Osmosis in Pocket Switched Networks

Pan Hui*†, Jérémie Leguay‡§, Jon Crowcroft*, James Scott†, Timur Friedman‡, Vania Conan§
* University of Cambridge
† Intel Research Cambridge
‡ Université Pierre et Marie Curie, Laboratoire LIP6–CNRS
§ Thales Communications

***Abstract*— The increase in the variety and capability of mobile communications devices carried by people today has made it possible to envision a new class of networks called Pocket Switched Networks (PSNs). In a PSN, because the nodes are constantly moving and their communication abilities are limited, the design of networking protocols and applications is challenging. This paper presents a new communications scheme for PSN, called Osmosis, based on the biological phenomenon. We show how this scheme can be applied to file sharing. This scheme uses epidemic routing to perform file lookup and achieves controlled flooding for file transfer. The flooding requires very little state information, which is collected during the lookup. This paper analyzes the performance of Osmosis in simulation studies based on real mobility traces.**

I. Introduction

The number of devices (cell phones, PDAs, music players) people carry every day is constantly growing, and the variety of networking capabilities allows us to envisage new applications such as self organizing peer-to-peer networks that take advantage of opportunistic contacts between people. We describe this networking scenario with the term Pocket Switched Networking (PSN) [1], [2]. Such networks fall under the more general heading of Disruption Tolerant Networking (DTN) [3]. DTN work provides solutions for networks that have very high delay links, or that suffer from frequent disruptions to connectivity, leading to topologies that are intermittently and partially connected. PSN focuses on mobile human scenarios, and aims to enable data delivery between connectivity islands (e.g., Wi-Fi at home and work) by exploiting both local and global connectivity and, crucially, also users' movements. Mobile users suffer from disconnection when they move out of their connectivity island such as their Wi-Fi coverage range at home or work. Partitioning makes traditional end-to-end connections impossible, and so store-and-forward is used for the communications. We refer the reader to the original PSN articles [1], [2] for more detailed information.

We can easily envision that applications such as file sharing will be popular in PSN, as in the Internet. But in the context of PSN, the network suffers from potentially poor and frequently disrupted connectivity, with complete partitioning being commonplace. Resource discovery is thus very challenging to design. In this paper, we propose Osmosis, a new communications scheme for PSN, based on the biological phenomenon. We show how this scheme can be applied to file sharing. This scheme forwards lookups in an epidemic fashion, and then takes advantage of the traces left behind to route reply messages. By analogy to the natural phenomenon of osmosis in biology, those traces represent the solute, and define the potential of nodes over the network. The reply messages then flow towards high potential areas, closer and closer to the original lookup sender.

The main contributions of this paper are: the introduction of the Osmosis approach to forwarding algorithms; its application to a file sharing protocol for PSNs; and the evaluation of this specific application with the help of mobility data. To the best of our knowledge, this is the first work on algorithms for PSN resource lookup services, specifically, in this case, for file sharing.

II. The Osmosis concept for PSNs

In this section we introduce Osmosis, a new communications scheme for PSNs.

A. The overall idea

In biology, osmosis [4] describes the natural flow of solvent molecules from an area of low solute concentration, through a semipermeable membrane, to an area of high solute concentration. Osmosis applies for instance in organic cells as the means by which water is transported into and out of cells.

We propose here a communication scheme to direct flows of messages in PSNs, based on the analogy with the osmosis phenomenon. We equate users carrying devices to cells. Lookup messages are considered to be the solute molecules, and reply messages to be solvent molecules. Each cell i is assumed to have a solute concentration C_i varying between 0 and 1. If the cell contains only the solute without any solvent, C_i is equal to 1 and the cell is said to have the highest potential. If the cell contains only the solvent, or no solvent at all without the solute, C_i is equal to 0 and the cell is said to have the lowest potential.

During the lookup phase, the lookup message is diffused out from the source to the neighboring devices (cells) epidemically [5]. This process establishes the solute concentrations C_i^k in the different cells i for the lookup k. These concentrations can be equal to a constant or can be attributed such that the closer the cell i to the source the higher the concentration. This second choice increases the attraction power of cells that are close to that of the requester. In this case, we define the *closeness* by a discrete distance function $d_s(x)$ that returns concentrations proportional to the distance of the node x to

the source s. Several obvious metrics can be used, such as the number of hops, or the measured delay.

