SocialGeoProtocol: A Socially and Geographically Aware Routing Protocol for Delay Tolerant Networks

João Francisco Ribeiro Silva

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisors: Prof. Paulo Rogério Barreiros D'Almeida Pereira

Examination Committee

Chairperson: Prof. Teresa Maria Sá Ferreira Vazão Vasques
Supervisor: Prof. Paulo Rogério Barreiros D'Almeida Pereira
Member of the Committee: Prof. Dr. Naercio David Pedro Magaia

September 2020

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

First and foremost I would like to thank my supervisor, professor Paulo Rogério Pereira, for being a true mentor and always providing the best insight and criticism I could ever ask for. It was a real privilege to have him share his scientific knowledge to steer this work in the right direction. Secondly, I would like to thank my family for always supporting me. Lastly, to Eduardo for always telling me what I was doing wrong, Bukkatuna for all the advice when I needed it the most, Teresa for bearing with me throughout this process and all my friends who made it possible for me to finish this work.

Abstract

Delay Tolerant Networks (DTNs) are wireless networks characterised by intermittent connections between nodes, usually caused by high mobility of the nodes. Under these circumstances, there is no guarantee that there will be an end-to-end path between nodes in the network. As such, the routing protocols must take this into consideration and they have to be very different from those used in ad hoc networks, due to the delay between connection times. They do this by using the store-carry-forward method in which nodes store the message to be delivered and carry it with them until they encounter another node, at which point the node must decide if they should forward the message or hold onto it.

Considering that nodes can be modeled as devices carried by real people, social relations between them can be inferred and ultimately used to predict which nodes are considered more popular (and thus encounter more nodes) or which nodes are friends with each other (which could indicate they meet often). Along with the social analysis, nodes also move freely and the knowledge of each other's location can also be used when making routing decisions.

These two different aspects have been studied before independently. In this sense, this thesis focuses on combining them both by developing a socially and geographically aware DTN routing protocol called SocialGeoProtocol. Starting from the concept of linearly combining these metrics, the developed protocol is then further refined in an iterating process to improve the algorithm's performance.

Even though the simulation results show that the chosen approach was not enough to achieve the expected performance, not all scenarios were tested and many more ways to reaching this goal can be explored. The addition of an initial spray phase to the algorithm can be refined, the way of cascading of the metrics can be further investigated and a dynamic selection between the social or geographical algorithms to make a routing decision can also be considered.

Keywords: Wireless Communications, Delay-Tolerant Networks, Social Routing, Geographical Routing

Resumo

Redes Tolerantes a Atrasos (DTNs) são redes sem fios caracterizadas por conexões intermitentes entre os nós, geralmente causadas pela alta mobilidade dos mesmos. Nestas circunstâncias, não há a garantia que existe um caminho que ligue todos os nós da rede. Assim, os protocolos de encaminhamento devem ter isso em consideração e devem ser necessariamente diferentes dos usados em redes ad hoc, devido ao atraso entre os tempos de conexão. Assim,utilizam um método chamado store-carry-forward, no qual os nós armazenam a mensagem a ser entregue e transportam-na consigo até encontrarem outro nó, momento em que o nó deve decidir se deve encaminhar a mensagem ou mantê-la.

Considerando que os nós podem ser modelados como dispositivos transportados por pessoas reais, podem ser inferidas relações sociais entre si e, em última análise, usar esses dados para prever que nós são considerados mais populares (e, portanto, encontram mais nós) ou que nós são amigos (o que pode indicar que se encontram frequentemente). Além desta análise social, os nós também se movem livremente e o conhecimento da localização uns dos outros também pode ser usado ao tomar decisões de encaminhamento.

Estes dois aspectos foram estudados anteriormente de forma independente. Nesse sentido, esta tese concentra-se em combiná-los através do desenvolvimento de um protocolo de encaminhamento DTN social e geográfico, denominado SocialGeoProtocol. Tendo como ponto de partida a combinação linear das duas métricas, o protocolo desenvolvido foi sendo aperfeiçoado num processo iterativo, com o objetivo de melhorar o desempenho do algoritmo.

Apesar dos resultados das simulações demonstrarem que a abordagem escolhida não foi suficiente para atingir o desempenho esperado, não foram testadas todas as possibilidades e existem várias formas de alcançar esse objetivo que podem ser exploradas. A adição de uma fase de spray inicial ao algoritmo pode ser refinada, a forma como é feito o encadeamento das métricas pode ser investigado mais profundamente e pode considerar-se fazer uma selecção dinâmica entre os algoritmos social e geográfico quando se faz a decisão de encaminhamento.

Palavras-Chave: Comunicação Sem Fios, Redes Tolerantes a Atrasos, Encaminhamento Social, Encaminhamento Geográfico

Contents

List of Tables

List of Figures

Acronyms

DTN Delay-Tolerant Network.

GPSR Greedy Perimeter Stateless Routing.

GUI Graphical User Interface.

IoT Internet of Things.

IPN Interplanetary Network.

MANET Mobile Ad-Hoc Network.

MoVe Motion Vector.

ONE Opportunistic Network Environment.

PRoPHET Probabilistic Routing Protocol using History of Encounters and Transitivity.

QoS Quality of Service.

RMBM Route Map-Based Movement.

SCF Store-Carry-Forward.

SCORP Social-Aware Opportunistic Routing Protocol.

SPMBM Shortest Path Map-Based Movement.

TTL Time to Live.

WDM Working Day Movement.

Chapter 1

Introduction

This introductory chapter provides an overview of the concept of Delay Tolerant Networks and the motivation behind studying them as an alternative way of communication that does not rely on the Internet's infrastructure. Following that, the structure of this document is outlined.

1.1 Motivation

These days, being able to communicate using an instant messaging application is taken for granted. This is commonly achieved by using traditional data communication technologies, such as the Standard Internet Protocol Suite commonly known as TCP/IP, which rely on persistent and bidirectional end-to-end paths, short round-trip times, low error rates and symmetric data rates. However, there are some scenarios where end-to-end connectivity may not be guaranteed. Some examples of networks outside of the Internet include civilian networks that connect mobile wireless devices, such as sensors for remote environmental and animal-movement tracking, military networks connecting troops, aircraft, satellites and sensors on land and in water, outer-space networks, such as the Interplanetary Network (IPN) Internet project [1]. These networks can be characterized by long and variable delays, but also by intermittent connectivity, asymmetric data rates or high error rates and these conditions make it so that the algorithms and protocols used for terrestrial communications could not be the same as the ones used for space communications. This issue was evident in IPN and a new network paradigm started to emerge. It was called Delay Tolerant Networking and, in this type of network, a message could arrive at its final destination hours after being created and relying dynamically on intermediate nodes as a result of significant changes that affected the network topology. This was a perfect fit for the IPN environment, where there was a need to establish contact across vast distances that resulted in long delays involving multiple hops and one could take advantage of the predicted trajectory of the nodes to schedule contacts [2]. These routing strategies revolved around disseminating messages to intermediate nodes that could retain them, transport them and deliver them to either the final destination or to other intermediate nodes. This relay mechanism is called Store-Carry-Forward (SCF) [3].

However, even though their initial purpose was to be used for interplanetary communications, the concepts behind DTNs were extended to a wide variety of applications on Earth that require delay tolerant support. Some examples of such applications are cargo and vehicle tracking, agricultural crop monitoring, in-store and in-warehouse asset tracking, mobile ad-hoc networks for wireless communication and monitoring, security and disaster communication and search and rescue communication [1]. What all these scenarios have in common is the lack of end-to-end connectivity at all times, thus making DTNs a good solution to maintain communications. Another area that can make DTNs shine is the fact that the number of devices capable of being connected to a network has been increasing rapidly with the emergence of the Internet of Things (IoT), with billions of devices being installed every year [4].

1.2 Thesis Goals

As researchers investigated in the field of Delay Tolerant Networking, several routing algorithms were proposed, focusing on taking advantage of a network's characteristics to achieve better performance and more reliable communications. This master's thesis focuses on two distinct aspects developed by other researchers. On the one hand, it takes into account the geographical positions of the nodes in a network, as well as their movement. On the other, it also analyzes the relationships between nodes when deciding if a message should be transmitted. Therefore, the scope of this master's thesis is to combine these two fields and provide a routing protocol that uses both geographical and social information to test if those two metrics combined can improve the performance of a DTN network.

The main goal of this thesis is to analyze the impact of merging these two fields on a protocol's performance. This will be achieved by studying their linear combination and with a cascading approach. Using the information obtained by studying various social and geographical protocols, different scenarios will be tested, where the focus will be on the social and geographical aspects of the protocol.

1.3 Thesis outline

This section provides a brief description of the contents covered on each of the chapters of this thesis.

Chapter 1: Introduction - The present chapter provides a brief overview of how DTNs started to be developed and the motivation behind studying them further, providing examples of where this technology is useful.

