Universidade do Minho
Escola de Engenharia

Pedro Jorge Alves Barbosa Vieira

Towards Social-Aware Opportunistic Network Datasets

Outubro de 2012

Universidade do Minho
Escola de Engenharia
Departamento de Informática

Pedro Jorge Alves Barbosa Vieira

Towards Social-Aware Opportunistic Network Datasets

Dissertação de Mestrado
Mestrado em Engenharia Informática

Trabalho realizado sob orientação de

Professor Doutor Joaquim Melo Henriques Macedo
Professor Doutor António Luís Duarte Costa

Outubro de 2012

Abstract

Delay-tolerant networks are wireless networks designed to be used in cases where network infrastructure is nonexistent or not available to be used. Because of this, there are several problems that need to be addressed in this environment, such as lack of continuous end-to-end connectivity and increased delay and error rates in data transfer. As such, conventional routing schemes aren't feasible in providing efficient solutions for these cases.

Since the nodes present in these kinds of networks usually possess very limited resources, opportunistic routing protocols should not only try to achieve a good message delivery probability, but also reduce the number of message replicas present in the network. This is done so as to avoid an unnecessary waste of storage and energy that comes from storing and transmitting messages to other nodes.

Some of the recent Delay-tolerant network routing proposals involve using social information to determine which node has a higher probability of successfully delivering a message to its intended destination. This seems to be a popular strategy, that achieves a good delivery probability while reducing the message overhead, when compared to simpler schemes.

One way to analyze the performance of a routing protocol is to use real opportunistic contact datasets to simulate a real life environment. This work focuses on providing a research on opportunistic network traces as a way to determine the contact patterns of Delay-Tolerant network nodes and their impact on routing algorithm performance, as well as proposing an architecture for a future data collection experiment.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Interest in Delay-Tolerant networking can be traced to the beginning of *Interplanetary Internet* research, which intended to develop a network architecture suitable for communication across outer space [1]. This environment proved to be often disconnected, error-prone and having high data transmission delay.

Also, when wireless technologies began to receive widespread usage, research has been done on terrestrial mobile and vehicular ad-hoc networking in challenging environments, where the network topology changes frequently due to the mobility of the nodes involved, while also suffering from similar problems as those found in interplanetary networking. Over the last years these subjects have been studied with great interest from the academic community.

Provided they have access to the Internet, users can easily communicate with devices situated anywhere on the world; this communication is typically error-free and has a relatively low delay. That is possible because the Internet nodes use a uniform protocol stack, which allows for easy communication between them. These protocols, such as TCP, facilitate end-to-end data transfer by implementing error-free mechanisms, such as packet reordering, retransmission of lost packets or flow control. Also, network routers are able to decide the most efficient path for a packet to take in order to reach its destination, even if a link in the network suddenly disconnects.

Additionally, we can state that the Internet operates based on some assumptions, namely the existence of an end-to-end path between sender and receiver, low error rates and short round-trip times.

In a delay-tolerant architecture, we cannot make these assumptions. Nodes in the network may constantly disconnect, packet errors may be frequent and may take a long time to reach their final receiver; furthermore, there is no access to a wired infrastructure, meaning that the mobile nodes must forward themselves the data in an ad-hoc manner, and only when they are in range of other devices (we call this opportunistic connections).

Even more so, we may have regions within the network that implement different protocol sets, which in turn requires the usage of an additional protocol layer in order to homogenize communication between different regions (the *Bundle Protocol Layer*).

All of these factors can impair the usability of the network; that's why we are required to employ a different set of techniques than those used on the Internet, in order to address these problems.

One of the most important approaches used in delay-tolerant networks is the *store-carry-and-forward* mechanism, in which messages are temporarily saved by the device in memory. When an opportunity arises, the message would then eventually reach its destination. (Note that since we can never predict when, or even if, the nodes in the network will be connected, there can be no guarantees that a message will ever arrive to its intended recipient). This implies further decisions to be made, in terms of the limited buffer space allocation and garbage collection, as well as the number of replicas of a message that are propagated through the network.

We can then conclude that the main problems related to a mobile delay-tolerant networking architecture relate to finding a balance between message delivery ratio and delay, in addition to the number of copies dispatched and the amount of resource consumption.

1.1 Context

There are several specific scenarios that can be considered when studying routing schemes for Delay-Tolerant Networks.

As discussed above, the Interplanetary Network consists of a network that possesses nodes spread over the space. As such, high transmission delay and constant disconnections are to be expected. Nonetheless, routing algorithms in this scenario have the advantage of being able to predict the periods in which there is available connectivity, by calculating the distance between nodes based on the planets' orbits.

Another possible usage for this technology is the vehicular networking scenario, where nodes are located inside mobile vehicles. These nodes can communicate when they approach each other, thereby creating a mobile network which can possibly provide traffic or weather information to the vehicle's driver. In this case, mobility patterns can be very useful to predict future encounters, since vehicles can only move in well-defined areas. This is especially true when considering nodes located on trains, which only travel on a fixed track.

In our case, we intend to focus our study on a specific delay-tolerant network instance, where nodes consist of mobile devices carried by humans (designated *Pocket Switched Networks*). Considering this scenario, we can explore the devices' mobility to physically carry the messages to another location; in this context, such devices are known as *Data Mules*.

One scenario that can be considered as a real-world application of our work could be in an urban context, where smartphone users may want to use this type of technology to communicate with each other, in the case that they are not allowed, cannot afford, or simply do not want to use the existing network infrastructure.

Since the network nodes are carried by people who will probably interact with each other on a regular basis, we believe social behavior information will be most useful in this environment. This observation can also allow for a more trusted communication channel, since the messages passed will usually be forwarded by nodes which have social ties with each other.

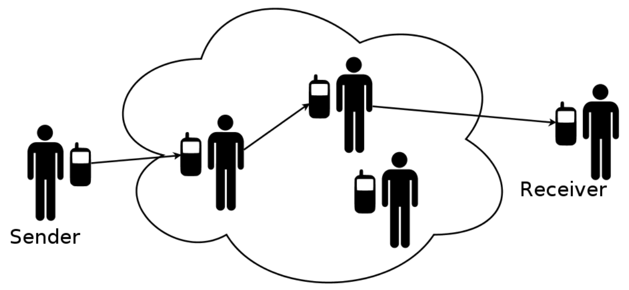


Figure 1.1: A scenario where social connections can be explored to route messages between mobile devices.

1.2 Motivation

The number of mobile devices with networking capabilities has increased significantly over the last years. As such, we have witnessed a growth on the demand for continuous wireless connectivity. Nonetheless, there are many parts of the world which cannot provide a network infrastructure that allows their residents to connect to the Internet. Environments such as developing countries, military zones or sensor networks may not allow for their mobile device users to use the Internet in a standard manner. Implementing a Delay-Tolerant networking scheme in these challenging zones can then enable communication where it was previously unavailable.

Meanwhile, we can observe that social network services are highly popular nowadays, with websites such as Facebook or Twitter having millions of users logged in at any time. These services can provide valuable information with respect to the social interactions of their users, which can then be overlaid on top of physical connection data, so as to explore possible connection opportunities.

1.3 Objectives

The proposed objectives of this work consist in studying and comparing different Delay-Tolerant network routing protocols and datasets. It is intended to compare and describe recent routing proposals in terms of the strategies they use, as well as analyzing their performance in terms of common metrics such as message delivery probability, delay and overhead.

The main goal is to perform a thorough analysis of real world opportunistic datasets collected in different scenarios. The idea is that, by trying to find common patterns regarding the contact opportunities of the participating nodes, we can gain a better understanding of how a scenario with specific network characteristics can influence the overall connectivity in that environment. To do so, different statistical analysis methods will be used, with emphasis on visualization techniques and the distribution of the nodes' connection opportunities, for duration the of the experiment.

We expect to also perform a more dynamic dataset analysis, by exploring the possibility of describing the intermittent connectivity of these environments by means of time-varying graphs. Since the nodes present in opportunistic networks usually have limited resources, it is planned to study the impact of these restrictions in terms of the ability to forward messages in the network.

It is also intended to propose an opportunistic dataset collection architecture that allows the retrieval of both physical connection data, as well as user data from a widely used social network service; in Chapter 2, we will observe that using social information can be an effective method of achieving a good message routing performance.

1.4 Document structure

The document is structured as follows: Chapter 1 introduces the problem of Delay-Tolerant networks in terms of message routing protocols, the motivation to perform research on this subject and some possible environments in which this type of network model can be applied, as well as listing the proposed objectives for this work.

Chapter 2 proposes a classification taxonomy for existing routing protocols. Also, two of the most referenced protocols on this subject are described. Finally, we analyze and discuss some recent related proposals on the subject of opportunistic network routing protocols in terms of their general strategy and performance.

In Chapter 3, different opportunistic network datasets, describing realistic connection opportunities, are subject to a statistical analysis and comparison. A number of techniques are applied, focusing on visualization, statistical distribution, and time series analysis.

Chapter 4 presents a time-varying graph model to describe the intermittent connectivity that is commonly observed on opportunistic networks. It also evaluates the impact of different network parameters (such as node buffer size and message time to live) on the performance of common routing protocols, by means of simulation experiments.

Chapter 5 some discussion is made regarding the problem of planning a data collection experiment in an opportunistic environment. It proposes an architecture for obtaining opportunistic network datasets that combine physical and social network information from a group of nodes. Lastly, a proof of concept example is described, in order to demonstrate the feasibility of collecting physical and social contact information.

Finally, Chapter 6 concludes this document with some observations regarding the work that was done, as well as discussing some possibilities for expanding this work on the future.

Chapter 2

Delay-Tolerant Routing Protocols

In the last few years, several protocols have been designed to deal with the message routing decisions used in delay-tolerant networks. In this chapter, some of these proposals will be discussed and compared, focusing our attention on recent social-aware approaches, since those are expected to be more closely related to our long term goal for this project.

Firstly, a hierarchical view of the several kinds of protocols that have been proposed for opportunistic networks will be presented, highlighting the most important aspects of each of them; afterwards, the Epidemic and PRoPHET routing schemes, the two most widely referenced protocols in this area, will be briefly described.

Then, some Delay-Tolerant routing algorithms will be analyzed, discussing the main concepts they introduce, as well as comparing them in terms of performance with some metrics obtained from simulations.

2.1 Classification of routing protocols

Routing protocols in this area of research are usually classified as being forwarding-based (only one copy of a message exists through the entire network), or replication-based (where multiple replicas of the same message are spread among the nodes) [2]. When a message is passed in the forwarding scheme, the original holder deletes it from its own memory buffer; in a replication scheme, both nodes end up having the same message.

The forwarding-based algorithms have the obvious advantage of wasting fewer resources in relation to the replication-based approaches, at the expense of usually achieving a lower message delivery ratio. In the context of Pocket-Switched Networking, one can deduce that multiple-copy routing approaches will probably be more adequate, since the connectivity issues discussed above imply some amount of message loss.

Additionally, there are different variants of the replication algorithm: namely *greedy replication* (which consists of passing the message to all of the encountered nodes, much like the Epidemic Protocol), *controlled replication* (implying that there is some kind of limit to the number of replicated messages, like the Spray-and-Wait scheme); or *utility-based* (where each node possesses a value that represents its feasibility of being able to carry the message to its destination).

Utility-based routing algorithms can also be classified as being *social aware*, meaning that the nodes have notion of the social relations with each other, like on of the proposals mentioned in the following section; or *mobility aware*, if information about the mobility patterns of the network participants is used in the routing decisions (such as the MobySpace model [3]). Also, in the social aware approach, we can have a *hierarchical* algorithm (if the nodes can be grouped in social communities, or *clusters*); or *flat*, if no such grouping is made. An example of an hierarchical approach is the BUBBLE protocol, while flat algorithms such as PROPICMAN or SimBet are discussed below.

There exists yet another kind of approach to these protocols, known as *message coding* or *coding-aware routing*: the network nodes, instead of simply forwarding the messages, can also process the received data in other ways, based on information theory concepts.

Coding-aware proposals possess two variants: in *source coding*, the sender node changes the original message to a code with additional information (for example, for error correction purposes); meanwhile, in *network coding* schemes, the intermediate nodes are able to join several received messages into only one, so that increased information output can be achieved. Once enough encoded messages are sent, the other nodes will then be able to decode the original message.

Finally, Delay-Tolerant Routing algorithms can be classified as being either *centralized* or *distributed*. A centralized algorithm requires knowledge of the entire network topology, as opposed to a distributed one, where the nodes update their context information when a connection is made to one of their neighbors.

Based on the work of [4], we will now expand the proposed routing protocol taxonomy with some of the characteristics that were discussed above, as shown in Figure 2.1:

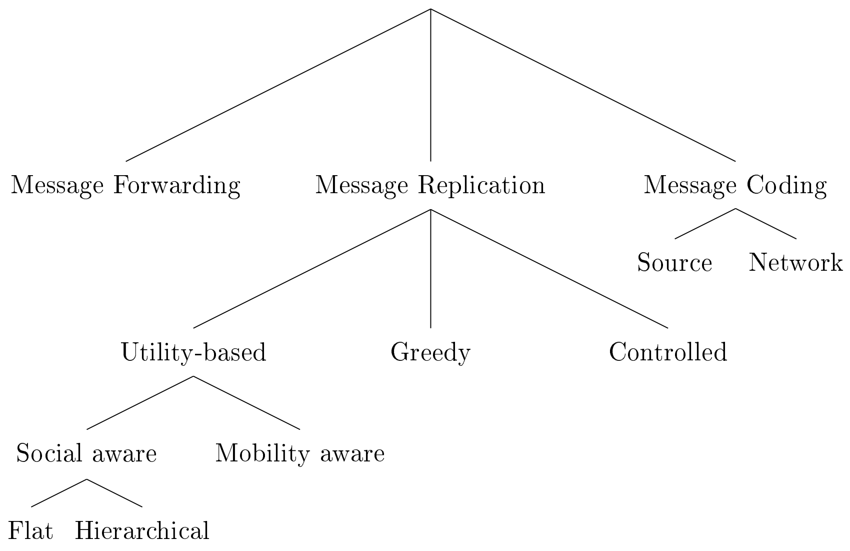


Figure 2.1: Routing protocols classification taxonomy.

2.2 Reference algorithms: Epidemic and PRoPHET

The Epidemic routing technique [5], in its simplest version, can be generally described as follows: a sender node propagates a message to all the neighbors that it encounters; each of the receiving nodes does the same and so on, until the message eventually reaches the intended receiver; this behavior presents some similarities with the manner in which a virus spreads through a community.

More specifically, each of the network nodes possesses a summary vector, which consists of a list of messages it currently stores. When a connection opportunity arises between two nodes, they enter a designated anti-entropy session. In this phase, a node compares its own summary vector with its pair's and requests the messages it doesn't possess locally, so that both participants share the same set of messages when this session ends.

Additionally, each message has a hop count field, indicating the maximum number of nodes a message can pass through before it is discarded. Although this approach ensures the best delivery ratio and minimizes the delay (since one of the copies will inevitably take the most efficient route), it may also saturate the network with useless replica overhead, thus wasting a lot of bandwidth and resources on the mobile devices, a situation that is undesirable in our scenario.

One proposal that seeks to improve this solution is the Spray and Wait scheme [6]. This protocol is a variation of the Epidemic algorithm, consists of two separate steps: the *Spray phase*, in which the sender transmits a fixed number of copies to its adjacent nodes; and the *Wait phase*, when all of the message holders will only forward the data directly to the destination if a direct path exists. This approach aims to lower the amount of messages present in the network at any given time by halting message propagation, while also trying to achieve a high transmission ratio and low latency, thus increasing the scalability in regards to "blind" Epidemic routing.

Another well-known routing scheme in this subject is PRoPHET (Probabilistic Routing Protocol using History of Encounters and Transitivity) [7]. This protocol's functionality is based on the assumption that the mobility pattern of the nodes is not entirely random; hence, it may be possible to determine the probability of a network node meeting another, by analyzing the history of past physical encounters that recently occurred between them.

PRoPHET achieves this by assigning a delivery predictability to each node pair, a value that indicates the likelihood of these two nodes meeting again in the future. This metric is recalculated every time two nodes meet, so that frequent encounters translate to an increased predictability value. This protocol also possesses the notion of aging (recent meetings are more important than distant ones), and transitivity (meaning that a node can be considered a suitable intermediary between two other nodes it usually meets).

Using this information, a node will then forward the message to another if their delivery probability is higher than the current message holder, in regards to its final receiver. This results in a significant decrease in message overhead, while maintaining comparable performance in relation to the Epidemic approach.

2.3 Social-aware Routing

The following section describes some recent context-aware routing protocols. These algorithms are able to predict future node encounters by some kind of social information (social network metrics or information about the owners, for example). In our case, they are particularly interesting, since they can illustrate some of the ways we can treat context information as a means to infer social ties between the nodes.

2.3.1 BUBBLE

The BUBBLE forwarding algorithm [8] operates by grouping the network nodes in terms of social communities, or *bubbles* (each node belongs to at least one bubble). Each of the participants in the community possesses two key attributes: the local ranking, which quantifies the node's popularity (or centrality) in relation to the others within the same community; and a global ranking, indicating the node's centrality value throughout the whole network.