When a device that holds the information requested receives a lookup, a reply message is created. This response represents the creation of a certain quantity of solvent molecules. This event changes the concentration of the current cell by increasing its solvent potential. Because the volume of the solution present at each node is assumed to be identical, the presence of a reply message in a pocket or cell i decreases the concentration C_i^k by an amount δ. Then, according to the osmosis process, it will flow out of its current cell to the others, reaching the cell of the lookup message's sender after a certain period of time. The flow is driven by the osmosis effect. If the reply message is at the node i having node j as current neighbor and if $C_i^k < C_j^k$, it is transferred to node j. By contrast, if $C_i^k \geq C_j^k$, the transfer is much more difficult. Specific constraints have to be defined as to how the osmosis analogy is applied. For instance, one can set a permeability probability to give a certain chance that the messages go through. Alternatively, one can say that transmissions affect the life spans of the reply messages, which is the approach we use in our application in Section III.

B. Relevance of Osmosis in PSNs

Intuitively, we believe that the Osmosis idea is applicable to PSNs because it takes advantage of social correlations between people. Humans form different social circles, such as families, workgroups or interest groups such as football clubs. These circles can be stable and permanent or can exist for a limited period. Human mobility is usually directed by these social circles. In addition to these social groups, people may meet the same strangers on the street every day, such as the people living in the same district, who move with regular synchronized mobility patterns (i.e. working hours from 9am to 6pm). These people are defined by Milgram as "familiar strangers" [6]. These circles are generally interesting for PSN routing because they represent groups within which individuals have regular contacts.

Osmosis takes advantage of these correlations between humans to achieve resource lookup. Lookup messages are first sent to the people around the senders, who can be expected to have some social correlation with those senders. If one is doing a lookup in a classroom, lookup messages are most likely to be transmitted to classmates, and then be spread to their friends, families and "familiar strangers." The process Osmosis employs of identifying the traces of lookup messages, and then using those traces to forward reply messages is effectively detecting those social correlations. "Good" contact opportunities mean that a lookup message is more likely to be received by a node holding the resource, and the resource is more likely to be provided to the requester, than under simple random forwarding. This idea has been previously highlighted in [2] by Hui et al. and has been the basis of several routing protocols designed for PSN or DTN [7], [8], [9].

Osmosis also naturally takes advantage of networking correlations. In the lookup/response process, we have a cause and result relationship: reply messages exist because of the lookup and reply messages are addressed to the lookup sender. Reply messages in Osmosis are *attracted* by traces left during the lookup process, as if a virtual reverse path exists.

III. Osmosis in PSN file sharing

Peer-to-Peer (P2P) file sharing has become one of the most popular applications in the Internet in recent years. Many very successful systems used in everyday life by millions of people have been implemented such as Gnutella [10] and BitTorrent [11]. We may reasonably expect that these applications will be adopted by PSNs users as well. Moreover, the peer-to-peer collaborative nature of the underlying PSN reinforces the fact that file sharing is potentially one of the main applications for these new networks. We show in this section how Osmosis can be applied to support file sharing applications in PSNs.

A. PSN file sharing

There are a number of characteristics of PSNs which make file sharing different from that on the Internet. In traditional Internet P2P file sharing, the usual strategy is basically a *four-step* process. First a lookup is launched to identify the hosts having the file which matches the query. The initiator of the request retrieves a list of potential peers. It can then target some of these peers to download files or parts of files called blocks, by directing their requests to them. This selection process is partly to choose the best performance peer. When their turns come around, the requested information is returned to the sender. In PSN, as in DTN generally, such chatty protocols could result in very large delays. These exchanges have to be reduced to a minimum. We require PSN file sharing protocols to maximize the chance for the users to be satisfied *at all*, while having a minimum impact on the network. We propose here a simple file sharing application that relies on a *two-step* communication process, consisting of just a lookup phase and a reply phase.