Chapter 2: State of the Art - Starting with a technical overview of DTNs and describing the underlying architecture of these networks, this chapter then proceeds to highlight the difference between traditional routing protocols and the ones used in a DTN. Afterwards, several protocols are described, focusing on three categories: traditional, geographic and social-based protocols. In the end, the simulator chosen for testing the protocols is described.

Chapter 3: Social and Geographic Metrics Evaluation - This chapter starts by presenting the fundamental concept that motivated this master's thesis: combining social and geographic metrics to build a routing protocol for DTNs. It then outlines how each metric is analyzed as a standalone metric and how the first version of the protocol is implemented by combining them linearly. The first simulations are also presented and their results analyzed.

Chapter 4: Social-Geographic Routing Protocol - Taking into account the results obtained in the previous chapter, the protocol is implemented with a non-linear combination of the metrics. Following that implementation, more simulations are done and analyzed, which lead to another effort of improving the protocol's performance by incorporating an initial spray phase. This last addition is then tested and its results analyzed.

Chapter 5: Conclusions - This chapter provides a summary of the most significant conclusions obtained in this study. Afterwards, suggestions for future work in this research area are presented.

Chapter 2

State of the Art

2.1 Delay Tolerant Networks

Delay Tolerant Networking is a recognized area in networking research, due in part to practical experiences with Mobile Ad-Hoc Networks (MANETs) that are required to operate in situations where continuous end-to-end connectivity may not be possible [5].

Delay-Tolerant Networks (DTNs) were introduced in 2003, in what would become RFC 4838 [3], for fighting the enormous delays involved in deep space communications (in the order of minutes, hours, or even days). The concept was then extended to terrestrial wireless networks which also suffer disruptions and delay, and the DTN architectural emphasis grew from scheduled connectivity in the case of space communications to include other types of networks and patterns of connectivity (e.g., opportunistic mobile ad-hoc networks with nodes that remain off for significant periods of time) [3].

Moreover, the architecture proposed by the Delay-Tolerant Networking Research Group (DTNRG) [6] was conceived not only to accommodate communications in environments where continuous end-to-end connectivity cannot be assured, but also to enable interoperability between dissimilar networks. To achieve this, the DTN architectural model is based in a transmission protocol called the Bundle Protocol, which implements store-carry-and-forward message switching in an overlay network environment making communications across application programs not rely on their specific protocols, allowing the interconnection of heterogeneous parts of the network [1].

This protocol defines a bundle layer transverse to all DTN nodes which is responsible for the encapsulation of messages of arbitrary length generated by DTN-enabled applications. At this point, the transformed data is called ”bundle” as it bundles together contiguous blocks of unaltered data tied with additional information required for a transaction [7]. Figure 2.1 depicts the stack architecture of this protocol. Bundles also contain a field called Time to Live (TTL), which is used to determine for how long the message is relevant. After this time passes, messages are discarded, meaning that too long of a delay will result in the message never reaching its destination.

With regard to the resilience of transmission, since the use of a common transport protocol in an end-to-end connection is not guaranteed, the end-to-end reliability can be enhanced at the bundle layer level by means of an optional custody-based mechanism. This custody transfer mechanism makes a DTN node accountable for holding a bundle until either this custodial responsibility is delegated to another node or the bundle’s TTL expires. Still considering methods that offer resilience, since the usual contact duration combined with the large size of the bundles can induce transmission failures, the protocol also contains fragmentation and reassembly mechanisms that enable the transfer of bundle fragments even over very short contact opportunities.

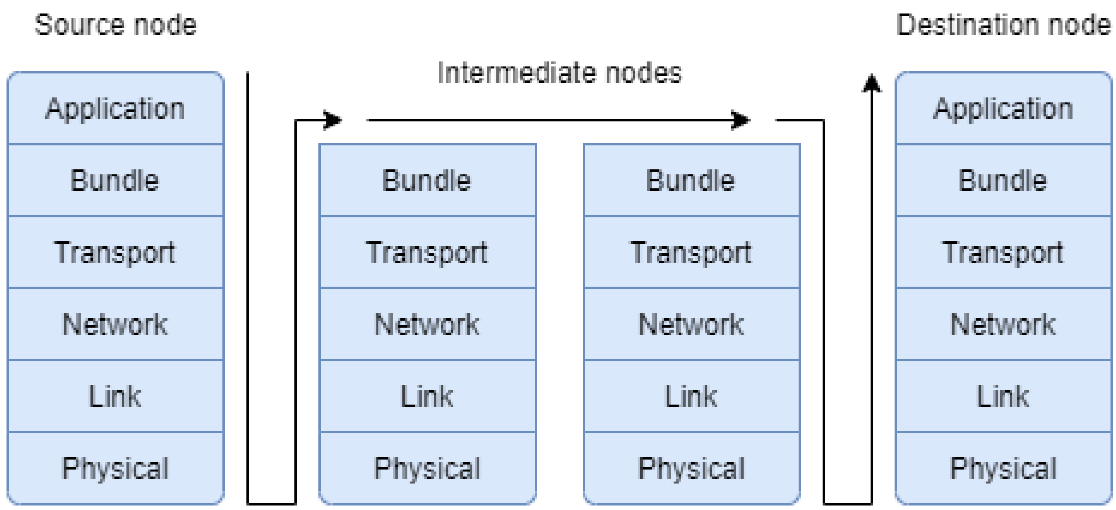

Figure 2.1: Architecture of a DTN Network (adapted from [1]).

The DTN architecture is designed to accommodate network connection disruption with a framework for dealing with heterogeneity. This allows for unique use cases, ranging from interplanetary communications to vehicular networks. It is worth mentioning that, at time of writing, the Covid-19 pandemic is active in the world and there have been many proposed solutions to track its evolution. One of them relies on a smartphone application which keeps track of other smartphones it encounters, using mechanisms similar to the ones a DTN node would use to find neighbours, keeping a log of contacts and their duration.

2.2 Routing Protocols in DTNs

As though as the routing problem in a DTN may at first appear as the standard problem of dynamic routing but with extended link failure times, this is not the case. For the standard dynamic routing problem, the topology is assumed to be connected (or partitioned for very short intervals), and the goal of the routing algorithm is to find the best currently-available path to move traffic end-to-end. In a DTN, however, an end-to-end path may be unavailable at all times; routing is performed over time to achieve eventual delivery by employing long-term storage at the intermediate nodes. This leads to long and variable delays, high error rates and intermittent connectivity, which are considered fundamental properties of DTNs, as these networks are designed to operate in such harsh conditions.

In conventional data networks, the assumption that there is a persistent path between the sender node and the receiver for the duration of a communication session is used. The sender acquires information about the network topology, defining a connected graph upon which the routing algorithm selects a shortest policy-compliant path [8]. Figure 2.2 shows a typical infrastructure supporting a conventional communications network.

The DTN routing problem amounts to a constrained optimization problem where edges (i.e., routes) may be unavailable for extended periods of time and a storage constraint exists at each node. This formulation reveals DTN routing to be a considerably different and more challenging problem [8].

Under the described conditions, in order for nodes to have their messages delivered to their destination, a non-conventional routing paradigm is used, called Store-Carry-Forward (SCF). Under it, nodes keep the messages until they find their destination or another node which is considered a better candidate to deliver them. To achieve this, nodes need persistent memory and only communicate with each

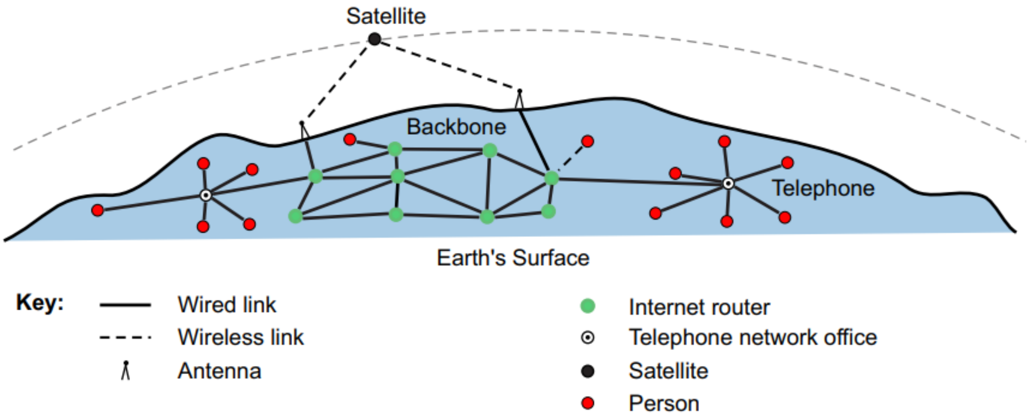

Figure 2.2: Traditional routing in communication with a persistent end-to-end path. Taken from [1].

other on opportunistic encounters [9].