In order to socially classify the network nodes, this protocol presents two different community detection approaches: by using existing centralized schemes; or the distributed version of the algorithm, called DiBuBB.

The centralized approach relies on the usage of two complementary community detection algorithms: K-CLIQUE [9] and Weighted Network Analysis [10]. These algorithms are divisive: starting from the whole network graph, they will iteratively split it into smaller clusters.

K-CLIQUE divides the whole network graph into complete subgraphs (or *k-clique communities*) that are accessible by a set of other adjacent k-cliques (k-cliques that share all but one node); here, the value k indicates the number of nodes that belong to a subgraph. The idea here is that increasing this value means that the detected subgraphs are smaller, but have stronger social ties (in this scenario). It is also possible to define a minimum value for link weight, in the case of a weighted graph; links which have a lower weight than the minimum are discarded, resulting in a similar effect to increasing the value of k.

These communities may overlap and so nodes can be part of several different k-cliques. Each node has a membership number (the amount of communities it is part of); the overlap size is the number of nodes that are part of two communities simultaneously; the community degree refers to the number of overlapping communities, while the size is simply the number of nodes a community has. These values can be used to describe the community structure of a network, so that is is possible to determine the most relevant connections between the nodes.

Meanwhile, Weighted Network Analysis assigns weights to the edges of the social graph, representing the importance of the social relationship between two nodes. By representing the network in a weighted graph, the betweenness values of all of the existing edges are calculated; this value is then divided by the weight of the edge it refers to. The edge with the highest of these values is then discarded from the graph, and the process is repeated, with the new network graph being used for the betweenness calculations.

In order to determine the number of splits to be made, the authors also introduce the *network modularity* concept [11]: this value represents the fraction of graph edges that belong to a certain community, subtracted by the fraction of the edges that would belong to those communities, should such edges be assigned at random. In other words, the higher the modularity value, the tighter social structure a particular network division offers; as such, the division with the highest modularity value will be the one chosen.

These two algorithms are used in conjunction to cluster the nodes together in communities with strong social bonds, meaning that contact opportunities will probably be greater in number within the same community.

On the other hand, the distributed version of this algorithm (DiBuBB) allows each of the network nodes to figure which community they belong to, and compute their own centrality values, without the need of a centralized entity; this means that the distributed version has a greater practical value, since the network nodes may experience frequent disconnections.

This is achieved by using a distributed version of the K-CLIQUE community detection algorithm; after that, a node computes the average *degree* (the number of unique nodes contacted) from previous, fixed-time windows; the authors call this strategy *Cumulative Window*. This degree is then compared to other nodes', when a connection opportunity is present; as such, a node will eventually be aware of its own centrality value in relation to the whole network.

The routing algorithm works in the following manner: any time a node wishes to send a message to another, this message will be forwarded to the other available nodes which own a higher global ranking than the sender; whenever the receiving node belongs to the same community as the receiver, the message will then be forwarded using the same logic, but using the local ranking, instead of the global one, as the value to be compared.

In conclusion, this proposal relies on the reasonable assumption that the more popular members of a social network are better suited to pass information than the least popular ones.

2.3.2 PROPICMAN

The PROPICMAN approach (Probabilistic Routing Protocol for Intermittently Connected Mobile Ad hoc Network) [12] assumes that a node's mobility isn't entirely random; as such, it uses information about the device's owner (*node profile*) to predict a device's future locations, in order to aid in message routing decisions.

This profile consists of several hashed key-value pairs (or *evidence-value*) that represent all the information available about a particular node (such as the name or residence of its owner). The routing algorithm is as follows: when a node wishes to send a message to another, first it sends only the message header to its adjacent neighbors.

This header is built by concatenating all of the evidence/value pairs of the receiving node's profile that the sender has; the sender can also attach *weights* to each of these attributes, which rank the attribute's relevance in regards to the rest of the evidences. The message header also possesses the MAC address of the sender and a message sequence number, in order to avoid duplicates.

Each of the header's receivers then calculates its delivery probability in relation to the message's destination node, by matching the received header's attributes with the information it currently stores about the destination. Then, each of the sender's neighbors will send this header to their own adjacent nodes (the 2-hop nodes, from the original sender's point of view, if they exist); the receivers also calculate their own delivery probability to the destination.

Finally, the sender node will receive all of the 1 and 2-hop node's delivery probabilities; the node with the highest probability will then be sent the message payload (or more than one, if we opt to sacrifice overhead for delay and reliability). This process is repeated until the destination node receives the message. Note that the intermediate nodes store the message in memory, in the event of future contact opportunities.

Interestingly, this protocol possesses some degree of in-built security, which cannot be said about many of the routing protocols in this area. Since the messages are encrypted with the destination's hashed key-value pairs the sender has, only the destination can decrypt the message content. As such, someone who can capture a message will not be able to read the payload, if the message isn't addressed for them.

The destination node's evidences are also somewhat protected, since the intermediate nodes can only acquire some information about the destination if the header's hashed values match their own hashed evidences (as such, only the destination can fully acquire all of the information, since it will be the only one to match all of the hashed attributes).

These security concerns represent an interesting approach to Delay-Tolerant Routing protocols; by increasing the security of the protocols, we can validate implementations of these algorithms in a real world scenario, where security and privacy are major issues in network communications.

2.3.3 PeopleRank

The PeopleRank forwarding scheme [13] is inspired by the well-know Pagerank algorithm used in the Google search engine [14], which rates Web pages by their importance based on the number of pages that link back to them. This approach aims to find the relative importance of the nodes present in an opportunistic network based on their social interactions, and then choosing the highest ranking nodes to forward a message.

These interactions are represented in a social graph, where the vertexes represent the network nodes and the edges denote social relationships between them (such as friendships or common interests). From this graph, we can then rank the nodes based on the number of edges they have. Nodes with a higher ranking possess more (or stronger) social connections with others, and so are probably better suited to carry messages across the network. PeopleRank classifies the nodes in a centralized or a distributed fashion.

The centralized node ranking is calculated as follows: for each node, its PeopleRank value is the sum of the ratio of each of its neighbors' own rank to the number of neighbors they possess. This value is then adjusted by the means of a *damping factor*, which quantifies the social importance a node should have, in relation to others (similar to PageRank's own damping factor, that denotes the probability that a person will stop browsing the Web).

The centralized approach assumes that the social graph is already defined and that the network topology is known in advance; this may be impossible to achieve in practical implementations.

In the distributed version of this algorithm, the nodes will update their social ranking when a connection opportunity arises between them, exchanging information regarding their own PeopleRank values and the number of neighbors they have; as such, two nodes that meet often will quickly increase their PeopleRank values, since they will be constantly updating their information.

This also means that the distributed approach has the advantage of deciding the best routing path dynamically, which is an important feature, considering that the topology can change very quickly in a delay-tolerant environment. If a message is held by one of the connection participants, it will be forwarded to its pair, if it has a higher PeopleRank value than the message holder.

2.3.4 SimBet

The SimBet proposal [15] relies in calculating the *centrality* (or popularity) of a node in order to determine the nodes which are most likely to successfully carry a message through a Delay-Tolerant network; a popular node is expected to have more connection opportunities in the future than one with lower centrality.

To do so, the authors introduce the notion of *ego networks* [16]: networks which consist of a central node (the *ego*) and the nodes which possess a direct path to it (the *alters*) and the links between them. This is done so that a node doesn't need to have knowledge of the whole network structure, which can be infeasible in a Delay-Tolerant scenario.

From this ego network, we can then calculate its *betweenness centrality* (the number of times the ego node belongs to a path between two other nodes that aren't directly connected) by representing the ego network as an adjacency matrix (which we will call A), representing the existence of a direct path between two nodes (1 if that's the case, 0 otherwise). A^2 represents the number of two-hop paths that are available between two nodes; $A^2[1 - A]$ then gives us the amount of two-hop shortest paths for two nodes.

The betweenness centrality is calculated by adding the reciprocal of each value of this last matrix (since the ego network is an undirected graph, we only add the values above the matrix diagonal). This calculation is done every time a node encounters new neighbors,so its centrality value stays updated.

The betweenness centrality value has been shown to possess a strong correlation with its sociocentric counterpart, meaning that a node with a high betweenness is expected to possess many social connections and thus be able to forward messages to their destinations with higher probability.

The other metric involved, called *node similarity*, is simply the number of shared neighbours between two nodes. In the adjacency matrix, this is equivalent to the sum of each of the rows, giving us the similarity between the ego and the alter the row refers to. The adjacency matrix can also be extended by adding columns representing other nodes that are accessible by an alter, so that other forwarding paths to non-neighbor nodes may be considered.

Note that the closeness metric is useless in an ego network model, since all the links from the ego to its alters have a length of 1.

SimBet then uses this values on *SimUtil* and *BetUtil*, which compare two of the nodes in terms of node similarity and betweenness centrality, respectively. These two parameters are then added (possibly with different weights) in the *SimBetUtil* function. SimBetUtil returns a value between 0 and 1, which represents the fitness (or utility) of a node being chosen to carry the message. As such, the node with the highest value has the most social connections.

The SimBet routing algorithm consists of the following steps: when a node encounters a new neighbor, it delivers any messages it may have, plus an encounter request, if the neighbor is their destination. The neighbor then sends its own list of previously nodes, which the first node uses to update its own node similarity and betweenness centrality metrics. Then, the nodes trade their *summary vectors*, which consist of the nodes they are carrying messages to, plus their calculated similarity and betweenness values. For every destination included in the neighbor's summary vector, a node calculates its own SimBet utility value: if this value is higher than the neighbor's, the neighbor will then forward the messages associated with that destination to its pair.

2.3.5 CAR

The CAR (Context-aware Adaptive Routing) protocol [17] supports message delivery synchronously (when there is a connected path between the origin and the destination of the message) or asynchronously (using a store-carry-and-forward approach), using social information as the basis for the routing decisions; nodes do not need to be aware of their own or others' location in order to send messages.

As in the previously mentioned approaches, the main objective is to try to predict which path a message should take in order to reach the receiver. The next node in the path to carry the message is chosen based on the calculated probability it has to meet the destination.

In order to do so, CAR uses *context information*, attributes regarding the node that can be used in routing decisions (like connectivity patterns or availability of resources, for example). This information is used by each node to compute its own *delivery probability* in regards to its neighbors, which is periodically broadcast to other nodes, for them to update their own routing tables; these tables, based on the DSDV (Destination-Sequenced Distance-Vector Routing) scheme, consist of the next hop node, the node with the highest probability, and the recipient node identifiers, in addition to the delivery probabilities of the known destination nodes and the distance to the destination.

After receiving the delivery probability, each of the nodes then regularly updates it, using local prediction techniques to determine its future value. This technique is interesting because a node can predict the future context information of neighbours that haven't connected in some time, which may be a frequent occurrence in Delay-Tolerant networks. Stale entries in the routing tables are periodically removed, if they haven't been updated after a certain amount of time, in order to save memory.

Similarly to the PROPICMAN protocol discussed above, each of the node's attributes is attached to a *weight* that rates the relative importance of each of the attributes. Although the attribute weights are the same for all of the nodes, such values can be dynamically tuned for all of them (*adaptive weighting*), by using three metrics that are able to classify the available context information: *range* (a function based on the possible values an attribute can possess), *predictability* (if the prediction technique is able or not to determine the future value of a given attribute) and *availability* (if current information about a specific attribute can be acquired). The predictability and availability values are binary: 1 means information is predictable or available, 0 otherwise. These three values are then multiplied for each of the existing attributes.

Since the goal of this protocol is to calculate the delivery probability of a message in the future, CAR employs a prediction model to guess the future values of a node's context information, based on the last known state (based on the Kalman filter method [18]); this technique alleviates the need of constant information exchange between neighbors, thus conserving valuable network and memory resources.

In other words, for a particular destination, the best message carrier is the one whose sum of its own weighted attributes has the highest predicted value. The carrier is then chosen to delegate the message asynchronously, until a connection opportunity with the message's destination is available.

2.4 Performance evaluation

Table 2.1 summarizes the main characteristics of the aforementioned protocols: the first row shows which specific scenario the proposal intends to address; next, we can see what kind of context information the protocols use for the routing decisions, and how this information can be acquired (as in a centralized or distributed way).

The fourth row indicates whether the protocol uses or not information from the different network layers (as specified in the OSI Network Model): for example, an algorithm may use information collected from the Application Layer (like the device's battery power, or social data inserted by the device's owner, for example) in addition to the Link or Network Layer information (such as the MAC identifier or IP address of the node) in order to calculate the fitness of a neighbor node being a good message carrier.

The Social clustering entry indicates if the protocols group the Delay-Tolerant Network nodes in an social hierarchy (or communities); if no such distinction is made, the algorithm may be classified as being flat, as discussed earlier in this chapter.

The next row identifies the datasets used for comparison with other routing approaches: BUBBLE used datasets collected from the Haggle Project (`http://www.haggleproject.org/`); more specifically, *Hong Kong*, *Cambridge*, *Infocom05* and *Infocom06*, in addition to *Reality*, gathered from the MIT Reality Mining Project (`http://reality.media.mit.edu/`).

The Infocom05 trace was generated on the IEEE Infocom conference, in Grand Hyatt, Miami. 47 mobile devices were distributed to the participants of the experiment, of which 41 yielded valid contact information, while the data from the remaining ones was discarded, due to hardware failure or loss of the device. This contact information was collected between March 7th and March 10th, 2005.

The Infocom06 dataset was collected during Infocom 2006 in Barcelona, from April 24th to April 26th, with 78 mobile devices (iMotes) being distributed to the attendants of the conference, in addition to 20 stationary devices installed on different points in the area. The 70 participants filled a questionnaire regarding some personal information (namely, residence, nationality and school attended) and then were given the remote device for communication during the conference. During the conference, all the contacts between the nodes were collected.

Meanwhile, the PROPICMAN authors used a custom simulation approach to the performance tests: a number of nodes were placed in a fixed-size area, with the nodes being randomly distributed across the partitions of the modeled space; each node has a different probability of being in one of these partitions and may move freely between them (based on probability).

PeopleRank uses several different datasets for their performance comparisons: *Mobi-Clique*, *SecondLife*, *Infocom06* and *Hope*, each of these possessing mobility patterns as well as social contact information.

Simbet used exclusively the *MIT Reality* dataset, mentioned above. From this information, the authors found that the calculated egocentric betweenness values followed closely the social network information provided by the participants in this project's data collection.

Finally, the CAR protocol authors, like PROPICMAN, used their own simulation models to analyze their algorithm's efficiency: network nodes were introduced in fixed spaces; these nodes followed a *Community-based* mobility model; additionally, the authors created a social network based on the *Caveman model*.

Each of the modeled spaces was split into fixed-size grids, with each social community being placed in one of them, with each host employing a Random Waypoint model to move to adjacent the sections.

The table also displays the performance comparison versus PRoPHET, in terms of the widely used network metrics: message delivery ratio, number of messages and delivery delay (when such information is available).

For each of the protocols, the average of the value was calculated for all of the datasets used in the tests. This was done so that we can have a rough estimate of the performance advantages of using social-aware protocols; since these values are obtained through graphic observation, we cannot accurately calculate these values. (In the case of the PeopleRank protocol, there wasn't any data regarding the PRoPHET protocol, so we cannot present this information).

In order to ensure more reliable information, some of the protocols employed some kind of statistical measures: namely, the PROPICMAN provides a confidence level of 95% and a standard deviation value of 0.975 milliseconds on their delay performance tests. On the other hand, CAR's performance data possesses a margin of error of 5%.

The table shows us that BUBBLE achieves nearly the same delivery ratio than PRoPHET, while reducing the number of messages in the network by roughly 50%; while PROPICMAN

Protocol	**BUBBLE**	**PROPICMAN**	**PeopleRank**	**SimBet**	**CAR**
Context	PSN	General DTN	PSN	General DTN	General DTN
Context Information	Clustered node ranking	Node profiling	Social Graph Analysis	Ego network metrics	Node attribute prediction
Context Calculation	Centralized or distributed	Distributed	Centralized or distributed	Distributed	Distributed
Cross Layer	No	Yes (Layers 2, 7)	Yes (Layers 2, 7)	No	Yes (Layers 2, 3, 7)
Social Clustering	Hierarchical	Flat	Flat	Flat	Flat
Datasets	Haggle, MIT Reality Mining	Custom Simulation	Multiple sources	MIT Reality Mining	Custom Simulation
Delivery Ratio (%)	[90-100]	Unavailable	Unavailable	[95-100]	[100-110]
Overhead (%)	[45-55]	[95-100]	Unavailable	[95-100]	[50-55]
Delivery Delay (%)	Unavailable	[60-70]	Unavailable	[80-90]	[90-100]
Statistics	Unavailable	95% confidence level, 0.975 ms standard deviation	Unavailable	Unavailable	5% margin of error

Table 2.1: Comparison of social-aware routing protocols

possesses a similar message overhead compared to PRoPHET, it manages to reduce significantly the delay a message suffers from traveling through the network.

