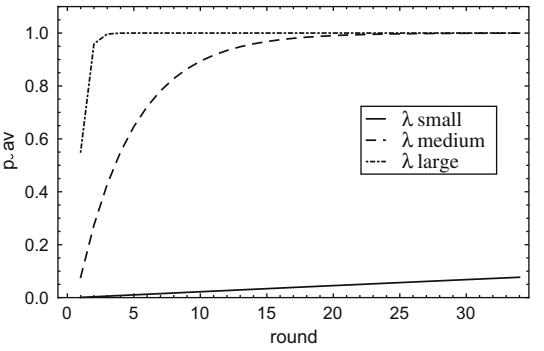


Fig. 14. Availability statistics as a function of the time.

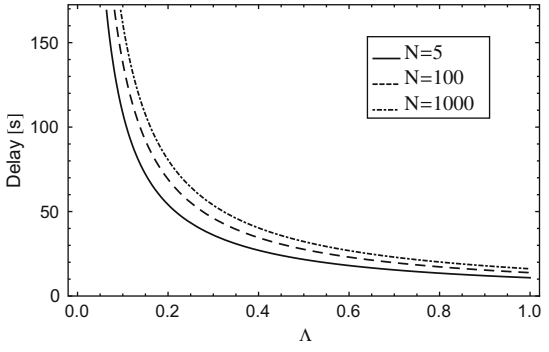


Fig. 13. Time required to meet all nodes.

$$p^n_{ac_j} = N^n_j / N^n = Pop(j). \tag{17}$$

Thus, on average, the statistics on the access probability give a precise estimate of the interests of the nodes. This is because, being all nodes equally likely to meet, the distribution of the interests on the subset of encountered nodes is equal to the distribution of interests themselves.

As explained in Section 6, we consider a simplified version of the availability of a data object. This availability can be either 0, if the data object has been seen by the traveller, or 1 otherwise. In our scenario, all data objects are available in both communities *A* and *B*, except for the new data object. Therefore, the availability of these data objects should be 1. However, during each roaming period the traveller meets only a subset of the nodes of the community. This implies that the statistics that the traveller collects are not exact until all nodes have been met. We want to compute the error on the estimation of the availability of the data objects available in the system before $t = 0$. Given a replication factor $f(j) = \frac{1}{N_j}$ for channel j, when the traveller meets N^n_j nodes having data objects for channel j, the probability that a given data object is seen is equal to $f(j)N^n_j$. The complementary of this quantity gives the probability of not having seen an object yet, i.e., of having a wrong statistic for that data object. After trivial substitutions, we found the following expression for the error on the availability after round nth:

$$e^n_{av_j} = e^{-\lambda n E[T_A]}. \tag{18}$$

Here the independence from the particular channel considered results from the fact that each data object is replicated on just one node (and therefore all data objects have the same probability of being found). In Fig. 14, the availability estimated by the traveller is plotted as a function of time (discretized in rounds) for different mobility configurations. When the frequency of meetings between nodes is high, the statistics converge fast. The opposite holds true for small values of λ, when it takes some rounds for the availability to become up-to-date.

6.2.2. Data object discovery by the traveller

Now that we have studied the evolution of the access probability and availability statistics, we can focus on how a given community is served by the traveller when it is outside. In particular, we need to find how data objects are discovered by the traveller, and then fetched. When the traveller roams in community *B*, it discovers all the data objects carried by the nodes that it meets. As we are interested in the average behaviour of the traveller, we simplify the dissemination process in community *B* assuming that the traveller selects the data objects to be put in its buffer only once per roaming period, and that this selection is performed on all the data objects discovered on average during that period. If the average roaming period lasts for $E[T_B]$, during this time the traveller meets on average $N(1 - e^{-\lambda E[T_B]})$ nodes, of which a fraction $Pop(j)$ having data objects for channel j. By denoting with Q the number of data objects carried by each local nodes, as all these objects belong to the channel in which the node is interested, the average number of data objects seen during a roaming period for each channel j is

$$M_j = \frac{QN(1 - e^{-\lambda E[T_B]})}{jH_C}, \tag{19}$$

where we have used Eq. (4) for $Pop(j)$.