As discussed, DTNs have the inherent property of not having a permanent end-to-end connection between all nodes. That being said, such networks can be represented as isolated nodes that sometimes, because they move and enter the range of transmission of one another, create links between themselves and destroy these links when they move away.

On the next subsections several routing protocols which explore the various ways to go about deciding if a node is a good candidate to deliver each message are presented.

2.2.1 Traditional DTN Routing Protocols

These protocols implement the basic functionalities required for a DTN to operate. Albeit simple and very similar, there are still some differences worth mentioning. The first two are single copy protocols, meaning only one copy of each message is present on the network, while others use multiple copies. This traditional approach does not take into account information that can be gathered as the DTN nodes move and establish connections.

- **Direct Delivery**

The first and simplest protocol to look into is called Direct Delivery. As the name suggests, when a message is generated on a node, it will only be delivered if that node encounters the message's destination [10]. In other words, the source waits until it comes in range of the destination and then delivers the message in this direct-contact situation. This scheme does not consume any additional resources and makes no additional copies of the data. The major limitation of this implementation is that the delivery delay can be infinitely large, as in many cases the source and destination may never come across each other [11].

- **First Contact**

The core concept of this protocol, called First Contact, revolves around each node forwarding as many packets in its buffer as possible while contact lasts, at every contact opportunity. The main disadvantage of this algorithm is its random character. By choosing the nodes randomly, the chosen node may not deliver the messages to their destination, possibly causing high drop rates and low delivery ratios [8]. Compared to the first protocol, the end result will be the same in terms of delivery ratio, while having a much larger overhead that originates on the many transmissions that ultimately will be deemed useless.

- **Epidemic**

The complete opposite approach to the Direct Delivery one mentioned earlier is called Epidemic. This flooding algorithm is also very simple, requiring no information about the network, as the nodes replicate all the messages stored in their buffers at every contact opportunity [12]. This makes it so that at least one copy will follow the shortest path towards its destination with minimum delay, since all paths are tested, as long as the receiving nodes still have memory to store the incoming messages. This means the Epidemic approach is only worth considering under ideal conditions, where all nodes have infinite transmission rate, to deal with situations where contacts occur in a limited time window; and infinite storage, to address the protocol's biggest setback, congestion due to the unlimited replication of messages at every node [13]. Since storage is not infinite, some messages will need to be discarded to make room for new ones and the shortest available path may not be tested.

- **Spray and Wait**

The Spray and Wait Protocol [14] is another one that uses more than one copy of each message in the network. As the name itself suggests, there are two different phases: the Spray phase and the Wait phase. On the Spray phase, the node that generates a message will forward *L* copies of that message to the first nodes it encounters, *L* being a configurable parameter. After that, the nodes enter the Wait phase where they will only forward the message if they find its destination. As this second phase is very straightforward, the original authors of this routing protocol defined some heuristics that can improve the first one. One of the proposed improvements is called Binary Spray and Wait and tries to solve the issue of how the *L* copies are to be spread initially. When a node encounters another, it forwards half of the remaining copies it has, switching to the Wait phase when it is left with just one copy. This hybrid system allows for a limited amount of copies to populate the network, avoiding congestion while also improving the delivery rates and delays [15] when compared to the other protocols analyzed so far, with the exception of the Epidemic protocol in terms of the delay, since as mentioned this protocol will always try to test the shortest available path (provided it has enough memory), at the cost of a very high congestion in the network that Spray and Wait tries to avoid.

- **PRoPHET**

The last traditional DTN routing protocol to be considered is called Probabilistic Routing Protocol using History of Encounters and Transitivity (PRoPHET) and is defined in [16]. It uses a probabilistic metric called *delivery probability* which indicates how likely it is that one node encounters another. To achieve this, this metric is updated every time the nodes meet and if two nodes do not meet for a while, the delivery probability starts getting lower again. Although this approach of assuming that nodes that have met often in the past will meet again in the future is not perfect [17], it does improve the delivery rates, network overhead and delays.

2.2.2 Geographic Routing Protocols

Geographic routing has deep research background that comes from the study of Vehicular Communications. This can be used in the context of Delay Tolerant Networks as they share some common aspects, such as the mobility of the nodes which may follow a specific pattern, in the case of public transports routes and work-home commutes, but also may not follow such patterns, e.g. taxi movements. Survey [18] presents several forwarding strategies for geographical routing in DTNs, of which Greedy forwarding and Directional forwarding stand out and will be used in this master's thesis. These concepts are defined below.

- **Greedy-DTN**

One of the most commonly used protocols involves a greedy algorithm and gets its name from it: Greedy-DTN. This is a single-copy protocol that prioritizes forwarding packets to the node that is closest to the destination at that time. It can be compared to a similar protocol used outside of the context of DTNs called Greedy Perimeter Stateless Routing (GPSR) [19]. In this protocol, nodes choose the neighbor geographically closest to the packet's destination for the next hop. However, similarities end here since if the node cannot find a neighbor closer to the destination than himself, while with GPSR the node will decide to forward the packet to a less optimal (in the greedy sense) neighbor, hoping it will be in range of a node in those conditions, Greedy-DTN utilizes the nodes' mobility, thus keeping the packet until either it finds the destination or it comes in range of a node that is closer to the packet's destination.

- **Motion Vector**

Besides the nodes' locations, some protocols also take into consideration the speed and direction of each node. The Motion Vector (MoVe) protocol proposed in [20] takes this information into account, analyzing the nodes' trajectories and tries to predict which one will move closer to the destination. Considering this information alongside the nodes' positions allows for better delivery rates and smaller delays [21].

2.2.3 Social-based Routing Protocols

The analysis of the literature on Social-based Routing Protocols led to the starting point of discovering how to categorize social relations between nodes. Unlike the geographic protocols, where the location and direction of movement are intrinsic properties of each node, social properties need to be defined. Surveys [22], [23], [24] and [25] have found that it is possible to enumerate at least five important social characteristics that protocols are based on: Community, Centrality, Similarity, Friendship and Selfishness.

One of the most important properties is **Community**. A community can be defined as a group of people living in the same place or having a particular characteristic in common [26]. In DTNs, instead of people we have nodes that communicate with each other and may create links between them. This can be useful as members of a community will have a higher chance of interacting with each other than with strangers. In order to build these dynamic communities several algorithms have been proposed, the most common one being k-clique. This algorithm adds new links to a node's graph as it meets other nodes, i.e. adds them to his community, if they interact for over a predefined amount of time.

Another property to be taken into account is **Centrality**, which represents a measure of how important a node is in a network. This importance can be categorized in three different groups: degree centrality, betweenness centrality and closeness centrality [27]. Degree centrality is the simplest as it is tied to the number of links a node has and can be calculated locally. Usually, nodes with the highest number of links are the ones other nodes want to deliver messages to, since they will most likely forward it to either the destination or closer to it. Betweeness centrality measures the degree to which a node is in a position of brokerage by summing up the fraction of shortest paths passing through it. Betweenness can be perceived as a measure of the load placed on a given node since it measures how well a node can facilitate communication among others [28]. Closeness centrality is defined as the total shortest path distance from a given node to all other nodes. Closeness can be perceived as a measure of how long it will take to spread information from a given node to all other nodes.

Another property is **Similarity**, which measures the degree of separation between individuals in social networks and implies that there is a higher probability of two nodes connecting if they have connections in common [29].

Friendship, as the name implies, represents if two nodes are friends or not. To measure this, nodes track their connection history to check if they maintain contact frequently, regularly and in long-lasting sessions [29]. Based on sociology context, nodes that are friends share some interests and therefore will interact more often when compared to strangers.

The last property is **Selfishness**, which can have a negative impact on the network. As said before, DTNs work based on the Store-Carry-Forward concept, which relies on the nodes being willing to cooperate. When a node acts selfishly, it acts against this premise. The motivation behind it can be related to the amount of resources to be spent forwarding a message - and this is where a node will negatively impact the network for its own selfish goals - or it can be a way to reduce the number of copies of a message on the network.

The following paragraphs will now describe different protocols that use some of the properties described in this sub-section.

- **BubbleRap Protocol**

The BubbleRap Protocol, defined in [30], combines the observed hierarchy of centrality of nodes and observed community structure with explicit labels, to decide on the best forwarding nodes and does so based on Label and Rank algorithms. The Label algorithm uses explicit labels to identify the communities they belong to, whereas the Rank algorithm forwards messages to nodes with higher centrality than the current node. Each of these two algorithms by themselves have some limitations. The Label algorithm is not capable of forwarding messages away from the source when the destination is socially far away and the Rank algorithm, in which each node does not have a view of global ranking, is not appropriate for big scenarios, as small communities will be difficult to reach [31]. This is the motivation BubbleRap's authors had to develop a new algorithm called BUBBLE. The first phase of this algorithm is to bubble messages to nodes that have a higher global rank until the message reaches a node with the same label as the destination's community. At this point the algorithm enters its second phase, where global ranking is not used anymore and instead messages are relayed based on local ranks, until either they reach their destination or expire.