As stated above, performance information comparing PeopleRank to PRoPHET does not exist, so we cannot make any assumptions on it. Meanwhile, SimBet performs similarly in terms of delivery ratio and amount of messages passed on the network, while CAR achieves roughly the same delivery ratio and delay as PRoPHET, while only spending roughly half of network resources.

We can then observe that social-aware protocols are able achieve a satisfactory performance when compared to a widely referenced Delay-Tolerant forwarding algorithm, in terms of the usual protocol metrics used in this specific environment; therefore, one can assume that exploring social information to aid in routing decisions is indeed an advantageous approach, while also validating our motivation on researching this subject.

2.5 Conclusion

In this chapter, an extended classification of Delay-Tolerant routing protocols was presented, with the aim of helping us understand the current general strategy these proposals follow. The Epidemic and the PRoPHET protocols were also briefly described, since these algorithms are widely referenced in this area and thus provide a reasonable basis for comparison with other Delay-Tolerant routing proposals.

We then characterize some of the recent routing protocols, with information regarding the routing algorithm itself, as well as other attributes that seem relevant, such as the kind of context information used, what network layers are used and the datasets used for comparison and those results.

After having analyzed some of the most widely referenced protocols in this area, as well as some of the state-of-the-art social-aware proposals for Delay-Tolerant routing, we can now conclude that algorithms which are able to use this kind of information about the nodes can, in fact, be useful in terms of calculating the most adequate path a message should take, usually presenting some kind of performance increase in regards to more oblivious schemes, whether in terms of message delivery ratio or delay, or in the amount of resources that the nodes have to use.

Chapter 3

Opportunistic Dataset Comparison

Opportunistic networking differs from more conventional architectures by the lack of existing network infrastructure, which can cause intermittent connectivity or increased communication delay between nodes. From a message routing perspective, solving these problems require a different set of techniques than those used in more traditional network schemes.

Forwarding algorithms in this area usually aim to improve performance metrics such as the ratio of successfully delivered messages, while trying to decrease the time a message takes to reach the destination node, as well as the number of copies replicated throughout the network.

A common approach used for testing the performance of opportunistic protocols relies on existing opportunistic contact traces. These datasets are widely available on the Internet, and provide a convenient way of simulating realistic usage scenarios. As such, studying the contact patterns between nodes can lead to useful observations to take into account on future experiments.

This chapter presents the results of a study on four different datasets [19]. First, we describe the main characteristics of each trace. Then, we propose a graphical representation of the contact behavior for each pair of nodes.

The next step was to perform an analysis in terms of the distribution of connectivity among nodes, having found that the contacts follow a roughly lognormal distribution and noting that a small group of nodes is usually much more popular than the rest. Lastly, we have made a temporal analysis over the duration of each collection experiment. It was noticeable that individual nodes have very similar contact patterns over time, as well as revealing some cyclic variation over time (namely over weekends).

3.1 Introduction

Opportunistic networks are usually characterized by the lack of conventional network infrastructure. As such, end-to-end paths between two nodes may not be always available, while also being prone to increased delay and error rates during data transfer, among other challenges. These problems motivate the need to design routing algorithms without guarantees of continuous connectivity, since traditional forwarding proposals do not usually take these constraints into account.

Several opportunistic routing protocols use *context information* (i.e., information that a node can acquire of its surrounding environment) in order to determine which of the available neighbors has the best chance of delivering a message to a destination node. For example, the PRoPHET protocol [20] uses the past contact history of a node to predict future connection opportunities, while other approaches explore social relationships between nodes in the network, for example. In recent years, this area has been subject of extensive research among the academic community, with a great number of proposals being made in regards to routing protocols and data dissemination strategies [21] [22] [23].

The performance of a routing protocol is an extremely important issue in these environments, as suggested by the challenges mentioned above. Some of the most important metrics in this subject include the delivery ratio, delay and message overhead.

One of the most popular tools to test the performance of opportunistic routing algorithms is the use of existing contact datasets, which consist of a record of all of the contacts made between the participants, during a data collection experiment. This information is then used to design more efficient forwarding schemes, based on real usage scenarios.

Contact traces are available for widely different scenarios, ranging from groups of university students, to bus systems in a city, among others. Nevertheless, several questions arise when comparing different datasets. Do the traces have any similar statistical characteristics? Do contact patterns between individual participants show significant variation? Would it be interesting to collect other types of information? Some of these concerns can be considered while planning future dataset collection experiments.

In order to test these kinds of proposal in terms of performance, several datasets are available online, which contain contact information between nodes collected during different experiments. These datasets can then be used to simulate a real-world application of the proposed algorithms.

This means that performing an analysis of existing opportunistic connection traces can be useful to determine what kind of connection patterns exist between the nodes, which can then be possibly used by new opportunistic network routing strategies.

The main objective of this work is to provide a comparison of different opportunistic contact traces, in order to determine what kind of information can be extracted from them, that is not immediately noticeable. Many of the datasets on this subject do not seem to justify some traits which characterize the experiments in which the traces were collected (for example, the number of nodes or the duration of the experiment); on the other hand, some amount of statistical information also could potentially be useful to plan future experiments, in order to produce a dataset with more relevant information.

In order to answer some of these questions, we apply a methodology consisting of statistical analysis regarding different subjects, such as data distribution, correlation between different metrics, and temporal analysis. We also present visual representations of the datasets in question, which can be used to summarize the interactions among the nodes in the network.[1]

The remainder of this chapter is organized as follows. Section 3.2 describes some recent proposals that were made related to this subject. Section 3.3 presents a description of the different datasets used on this work. Section 3.4 presents visual representations of each node's contact patterns. Section 3.5 explores the statistical distribution of the data. Section 3.6 provides an analysis of the data over the duration of each experiment. Section 3.7 presents our conclusions on the work that was made, in addition to proposing future work on this subject.

3.2 Related work

As seen before, the performance of Delay-Tolerant routing algorithms is usually tested by means of simulations. By using real-world connection traces, it is possible to simulate a

[1]Due to space constraints, it is not possible to present all of the generated figures on this document. Additional images are available at `http://marco.uminho.pt/projectos/oppdatasets/home/`.

realistic opportunistic network scenario. Here, some recent dataset analysis proposals are discussed.

Belblidia et al. [24] propose the *surround indicator* metric, which describes the spacial dimension of a contact in wireless networks; in other words, it indicates the density of nearby nodes in the network. This metric could be used in conjunction to the more widely used temporal dimension metric as information for opportunistic routing protocols.

Xu et al. [25] present a social community detection strategy for opportunistic datasets; this algorithm groups the nodes in different communities (or clusters), based on the duration and frequency of contacts a node has with others. Community information can then be used by routing protocols to forward messages more efficiently (for example, two nodes from the same community may have a higher chance of meeting again in the near future than two nodes of different communities). This algorithm was proven, in fact, to detect the social structure, as well as changes in the network structure over time, which can be frequent when considering opportunistic networks.

Yoneki [26] also uses different community detection algorithms, namely K-CLIQUE, Weighted Network Analysis, and Fielder Clustering to visually present the existing community structure of the datasets [27]. This work demonstrates the use of distributed and centralized techniques to group the participating nodes in social communities based on their contact patterns. Besides showing both flat and hierarchical community structure for some connection traces, the authors also exhibit how the hub nodes (the most influential nodes on the network) change position over a period of time.

Chen et al. [28] propose an algorithm to recover censored contacts (contacts that start during the measured time but end after the end of the measurement); these contacts were shown to imply a skewed statistical distribution if ignored. Using the recovered contact information, the authors then provide a thorough statistical and graphical analysis of different opportunistic network traces, noting the existence of strong self-similarity (meaning that a subset of the data possesses the same statistical properties as the whole trace).

Ristanovic et al. [29] observe that opportunistic contact simulations may not adequately reflect the characteristics found in datasets with more realistic scenarios. As such, the authors develop a data collection experiment with real users and compare the collected trace with simulated contact information. One of the main conclusions is that simulations tend to be more optimistic than real opportunistic network applications, regarding performance metrics such as delivery ratio (percentage of messages that successfully reach the destination) and the delay (the time it takes for a message to reach the final node); one fact that may justify this conclusions is the assumption of infinite cache sizes in the simulation experiments. The authors also note that using a network backbone can result in increased performance.

3.3 Dataset description

This section describes the four studied datasets that illustrate the contact opportunities, usually between users of mobile devices, during different data collection experiments, which can prove useful in finding relevant metrics applicable to opportunistic routing protocols. The datasets used in this work are: *unimi/pmtr* [30], collected at the University of Milano, Italy; *upmc/rollernet* [31], obtained from a rollerblade tour in Paris, France; *st_andrews/sassy* [32], from Scotland; and *upmc/content* [33], collected around Cambridge, England.

These datasets were obtained from the CRAWDAD repository[2], which provides a vast collection of wireless network traces for analysis, as well as providing a number of tools to process and analyze the offered traces. This data consists of experiments made on diverse kinds of scenarios, and provide useful information regarding the contact patterns between the participants.

The collected datasets are presented in tabular form, each row representing one contact between two nodes. The first two columns denote the identifier of the nodes involved in the contact, while the following two values represent the beginning and ending of a contact opportunity, respectively. The *upmc/content* dataset splits the nodes into separate files, so it only possesses a single node identifier in each file (that is, the node contacted by n, in the file $n.dat$).

Some of the datasets also have two more columns: the fifth one indicates the number of previous contacts made between those two nodes, while the sixth column presents the time interval between the end of the last contact and the beginning of the current contact between the nodes. Appendix A presents some examples of the format used in the datasets.

The *st_andrews/sassy* dataset provides slightly more information regarding the contact opportunities (in addition to the aforementioned data). The fifth column indicates the timestamp of the information upload for that contact, the sixth value is the signal

[2] `http://crawdad.cs.dartmouth.edu/`

strength of the device (*RSSI*) and the last column presents the maximum possible difference between the upload timestamp and the actual start of the contact opportunity (due to unsynchronized clocks after a device resets or battery failure). *st_andrews/sassy* also presents an additional file illustrating the participants' Facebook social information (named *self-reported social network*, as opposed to the *detected social network* that lists the physical contacts between the mobile devices). In other words, it simply lists the node pairs that are friends on Facebook. Even though there seems to be a good correlation between the physical connections and the social network, there isn't enough information to allow us to perform a more complete analysis. As such, we believe that it would be interesting to have access to other kinds of social data, such as the number or the date and time of the contacts between the pairs of nodes featured in the data collection experiment.

Table 3.1 displays the most important information about the observed datasets. The first row shows the name of the dataset. The second row indicates the scenario in which the experiment was conducted. The third row indicates the number of participants[3] involved in each experiment. The fourth row shows the total duration of the experiment, while the fifth row denotes the total number of connections in each of the traces (which is the same as the number of entries present in the file). The sixth row presents the average pairwise (that is, each different pair of nodes) of the number of contacts a node establishes with others. The seventh row shows us the average pairwise time duration of connections between two nodes, and the eight row the average interval between contacts for each pair of nodes.The ninth row denotes the average degree of each node (in other words, the average number of different nodes that have contacts with a single node). The last row displays the number of articles that reference the dataset (this information was collected from the CiteULike[4] scientific reference service, by searching for articles that are tagged with the trace's name). We believe that these traces provide sufficient variation in terms of sample size, duration and number of contacts to be able to provide a meaningful comparison between them.

Moreover, we have used the ONE simulator [34] to generate 4 additional datasets, with the same number of nodes and duration as each of the traces mentioned above, using the Random Waypoint movement model. We number these simulations from 1 to 4, following the same order as the aforementioned datasets. This will allow us to make more relevant

[3]We do not consider contacts with nodes that don't belong to the experiment, since the generated heatmaps would become too big and sparse for visual interpretation.

[4]`http://www.citeulike.org/`

Name	*unimi/pmtr*	*upmc/rollernet*	*st_andrews/sassy*	*upmc/content*
Scenario	University of Milano mobility traces	Rollerblade tour contacts	St. Andrews University contacts	Cambridge City traces
Number of nodes	44	62	27	54
Duration	19 days	3 hours	79 days	54 days
Number of contacts	11895	132511	112265	40164
Average pairwise contacts	10.7727	31.8065	319.8405	7.5982
Average pairwise duration (seconds)	4905.3	205.1988	1458.4	8064.2
Average pairwise interval (seconds)	393022	6330	943376	182907
Average node degree	27.5909	60	11.4815	23.8519
Number of references	3	11	6	13

Table 3.1: Comparison of opportunistic contact datasets

comparisons between real and simulated traces.

It is only possible to present information regarding the connection times of the nodes, since that is the only type of data we have access to. We believe that additional information (such as connection bandwidth or location data) could be explored to provide a more thorough analysis.

3.4 Dataset visualization

Based on the work of Phanse et al. [35], we rank the network nodes of each dataset in regards to their degree (the number of different nodes contacted) and connection time (the total connection time with other nodes). Figure 3.1 presents the node rankings for *upmc/rollernet*.

We observed that that the degree distribution of the nodes is approximately linear, while the connection time distribution is heavy-tailed across all of the datasets. This implies that

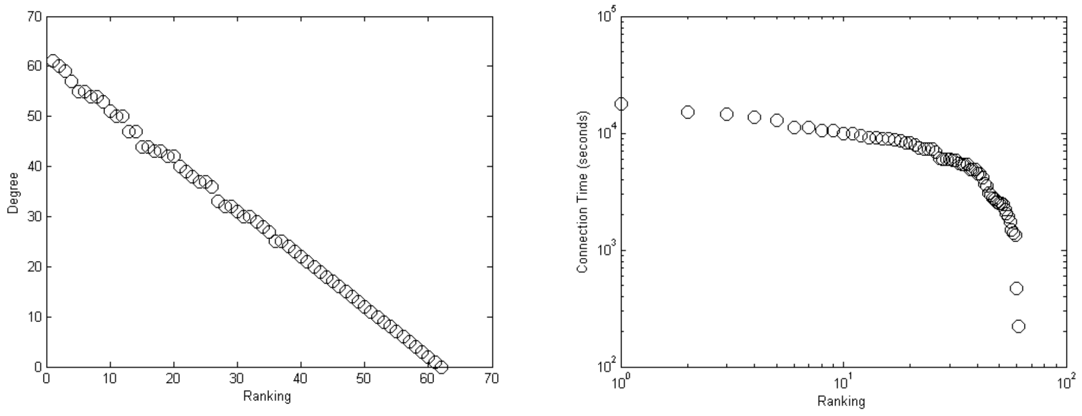


Figure 3.1: Degree and connection time node ranking for *upmc/rollernet*.

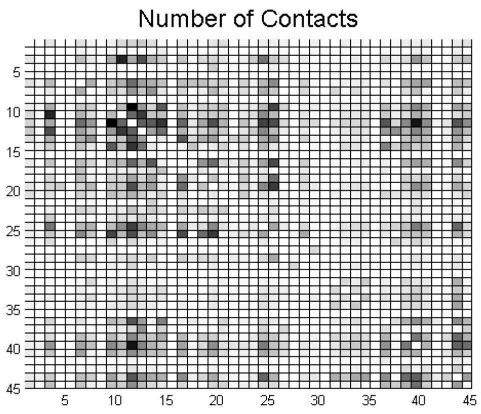
there are generally a few nodes that are much more popular than the remaining ones. This assumption seems reasonable, since in real life people have different amounts of popularity and may spend more or less time communicating with their peers, depending on the social bond between them. All of the simulations seem to generate a perfectly linear degree distribution, given enough time (such was not the case for the *upmc/rollernet* experiment, which only had a 3 hour duration); in terms of the connection time distribution, the results were similar to the real traces.

Additionally, for each of the referenced datasets, three additional n-by-n matrices were generated (n being the number of nodes in the dataset), which contain the total number of connections between nodes i and j, and the total connection time between two nodes, and the accumulated time interval between contacts, respectively. These matrices are symmetric because the we consider that connections between nodes are bidirectional. The matrices refer to the number of connections, accumulated contact duration and accumulated interval between contacts for each node pair, for the four datasets presented.

From these matrices graphical heatmaps were generated, with black and white squares being the ones with the most and the least intensity, respectively. These images provide a graphical representation of the connection patterns of the network, and can be useful to provide visual information regarding the datasets, namely the number of nodes (related to the number of squares in the image) or the distribution of connectivity among nodes (a higher image contrast implies a bigger difference).

Figure 3.2 shows the generated heatmaps for the number of contacts for each pair of nodes in *unimi/pmtr* and its simulated counterpart. From these figures, we observed that the connectivity rate is more evenly distributed among the first two datasets than in the last two, since the heatmaps from the last two datasets are much sparser (for example, *upmc/content* possesses very few contacts for nodes with IDs greater than 36). It is clearly noticeable that the simulations show much more homogeneous interactions between pairs of nodes than the real datasets. We can also deduce that the connection time may be evenly distributed among all of the connections, since the connection time heatmaps show similar patterns in regards to the heatmaps representing the number of contacts.

Figure 3.3 shows a heatmap representing the self-reported social network for the dataset st_andrews/sassy. When compared to the number of contacts heatmap for the same data, it is observable that the detected social network has more information than the self-reported one: that is because the self-reported social network only gives binary information regarding the relationship between two nodes. Also, these two sets of information do not appear to present a strong correlation, judging by the contact heatmap patterns. Given kind of data provided it may not be possible to further elaborate on this case.