As it is only a two-step process, we use only two message primitives: GET and PUT. GET messages are generated during the query phase and PUT messages are issued during the reply. A time to live (TTL), is used in these messages to limit their life in the network. TTL is not simply a hop count, but rather actually refers to wall-clock time. After a certain time, the issuer of a GET message may no longer be interested in the file he requested. Also, since the correlation between the number of hops and the delay experienced by a packet does not exist in PSN, an expiration date is needed to prevent old queries from remaining in buffers of nodes. We also employ a counter to uniquely identify each GET message. The counter is increased by the sender each time a new query is issued.

A node that receives a GET message and has the file available in its buffer, creates a PUT message and sends it towards the issuer of the GET. Of course, it is possible that multiple nodes hold the requested file, and such a strategy could lead to a reply storm. Several ways can be imagined to limit such a storm. For now, we assume that when two PUTs

meet each at a node, the first one that arrives is processed, while later ones are discarded. We retain the request state traces to provide this filtering.

As we know, the PUT messages can be much larger than the GET messages since a PUT message contains the requested file. We propose an epidemic scheme to distribute the lookup messages. This increases the chance of reaching a node with the file of interest. A TTL value is added to limit the flooding. For the reply process, it is a bit more complex: the goal is to return the file back with a certain level of reliability while not overloading the network. An epidemic distribution of the reply is thus not realistic. Osmosis limits the flooding of the reply towards the requester as described in next part.

B. p-Osmosis: Penalty-based Osmosis

We present here an instance of Osmosis, called p-Osmosis (for *penalty-based* Osmosis), that is used to drive reply messages in the PSN file sharing application presented previously and evaluated in Section IV.

The analogy is applied in this way: We consider mobile devices to be cells; GET messages of the file sharing application are considered to be the solute molecules, and PUT messages are equivalent to the solvent molecules. Also, in p-Osmosis, when the solute is distributed during the lookup phase, every node i containing the solute, i.e. the GET message, has the same solution concentration C_i. This instance of the Osmosis concept thus leads to only 3 different concentrations for the solution: 0 when only the solvent (PUT message) is present or when neither the solvent and solute (GET message) is present; 0.5 when both solvent and solute are present at a node, which comes from the fact that the quantity of molecules of solute present in a GET message is equal to the quantity of molecules of solvent in the PUT message; and finally 1 when only the solute is present at a node.

A PUT message is generated at node i, if it owns the file, and will be transferred to neighboring cells having lower solvent potentials. When the PUT message is created at node i, the solution concentration become 0.5. The message will then flow naturally toward nodes having concentration of 1, i.e. the nodes that have seen the lookup messages before. There is a possibility of the message flowing to high potential areas, depending on the permeability of the cells involved in the process. To make the protocol more flexible and tolerant, we consider that molecules can flow toward high potential areas, but with an effect on them. We implement this effect as a penalty p attributed to their time-to-live, hence the name, p-Osmosis. As a consequence, when PUT messages flow in the wrong direction, their time-to-live will be reduced in order to lower their importance. The parameter p can be tuned to kill those messages more or less rapidly.

To describe the penalty attribution algorithm formally, assume that r_{ji} is the time remaining (or TTL) of the PUT message j at node i. P_i and G_i represent respectively the set of PUT messages and GET messages seen by the node i. Note that the PUT message in reply to the GET message has the same message ID as the GET message j. Algorithm 1 describes the penalty attribution algorithm used by p-Osmosis.

Algorithm 1: Penalty attribution in p-Osmosis.

```
begin
    if j ∉ P_i then
        if j ∉ G_i then
            P_i ← P_i ⋃ {j};
            r_ji ← r_ji − p;
end
```

In p-Osmosis, PUT messages can be triggered voluntary when the resource requested by a given GET message j relayed by a node A is present in one of PUT messages currently stored in its buffer. It only applies if the corresponding PUT message of j has not been already relayed by that node. Note also that a GET message j is not relayed by node A as soon as it has seen the corresponding PUT message j.