- **dLife Protocol**

According to [32], the dLife Protocol was developed to have a better picture of social structures and to take into account the dynamics of users' behaviour resulting from their daily routines. This algorithm is able to capture such dynamics of the network, represented by time-evolving social ties between pairs of nodes as a weighted contact graph, where the weights (i.e., social ties strength) express how long a pair of nodes was in contact over different periods of time.

Based on [32], when a node wants to forward a message using dLife and finds a node in a daily sample, the sender will receive a list of all neighbours of that encountered node and the weights to reach them. If the weight from that node towards the destination is higher than the weight from the sender to the same destination, the sender will replicate the message. If not, the sender will receive the importance of the encountered node. If that value is higher than the sender's importance, the message is replicated as well [31].

- **SCORP Protocol**

The Social-Aware Opportunistic Routing Protocol (SCORP) is an algorithm proposed by the same authors as dLife that works in a similar way, while also taking into consideration the contents of the messages, in the form of the interests of the nodes. The authors define SCORP as based on a utility function that reflects the probability of encountering nodes with a certain interest among the ones that have similar daily social habits [33].

According to [34], SCORP nodes receive and store messages considering their own interests as well as interests of other nodes with whom they have interacted before. Data forwarding takes place by considering the social weight of the encountered node towards nodes interested in the message that is about to be replicated. This protocol acts on a different forwarding paradigm, as the concept of sending a message from a source to a destination (unicast) is not present here, but instead the message comes from a source to all nodes interested (multicast).

- **FinalComm Protocol**

The FinalComm Protocol described in [31] is a social-based routing protocol for DTNs which heavily relies on information about each nodes' communities to relay messages. Using BubbleRap as a starting point for development, the main concept is that nodes should have a bigger vision of the network, even though that vision could not always be the most updated one. To implement this, every time two nodes meet they exchange each others' communities (as that is their most updated information) and then both nodes compare it with the information in their own storage to get the more recent information about existing communities.

To create these communities, this protocol brings some changes when compared to BubbleRap: Node B belongs to node A's community if the total contact time between A and B is superior or equal to a threshold (*commfamiliar*), having in mind all the contact time with all other nodes in the network that A has had.

By prioritizing nodes that belong to the same community as the destination of a message, this protocol improves the delivery rate while decreasing overhead, when compared with the other social-based algorithms discussed earlier.

2.3 The ONE Simulator

This section provides a brief overview of the chosen simulator, showcasing how it will be used as the simulation environment for testing the proposed protocol. Afterwards, a detailed explanation on the diverse movement models that nodes can use is provided.

2.3.1 Brief Overview

Analyzing the efficiency of all these protocols can require weeks, months or even years of field testing. In order to avoid this, several simulators have been developed to mimic those scenarios. The authors in [35] offer a review of the following widely accepted simulation tools in opportunistic networks: Adyton, OMNeT++, The Opportunistic Network Environment (ONE) and ns-3. Out of all of these tools, The ONE Simulator stands out for being specially tailored to evaluate routing strategies in opportunistic networks encompassing ready-to-use movement models, the implementation of well-known DTN routing protocols and a powerful graphical user interface.

The Opportunistic Networking Environment Simulator was proposed in [36], a paper that describes how it can be used to evaluate and compare DTN protocols. This section will provide a brief overview of how The ONE simulator does so.

The ONE is an agent-based discrete event simulation engine developed in Java and designed in a modular fashion. In it, the agents are called nodes, the same as described in the previous sections, and they are capable of simulating the store-carry-and-forward concept. As it is open source and publicly available at [37], it allows the integration of complementary functionalities developed by any user. While this is a powerful tool to test routing protocols, this simulator also allows external event traces to be imported and used to run simulations. Since these traces can be collected from real world measurements, the ONE simulator can be used for validating the conceived synthetic models.

To replicate an opportunistic network environment when using The ONE, nodes have a set of configurable capabilities to be able to cope with a realistic store-carry-and-forward paradigm during simulation. These include the modeled radio interfaces, persistent storage, movement and energy consumption which in turn can be fine-tuned (e.g, storage capacity) or through specialized modules (e.g, different mobility models). When modeling inter-node contacts using various interfaces, since lower layers of communication are not fully modelled, wireless link characteristics are simplified: two nodes can communicate if they are within a certain radio range and that communication will follow a predefined data rate. These parameters can also be tweaked by the user in order to better match the real world characteristics of the simulation environment.

The simulator also has a report module that generates reports of simulated events, such as node movement and general Quality of Service (QoS) measurements, that are exported to text files and can be post-processed using external tools. There are several default report classes already implemented which make several metrics used to evaluate the performance of the protocols promptly available for analysis.

The ONE's release used for this thesis was v1.6.0, which has the following native implemented DTN protocols: Direct Delivery, First Contact, Spray and Wait, Epidemic, MaxProp and PRoPHET. However, as stated before, it is possible to find the work of other users and add their functionalities to The ONE. Such is the case of the dLife and SCORP protocols, found on the official The ONE page [37].

2.3.2 Movement models

The ONE already has several modules with implemented movement models. These consist in a set of algorithms and rules and are used to describe the mobility of nodes. By default, the map that is used is the one of the city of Helsinki, in Finland. The most relevant models for the proposed SocialGeoProtocol are described in this subsection.

- **Shortest Path Map-Based Movement (SPMBM)**

Nodes assigned the Shortest Path Map-Based movement model move randomly, with a seed provided by the user. A node selects a destination and proceeds to follow the shortest path along the selected map to reach it. By default, the map used is one of Helsinki's downtown. Upon reaching its destination, the node selects another destination and the cycle repeats itself until the simulation is finished.

- **Route Map-Based Movement (RMBM)**

Nodes use pre-determined routes to follow. This is used to simulate the movements of public transports, such as trains and buses.

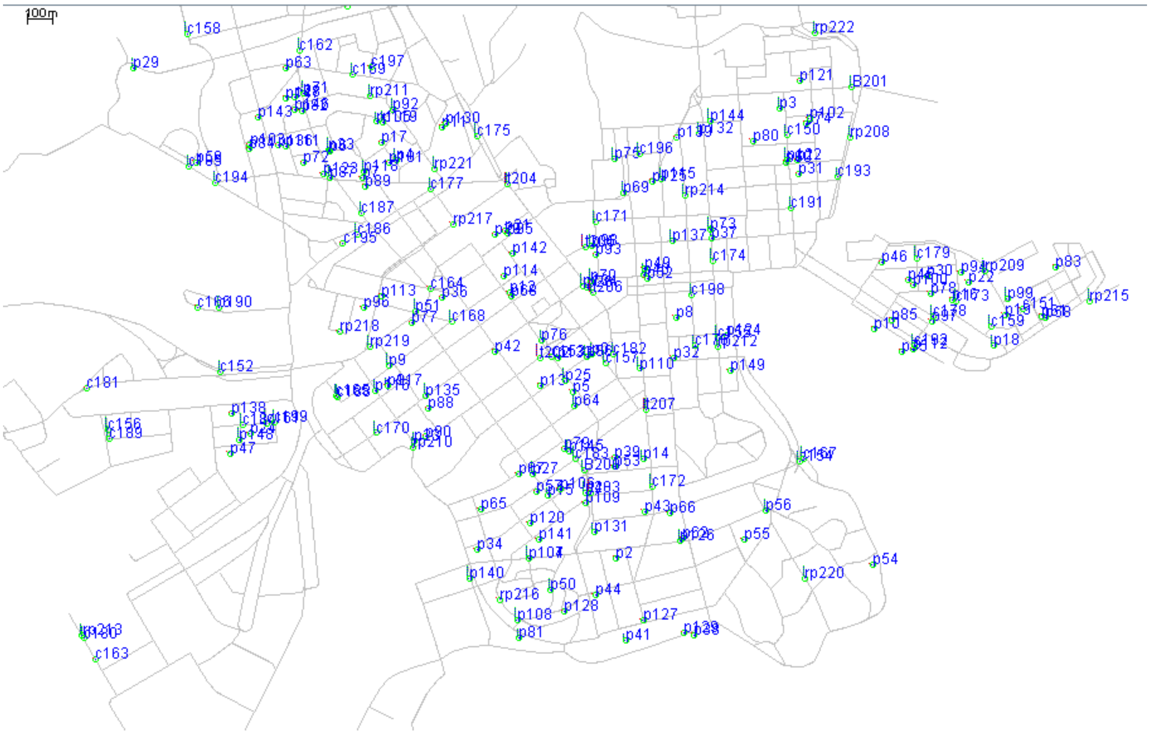

Figure 2.3: ONE Simulator GUI depicting DTN nodes using the WDM model while at their homes.