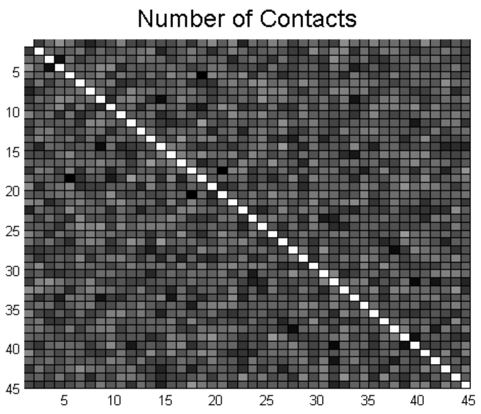


Figure 3.2: Heatmaps for the number of contacts on the *unimi/pmtr* dataset and its respective simulation.

The image energy (average of the squared values of the pixels) corresponding to the heatmaps is presented in table 3.2.

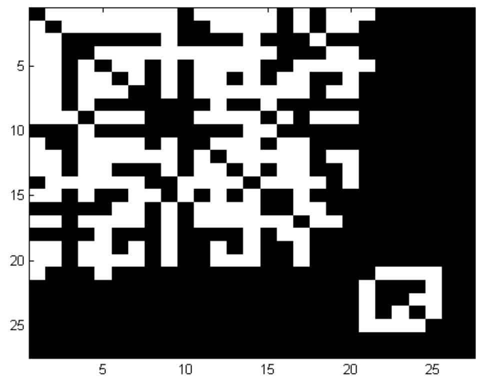


Figure 3.3: st_andrews/sassy self-reported social network.

Table 3.2: Image energy for the heatmaps.

Name	*unimi/pmtr*	*upmc/rollernet*	*st_andrews/sassy*	*upmc/content*
Number of contacts heatmap energy	1.3789	2.8743	0.3205	1.1438
Contact time heatmap energy	0.6035	0.7706	0.3243	1.0693

3.5 Statistical distribution

We have generated histograms for the number of contacts on each data trace. The histograms visually represent the distribution of the data; for this case, the x axis represents the number of contacts between two nodes, and the y axis shows the number of occurrences for that number of contacts. The histograms clearly attest to the sparsity of contacts; it is also observable that there are much more nodes that possess few contacts than those that have many contacts, for all of the datasets. Again, this conclusion seems reasonable, since each node probably contacts a small number of its peers much more frequently than others. Figure 3.4 shows the contact histogram for the *unimi/pmtr* trace.

The probability distribution can be defined as a function that denotes the probability of occurrence of a certain variable. This can be useful to generally describe the expected distribution of values present in a dataset.

For each of the datasets (using both the number of contacts and the total contact time, for each node), we have performed the one-sample Kolmogorov-Smirnov and the Lilliefors

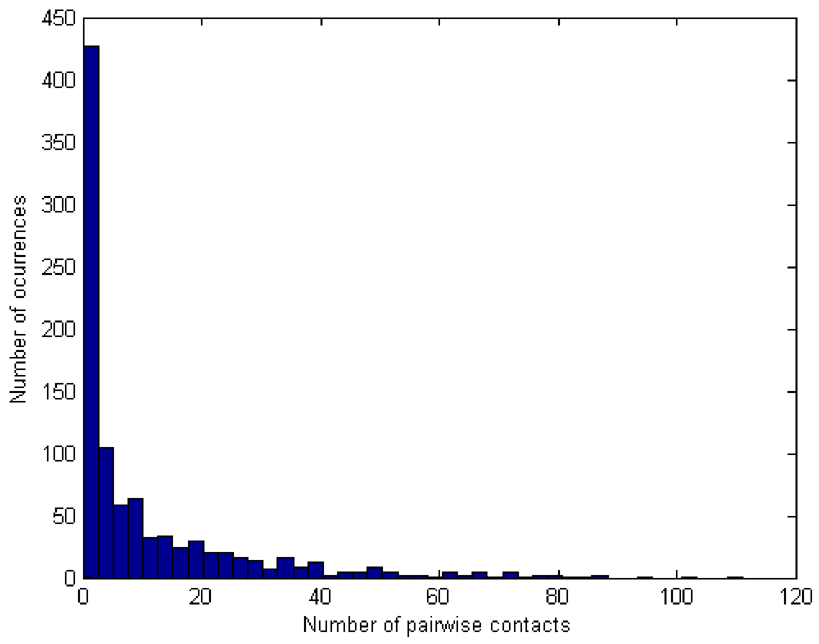


Figure 3.4: *unimi/pmtr* contact histogram.

tests to determine the validity of the null hypothesis that the data follows a standard normal distribution, also using the logarithms of the values to test for fitness of a lognormal distribution, which was not the case for both evaluations (with 5% significance level). We have also used the two sample variant of the Kolmogorov-Smirnov test to compare the datasets with different values randomly generated from a continuous distribution (normal, lognormal, Weibull and exponential) to find if these two different kinds of data followed the same distribution; also, in this case, the test rejected the null hypothesis.

Finally, we have generated probability plots to visually compare the data to a number of distributions (normal, lognormal, exponential, extreme value, Rayleigh and Weibull). These plots draw a reference line to aid in determining the closeness of the data to one of the mentioned distributions; if the data points are close to the line, the data can be assumed to fit appropriately the distribution in question. From the generated plots, we found that the data follows more closely the lognormal distribution than others, which may be assumed to fit the actual data reasonably well. This is true for the 4 different datasets. Similar results were observed for the simulations (with the exception of the *upmc/rollernet* simulation). This distribution assumes that the natural logarithms of the observed values follow a normal distribution. Image 3.5 shows the probability plots of the *st_andrews/sassy* dataset for the lognormal distribution.

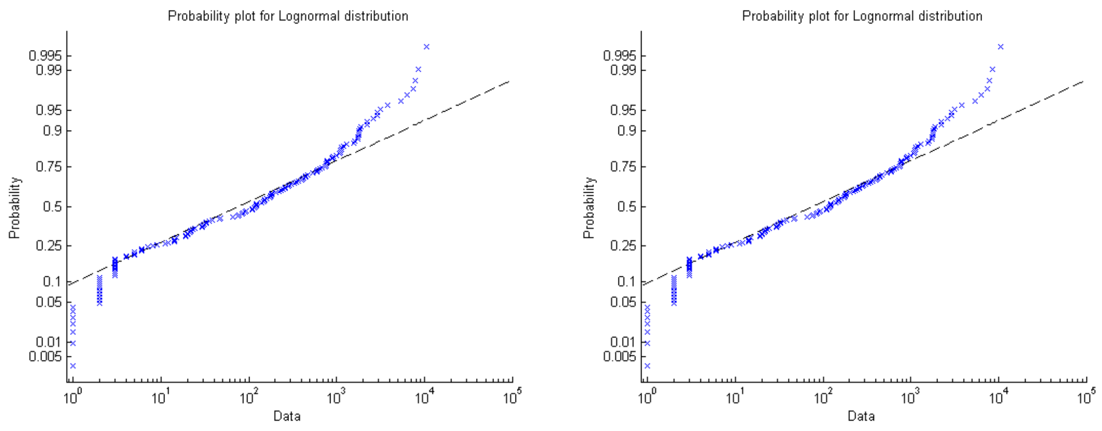


Figure 3.5: Probability plots for *st_andrews/sassy*.

Moreover, it may also be interesting to inspect the distribution of values among individual nodes, rather than for each pair of nodes. Figure 3.6 exhibits the box plot graph for *unimi/pmtr* and the respective simulation. Each box denotes the distribution of contacts for an individual node summarizing the following statistics: the height of the box ranges from the 25 to the 75th percentiles (splitting the data through the lower and the upper quantiles, respectively); the center marker denotes the median (the middle value from an ordered set of observations), and the whiskers indicate the range of values that aren't considered outliers. The data outliers are marked individually, above the boxes.

We can see *unimi/pmtr* has a few nodes that possess much more contacts than others. It is also clearly noted that some traces possess nodes with a similar contact distribution, like *upmc/rollernet*, while others have nodes with very few contacts (as in *st_andrews/sassy*), even of there are a number of outliers that are located well beyond the expected range of values. Lastly, for *upmc/content*, the last nodes possess almost no contact with others. The individual nodes in each the 4 simulation experiments show little variation between them in terms of the number of contacts.

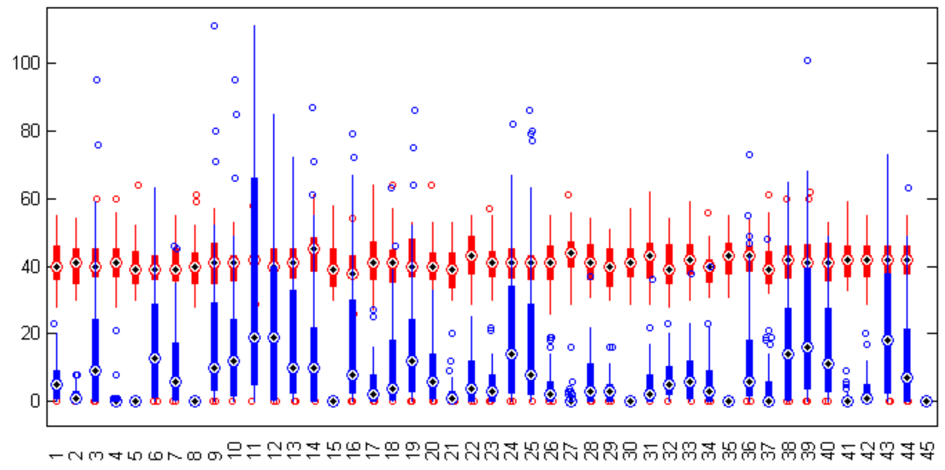


Figure 3.6: Box plot for *unimi/pmtr* (blue) and its simulation (red).

3.5.1 Correlation coefficient

The *Pearson product-moment correlation coefficient* measures the correlation between two sets of data, in order to identify and rate the magnitude of statistical relationship between two variables. In other words, it assigns a value that represents the linear dependence between two variables (1 if the variables show a perfect correlation, -1 if there is a perfectly inverse correlation, and 0 if there is no relation whatsoever). Table 3.3 presents the correlation coefficients for the data acquired from the datasets, based on the statistical information mentioned above.

It was observed that the number of contacts and the duration of those contacts has a generally strong correlation, as expected (with the exception of *unimi/pmtr*), so we can conclude that these values may be related in some way. The same cannot be said for the contact duration and the time between contacts, so it is not possible to make any

Table 3.3: Correlation coefficients for the datasets.

Name	*unimi/pmtr*	*upmc/rollernet*	*st_andrews/sassy*	*upmc/content*
Number of contacts vs Contact time	0.5912	0.8786	1	0.9110
Contact time vs Time interval	0.4467	0.2772	0.5297	0.6312

assumptions regarding the relationship between these two sets of data.

3.6 Time series analysis

A time series is an ordered sequence of values measured in successive times; these values are then grouped in uniform time intervals. By arranging the available data in time series, one can have a better understanding of the contact patterns throughout the total duration of the experiments that resulted in these datasets. Figure 3.7 presents a graphical representation of the number of contacts during each time interval, for the *unimi/pmtr* and *st_andrews/sassy* traces, and their respective simulations with the same duration and number of nodes.

Since each dataset was collected during experiments with different time durations, the time interval was adjusted in order to scale the graphics along the x-axis, enabling a more complete view of the data (the intervals are 12 hours for *unimi/pmtr*, 5 minutes for *upmc/rollernet*, 24 hours for *st_andrews/sassy* and 12 hours for *upmc/content*). The blue line indicates the actual number of contacts during a time interval, while the black line represents a simple moving average of the last 7 values, in order to minimize the short-term fluctuations and to simplify the visual analysis of the general data trends. The simulated dataset results are marked in red.

The *unimi/pmtr* experiment figure clearly demonstrates that the number of contacts drops to almost zero on two days, which were observed to correspond to the weekend periods. The *upmc/rollernet* data shows a noticeable increase in contacts during the first hour of the experiment; the number of contacts then starts to steadily decrease after that point.

st_andrews/sassy possesses a significant variation over time, regarding the contacts made; nonetheless, it is also clearly observable that the contacts drop significantly during the weekends. The contacts also practically cease after about 45 days, which can probably be explained by the beginning of the Spring vacation in the St. Andrews University. The number of contacts in the *upmc/content* dataset also significantly drops starting at the middle of November.

Since additional information about the experiments is not available, we cannot guarantee that these assumptions are correct; on the other hand, the idea that people have less contacts during weekends and holidays seems to make sense, since they probably spend less time with their colleagues or friends in their usual workplace environments.

The time series also presents a noticeable descending trend for all of the traces, meaning that the total number of contacts by interval usually decreases over time; this may be explained by the fact that the attendants usually become less and less interested in participating on the experiment. Also, there seems to exist a certain degree of seasonality in some of the traces (most notably on the decrease on contacts during weekends), even though the data was collected during a relatively short period.

The simulations present much more stable time series; the data varies less between different points in time, and the range of observed values is much shorter when compared to the real traces. Moreover, the first 2 simulations show a growing trend, while the other two do not present a significant trend at all. Furthermore, there are no clearly identifiable signs of seasonal patterns, as observed in some of the real traces.

Figure 3.8 shows the correlogram for *unimi/pmtr*. It is possible to clearly identify a high degree of similarity for successive time intervals, especially during the final half of the experiments. These plots are also useful for determining the randomness of a data; the fact that some of the observed autocorrelation values aren't close to zero leads us to believe that the data in question is not completely random (following the same logic as the correlation coefficient explained above).

We have also made time series plots for individual nodes; for each dataset, 3 different nodes were selected : one with high, other with medium and another with a low number of contacts.[5] The node numbers chosen were 39, 19 and 42 for *unimi/pmtr*, 45, 21 and 27 for *upmc/rollernet*, 9, 7 and 20 for *st_andrews/sassy*, and finally 19,30 and 54 for *upmc/content*, listed from highest to lowest number of contacts.

In comparison to the plots for all the participating nodes, presented above, we have observed that the visual shape of the graphs bear a strong resemblance between them. This means that the contacts patterns between nodes are usually similar, independently of the popularity of the node in question. Again, the correlation coefficients can probably reinforce this observation. It was observed that the correlation coefficients for some of these pairs is close to 1, hinting that the contact patterns these nodes possess may be related. However, the nodes with low number of contacts might not have enough data to be statistically

[5]This approach was not used on the simulations, since the individual nodes are extremely similar in terms of number of contacts.

significant.

Finally, for each trace, an additional plot was generated, but this time the y-axis has the degree (number of different nodes contacted during a period) of a singular node, instead of the number of contacts that node has made with others. In this case, it is possible to conclude that the degree of the node can have much variation for a sufficiently small time window. Naturally, the degree plots share a similar pattern to the previous number of contacts graphics referenced above; however, having a large number of contacts doesn't necessarily imply a great number of different nodes were contacted (this is especially noticeable on the *st_andrews/sassy* trace, where higher nodes have thousands of contacts, but with barely over a dozen of other nodes). The simulated experiments tend to have individual nodes with lesser variation of degree, both over the course of time and between individual nodes.

3.7 Conclusions and Future Work

This chapter analyzes four different datasets collected during opportunistic network experiments. It summarizes the main characteristics of the experiments and presents some insights about them. It also shows visual representations of the contact patterns between the various devices present in the experiments. One of the main objectives of this work is to identify some characteristics that can be useful to consider during the planning of a new data collection experiment.

After describing the main characteristics of each trace, the next step was to calculate some average metrics regarding contact patterns contained in each dataset, namely the number of contacts, the duration of contacts and the time interval between them. Then, we ranked the nodes in terms of degree and connection time, based on related work that has been done previously; along with the generated histograms for the number of contacts, we were able to make some observations regarding the social behavior of the nodes; namely, the existence of a few nodes much more popular than the majority.

The heatmaps for the number of contacts and their duration for each pair of nodes visually summarizes the distribution of contacts, while also attesting to the sparsity of contacts.

Afterwards, some statistical analysis has been done, in terms of correlation between these metrics (a strong correlation was found between the number of contacts and the total duration of contacts between two nodes), as well as the data distribution that seems to have the best fit for the contact information, which was found to be the lognormal distribution.

Additionally, some time series analysis has been made; the plots presented can give us some information about how the contact patterns seem to behave over the duration of the data collection experiments. It was noted that the contacts share a descending trend for all traces, while in some cases seasonal patterns were clearly observable (the autocorrelation plots also seem to imply that there is a periodic pattern over time).

One of the main conclusions reached during this work is that the scenarios in which these experiments have been done seem to be one of the determining factors regarding the contact behavior of the majority of the nodes. In other words, the overall population of participants seems to be somewhat homogeneous in terms of contacts made over the duration of the data collection.

On the other hand, it has proven difficult to determine other metrics that differentiate these distinct traces; this challenge can be explained by the lack of other kinds of data associated with network connections; for example, information related to the movement patterns or the energy levels of the mobile devices could prove useful for a more detailed analysis.