IV. Evaluation methodology

We have implemented a stand alone simulator to evaluate the performance of Osmosis. This simulator only implements the transport and network layers. It makes simple assumptions regarding lower layers described later in this section. We give here an overview of the general parameters of the scenarios we used for simulations.

Mobility traces used

We used mobility traces from the WiFi access network of Dartmouth college [12] to model the movements of users in our simulations. Dartmouth college's wireless network is composed of about 550 access points (APs) and is used daily by thousands of users. It covers the academic buildings, the library, the sport infrastructures, the administrative buildings and the student residences. As in [9], to evaluate Osmosis, we have used mobility traces recorded between January 26$^{\text{th}}$ 2004 and March 11$^{\text{th}}$ 2004 because it corresponds to a class period where users make an intensive and regular use of the network.

These mobility traces consist of logs of associations or dissociations of nodes to APs. We have adapted the WiFi traces from Dartmouth to make them usable in a PSN scenario which focuses more on the interaction, e.g. contacts, between nodes. We assume that two nodes are able to communicate with a low range device (using Bluetooth or WiFi for instance), if they are attached at the same time to the same AP. As one may see, WiFi traces have a number of characteristics that make them different from PSN traces. We must admit that there are shortcomings in using this data, since nodes that are attached to two different APs that are close to each other might be able to communicate directly, or by contrast, two nodes connected at the same AP might be out of range of each other. Nonetheless, this is the best approximation we can make with the data at hand. Although these kind of traces presents a coarse-grained sample of human mobility, we believe that they do provide useful support for early evaluations of PSN protocols or applications.

People that use the wireless access network are not connected with their WiFi equipment during the entire day. Thus these data represent only a sample of their mobility, which might not be representative. However, one may argue that users using their PSN file sharing application will tend to perform queries while they are stationary, and may be sitting down at the cafeteria or in a lecture theater, just as many of the Dartmouth WiFi users would be.

We have also used a slightly different data set than in [9]. We selected all the users in the data that have been active during 31 days. 31 represents the number of days between January 26th 2004 and March 11th 2004 (45 days) minus the Saturdays and the Sundays. We tried here to capture two classes of users that make an intensive use of the network but who do not have the same habits. The first class contains the users that might live in the campus, i.e. people that are present in the data all 45 days. The second class is formed by users that come on week days at the college to work or to study.

Traffic generation

The way traffic is generated in simulations is important for accurate protocol evaluation. We envision that PSN file sharing may have many similarities with Internet P2P, but showing some changes of emphasis, due to the features of devices and the social context in which they are used.

Because traffic is driven by application usage, we are interested in envisioning future usage of PSN file sharing. For this reason, we have conducted an informal on-line survey that was advertised during the month of November 2005 on USENET newsgroups and forums dedicated to mobile computing. We would like to emphasize the informal character of that survey: it has been only performed to learn the flavor of the results we might expect. We received 145 answers from experienced users who were using, for 57.93%, PDAs and, for 42.07%, smart phones. Part of the survey result shows that the most popular files on the mobile devices are software, MP3, pictures and documents (other). Nowadays, PDAs and smart phones can be used to play music and most of them have a built-in camera. Most of the people seem to hold 30 to 100 files on their devices. We observe that the median file size distribution follows a Zipf's distribution with exponent -1.569 as a basis, but with a bump around the typical MP3 size, i.e. 3-4 MB.

We introduce here the traffic model we used for the simulations that attempts to capture both observations from previous studies conducted on Internet P2P systems [13], [14] and future behaviors that can be inferred from the survey we conducted. The distributions of the popularity of the ownership and of the queries follow Zipf's laws that can be describe in a generic fashion by the following equations:

$$f(k;e,N) = \frac{1/k^e}{\sum_{n=1}^{N} 1/n^e} \text{ and } \sum_{k=1}^{N} f(k;N,e) = 1 \quad (1)$$

where N is the number of files, k is their rank, and e is the exponent characterizing the distribution.

In order to find the exponent characterizing each distribution, we fix a parameter for each of them which is $f(0;s,N)$. This is the popularity of the most popular file. Note that the rank attributed for the ownership is not the same as for the queries.