- **Working Day Movement (WDM)**

The Working Day Movement (WDM) model is used by most of the nodes. Each of these nodes acts as a person with special features: each has a randomly generated home, as depicted in figure 2.3, and is assigned an office from a user predefined list of offices, as shown in figure 2.4. Every ”morning”, each of these nodes has a configurable chance of using its private vehicle to move to its designated office. The other options include taking public transport (bus or train) or by walking. The nodes stay at the office for 8 hours, simulating a typical working day, where they interact with their coworkers. After those 8 hours pass, every node has a configurable chance to visit a social gathering place (e.g. shopping) before returning to its home. This cycle repeats itself every 24 hours.

Figures 2.3 and 2.4 show the ONE Simulator's Graphical User Interface (GUI) running a simulation. Nodes are green dots, identified by a label. In the case of figure 2.3, the simulation is paused at a time when nodes using the WDM model, identified by the letter *p* in their label, are in their respective homes (they are very scattered and isolated) and figure 2.4 shows those nodes during their working hours, clustered together in their respective offices.

Delay Tolerant Networks have been thoroughly researched since RFC 4838 [3] defined their architecture. The studies mentioned in this section have shown that there are several routing algorithms that can improve a network's reliability, even in the harsh conditions usually seen in a DTN environment, by using either the previously described social properties that can be inferred from the nodes' behaviour and contacts or a node's location and its knowledge of other nodes' positions. However, the combination of these two approaches has not yet been researched. Therefore, the focus of this work will be to take both the geographical and social approaches, which have been proven to perform better than other traditional protocols, and investigate if by merging these two concepts together in a single algorithm it is possible to achieve even better performance.

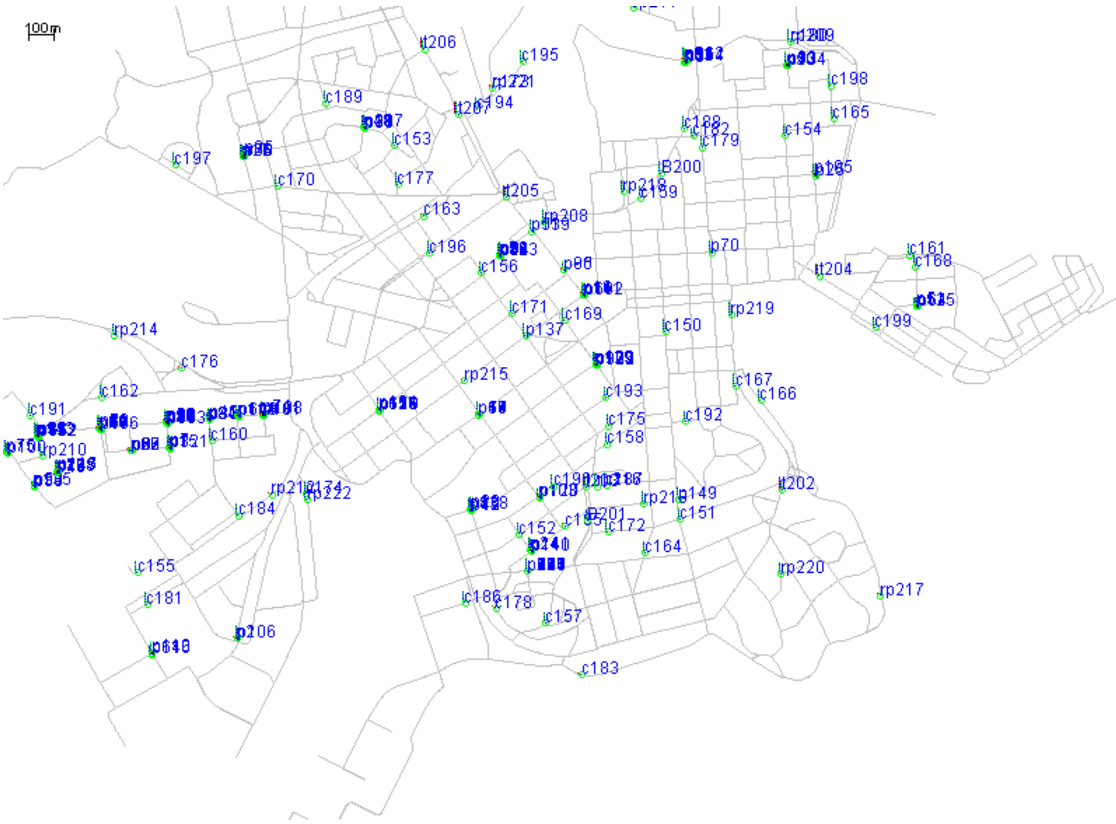

Figure 2.4: ONE Simulator GUI depicting DTN nodes using the WDM model while at their offices.

Chapter 3

Social and Geographic Metrics Evaluation

This chapter starts by laying out the motivation behind using both Geographical and Social metrics combined in a single DTN Protocol. Afterwards, a deep description of the implementation of a protocol with these two functionalities is presented. Following that, the simulation settings to test the protocol are presented. Finally, the initial tests made are presented and their results analyzed.

3.1 The Concept

Both geographical and social metrics have been widely used to improve the performance of DTN routing protocols with a high rate of success. The SocialGeoRouter proposed here is a protocol that attempts to unite both of these fields in order to achieve even better results.

The starting points to develop this protocol were FinalComm, described in section 2.2.3, as the Social protocol and Motion Vector (MoVe), described in section 2.2.2, as the Geographical one. This choice was based on the research over the literature which led to the conclusion that combining Community and Centrality was a solid approach to obtain social information of the nodes, while on the geographic case the MoVe protocol was the one that provided significant insight into the nodes' geographical properties but did not require an enormous amount of extra storage space to achieve its goal.

The ONE Simulator provides a wide range of movement models, as described in subsection 2.3.2. To better emulate a real world scenario, all simulations used a combination of those movement models with a particular emphasis on the Working Day Movement model.

The following sections describe how each algorithm was implemented and then how the algorithms were combined to generate the proposed SocialGeoProtocol.

3.2 Merging Socially and Geographically Aware Protocols

This section provides insight towards the first iteration that was required to develop the SocialGeoProtocol.

Since both FinalComm and MoVe already use their own data structures to store information relevant to their operation, the first step taken was to define a new data structure that would encapsulate both data sets. On the FinalComm side, for the protocol to work it needs to keep records of the nodes' community, the number of encounters with each node and their duration. On the MoVe side, it requires the last known position of other nodes in the network. All this information was aggregated on a single data structure that will be referred to as a dictionary.

Additionally, the Decision Engine already implemented in the FinalComm protocol had to be redesigned to face the new challenges. This Decision Engine is used to take load out of the original update method defined for The ONE simulator. It maintains and manages a list of active connections and messages. There are four events that can cause this list to be updated: a new message generated from this node, a new received message, a new connection going up or a connection going down. On the event of a new message being detected (either from this host or received from a peer), the collection of active connections is examined to see if the message should be forwarded along them. If so, a new entry is added to the list, representing that new connection. When a connection goes up, the collection of messages is examined to determine if any of them should be sent to this new peer, adding a new entry to the list. When a connection goes down, any pair of message-connection associated with it is removed from the list.

To further detail the operation of this SocialGeoProtocol, a typical scenario is presented. All nodes can carry messages, storing them in their buffer and nodes A and B are part of a vast collection of DTN nodes in a network. As any two nodes meet, the first action they perform is exchanging their dictionaries, comparing each entry. If node A finds an entry on node B's dictionary that was obtained more recently than the one it had previously stored, node B's information is considered more accurate and updates its dictionary. This is based on the assumption that, as nodes move, geographic location obtained more recently should reflect a more accurate position of a node's current location and, as the Community and Centrality metrics evolve over time, newer data is always closer to the reality in the network at time of contact.

After exchanging information about the current status of the network, nodes start looking through their message queue. For each message *m* carried by node A the first check is to see if node B is this message's destination, delivering if it is and moving on to the next message in the queue. Provided that it is not, it then checks if B already has *m* in its buffer as if it already received message *m* it will not need to forward or replicate it. These initial checks are depicted in figure 3.1.

3.2.1 Socially and Geographically Aware Algorithms

After the initial checks pictured in the flowchart in figure 3.1, the SocialGeoProtocol starts going through the Social and Geographic metrics to decide which node is the better candidate to deliver each message *m*. For each metric there's a specific algorithm, sketched in figures, 3.2, 3.3 and 3.4, and a brief explanation is provided on the next subsections.