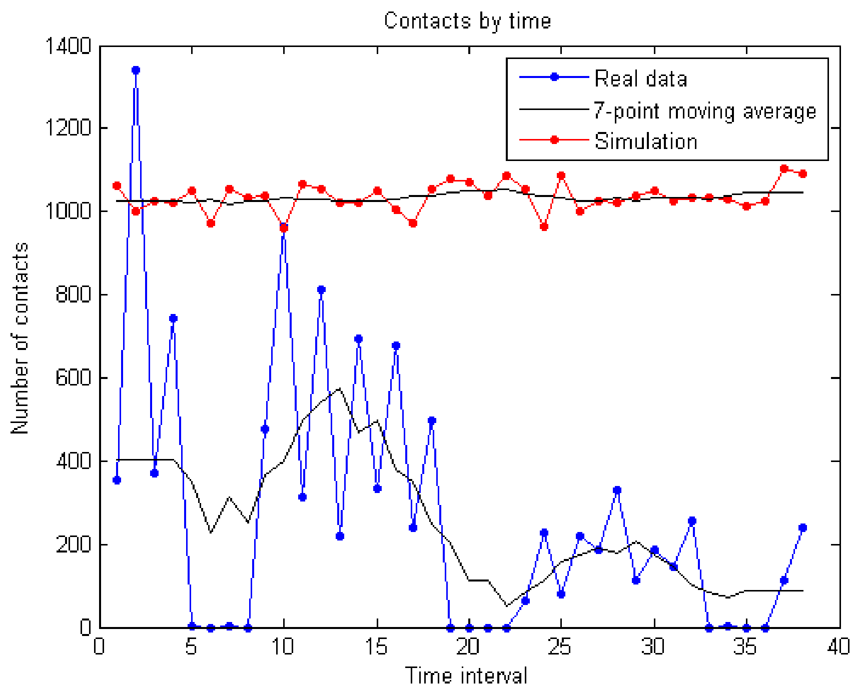
Another difficulty relates to the widely different variables present in each dataset, namely the number of nodes, total duration of the experiments, and number of contacts, which hamper the ability to find relevant metrics which differentiate each collection of data. Therefore, it would be interesting to analyze other real datasets that were collected in a similar time frame, which would make statistical comparisons much more relevant.

Even so, this work has managed to determine some meaningful observations related to the datasets. The participants usually possess a narrow group of other nodes which they frequently contact. The number of contacts and the duration of these contacts are probably related in some way; the same observation also seems to holds true for the amount of contacts between some of the individual nodes.

The same observation also seems to be valid in terms of the duration of the experiments, which again appear to be strongly related to the specific experiment scenario in question: some of the datasets provide little to no relevant information past a certain point in time; on the other hand, it is possible to identify seasonal patterns for the time series graphs, which can be used to determine a sufficient amount of time periods for future data collection experiments after which the amount of information provided would be somewhat redundant.

The inclusion of simulated contact traces allows us to verify that the contact behavior of their nodes is much less diverse than their real counterparts. One possible explanation for this fact is that a simulation only allows us to adjust a strict number of parameters; naturally, some other possible social factors cannot be accurately portrayed on these environments.

Future work includes expanding our analysis to a greater number of datasets, as well as studying additional ways of visually representing the data contained in them, such as community detection techniques. Another interesting approach would be to continue analyzing contact traces using opportunistic network simulators, to better identify the differences between a computer-generated datasets and real ones, collected from human participants.



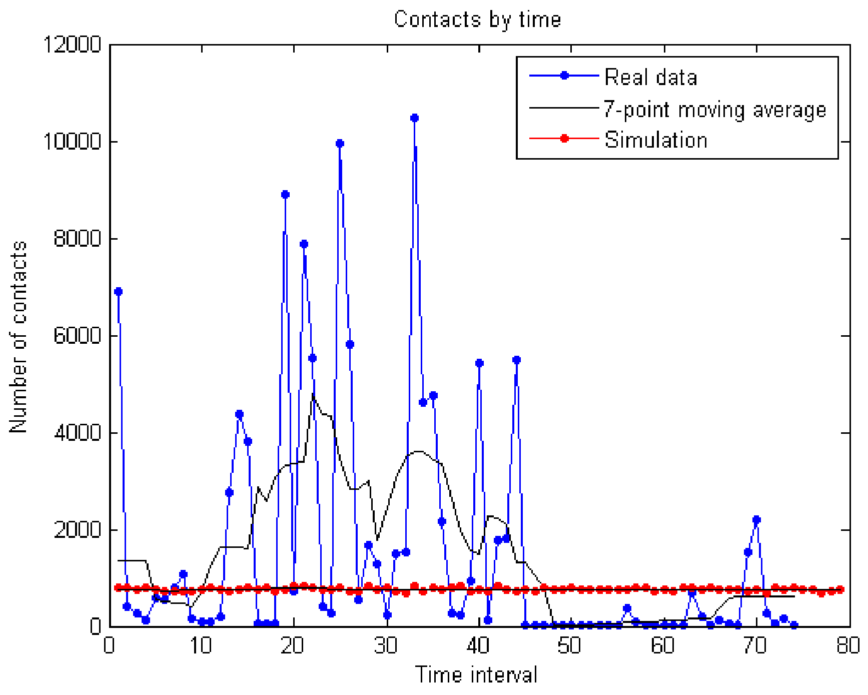


Figure 3.7: Time series plots for *unimi/pmtr* and *st_andrews/sassy*.

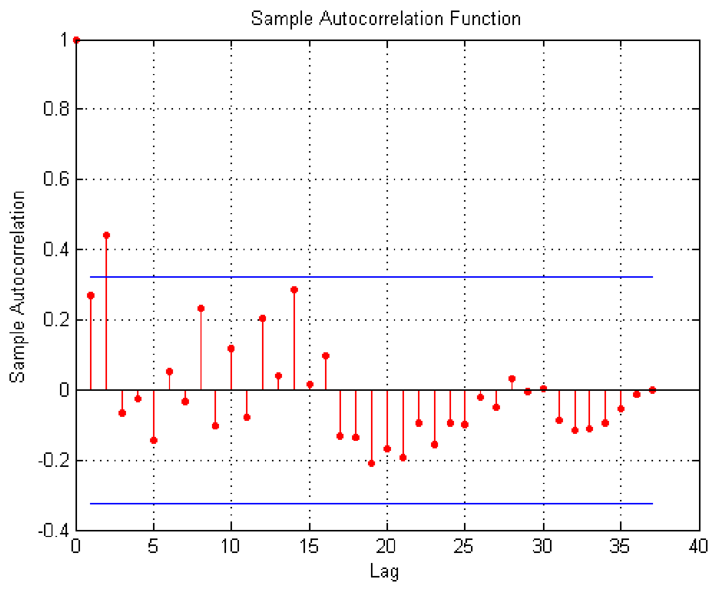


Figure 3.8: Autocorrelation plot for the number of contacts on *unimi/pmtr*.

Chapter 4

Dynamic Dataset Evaluation

Delay-tolerant networks are characterized by intermittent connectivity and increased delay and error rates, when compared to more conventional networking schemes. Due to the lack of dedicated network equipment, the nodes themselves have the responsibility to decide to whose neighbors shall it forward a message to. This scenario implies additional challenges, since the network topology tends to be highly dynamic, and the nodes possess limited resources, such as energy and buffer sizes.

Because the operation of forwarding messages can be relatively costly to the mobile nodes, it is important to restrict the number of messages that a node replicates to its neighbors. As such, several routing algorithms have been researched, with many of them using context information to determine which nodes possess a higher probability of delivering a message to a specific receiver.

Performance of opportunistic routing algorithms is usually tested via simulation of a real-world scenario. To this end, several opportunistic connection datasets are available online, collected in widely varied circumstances. Because of the aforementioned problems associated with this kind of networking scheme, the main goal of this work consists using realistic connection traces in order to perform a series of simulations and analyze the impact of different network parameters in Delay-tolerant routing algorithms.

4.1 Introduction

The scarcity of contact opportunities in Delay-tolerant networks mean that it is infeasible to assume that direct paths between are usually available. To overcome this challenge, Delay-tolerant network nodes usually employ the *store-and-forward* technique: when a node generates or receives a new message, it is stored in memory. The node will then physically carry the message until it eventually forwards it to another node (or discards it, if another node isn't found within a predetermined period of time).

Due to the limited resources possessed by the nodes, routing performance is an extremely important factor in Delay-Tolerant networks. Not only it is desirable to achieve a high delivery ratio (percentage of messages that successfully reach their destination) and a low delay (the time it takes for a message to arrive at the destination), but also message overhead (the ratio of replicated messages) should be kept to a minimum.

As an example, the Epidemic routing algorithm achieves a good delivery ratio and usually small delays, but the amount of generated message replicas increases exponentially with the number of neighbor nodes, leading to an unnecessary waste of resources that is not acceptable in a practical application.

It then makes sense to restrict the number of replicas generated by a node, when it wants to forward a message. One common approach is to use *context information* (information that nodes exchange between themselves when there exists a connection opportunity between them) to evaluate the fitness of a given node to meet the receiver of a message. Context-aware forwarding algorithms frequently achieve good message delivery ratios, while imposing a much smaller burden on the nodes' resources, when compared to more oblivious strategies.

Performance evaluation of Delay-tolerant network routing algorithms is mainly done by simulating a realistic usage environment, where a number of nodes experience contact opportunities over time. To do this, a number of different tools are available: network simulation software, such as the ONE (Opportunistic Network Environment simulator) or Ns (Network Simulator) allow us to generate a network environment and configure its parameters without the need of a fully-featured test bed.

On the other hand, several opportunistic contact traces are obtainable on the Internet. These consist of a record of all the contact opportunities the nodes possess with each other for the duration of the data collection experiment. Connection datasets are collected in a variety of different scenarios, providing a realistic example of how an opportunistic network behaves under significantly different circumstances.

The main objective of this chapter is to evaluate different opportunistic network characteristics and study their importance in regards to performance in routing algorithms. In order to do so, we will use a number of different opportunistic connection datasets, as well as performing network simulations using some of the tools mentioned before.

4.2 Time-Varying graphs

Usually, communication networks can be modeled as a graph structure, where vertexes represent the nodes and the edges denote the connections between them. The edges may also possess a weight, which represents the cost (or distance) between two nodes. Since opportunistic networks are known to have a highly dynamic topology, it is possible to represent the changes in the network structure by using a time-varying graph [36]. This graph is undirected since connections are considered to be bidirectional. In this case, a *journey* represents a path that exists over time between two nodes. Figure 4.1 illustrates a simple example of a time-varying graph.

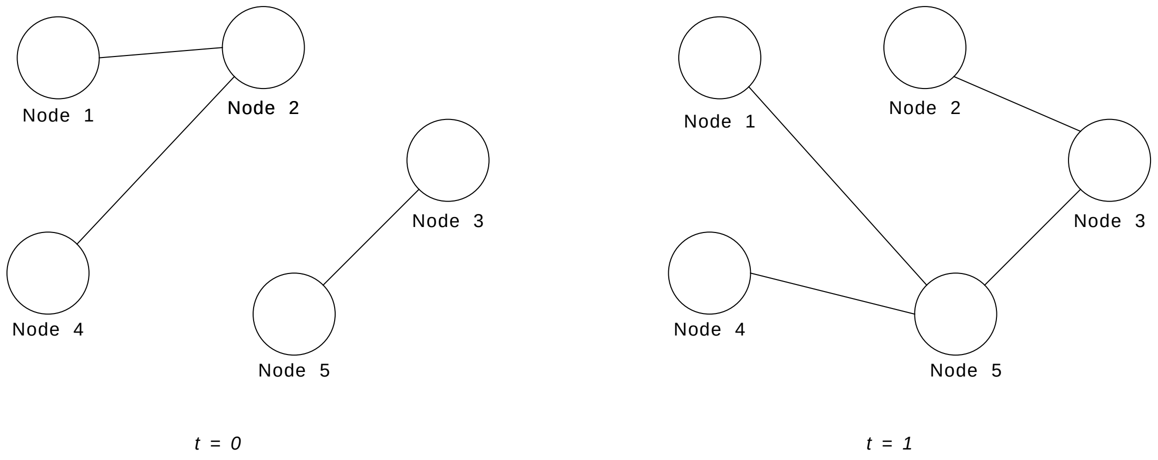


Figure 4.1: Variations in the network topology on two different periods.

We have also used the DTNTES (Delay Tolerant Network Trace Evaluator System) tool [37] to obtain a number of network metrics related to the evolution of the connectivity of the networks, as presented in table 4.1. Besides the usual metrics, such as number of nodes and edges present over the duration of the traces, there also exists information regarding the *foremost* journeys (the journey that reaches the destination at the earliest possible time) and *shortest* journeys (that is, the journeys with the least number of hops), for a given pair of nodes.

It is observable that thousands of connections between nodes are made over the du-

Table 4.1: Time-varying graph metrics for the studied datasets.

Name	*unimi/pmtr*	*upmc/rollernet*	*st_andrews/sassy*	*upmc/content*
Number of nodes	39	62	25	52
Number of edges	1214	3720	310	1288
Number of connections	20382	119204	82328	5180
Last working time	1632795	10141	6413284	987530
Average foremost transit time	154647.79	208.11	151963.66	64989.66
Average shortest journey size	1.15	1	1.44	1.58
Maximum foremost transit time	1259918	1504	3719348	646033
Maximum shortest journey size	4	2	3	4

ration of the experiments, which confirms the extremely dynamic topology of these kinds of networks. There also seems to be a significant difference between the average and the maximum values of the foremost journey times, which reinforces the idea that the paths between two nodes may take a long time to be formed. However, the hop count of the paths is very low, with the average value being usually between 1 and 1.5.

4.2.1 Static Subgraph Model

In this work, each dataset was modeled as a sequence of *footprints*. In other words, each experiment was split into intervals with a fixed duration (the same duration described in Section 3.6); for each of these intervals, a static subgraph aggregating all of the contacts between nodes during that period was calculated, with the edge weights being the inverse of the aggregated number of contacts between the two nodes in question, during that time frame. The logic behind this is that two nodes that have a high number of contacts between them will probably have a high chance of meeting again in the future. Once again, granularity is an important issue: a time window that covers the entire duration does not provide us with any meaningful information; on the other hand, choosing a duration which corresponds to the smallest time unit (in our case, one second) would provide a realistic model of the network. It is important to realize that this approach is a clear simplification of the reality, since there is no notion of the order of contacts inside a given period; nonetheless, it is a simple method that can be used to calculate relevant network metrics.

4.3 Performance analysis

By modeling an opportunistic network as a time-varying graph, it is then relevant to study different scenarios via simulations.

After generating the subgraphs, 50 different messages with random source and destination node identifiers, originating on a random time interval, were created. For each message, we use Dijkstra's algorithm to find the shortest path between the source and destination, if it exists; otherwise, the same algorithm is executed in the next time interval. The messages are replicated to all of the original sender's neighbors, and so on (similar to an epidemic routing algorithm; this effectively improves message availability, although it requires more resources from the mobile devices). Figure 4.2 shows the distribution of delivery ratio for the four datasets over a span of thirty simulations. It is clearly noticeable that *upmc/rollernet* is the scenario with the best connectivity, with the other datasets having a very low performance in terms of delivered messages.

The next approach was to find the most popular nodes and to analyze the impact they have on the usual performance metrics. When a message is successfully forwarded to the destination, all the nodes in the path are saved. The idea is that, after a reasonable amount of forwarded messages, it is possible to find out the set of critical nodes (in other words, nodes that would impair connectivity if they were removed from the graph).

This simulation is repeated for all of the datasets, but this time the 3 nodes that appear more frequently in the previous simulation are removed from the subgraphs. We then compare the 2 simulations in terms of message delivery ratio (percentage of messages that successfully reach its destination) and the average number of hops. Table 4.2 presents this information. It was observed that there was a general decrease in the message delivery ratio, and an increase in the average number of hops, for the simulations without the critical nodes.

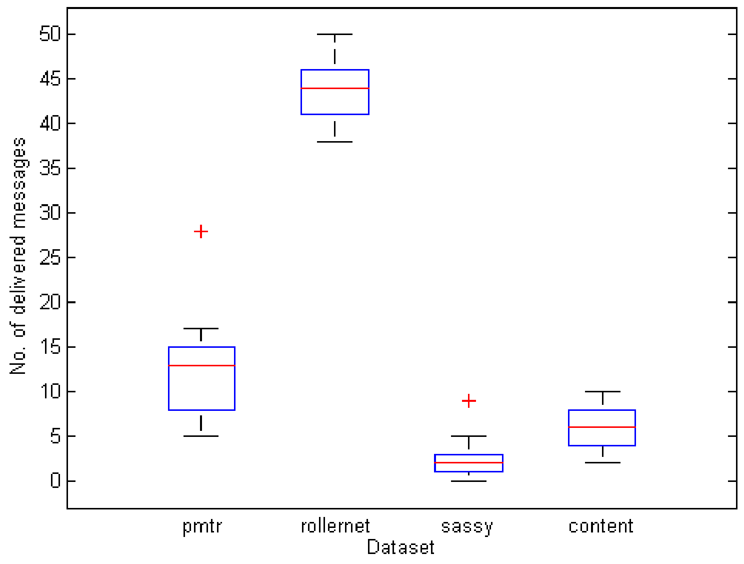


Figure 4.2: Delivery ratio distribution over thirty simulations.