Regarding the file size distribution, we also used a Zipf's distribution but distorted it around 3 MB. According to the survey results, we assume here that MP3s represent 30% of the files in the network. Thus, after established n representative file sizes S from 0.5 MB to 50 MB (3 MB excluded), we established that the Zipf's distribution f_s should satisfy $\sum_{k \in S} f_s(k;n,e_s) = 0.7$. Figure 1 shows the artificial file size distribution we used in comparison to the one observed in the survey.

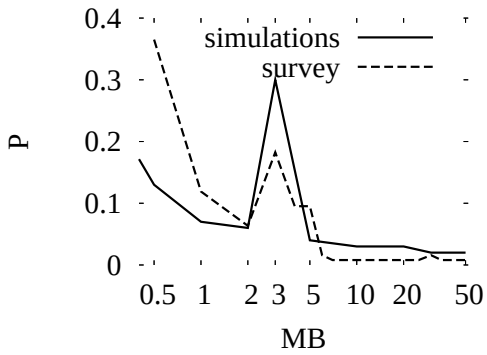

Fig. 1. File size distribution used.

Protocol comparison

We compare the performance of p-Osmosis against the following approaches:

- *Wait-for-Destination*: A node waits to meet a node that owns the file that it is interested in. The main advantage of this method is that it involves only one transmission per file. The delivery relies just on the mobility of nodes and their contact opportunities.
- *Epidemic routing*: the file is sent to the requester in an epidemic fashion. If two epidemic answers meet, the last arrived is discarded.

These two protocols represents two extremes. Wait-for-Destination leads to the minimum communication overhead by making no efforts to look for the data requested father than one hop. Epidemic routing is the opposite case, an extensive effort is performed to fetch the data, leading to a high network overhead. A good file sharing protocol for PSN would be one that achieves the same performance as Epidemic in terms of amount of information fetched, while incurring a reasonable communication overhead.

Performance metrics

For all the simulations we have conducted for the p-Osmosis evaluation, we have measured the following metrics:

- *Success rate*: The proportion of queries that have been satisfied out of the total of queries issued.
- *Delay of retrieval*: The amount of time the user has to wait to get the requested file. We have computed the average and the median values for the delay.
- *Average number of hops for the replies*: The number of hops needed for a reply to be delivered. This metric gives an idea of the social distance between the client and the server.
- *Number of expired replies*: This metric measures the number of reply messages that have expired because of their TTL.
- *Communication cost*: To measure the communication cost, we implemented two metrics, the number of medium accesses and total amount of data transferred. Each time a node initiates communication with another node, we refer to this event as a medium access.
- *Average maximum buffer utilization*: For each node, we monitor the buffer occupancy during the simulations. This statistic is the average over all the participants of the maximum buffer occupancy.

Common simulation parameters

All the scenarios share the common parameters that can be found in Table I. For computational reasons, we picked at random for each simulation run 300 nodes participating in the PSN in which 50 nodes are considered to be *active*. A node is said to be active if it issues search queries from the file sharing application. Each of the active nodes generates a total of 30 different searches during the 45 days of data replayed. We assumed that the number of files available in the PSN is 200 and that 800 copies of these files are spread across the nodes before the simulation starts. The size of the request messages and the size of the reply headers are considered to be 1 KB. The time to live (TTL) of a file sharing session is 2 weeks and the penalty for Osmosis is 2 days. The launch of the search queries is spread over the 4 first weeks of the period replayed to let them have time to be satisfied or not. Finally, we used a time step of 1 sec. for the simulations.

Parameter	Value	Parameter	Value
Total nodes	2551	Total number of files	200
Total locations	533	Total number of copies	800
Users sampled	300	Time step	1 s
Users generating traffic	50	Size of requests	1KB
Simulation duration	45 days	Size of replies' header	1KB
Requests per active user	30	TTL	2 weeks
		Penalty	2 days

TABLE I
SIMULATION PARAMETERS.