3.2.1.1 Centrality Metric

The Centrality metric takes into account which node has a higher Centrality (i.e., has had contact with more nodes in the whole network). Node A will transfer message *m* to node B only if node B has a higher centrality. As there are several ways to achieve this, the chosen approach was to estimate the node's centrality by maintaining its degree (amount of unique past encounters). However, computing this value for all time was proven challenging and inefficient and as such, an alternative implementation was used, based on Window Centrality, where simulation time is divided in six hours periods called epochs. The specific algorithm used is called Average Degree Centrality and it computes the average node degree by breaking the entire past connection history into this series of time windows, computing the nodes' degree centrality in each epoch and then calculating the average.

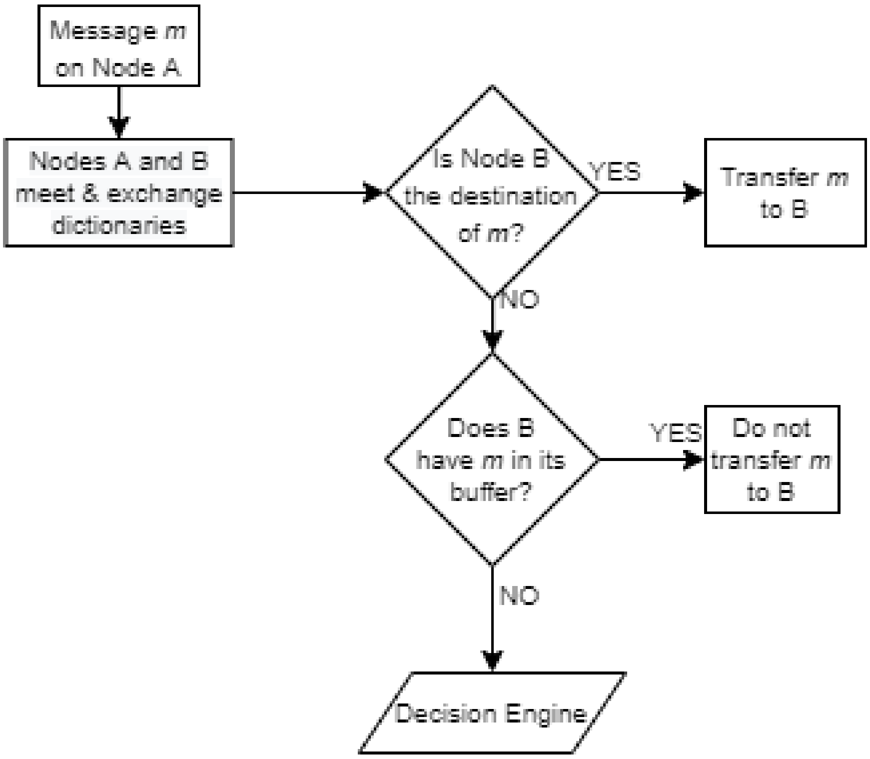

Figure 3.1: Flowchart of the initial checks.

3.2.1.2 Community Metric

Every time a node encounters another, it stores that information, along with the contact duration. As time passes and nodes spend more time in range of communication and encounter each other, a threshold is surpassed and they are considered to be part of the same community, as detailed in section 2.2.3. This is how nodes become familiar with each other and was already implemented and studied in great detail in [31], where the optimal value for this familiar threshold was found. When node A encounters node B, it checks if B is on the message's destination community and A is not, transferring message *m* if that is the case. On the other hand, if A is on message *m* destination's community and B is not, A keeps the message. Should both nodes be on the destination's community, further calculations need to be made and the node that had more contacts with nodes in the destination's community, meaning it beats the threshold to belong to that community by a larger margin gets the message. If, however, neither is on the destination's community, the node that is closer to meet the requirement to belong to said community is the one that gets *m*.

3.2.1.3 Geographic Metric

For the geographic metrics, there are two fundamental states for the nodes that established a connection, as they can either be moving or static. If node A is static and node B is moving, the algorithm checks if B is moving away or closer to the destination, transferring *m* if B is moving closer to the destination and keeping it otherwise. If B is static and node A is moving, the same logic applies with inverse consequences. In case both nodes are moving, the algorithm performs additional calculations to determine which one is moving closer to the destination of *m* and assigns the message to it. This is depicted in figure 3.5, which shows nodes A and B with their respective speed vectors, vA and vB, and vectors from their own position to the last known location of the message's destination, AD and BD. vA and vB represent the actual movement of each node, accounting for their speed, while AD and BD

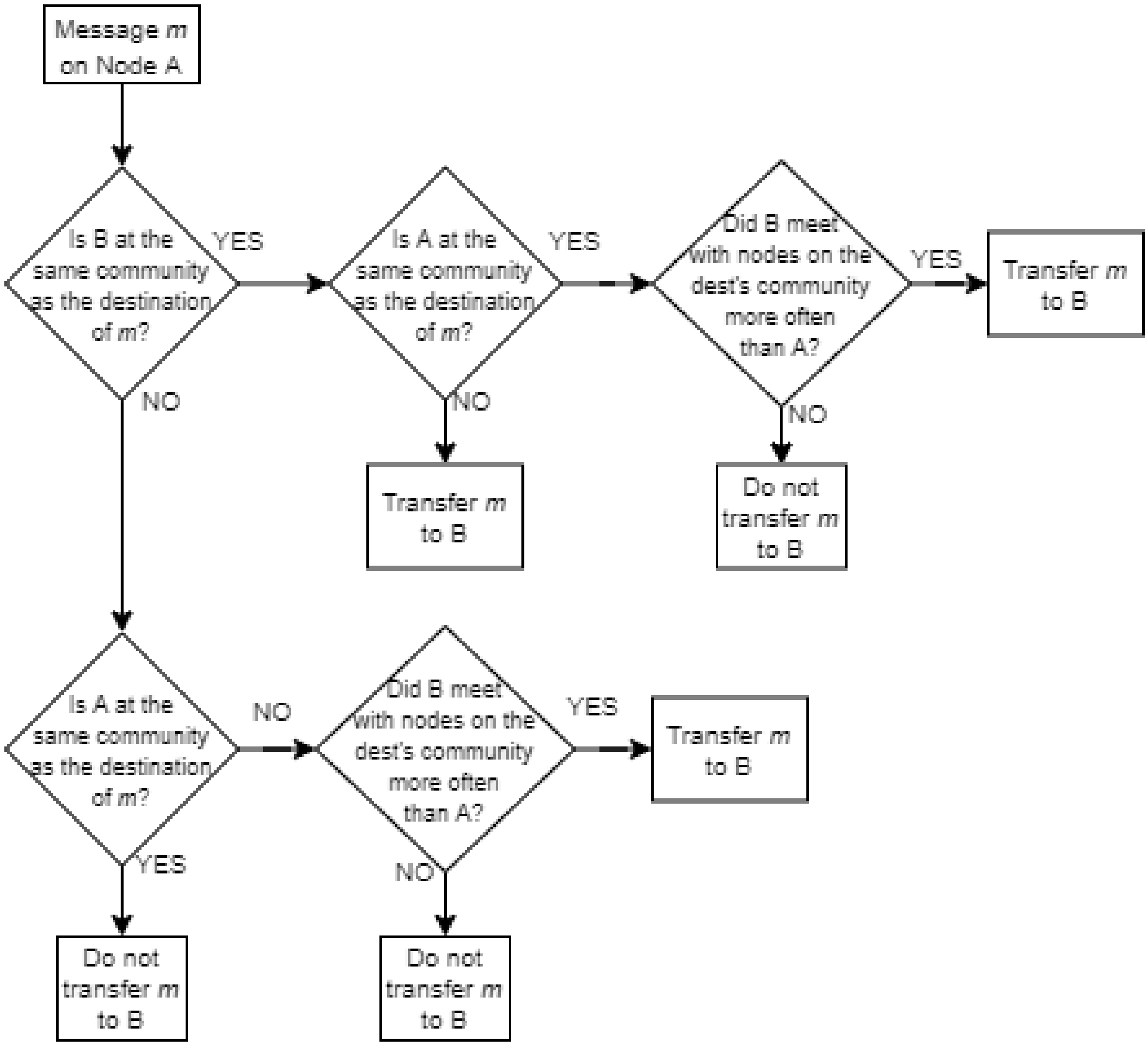

Figure 3.2: Flowchart of the community check algorithm executed for each message *m* in node A when a contact with node B is established.

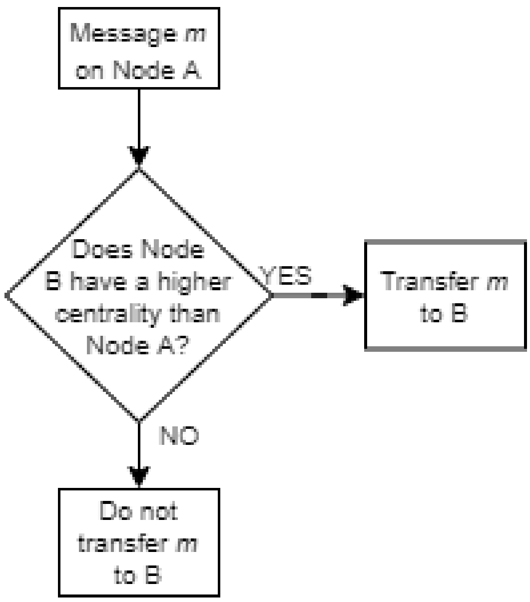

Figure 3.3: Flowchart of the centrality check algorithm executed for each message *m* in node A when a contact with node B is established.