This is more noticeable in the first 2 mentioned datasets, which also possess much higher delivery ratios; this fact may be explained by the more homogeneous distribution of contacts in these traces, as noted in Section 3.4.

4.3.1 Comparison with the ONE simulator

An additional experiment was also performed using the ONE network simulator, with similar parameters to the previous one; that is, messages were generated with random node source and destination ID's, during evenly-spaced time intervals. The nodes present in the dataset were modeled as having a stationary movement (since the datasets do not provide any kind of information about the movement patterns of the nodes), and the epidemic algorithm was chosen to forward the messages between the nodes.

The initial results of this experiment presented an extremely low ratio of messages that were successfully delivered (between 2 and 4%), with the exception of *upmc/rollernet*, that resulted in a delivery ratio of nearly 40%. Even by repeating the experiment with 100 and 200 random messages, the message delivery ratio remained practically unchanged. However, by increasing the simulated message TTL, the results obtained were closer to the ones of the first simulation experiment. Even so, these observations may suggest that the previous

Table 4.2: Statistics for the simulations on time-varying graphs (with and without critical nodes).

Name	*unimi/pmtr*	*upmc/rollernet*	*st_andrews/sassy*	*upmc/content*
Delivery Ratio (Complete)	0.68	0.96	0.04	0.24
Average no. of hops (Complete)	2.18	2.83	2.50	2.50
Delivery Ratio (Without critical nodes)	0.64	0.56	0.06	0.18
Average no. of hops (Without critical nodes)	4.34	4.50	1.67	2.56

simulation could be too optimistic in relation to a real-world scenario; in other words, the aggregation of contacts by time windows may not provide a model which perfectly describes contact patterns between nodes, for the chosen time periods. Table 4.3 presents these results.

The next step was to analyze the performance of another common routing algorithm, when considering a time-varying graph model. To do so, for each dataset, a simulation with 50 messages with random source and destination nodes, originating on a random time interval, was performed. A simple epidemic routing approach was used on these simulations: although it is a simple algorithm, it generates a big number of replicas that consume the limited resources of the nodes. A more efficient approach that was implemented was the Spray-and-Wait scheme, proposed by Spyropoulos et al [6]. This routing algorithm consists of two phases: the *spray phase*, where a predefined number of replicas (in our case, ten) are created and forwarded the the neighbors of the sender; and the *wait phase*, during which a node only forwards the message directly to the destination, if it is found. Table 4.4 shows the performance results in terms of delivery ratio, average hop number and number of generated replicas.

It can be concluded that the simulation produces widely different results in terms of

Table 4.3: Time-varying graph model and ONE performance comparison.

Name	*unimi/pmtr*	*upmc/rollernet*	*st_andrews/sassy*	*upmc/content*
Delivery Ratio (Complete)	0.68	0.96	0.04	0.24
Average no. of hops (Complete)	2.18	2.83	2.50	2.50
Delivery Ratio (ONE)	0.14	0.39	0.035	0.04
Average number of hops (ONE)	2.2	1.9	3.18	2.74

the message delivery ratio; this may be explained by the fact that the datasets characterize very different scenarios and network topologies. The average hop count is usually between 2 and 3, which implies that the messages do not usually need to be forwarded many times to reach the receiving node. However, the number of replicas using epidemic routing is very high for all of the datasets, which implies a waste of resources on the devices. On the other hand, the Spray-and-Wait scheme seems to produce generally similar results in terms of delivery ratio; the hop count increases, but with a noticeable decrease in the number of replicas present throughout the network.

After this, it would be interesting to observe the impact of different network parameters on the performance of a routing algorithms. To do so, the nodes were given a limited buffer size and the messages were set to be discarded after a specific number of hops. When a node receives a message when is buffer is full, it discards the oldest stored message.

Another relevant aspect of opportunistic networks is related to the amount of resources that are available to the mobile nodes. Again, using our time-varying graph model, 30 simulations were made with 50 random messages, using the epidemic algorithm. However, both the maximum number of messages a node can store and the time to live (maximum number of hops) of a random message are restricted. Figure 4.3 presents the average delivery ratio for the datasets with varying buffer size and message time to live.

It was observed that, past a certain point, increasing the buffer size does not seem to produce a noticeable impact on the message delivery ratio (around 10 stored messages for

Table 4.4: Performance metrics of Epidemic and Spray-and-Wait.

Name	*unimi/pmtr*	*upmc/rollernet*	*st_andrews/sassy*	*upmc/content*
Delivery Ratio (Epidemic)	0.68	0.96	0.04	0.24
Average number of hops (Epidemic)	2.18	2.83	2.50	2.50
Total number of replicas (Epidemic)	537	114	90	401
Delivery Ratio (Spray-and-Wait)	0.66	0.92	0.12	0.2
Average number of hops (Spray-and-Wait)	5.9	3.8	3.16	4
Total number of replicas (Spray-and-Wait)	369	108	50	292

unimi/pmtr, 5 for *upmc/rollernet*, and 20 for *upmc/content*). One notable exception is *st_andrews/sassy*, in which the delivery probability is so low that the amount of stored messages does not seem to be an important factor. Meanwhile, the maximum time to live of a message does not seem to influence heavily the delivery ratio after around five or six hops. The datasets which have greater connectivity also seem to be more easily influenced by small changes of message time to live; for the others, the connectivity is so low to begin with that these parameters do not seem to have a great impact on the delivery ratio.

The same experiment was performed using the ONE simulator; the only difference being that the message time to live was restricted in terms of actual time (in minutes) rather than hop count. This was done because the ONE does not support a maximum message hop count; also, restricting the time to very small periods would not affect the simulations on the time-varying graph model. The time to live follows a logarithmic scale, so that it is possible to observe the approximate optimal values across the four datasets. Figure 4.4 presents these results.

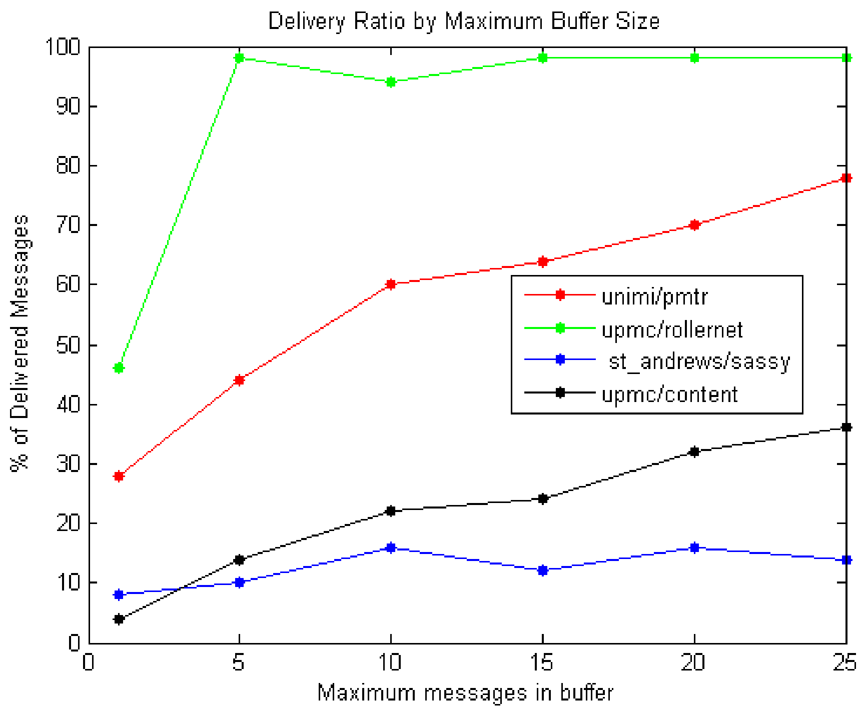
The results obtained were lower when compared to the time-varying graph simulations. This implies that the proposed time-varying graph model does not perfectly describe the reality (at least in terms of these metrics); this was expected, since our simplified model may wrongly assume that there are paths between the nodes in certain moments that do not correspond to the truth.

4.4 Conclusions and Future work

This chapter proposes the use of time-varying graphs to describe the dynamic nature of opportunistic networks. In our case, a model using static subgraphs was used, which is a simple approach, but at the cost of a lack of precision in terms of the contact opportunities of the nodes within a time period.

Using this model, a number of simulations were done in order to test the impact of different network characteristics on the performance of common routing algorithms, by using different opportunistic network traces. It was shown that the most popular nodes of a network are not always important to the performance of a routing algorithm, especially if there is a low amount of connectivity over time. The same conclusion can be made when observing the delivery ratio for reduced buffer sizes and message time to live; that is, the scenarios with the best connectivity seem to be more sensible to restrictions on the nodes' resources.

In terms of future work, it would be relevant to research other kinds of time-varying graph models, since they seem to properly represent the problem at hand. Another interesting approach would be to simulate more scenarios in order to study other kinds of network characteristics. For example, it would be interesting to simulate the energy levels of mobile devices, by assigning a cost to forwarding and device scanning operations; a node with lower energy could then be assumed to be less likely to successfully deliver a message to the destination node.



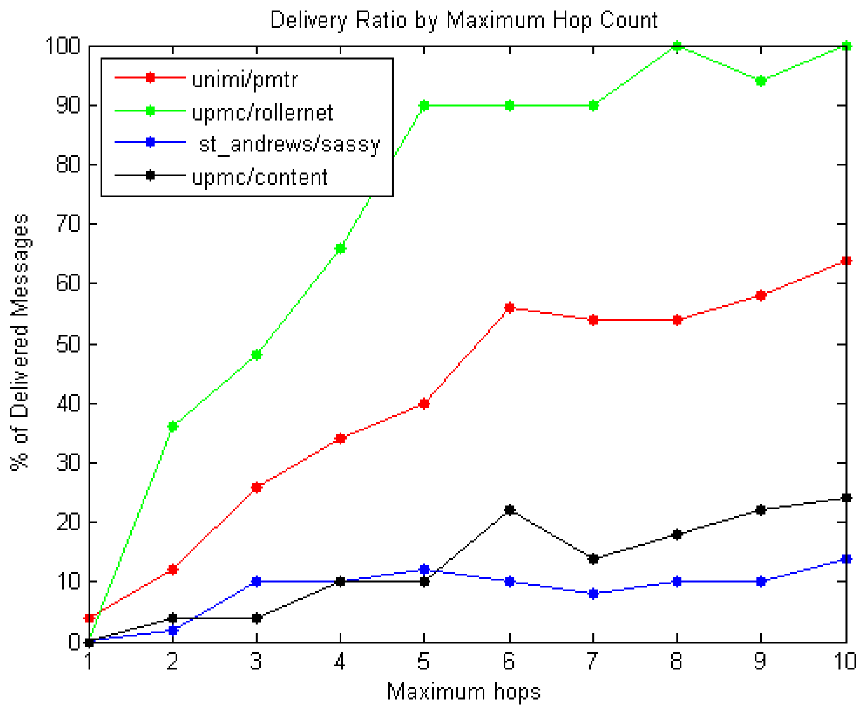
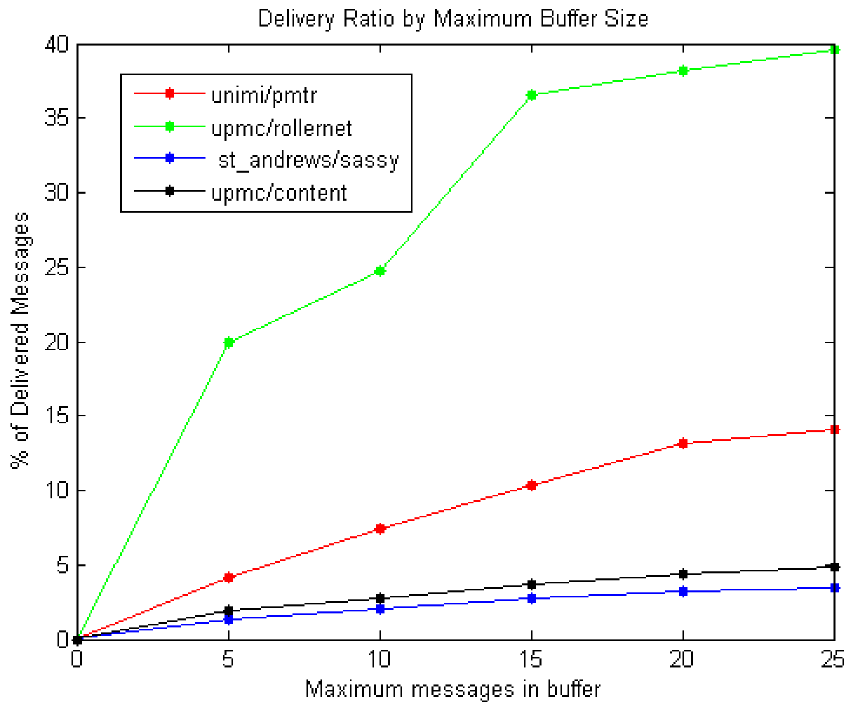


Figure 4.3: Delivery ratio with restrictions on buffer size and time to live (time-varying graph simulation).



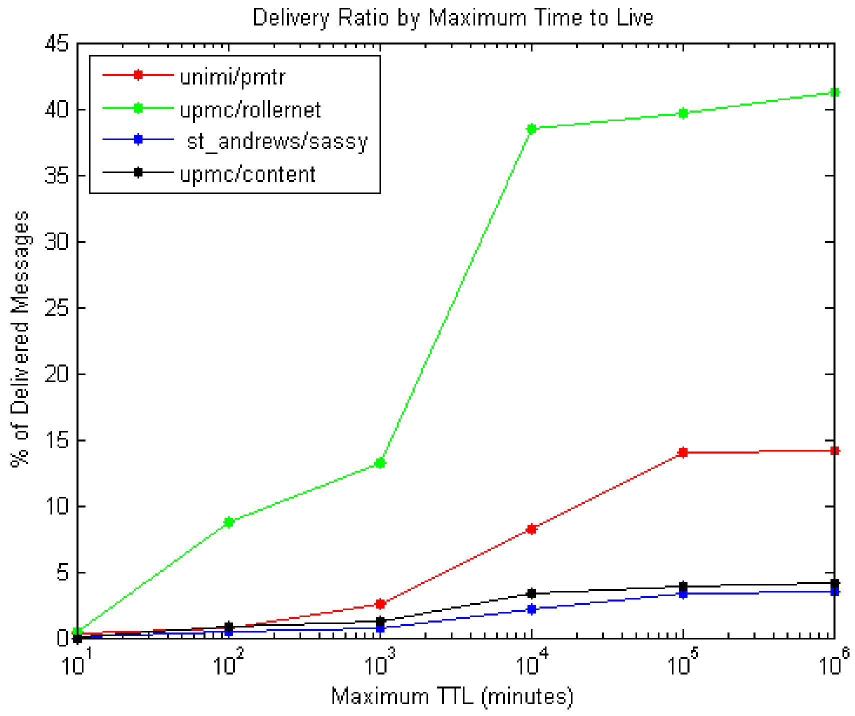


Figure 4.4: Delivery ratio with restrictions on buffer size and time to live (ONE simulation).

Chapter 5

Data Collection Issues

5.1 Introduction

One of the future objectives of this work is to design a social-aware forwarding protocol suitable for Delay-Tolerant networks. This protocol would be able to use combine existing social and physical contact information, in order to decide the most efficient path for the messages to be transmitted to other nodes. Because of this, planning a data collection experiment would prove very beneficial, since then it would allow us to obtain a suitable dataset for performance testing purposes, as well as possibly enabling us to make some initial observations (regarding both the social and the physical aspect of the contacts) that could prove to be useful when beginning to conceive the proposed forwarding algorithm.

However, due to constraints related to time and number of available participants mean that, at this time, it would not be possible to collect a sufficient amount of data for a meaningful statistical analysis, like the one done on Chapter 3. Not only it is difficult to deploy the necessary architecture for the experiment, but it is also highly unlikely to recruit a reasonable number of volunteers that are willing to provide their social network data in addition to information about physical encounters. In other words, putting this experiment into practice would only serve as a proof of concept, which would not provide us with a complete dataset of social and physical connections, with a reasonable duration and number of nodes, which was our main goal. Even so, it would be interesting to discuss some of these aspects regarding a data collection experiment, that will possibly be achieved in the future.

This chapter describes a proposed approach for a data collection experiment. First, some related ideas and problems will be discussed on a higher level of abstraction. Afterwards, each part of the experiment will be reported in a more specific and detailed manner, focusing on the technologies used and the design of the architecture.

5.2 Concepts

When planning a data collection experiment, several questions need to be addressed. One of the most important questions is, obviously, what kind of data we actually want to obtain. In our case, the objective is to acquire both social information collected from a social networking service, as well as physical contact data from mobile devices.