Note that while capturing some important characteristics of what might be the traffic for PSN file sharing, our model does not capture the social behavior that might be induced by social communities. One can imagine that PSN traffic for file sharing might be divided into two classes. One that exists within one or several social communities that are very related, and one that exists at the scale of the PSN. The model used here may only be representative of the second class, leading to performance that are not exceptional. We expect Osmosis to achieve better performance using a traffic model capturing the effect of social communities.

V. RESULTS AND DISCUSSION

We performed five runs for each scenario, and, in the following tables, present mean results with confidence intervals at the 90% confidence level. As five runs is particularly low for the calculation of confidence intervals, we used the Student t distribution, which is suitable for small numbers.

A. Simulation Results

To obtain an understanding of the performance of p-Osmosis that was not conditioned on too many simulation parameters, we started by studying a scenario with two simplifying assumptions: infinite buffers in the nodes, and unlimited bandwidth between the nodes. Table II shows the simulation results.

	Wait-Destination	Osmosis	Epidemic
Success rate (%)	31.0 ± 2.8	54.7 ± 4.6	61.5 ± 4.5
Avg retrieval delay (days)	3.4 ± 0.1	2.4 ± 0.1	2.0 ± 0.1
Med retrieval delay (days)	1.5 ± 0.2	0.6 ± 0.1	0.1 ± 0.0
Avg hop count for replies	1.0 ± 0.0	2.6 ± 0.1	3.9 ± 0.2
Medium accesses (10^3)	0.9 ± 0.0	121.1 ± 15.0	208.2 ± 16.4
Data transferred (GB)	1.9 ± 0.4	482.3 ± 107.6	839.1 ± 212.7
Avg Max buffer utilisation (MB)	13.8 ± 2.8	309.3 ± 44.7	1145.7 ± 247.7
Replies expired (10^4)	0.0 ± 0.0	10.8 ± 1.4	19.2 ± 1.6

TABLE II
RESULTS WITH INFINITE RESOURCES.

We first observe that Epidemic satisfies 61.5% of the requests while Wait-for-Destination satisfies 31.0% of them. These results bound our expectations for the performance of any other algorithm. In this mobility scenario, based on the Dartmouth data, it is simply not possible to satisfy more than 61.5% of the requests. And, while it is possible for an algorithm to perform worse than Wait-for-Destination, given Wait-for-Destination's minimal investment of resources it is hard to imagine another algorithm that could provide a benefit that would compensate for having to accept a worse request satisfaction level. Osmosis performs much closer to Epidemic than to Wait-for-Destination, satisfying 54.7% of the requests.

In terms of delays, Wait-for-Destination shows the highest, at 3.4 days, and Epidemic the lowest, at 2.0 days. Osmosis achieves a retrieval delay closer to Epidemic than to Wait-for-Destination, at 2.4 days. Results concerning the delays are nuanced by the fact that it is computed on the files that have

been effectively retrieved. To add further perspective, Table II shows also the median delay. We can observe that the results concerning the median delay are better. Wait-for-Destination shows the highest, at 1.5 days, Osmosis and Epidemic show respectively 0.6 and 0.1 days. This means that most of the lookup can be satisfied by Epidemic and Osmosis within 1 day.

Looking at the results for the average number of hops needed for the replies to reach the requester, Wait-for-Destination naturally performs the best, as by definition it requires just 1 hop. Osmosis and Epidemic need 2.6 and 3.9 hops respectively. We see the poorer performance of Epidemic compared to Osmosis as the price that Epidemic pays in order to achieve its 6.8% margin over Osmosis in terms of request satisfaction. Epidemic is successful in retrieving files that are farther away, and these are files that Osmosis must be dropping because they, along some paths in their trajectories, are not flowing toward a region of high solute potential.

In summary, the performance of Osmosis is close to Epidemic in terms of request satisfaction. The costs to pay are the slight reduction in satisfaction, and a higher average delay for the requests that are satisfied. On the other hand, the average number of hops required to satisfy requests is lower for Osmosis than for Epidemic. This points to lower resource consumption by Osmosis, and we look to Table II to see if this is the case.