Out of these M_j data objects, on average, $W^n_j = e^n_{av_j} M_j$ are the data objects for which the traveller has wrong statistics. As the traveller has not seen these data objects yet, their availability is zero, although they are actually spread in the community. These data objects with wrong statistics are the ones that can slow down the retrieval of the new data object, because the traveller can erroneously think that some of these data objects are more important than

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001

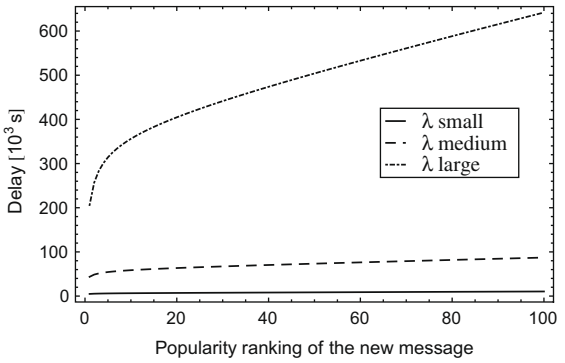


Fig. 15. Expected delay during the transient regime.

the new one. This is actually the case when the old data objects with wrong statistics belong to channels that are more popular than the channel to which the new data object belongs. In fact, in the average case, the new data object is fetched by the traveller as soon as the number of data objects with wrong statistics discovered during a round and more popular than the new data object is less than the total size K of the traveller's buffer. We assume that the new data object is considered less important than the already existing data objects for the same channel.[5] The number of rounds $n_{first}^{ch_{new}}$ for the traveller to fetch the new data object in the average case is thus:

$$n_{first}^{ch_{new}} = \min_{n} \left\{ \sum_{i=1}^{ch_{new}-1} e_{av_i}^{n} M_i < K \right\}. \tag{20}$$

Note that the inequality in Eq. (20) has always a solution. In fact, $e_{av_j}^{n}$ is always decreasing with n, as it represents the probability of not seeing each particular data object, and M_j does not vary with n. Therefore $\sum_{i=1}^{ch_{new}} e_{av_i}^{n} M_i$ is decreasing, and it goes to 0 for $n \to \infty$. This implies that we can find n by solving equation $\sum_{i=1}^{ch_{new}} e_{av_i}^{n} M_i = K$ for n, whose solution is:

$$n_{first}^{ch_{new}} = \frac{\ln\left(\frac{e^{E[T_B]\lambda}(e^{E[T_B]\lambda}-1)N(B)H_{ch_{new}}}{H_C}\right)}{E[T_A]\lambda}. \tag{21}$$

The time up to round $n_{first}^{ch_{new}}$ gives the expected delay for retrieving an object when the context information is not in its steady state. For computing this delay, we have to consider both the case of the traveller starting from community A and that of the traveller starting from community B. Following the same line of reasoning of Section 6.1 for the computation of $E[D]$, we obtain:

$$E[D_{fetch}|S_T = C_A] = T'_{cycle} + \left(n_{first}^{ch_{new}} - 1\right) T_{cycle}, \tag{22}$$

$$E[D_{fetch}|S_T = C_B] = T'_{cycle-short} + n_{first}^{ch_{new}} T_{cycle}. \tag{23}$$

Once the traveller has a copy of the data object, it will not drop it until the destination in community A has received the data object. As the expressions for $P(S_T = C_A)$, $P(S_T = C_B)$, and $E[D_{local}]$ are the same as in Section 6.1, we can easily compute $E[D]$.