In terms of the social component, the main objective is to collect the social interactions of a number of users of a social networking service and using it to infer the strength of the social relationship between them. Information such as exchanged messages, shared multimedia content or mutual interests can be useful to assess how much two users interact with each other and, possibly, how likely it is for them to meet in a real world scenario. Additional data can also prove to be useful; for example, the comment timestamps can be used to represent the notion of aging (recent comments are more important than older ones, for example).

One of the most critical concerns relates to the privacy issues. Social networking service users usually share personal information, such as name, address, and contact information, not to mention extra information that can be deduced from the user's regular online activity. It is then imperative that the collected data cannot be used to identify the user in question; moreover, the actual content of the shared messages should be ignored.

On the other hand, the physical contact information consists of data describing contact opportunities between a set of mobile devices. Besides the usual node identifiers and timestamps for the beginning and ending of contact opportunities, additional data can be used to make routing decisions. For example, a node can have a lower probability of delivering a message if it has a low energy level and vice versa. Additionally, the inclusion of a node's location during a contact could be used to determine the presence of nearby devices (some social networking services also enable its users to associate a location to a specific message; this can possibly enable a way of combining the data from the two different sources).

Other question that can be discussed is related to the number of participants and the duration of the experiment. As such, it is necessary to adjust the duration of the experiment, since it should be long enough that makes it possible to identify trends and seasonality patterns; however, users taking part on this kind of experiments usually lose interest after a certain period, meaning that they will not provide a high amount of relevant information past a certain point in time.

We plan to use the social networking service Facebook to collect social data from a set of users. Facebook treats all of the users, events, shared content and connections between them as objects on this graph, and provides an API for querying that information (Graph API).

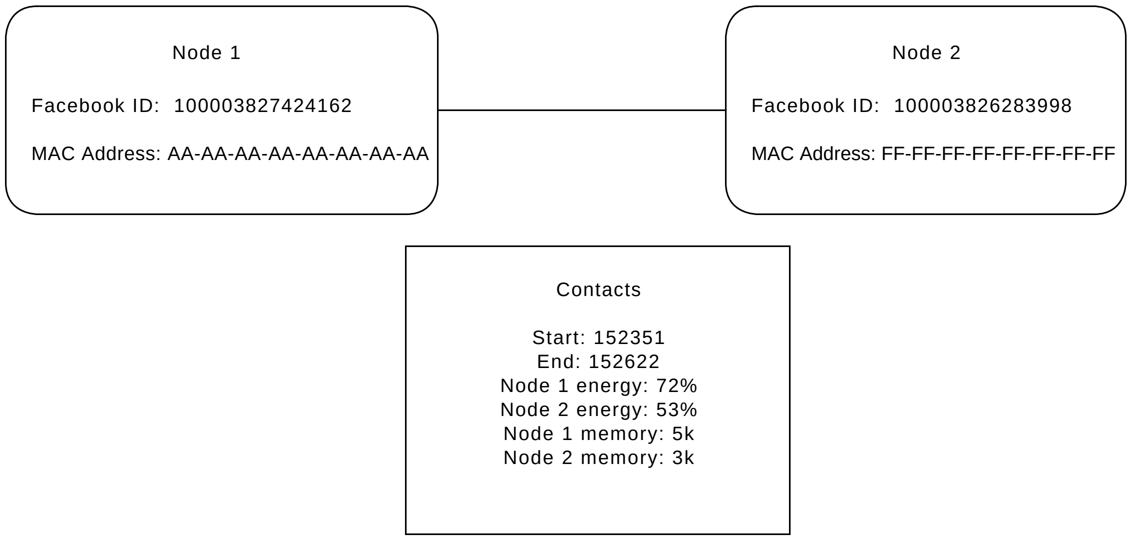

Node 1
Facebook ID: 100003827424162
MAC Address: AA-AA-AA-AA-AA-AA-AA-AA
Node 2
Facebook ID: 100003826283998
MAC Address: FF-FF-FF-FF-FF-FF-FF-FF
Contacts
Start: 152351
End: 152622
Node 1 energy: 72%
Node 2 energy: 53%
Node 1 memory: 5k
Node 2 memory: 3k

Figure 5.1: Graph with physical connection timestamps between two nodes.

It would certainly be interesting to analyze the social data in order to find relevant information regarding the way that people interact with each other in a social perspective. However, it would not be possible to perform the exact same analysis which was done on Chapter 3, for a number of reasons: namely, there is no notion of contact duration, which was is an important aspect of the physical connection traces; also, communication on social networking services is usually asynchronous, meaning that users do not need to be using the service at the same time in order to send and receive messages between them. Even so, this data can be easily modeled as a network graph, meaning that it is feasible to analyze it in terms of usual network theory metrics, as well as studying other statistics about the participating users and their social behaviour or using visualization tools to detect other types of patterns.

After collecting the social network information from a set of users, it would then be possible to add physical contact data to the social graph. The basic idea is to retrieve the duration and identifiers of the Bluetooth-enabled devices owned by the participating nodes, when a contact opportunity arises. The proposed approach involves adding a list of contacts to the edges that link two different users together. These contacts consist of a pair of timestamps that denote the start and the end of a connection opportunity between two users, as illustrated in Figure 5.1, similar to the time-varying graphs discussed in Section 4. Because this format is similar to the ones used in opportunistic connection datasets, it would be simple to export the physical contact data to be used in simulations or further analysis. In addition, other information could be collected regarding the devices, such as energy level, bandwidth and storage capacity, for example.

5.3 Proposed scenario

In terms of the data collection architecture, the ideal scenario would be to have an application installed on a number of locations that would scan and record contact opportunities with nearby wireless devices. An university campus would be a good example, since a lot of students would be able to provide information from their social networks as well as their mobile devices. Another advantage of this approach is that the users would usually spend a lot of time in the campus, thus increasing the opportunity to collect a significant amount of contact information.

This data would be transferred to a centralized machine, which would then allow us to group, anonymize and store all the information collected during the experiment, for further analysis. In this manner, the users would be generating real mobile device connection data, as in our proposed environment. An example of this scenario is shown by Figure 5.2.

To this end, a number of Bluetooth data collection applications, installed across the campus on the participants' laptops, would scan the environment and record the contact information of discovered mobile devices. This would make it possible to infer the movement patterns of the mobile devices, while also allowing to collect data outside of the original area, if the application detects external devices. This data is sent to a server application, which then updates the information stored in the graph database.

The users would also provide us with their Facebook user identifiers, so that the physical connection information could be complemented with the users' social data. The social data component would then query the social networking service in order to retrieve relevant data about the participants, which would also be inserted in the graph database.

With the obvious privacy issues that arise with this proposal, it is necessary to protect all the data that can be used to identify the users. One way of dealing with this problem would be to encrypt the user data with a cryptographic hash function, making it infeasible to obtain the protected information, while also giving stronger guarantees of integrity. Another important aspect is the security of the network channels between the different components. As such, communication between the clients, server and database should be performed through the encrypted Secure Sockets Layer protocol.

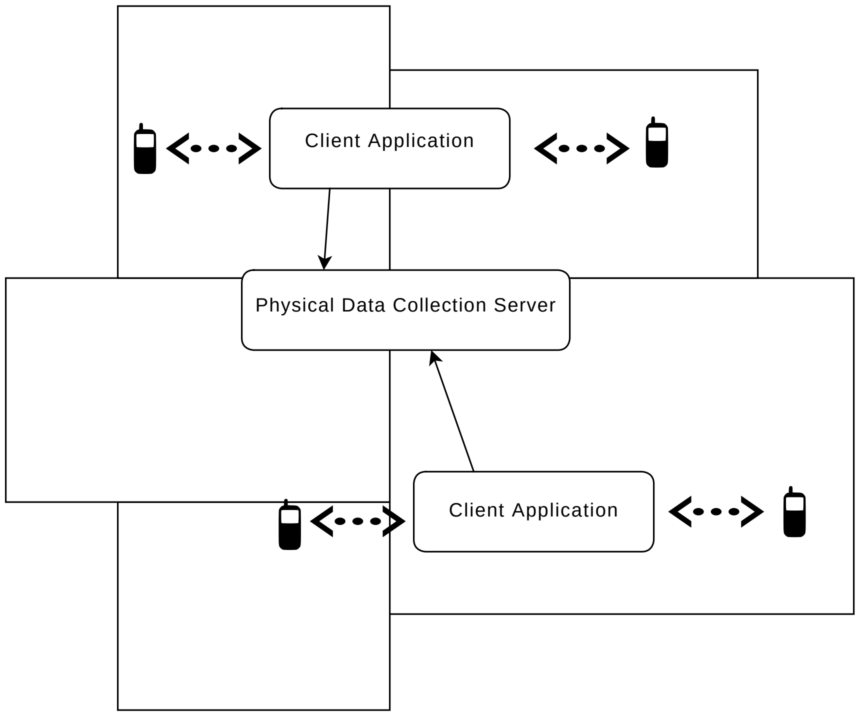


Figure 5.2: Example of the collection experiment on a campus environment.

5.4 Proof of Concept Experiment

In order to demonstrate the feasibility of the data collection experiment discussed above, a prototype was created that retrieves both social and physical device data. Since this small experiment was not performed on a realistic scenario, it does not provide sufficient information to perform a more thorough analysis; it is only intended to serve as an example of the architecture.

On Facebook, it is only possible to access information regarding its users by means of the Open Graph API. Only information that the users have made public is available by default. To access private information about a user, the Open Graph API requires an *access token*, which is an unique string for each user that allows secure access to what information the user has allowed to be queried, when authenticating an application. To make a call to the Facebook Graph API, we need to simply make a HTTPS request to https://graph.facebook.com/object_id, where *object_id* refers to the unique identifier of a particular Facebook resource, such as an event, an application, or a photo album.

The Facebook API returns a JSON (JavaScript Object Notation) object of the queried resource, which is then parsed and stored in a database by a Java application. An example of a Facebook object is shown by Listing 5.1.

Listing 5.1: JSON object example

```
{
    "name": "Facebook Platform",
    "website": "http://developers.facebook.com",
    "username": "platform",
    "founded": "May 2007",
    "company_overview": "Facebook Platform enables anyone...",
    "mission": "To make the web more open and social.",
    "products": "Facebook Application Programming Interface...",
    "likes": 449921,
    "id": 19292868552,
    "category": "Technology"
}
```

To retrieve an user's access token, a Facebook app was made, with a registration form that registers the users' name and contact information. This form is hosted on a web page that the users connect to, if they wish to participate in the experiment. First, the user is prompted to log in to their Facebook account; then, when authorizing the app, the user is asked if he allows additional info to be collected; in our case, that means having the permission to read the user's news feed (in other words, the messages, links or photos a user shares with his friends, as well as the associated comments and "likes"). The users would also provide the MAC address of their wireless devices, in order to correctly associate the physical and social contact data. The Facebook access token is also automatically retrieved, to enable further Open Graph API queries. This token is written on a web socket, to be later read by the actual querying implementation. The login flow is exemplified on Figure 5.3

The collection mechanism is hosted on the same system as the authentication web server. It consists of a daemon which listens in a web socket for a new access token. In that case, the daemon performs a series of API calls using the obtained access token. These calls request a number of different information about the user, such as the users' friends list, comments, "likes", and photo and video tags.

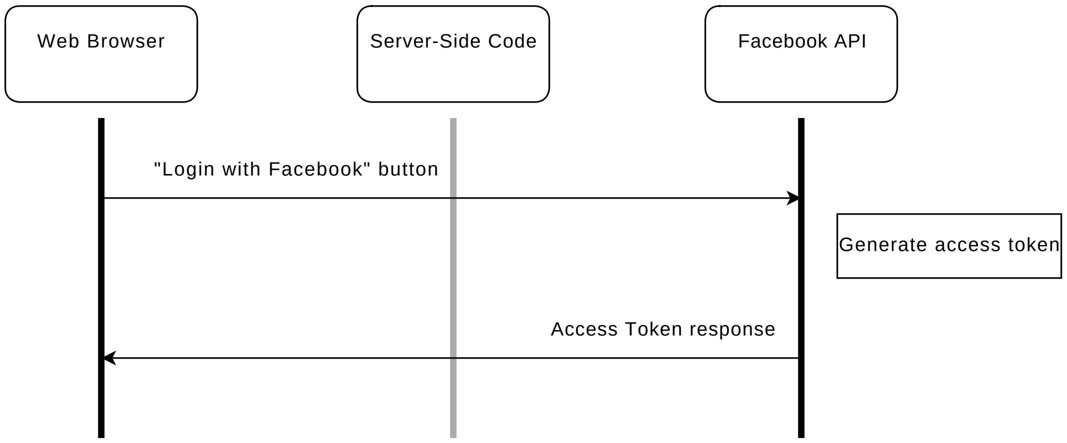


Figure 5.3: Facebook login architecture.

Since Facebook itself treats the data as a social graph, it makes sense to use a similar scheme to save the collected information from the users. The collected data is saved in a *Neo4j* database `http://neo4j.org/`, a high-performance, open-source NoSQL graph database, which stores the data as a graph structure, as opposed to the usual database tables. As the network topology is highly connected, a graph database seems to be a good choice, due to its flexibility. This allows to represent the users as nodes in the graph, while comments, "likes", shared content, and the contact opportunities between them are recorded in the edges between two nodes. Additional data the nodes possess, such as MAC address or Facebook ID, is stored as an *attribute* (a key-value pair, in Neo4j).

To collect physical contact information, a client application was built using the BlueCove Java API (`http://bluecove.org/`), which provides an interface to interact with an underlying Bluetooth protocol stack, regardless of its implementation. The physical data collection application searches for nearby Bluetooth enabled devices at regular and configurable time intervals. When a contact opportunity is found, the application records their MAC addresses, as well as the time and the duration of the contact opportunity.

This application also collects additional information such as the network interfaces, bandwidth, battery level, available memory and the number of cores of the system where it is hosted; this data is certainly relevant when considering the problems associated with opportunistic network routing schemes. This data is stored in three different files: a configuration file, which stores the application configuration parameters (such as interval between scans and server uploads, as well as proxy server hostname and port number), one file that saves the application state between different executions, and the actual connection information file, which stores the contact opportunities with other Bluetooth devices, as well as the energy levels and available memory space.

All of the collected contact information is then sent to a centralized server in regular time intervals, which stores it in the aforementioned Neo4j graph database as labels on the edges of the graph, as seen on Figure 5.4.

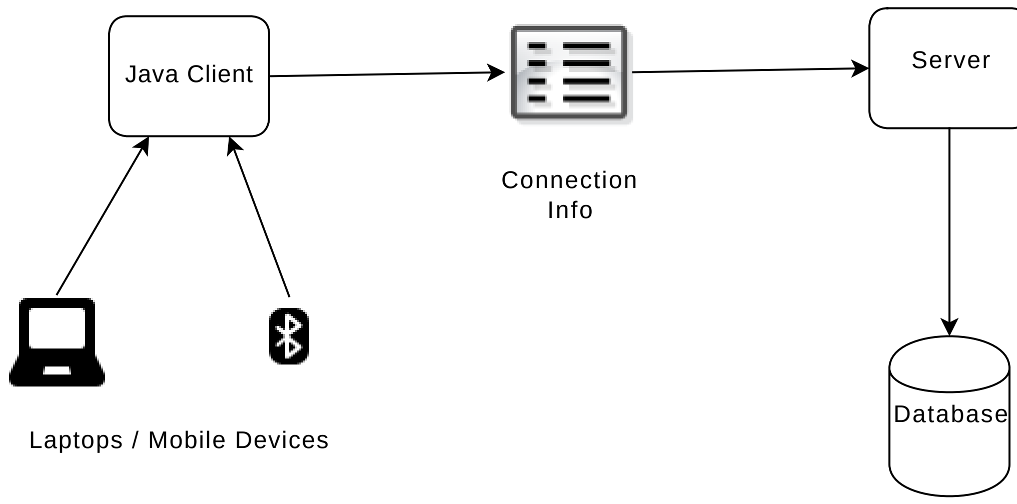


Figure 5.4: Physical data collection architecture.

The complete architecture for the data collection experiment, with the components mentioned above, is shown by 5.5. The users would authenticate the Facebook application in order to authorize access to the information that they post on their profile feeds. When first authorizing the Facebook app, we would gain access to the access token of that particular user. This token is the used to make a series of requests to the Facebook Graph API. The responses, in JSON format, are then parsed and stored in the graph database.