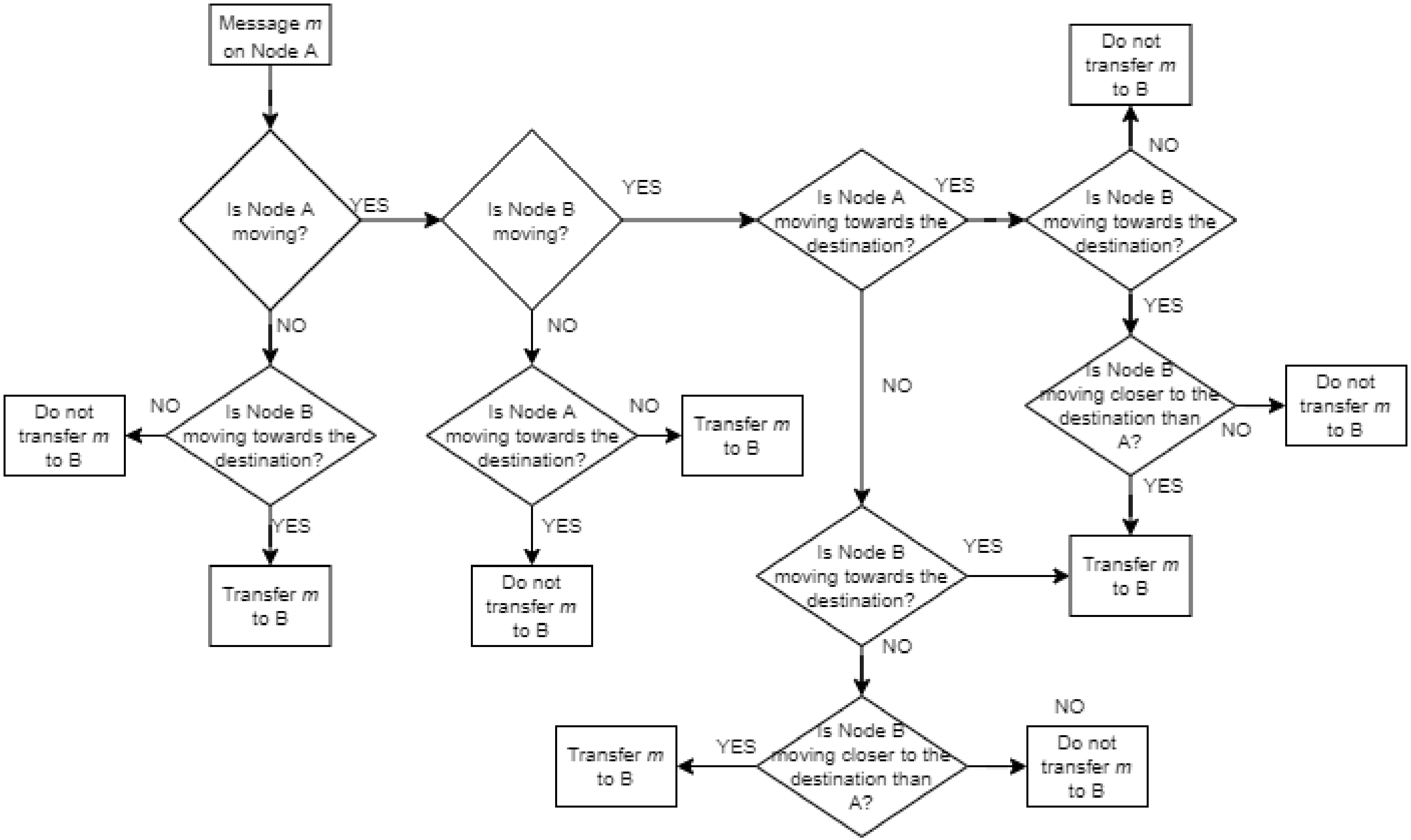

Figure 3.4: Flowchart of the geographic check algorithm executed for each message *m* in node A when a contact with node B is established.

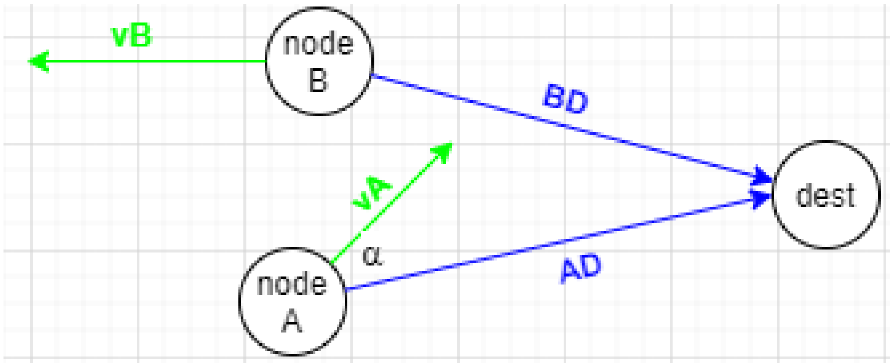

Figure 3.5: Demonstration of the angleMove routine, used within the geographic part of the SocialGeo-Protocol.

represent the distance towards the destination. Using simple trigonometry described in equation 3.1, it's trivial to find α. This angle determines how close to the destination's position each node is going. In this particular case, the angle found for node B would be greater than 90º, which means it is moving away from the destination, while α is smaller than 90º, making node A the most suitable one to keep message *m*. However, if both nodes were approaching the destination, i.e. both angles smaller than 90º, further calculation is performed combining two factors: how close the nodes are to the destination and how close they will come by the destination's last known position.

$$\alpha = \arccos \frac{\langle vA, AD \rangle}{\|vA\|.\|AD\|} \tag{3.1}$$

3.2.2 Combining the metrics

With the individual algorithm for each metric in place, the next step was to define how to merge them in a useful and meaningful way, as there are several options to do so. Since the protocol must evaluate each metric on its own and the goal was to combine them, initial tests had to be conducted to find out if any of the metrics performed better than the rest. This was achieved by, instead of forwarding or keeping a message and moving on to the next one in the buffer as soon as the first algorithm decided to do so, that information was stored locally while all the individual algorithms reached their conclusion. Once all of them had a chance to provide a possible routing decision, each result was combined in a formula, shown in equation 3.2, and the result of that calculation was the one that ultimately made the routing decision.

$$metricsCombined = (commF \times commWeight) + (centrF \times centrWeight) + (geoF \times geoWeight) \tag{3.2}$$

In equation 3.2, commWeight, centrWeight and geoWeight are the weights given to each metric and commF, centrF and geoF represent, on a scale of 0 to 100, how much node B is a good candidate to deliver message *m* according to each individual metric and thus *m* should be forwarded to it.

3.3 Simulation Settings

This section covers the simulation settings that were used in all tests, unless stated otherwise. As previously mentioned, the default scenario on The ONE relying the map of downtown Helsinki was used. However, several other settings diverged from the default parameters. These are detailed below and summarized in table 3.1.

A total of 223 nodes were created, divided into seven different groups: The first group was the core of this study and was comprised of 150 nodes, classified as pedestrians, with a walking speed of 0.5 to 1.5 m/s and moving according to the Working Day Movement Model as described in section 2.3.2. The specific settings this group's movement model were as follows: there were a total number of thirty offices and ten meeting locations. After the simulated work day, each node had a 50% chance of visiting one of these locations before returning to its assigned home. The second group had 50 nodes that represented cars, moving with a speed of 2.7 to 13.9 m/s according to the Shortest Path Map-Based Movement Model. The third group had two buses, moving at a speed of 7 to 10 m/s, following a predefined route. Groups number four, five and six represented two trams each, for a total of six trams, where each group had a predetermined route and moved at the same speed as the buses. Finally, the seventh group had 15 pedestrians which also moved at a speed of 0.5 to 1.5 m/s but followed the Shortest Path Map-Based Movement Model.

This distribution of nodes and their movement models was designed to emulate a typical working day in real world town. Most nodes in the network used the WDM model, which ties directly to the socially aware part of the protocol, highlighting the creation of communities. The other nodes were intended to work as proxies to move information from one cluster (office or shopping area) to another. This is also where the addition of the MoVe protocol should increase the performance of the algorithm as a whole.