We conclude our analysis discussing a plot for the delay experienced on average by the new data object during the transient regime. Fig. 15 shows $E[D]$ varying the popularity of the channel to which the new data object belongs. When the channel is very popular the delay is small because the data object is retrieved in the first few rounds or even in the first one (when λ is big). On the other end, when the data object belongs to an unpopular channel, the expected delay increases. The notable thing is, however, that the number of rounds required to get the new data object is *finite* for all channels. This is the main advantage of the Future policy with respect to the Greedy. In fact, the Greedy policy can result in lower delays for certain channels but in infinite delays for others, as shown in Section 5.2. With the Future policy the service is guaranteed to all channels, but the quality of that service in term of, in this case, the expected delay depends on the popularity of the channel. Note also the difference in the delays shown in Figs. 10 and 15: the inaccuracy of context information worsens the performance of the Future policy.

7. Conclusions

In this paper we have focused on data dissemination issues for opportunistic networks. Specifically, we have presented and evaluated ContentPlace, which is a data dissemination system exploiting information about the social behaviour of the users in order to drive the dissemination process. We have provided extensive simulation results showing the advantage of social-aware policies over naïve policies that do not take social information into account. In general, social-aware policies turn out to provide higher hit rate, to be more fair, and to be quicker to disseminate data objects with respect to the other policies. Among the social-aware policies, we have identified Future as the best one. Broadly speaking, Future takes into consideration the set of users (communities) each particular user will be in touch with in the near future, and carries data objects of interest to them. This simple behaviour allows Future to serve all users in an efficient and fair way. The behaviour of Future has been investigated both through simulation and analytical models.

References

[1] A. Balamash, M. Krunz, An overview of web caching replacement algorithms, IEEE Commun. Surv. Tut. 6 (2) (2004) 44–56.
[2] A. Balasubramanian, B.N. Levine, A. Venkataramani, Dtn routing as a resource allocation problem, in: ACM SIGCOMM, 2007.
[3] C. Boldrini, M. Conti, A. Passarella, Users mobility models for opportunistic networks: the role of physical locations, in: IEEE WRECOM, 2007.
[4] C. Boldrini, M. Conti, A. Passarella, ContentPlace: social-aware data dissemination in opportunistic networks, in: ACM MSWiM, 2008.
[5] C. Boldrini, M. Conti, A. Passarella, Context and resource awareness in opportunistic network data dissemination, in: IEEE AOC, 2008.
[6] C. Boldrini, M. Conti, A. Passarella, Exploiting users' social relations to forward data in opportunistic networks: the HiBOp solution, Pervasive Mob. Comput. (2008).
[7] C. Boldrini, M. Conti, A. Passarella, IIT TR-08/2009, Technical Report, IIT-CNR, 2009. <http://www.bruno1.iit.cnr.it/chiara/pubs/comnet09tr.pdf>.

[5] For the case in which the new data object is considered more important than the already existing data objects for the same channel, just substitute $ch_{new} - 1$ with ch_{new} in the summation in Eq. (20).