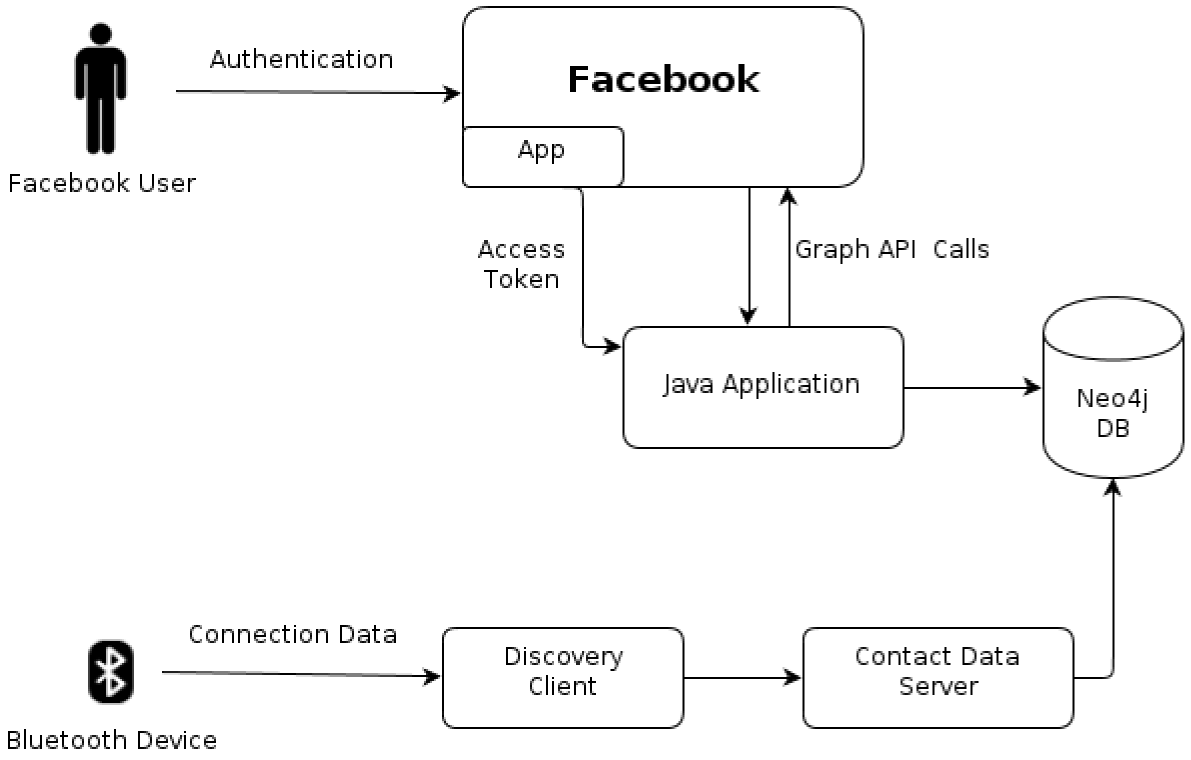


Figure 5.5: Proposed architecture of the data collection experiment.

For this small experiment, only the users' profile feed data was collected; however, the Facebook Graph API enables querying of other objects, such as liked movies, music, books, photo and video tags, or attended events. Since the Neo4j database does not follow a rigid schema, it is possible to add other types of objects to the graph. This information could be interesting to collect, since two users with the same interests could possibly have a stronger social bond, and subsequently having a higher chance of having a contact opportunity; on the other hand, two users who have attended the same event or have been tagged in the same photo or video can be assumed to have had a contact opportunity during that moment.

Naturally, this information is extremely delicate, so privacy must be guaranteed.

5.5 Conclusions and future work

As stated previously, it was not possible to use the discussed architecture and technologies in a real data collection experiment, since the available resources and number of participants would be too low to achieve any kind of statistical importance. However, a possible architecture and scenario for a real data collection experiment were proposed. Also, a proof of concept example was presented that integrates both social and physical data in a graph database. The software components described are working properly and are ready to be deployed.

We believe that, given the opportunity, obtaining a dataset that contains both social and physical connection data would prove to be very interesting, not only in terms of statistical analysis, but also as a tool used to test social-aware opportunistic routing protocols. The collected data could also prove useful when designing a social-aware routing protocol, since it could provide information about how the users connect to each other, in both physical and social environments. As such, we consider the proposed data collection experiment as one of the main goals of our future work.

Chapter 6

Final Conclusions

The documented work had the objective of performing a study on the subject of Delay-Tolerant networking. Yet, its focus was more directed to the widely available datasets than to forwarding protocols. Although this is a less common procedure, it also seems to be relevant, since it provides a different perspective on this problem.

A forwarding protocol hierarchy was presented, so as to classify the existent proposals in terms of message replication and the type of information they retrieve from the nodes themselves. Some recent social-aware routing algorithms were compared, taking into account the specific type of context information used to choose the best path for a message to reach its intended destination. Two of the most well-known Delay-Tolerant routing algorithms, Epidemic and PRoPHET, were also briefly described, since they are popular candidates for comparison of newer proposals in terms of performance metrics. It was observed that the use of context information usually seems to reduce the strain of the nodes' resources by reducing the message overhead, while maintaining a good performance in terms of delivery probability, when compared to more oblivious forwarding methods. As such this seems to be a viable strategy for the problem at hand.

One of the main contributions of this project was to perform a statistical analysis on opportunistic datasets. Four different traces, collected in different environments, were used. These traces describe the contact opportunities between the participating nodes of the experiment in question. This work included different procedures including visualization, statistical distribution and time series analysis. After describing the datasets in general regarding the scenario at hand (such as number of nodes and duration), some visualization techniques were employed. These allow us to observe that the registered contact opportunities are usually sparse and heterogeneous, when compared to the complete set of participants. It was also concluded that a small number of nodes are much more popular than the rest (in other words, they have contacts with a lot of different nodes), which seems to imply that these are very important when exchanging messages between a pair of nodes not connected otherwise. Another conclusion was that the number of contacts seems to have a good correlation with the duration of the contacts; this means that a higher contact frequency between a pair of nodes usually results in a longer connection opportunity between them.

The contact distribution seems to fit best the lognormal distribution, for all traces. The time series analysis that was made revealed some interesting results: the observed seasonality patterns seem to indicate that the specific scenario of the experiment is a deciding factor in terms of the contact opportunities that are made, since the connectivity patterns of individual nodes appears to be very similar, regardless of their popularity. Generally speaking, the main objective of this chapter was to detect connectivity patterns in the datasets that were not easily noticed at first sight, and to provide some observations that could potentially be useful when planning a new opportunistic network contact collection experiment. It was concluded that these traces characterize very different scenarios, and that these seem to be of extreme importance in terms of the connectivity behavior of the participating nodes.

Some discussion was also done in terms of time-varying graphs, since these seem to fit perfectly into the opportunistic network paradigm. The highly dynamic topology of Delay-tolerant networks can be described as a graph that is subject to constant additions and removals of edges, over time. A simple time-varying graph model was used, which divides the whole duration of the experiment into smaller subgraphs which aggregate all of the contact opportunities between the nodes, on a given time period. Although this approach implies a loss of precision, it is a simple way of comparing different datasets in terms of message routing performance metrics.

It was observed that the datasets portray very different realities, in terms of the specific scenario in which they were collected. It also seems that the more popular nodes seem to have a higher importance when there is a higher connectivity overall. As expected, the Spray-and-Wait protocol generates fewer replicas throughout the network, while achieving a similar delivery ratio to that of the Epidemic routing scheme. In addition, each trace has different optimal message time to live and buffer size values, again proving that different opportunistic environments have a great impact in terms of the underlying network parameters.

Finally, a data collection architecture was proposed, with the intention to obtain a dataset that combines both physical and social information about a set of nodes. Even though it was not possible to perform a complete data collection exercise, a proof of concept example was described, with hopes of performing a real experiment in the future.

6.1 Future Work

Naturally, there are several different approaches that can be made in terms of future work. These include, for example, extending our research to other datasets or exploring other techniques of network analysis or visualization techniques. However, one of the intended future objectives is to collect a dataset that possesses both physical contact and social network information.

Obtaining such a trace would then allow us to perform a statistical analysis such as the one made on Chapter 3. If a correlation is found between the contact frequency of the physical and the social data, then it is reasonable to think that the social network contact information can be useful in regards to a Delay-Tolerant network routing protocol. This observation would then open the doors to planning a social-aware routing algorithm that combines these two kinds of data.

One of the discussed proposals is to design a probabilistic routing algorithm based on social network interactions in addition to past physical device encounters. The main idea is that if two people interact commonly on a social network (e.g. exchanging messages, appearing in the same photographs, attending the same events, etc.), they will probably meet physically in the near future, thus allowing connection opportunities between the mobile devices carried by them.

This data, in conjunction with the history of connections of their devices, would then allow this protocol to attribute a metric which states the feasibility of a node being the next network hop (similarly to the PRoPHET protocol discussed earlier), thereby exploiting the users' mobility in order to forward a message through the network using the least resources possible, while also attempting to achieve a high probability of a message reaching its destination. It would then required to run performance tests for our protocol, in comparison to other well-know existing routing algorithms (such as Epidemic or PRoPHET), using network analysis tools, such as the ONE simulator.

An additional approach would be to implement this protocol on an actual mobile device. As an example the Bytewalla project (`http://www.tslab.ssvl.kth.se/csd/projects/092106/`) features a Delay-Tolerant network implementation for the Android mobile platform with the PRoPHET routing protocol, while also allowing developers to implement their own routing schemes.

In retrospective, the work that was made seems to be relevant to the problem in question, which fundamentally consists of providing connectivity to challenging network environments. We believe that the proposed future work on this subject makes sense, in the sense that examining available opportunistic contact traces would result in some observations that can be relevant when planning a new data collection experiment. Obtaining a trace that combines both physical and social information could prove useful when designing a new routing protocol, since recent social-aware proposals seem to have given good performance results.

Appendix A

Dataset samples

This appendix presents a small part of the datasets obtained from Crawdad, that were used in Chapters 3 and 4.

Listing A.1: unimi/pmtr dataset sample

```
id_source, id_destination, t_start_contact, t_end_contact
38         1               35869            35887
3          1               35860            35890
11         1               35901            35951
10         1               36067            36099
```

Listing A.2: upmc/rollernet dataset sample

```
1          2          1156084891          1156084891          1          0
1          2          1156085092          1156085092          2          201
1          2          1156085110          1156085110          3          18
1          2          1156085190          1156085190          4          80
```

Listing A.3: st_andrews/sassy dataset sample

```
device_having_encounter, device_seen, rawtime_start, rawtime_end,
timeuploaded, rssivalue, errorval
2,  1,  1203082300.0, 1203082393.0, 1203086114.0, 21.0, 0:35:14
2,  10, 1203082491.0, 1203082491.0, 1203086114.0, 229.0, 0:35:14
10, 18, 1203082494.0, 1203082494.0, 1203086114.0, 225.0, 0:35:14
17, 19, 1203082497.0, 1203082504.0, 1203086114.0, 224.0, 0:35:14
```

Listing A.4: upmc/content dataset sample for node 3 (file 3.dat)

```
19              1130495571              1130495571
952             1130496163               1130496163
434             1130496163               1130496163
9               1130495578              1130496752
```

Bibliography

[1] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss, "Delay-tolerant networking: an approach to interplanetary Internet," *IEEE Communications Magazine*, vol. 41, pp. 128–136, June 2003.

[2] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Single-copy routing in intermittently connected mobile networks," in *2004 First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004.*, vol. 00, pp. 235–244, IEEE, 2004.

[3] J. Leguay, T. Friedman, and V. Conan, "Dtn routing in a mobility pattern space," in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pp. 276–283, ACM, 2005.

[4] T. Spyropoulos, R. N. B. Rais, T. Turletti, K. Obraczka, and A. Vasilakos, "Routing for disruption tolerant networks: taxonomy and design," *Wireless Networks*, vol. 16, pp. 2349–2370, Sept. 2010.

[5] A. Vahdat, D. Becker, *et al.*, "Epidemic routing for partially connected ad hoc networks," tech. rep., Technical Report CS-200006, Duke University, 2000.

[6] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait," in *Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking - WDTN '05*, (New York, USA), pp. 252–259, ACM Press, 2005.

[7] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 7, p. 19, July 2003.

[8] P. Hui, J. Crowcroft, and E. Yoneki, *Bubble rap.* New York, USA: ACM Press, 2008.

[9] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, pp. 814–818, 2005.

[10] M. Newman, "Analysis of weighted networks," *Physical Review E*, vol. 70, pp. 1–9, Nov. 2004.

[11] M. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, pp. 1–15, Feb. 2004.

[12] H. A. Nguyen, S. Giordano, and A. Puiatti, "Probabilistic Routing Protocol for Intermittently Connected Mobile Ad hoc Network (PROPICMAN)," in *2007 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, pp. 1–6, IEEE, June 2007.

[13] A. Mtibaa, M. May, C. Diot, and M. Ammar, "PeopleRank: Social Opportunistic Forwarding," in *2010 Proceedings IEEE INFOCOM*, pp. 1–5, IEEE, Mar. 2010.

[14] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: Bringing order to the web.," Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[15] E. M. Daly and M. Haahr, "Social network analysis for routing in disconnected delay-tolerant MANETs," in *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing - MobiHoc '07*, (New York, New York, USA), p. 32, ACM Press, 2007.

[16] M. Everett and S. Borgatti, "Ego network betweenness," *Social Networks*, vol. 27, pp. 31–38, Jan. 2005.

[17] M. Musolesi and C. Mascolo, "CAR: Context-Aware Adaptive Routing for Delay-Tolerant Mobile Networks," *IEEE Transactions on Mobile Computing*, vol. 8, pp. 246–260, Feb. 2009.

[18] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems 1," vol. 82, no. Series D, pp. 35–45, 1960.

[19] P. Vieira, A. Costa, and J. Macedo, "A comparison of opportunistic connection datasets," in *The 3-rd International Conference on Emerging Intelligent Data and Web Technologies (EIDWT-2012)*, 2012.

[20] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 7, pp. 19–20, July 2003.

[21] L.-J. Chen, C.-H. Yu, T. Sun, Y.-C. Chen, and H.-h. Chu, "A hybrid routing approach for opportunistic networks," in *Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, CHANTS '06, (New York, USA), pp. 213–220, ACM, 2006.

[22] R. Ramanathan, R. Hansen, P. Basu, R. Rosales-Hain, and R. Krishnan, "Prioritized epidemic routing for opportunistic networks," in *Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, MobiOpp '07, (New York, NY, USA), pp. 62–66, ACM, 2007.

[23] C. Boldrini, M. Conti, J. Jacopini, and A. Passarella, "Hibop: a history based routing protocol for opportunistic networks," in *World of Wireless, Mobile and Multimedia Networks, 2007 (WoWMoM 2007)*, pp. 1–12, IEEE, 2007.

[24] N. Belblidia, M. D. Amorim, J. Leguay, V. Conan, J. Crowcroft, and S. Fdida, "Contact surround in opportunistic networks," in *Proceedings of the IEEE 21st International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC 2010, 26-29 September 2010, Istanbul, Turkey*, pp. 2499–2504, IEEE Press, 2010. INT LIP6 NPA.

[25] K. Xu, G.-H. Yang, V. O. K.Li, and S.-Y. Chan, "Detecting dynamic communities in opportunistic networks," in *Proceedings of the first international conference on Ubiquitous and future networks*, ICUFN'09, (Piscataway, NJ, USA), pp. 159–164, IEEE Press, 2009.

[26] E. Yoneki, "Visualizing communities and centralities from encounter traces," in *Proceedings of the third ACM workshop on Challenged networks*, CHANTS '08, (New York, USA), pp. 129–132, ACM, 2008.

[27] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: social-based forwarding in delay tolerant networks," in *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '08, (New York, USA), pp. 241–250, ACM, 2008.

[28] Y. C. Chen, P. Sreedevi, K. T. Chen, L. J. Chen, and A. Sinica, "Analysis of Opportunistic Networks based on Realistic Network Traces,"

[29] N. Ristanovic, G. Theodorakopoulos, and J. Le Boudec, "Traps and pitfalls of using contact traces in performance studies of opportunistic networks," in *INFOCOM, 2012 Proceedings IEEE*, pp. 1377–1385, IEEE, 2012.

[30] P. Meroni, S. Gaito, E. Pagani, and G. P. Rossi, "CRAWDAD data set unimi/pmtr (v. 2008-12-01)." Downloaded from `http://crawdad.cs.dartmouth.edu/unimi/pmtr`, Dec. 2008.

[31] J. Leguay and F. Benbadis, "CRAWDAD data set upmc/rollernet (v. 2009-02-02)." Downloaded from `http://crawdad.cs.dartmouth.edu/upmc/rollernet`, Feb. 2009.

[32] G. Bigwood, D. Rehunathan, M. Bateman, T. Henderson, and S. Bhatti, "CRAWDAD data set st_andrews/sassy (v. 2011-06-03)." Downloaded from `http://crawdad.cs.dartmouth.edu/st_andrews/sassy`, June 2011.

[33] J. Leguay, A. Lindgren, J. Scott, T. Friedman, J. Crowcroft, and P. Hui, "CRAWDAD data set upmc/content (v. 2006-11-17)." Downloaded from `http://crawdad.cs.dartmouth.edu/upmc/content`, Nov. 2006.

[34] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE Simulator for DTN Protocol Evaluation," in *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, (New York, USA), ICST, 2009.

[35] K. S. Phanse and J. Nykvist, "Opportunistic wireless access networks," in *Proceedings of the 1st international conference on Access networks*, AcessNets '06, (New York, USA), p. 11, ACM, 2006.

[36] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro, "Time-varying graphs and dynamic networks," *International Journal of Parallel, Emergent and Distributed Systems*, vol. 27, no. 5, pp. 387–408, 2012.

[37] A. Goldman, P. Floriano, and A. Ferreira, “A tool for obtaining information on DTN traces,” *The 4th Extreme Conference on Communication*, 2012.