Naturally, Wait-for-Destination requires negligible communication costs in term of medium access and data transferred compare to Osmosis and Epidemic. We see that, under either Osmosis or Epidemic, considerable resources are consumed in improving the level of request satisfaction compared to this base level. However, and this is significant, Osmosis offers a significant reduction in communication costs compared to Epidemic. It reduces by 41.8% the accesses to the medium and by 42.5% the total amount of data transferred. Since Osmosis and Epidemic benefit from having users relay information for each other, their buffer utilization must be larger than Wait-For-Destination, and this is borne out by the results. However, Osmosis reduces the maximum occupancy of the buffers by 73%.

We also measured the number of replies expired for each protocol. In Epidemic, the basic TTL of 2 weeks impacts the expiration of replies. In Osmosis, replies can be killed early because their TTL has been decreased by penalty attributions. We might have expected that more replies would expire with Osmosis than with Epidemic. This result is simply explained by the fact that in Osmosis replies expire earlier, stopping the spreading of replies to low potential areas. Thus, the total number of replies that are present in the network is lower.

VI. Conclusion and future work

In this paper, we have introduced Osmosis, a new communications paradigm for PSNs, based on an analogy with the well known biological phenomenon of the same name. We have applied this concept to PSNs file sharing to take advantage of the underlying relationships between the lookup and the reply messages. Lookup messages create a potential at each node they traverse, that attracts reply messages so that they will flow back toward the lookup originator. We have applied this concept for file sharing through the design of a specific protocol, p-Osmosis for penalty based Osmosis, and we have evaluated this by replaying mobility traces that we inferred from the Wi-Fi access network from Dartmouth College.

We have shown that p-Osmosis leads to a lookup satisfaction close to the one of a basic epidemic scheme for replies, by strongly reducing the overhead in terms of communication costs and buffer utilization.

Note that the Osmosis concept might also be applied in PSN or DTN to other purposes, such as to create packet erasure protocols. The strategy would be to send the data epidemically and then to send a sort of CLEAN packet attracted with Osmosis to the area where data need to be erased.

Finally, we expect to evaluate p-Osmosis and to continue the investigation of the Osmosis concept with the help of other mobility traces like the one acquired with the iMotes [1] within the Intel's Haggle project or the one of the Reality Mining project [15] captured with mobile phones. These data sets provide information about fine-grained interactions between people instead of their co-presence in a coarse-grained area.

References

[1] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot, "Pocket switched networks and human mobility in conference environments," in Proc. WDTN, 2005.

[2] P. Hui, A. Chaintreau, R. Gass, J. Scott, J. Crowcroft, and C. Diot, "Pocket switched networking: Challenges, feasibility, and implementation issues," in Proc. WAC, 2005.

[3] K. Fall, "A delay-tolerant network architecture for challenged internets," in Proc. SIGCOMM, 2003.

[4] "Osmosis definition. website: `http://en.wikipedia.org/wiki/Osmosis`," .

[5] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Tech. Rep. CS-200006, Duke University, April 2000.

[6] S. Milgram, "The familiar stranger: An aspect of urban anonymity," pp. 51–53, 1977.

[7] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," in Proc. SAPIR, 2004.

[8] B. Burns, O. Brock, and B. N. Levine, "MV routing and capacity building in disruption tolerant networks," in Proc. Infocom, 2005.

[9] J. Leguay, T. Friedman, and V. Conan, "Evaluating mobility pattern space routing for DTNs," in Proc. INFOCOM, 2006.

[10] "Limewire," http://www.limewire.net.

[11] "Bittorrent," http://www.bittorrent.com.

[12] T. Henderson, D. Kotz, and I. Abyzov, "The changing usage of a mature campus-wide wireless network," in Proc. Mobicom, 2004.

[13] A. Klemm, C. Lindemann, M. K. Vernon, and O. P. Waldhorst, "Characterizing the query behavior in peer-to-peer file sharing systems," in Proc. IMC, 2004.

[14] S. Zhao, D. Stutzbach, and R. Rejaie, "Characterizing files in the modern gnutella network: A measurement study," in Proc. SPIE, 2006.

[15] N. Eagle and A. Pentland, "Social serendipity: Mobilizing social software," in Proc. PerCom, 2005.