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001

[8] P. Costa, C. Mascolo, M. Musolesi, G. Picco, Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks, IEEE JSAC 26 (5) (2008) 748–760.
[9] K. Fall, A delay-tolerant network architecture for challenged internets, in: ACM SIGCOMM, 2003, pp. 27–34.
[10] R. Groenevelt, P. Nain, G. Koole, The message delay in mobile ad hoc networks, Perform. Evaluat. 62 (1–4) (2005) 210–228.
[11] P. Hui, J. Crowcroft, How small labels create big improvements, in: IEEE ICMAN, 2007.
[12] P. Hui, J. Crowcroft, E. Yoneki, Bubble rap: social-based forwarding in delay tolerant networks, in: ACM MobiHoc, 2008, pp. 241–250.
[13] P. Hui, E. Yoneki, S.-Y. Chan, J. Crowcroft, Distributed community detection in delay tolerant networks, in: ACM/IEEE MobiArch, 2007.
[14] A. Jindal, K. Psounis, Contention-aware analysis of routing schemes for mobile opportunistic networks, in: ACM/SIGMOBILE MobiOpp, 2007.
[15] H. Kellerer, U. Pferschy, D. Pisinger, Knapsack Problems, Springer, 2004.
[16] V. Lenders, G. Karlsson, M. May, Wireless Ad hoc podcasting, in: IEEE SECON, 2007.
[17] S. Milgram, The small world problem, Psychol. Today 2 (1) (1967) 60–67.
[18] L. Pelusi, A. Passarella, M. Conti, Opportunistic networking: data forwarding in disconnected mobile ad hoc networks, IEEE Commun. Mag. 44 (11) (2006).
[19] J. Scott, P. Hui, J. Crowcroft, C. Diot, Haggle: a networking architecture designed around mobile users, in: IFIP WONS, 2006.
[20] A. Vahdat, D. Becker, Epidemic Routing for Partially Connected Ad Hoc Networks, Technical Report, CS-2000-06, 2000.
[21] D. Watts, Small Worlds: The Dynamics of Networks Between Order and Randomness, Princeton University Press, 1999.
[22] L. Yin, G. Cao, Supporting cooperative caching in ad hoc networks, IEEE Trans. Mob. Comput. 5 (1) (2006) 77–89.
[23] E. Yoneki, P. Hui, S. Chan, J. Crowcroft, A socio-aware overlay for publish/subscribe communication in delay tolerant networks, in: ACM MSWiM, 2007.

Chiara Boldrini is a Ph.D. candidate at the IIT Institute of the Italian National Research Council (CNR), Italy. She holds a M.Sc. degree (2006) in Computer Engineering from the University of Pisa. She is working on social-aware data dissemination, routing protocols, and mobility models for opportunistic networks.

Marco Conti is a research director at IIT, an institute of the Italian National Research Council (CNR). He published in journals and conference proceedings more than 200 research papers related to design, modelling, and performance evaluation of computer-network architectures and protocols. He co-authored the book "Metropolitan Area Networks" (1997) and is co-editor of the books "Mobile Ad Hoc Networking" (2004) and "Mobile Ad Hoc Networks: From Theory to Reality" (2007). He is the chair of the IFIP WG 6.3 "Performance of Communication Systems". He is the Editor-in-chief of the Computer Communications Journal and Associate Editor-in-chief of Pervasive and Mobile Computing Journal; he is on the editorial board of IEEE Transactions on Mobile Computing, Ad Hoc Networks, Journal of Communications Systems, and Wireless Ad Hoc and Sensor Networks: An International Journal. He served as TPC chair of IEEE PerCom 2006, and of the IFIP-TC6 Conferences Networking 2002 and PWC 2003, and as TPC Co-chair of ACM WoWMoM 2002, WiOpt'04, IEEE WoWMoM 2005, and ACM MobiHoc 2006. He served as general chair of ACM REALMAN 2006 and IEEE MASS 2007, and as general Co-chair of IEEE WoWMoM 2006, and ACM MobiOpp 2007. He is currently serving as general Co-chair of IEEE PerCom 2010 and as program Co-chair of ACM MobiOpp 2010.

Andrea Passarella is with IIT-CNR, Italy. Previously, he was a Researcher at the Computer Laboratory, Cambridge, UK. He holds a Ph.D. in Computer Engineering (Pisa University, Italy). He is working on opportunistic self-organising networks, with emphasis on p2p, services, routing protocols, mobility models. He serves in the Program Committes, among others, of IEEE WoWMoM and PerCom. He was Co-chair of IEEE AOC 2009, TPC Co-chair of ACM MobiOpp 2007, Vice-chair for ACM REALMAN (2005/2006), and IEEE MDC (2006). He is Workshops Co-chair for IEEE PerCom and WoWMom 2010. He is an Associate Technical Editor for IEEE Communications Magazine, and is in the Editorial Board of the Inderscience IJAACS Journal.

Please cite this article in press as: C. Boldrini et al., Design and performance evaluation of ContentPlace, a social-aware data dissemination system for opportunistic networks, Comput. Netw. (2009), doi:10.1016/j.comnet.2009.09.001