All the nodes used the same transmission interface, with a constant speed of 250 kbps and a communication range of 10 meters. Additionally, the train groups (numbers 4, 5 and 6) have another transmission interface, representing a high speed WiFi interface, with a constant speed of 10 Mbps and a communication range of 10 meters. Although in a real world scenario the transmission range is not constant due to attenuation caused by obstacles, the value of 10 meters was defined as an approximation to account for such situations. All nodes have a buffer size of 5 MB, except for buses and trains (groups 3 through 6) which have a buffer size of 50MB.

Table 3.1: Simulation settings for each node group.

Group	#Nodes	Speed [m/s]	Buffer Size [MB]	Interfaces	Movement Model
1: Pedestrians (p)	150	0.5 to 1.5	5	1	WDM
2: Cars (c)	50	2.7 to 13.9	5	1	SPMBM
3: Buses (B)	2	7 to 10	50	1	RMBM
4: Trams (tA)	2	7 to 10	50	2	RMBM
5: Trams (tB)	2	7 to 10	50	2	RMBM
6: Trams (tC)	2	7 to 10	50	2	RMBM
7: Pedestrians (rp)	15	0.5 to 1.5	5	1	SPMBM

To generate messages, the Message Event Generator included in The ONE was used. A new message with a size of 250 kB and a Time to Live (TTL) parameter of 1440 minutes was created every 30 seconds. This value for the TTL of 24 hours allows the message to persist for a full day cycle unless it is discarded. The simulator then chooses the source and destination of messages using a random number generator derived from the nodes' group prefix. Finally, the simulations ran for 610 000 seconds, which corresponds to approximately 7 days.

3.3.1 Evaluating the performance

To evaluate the performance of the proposed SocialGeoProtocol, The ONE Simulator was used. The chosen scenario was the default map of Helsinki and the key parameters taken into account are the Delivery Ratio, Overhead, Average Delay and Average Hop count. These parameters are defined as follows:

Delivery Ratio - the ratio between successfully transmitted messages and the total number of messages that were created during a simulation. Equation 3.3 shows how it is calculated.

Average Delay - also called average latency, it represents the amount of time before a message is delivered.

Overhead - defined in equation 3.4, it is the difference between the number of relayed messages (i.e., messages that are transmitted in the network) and the number of delivered messages, divided by the number of delivered messages. This parameter indicates how saturated the network is.

Average Hop Count - average number of transmissions (called hops) that a message takes before reaching its destination.

$$Delivery\ Ratio = \frac{\#\ of\ delivered\,messages}{\#\ of\ created\,messages} \tag{3.3}$$

$$Overhead = \frac{\#\ of\ relayed\,messages - \#\ of\ delivered\,messages}{\#\ of\ delivered\,messages} \tag{3.4}$$

3.3.2 Initial Tests

The first tests were made using the weights in equation 3.2 as a way to run the protocol using only parts of the algorithm. This was accomplished by setting the values for the weights as shown in table 3.2 and running five simulations with five different seeds for each weight combination. In all cases, a 95% confidence interval is presented. The other columns in this table show the simulation results, with the average value for the delivery rate, network overhead, latency in seconds and hopcount, under the described circumstances. The most relevant output of these tests was the delivery ratio, as the goal here was to identify the most promising approach to efficiently merge all the metrics by analyzing how well they performed when considering only message delivery. The results for other constraints such as overhead, delay and hopcount are presented for a better comparison with other tests. The goal would be to first find a good combination of the metrics that delivered messages efficiently and afterwards improve the remaining evaluation parameters by refining the algorithm.

Table 3.2: Initial tests designed to assert the performance of each metric when using them with a standalone approach and combined with others.

Test number	comm Weight	centr Weight	geo Weight	Delivery rate	Overhead	Avg Latency [s]	Avg Hopcount
1	1	1	1	51.4% ± 2.16	531 ± 33	18472 ± 881	5.82 ± 0.08
2	1	0	0	65.5% ± 2.80	286 ± 22	16190 ± 799	5.16 ± 0.06
3	0	1	0	42.3% ± 1.67	328 ± 22	15631 ± 958	5.66 ± 0.06
4	0	0	1	52.4% ± 1.88	566 ± 39	13790 ± 633	6.47 ± 0.04
5	1	1	0	59.4% ± 1.92	291 ± 22	16157 ± 506	5.11 ± 0.05
6	0	1	1	50.3% ± 1.69	504 ± 46	17904 ± 545	5.76 ± 0.05
7	1	0	1	57.1% ± 1.83	350 ± 31	14284 ± 985	5.42 ± 0.06
8	2	1	1	62.3% ± 1.11	271 ± 36	17039 ± 1274	4.99 ± 0.04
9	1	2	1	38.9% ± 2.68	338 ± 30	15658 ± 1033	5.43 ± 0.06
10	1	1	2	55.8% ± 2.43	556 ± 35	13874 ± 519	6.81 ± 0.04

The initial tests are shown in table 3.2 and range from 1 to 7. Focusing on the delivery ratio, their results showed that the best standalone metric was the community one, followed by the geographic one and only then the centrality one (tests 2 through 4). By combining two metrics with the same weight, test 5 had the best result, by combining community and centrality, instead of test 7 that combined the two metrics that performed better individually (community and geographic). This result was expected, as the two metrics under analysis in test 5 are the ones used in FinalComm and BubbleRap, which have previously shown good results. However, the community metric alone performed better than along with the centrality one. Test 6 shows a decent improvement when compared to test 3, showing the benefit of adding geographic to the centrality metric, with only a marginal decline in performance when compared to test 4, where only the geographic metric is used. Test 7 shows that the community metric used along with the geographic one did not perform as well as when used alone, however this test had better results than test 4 where only the geographic metric is used. Finally, test number 1 shows that simply combining all three metrics linearly and with the same weight is not a good solution, as better delivery ratios were achieved by using a single metric or a combination of just two.

After these initial results, tests 8, 9 and 10 were performed, using all three metrics simultaneously while setting one of the weights to a higher value, as shown in table 3.2. The results all showed that the delivery rate approached the value previously obtained when running the metrics individually.

Even though the main focus of these initial tests was to compare the delivery ratios obtained under different conditions, the remaining parameters also provide some insight regarding the performance of the linear combination of the metrics. Tests using the social metrics tend to have less overhead, with the community metric impacting this parameter the most. Considering the average latency, tests that used the geographical metric performed better, resulting in less delay when delivering messages. However, this seems to have been achieved at the cost of more hops on average.

The results of the initial tests revealed that these metrics could not simply be combined linearly to obtain a better delivery ratio, thus requiring further refinement. However, the knowledge of how each individual metric impacts the delivery rate will be useful in the next section, as the protocol is improved and more tests are performed.

Chapter 4

Social-Geographic Routing Protocol

With the SocialGeoProtocol implemented and the initial results showing that the linear combination defined in equation 3.2 did not improve the delivery ratio of the protocol when compared to using the metrics by themselves, further improvements were made to the protocol.

This chapter outlines these improvements, presents new tests with adjusted simulation settings and analyzes their results as the iterating process of refining the algorithm unfolds.

4.1 Cascading the metrics

The results obtained in the end of chapter 3 were useful in the sense that they indicated how each metric behaved when analyzed by itself and how it interacts linearly with the others. Since the linear combination did not prove to be the most useful, this knowledge was then used to redesign the protocol. This meant disregarding equation 3.2 and cascading the metrics.

The idea of combining different metrics and using each of them sequentially has been studied to develop protocols such as BubbleRap, detailed in section 2.2.3. BubbleRap uses both community and centrality, focusing on forwarding messages to nodes with high centrality until a node belonging to the same community as a message's destination is found. It then switches to its second phase, where messages are forwarded to other nodes in the community until they reach the destination. The approach taken to further develop the SocialGeoProtocol was similar but relied on the delivery ratios shown in table 3.2. These led to the conclusion that the first metric to be accounted for should be the community one, as it had the best performance when considered alone. The second best performing metric was the geographical one, making it so that the protocol only analyzed this metric if the community one was not enough to reach a routing decision. If both these metrics failed to provide sufficient insight, centrality was used. The core of the algorithms for each individual metric remained as described in section 3.2.1, with the appropriate modifications pictured in figure 4.1 and detailed below.

At this stage, the main change to the algorithm is that it no longer evaluated all the metrics one by one, storing each output and combining them in the end to achieve the routing decision. Instead, if the decision was to forward a message to the node that established a connection, the messages were transmitted as soon as that decision was made and the node moved on to the next message in its queue.

However, if by evaluating the community metric as described in section 3.2.1.2, the algorithm did not obtain a positive outcome and the node would decide to keep the message to itself, it now moves on to the geographic algorithm detailed in section 3.2.1.3. Following the same logic, if the node finds it still should keep the message, it moves on to the centrality metric described in 3.2.1.1. If the output of this third computation is that the node should keep the message, it finally does so and repeats the whole process until there are no more messages in its queue or the connection is broken.

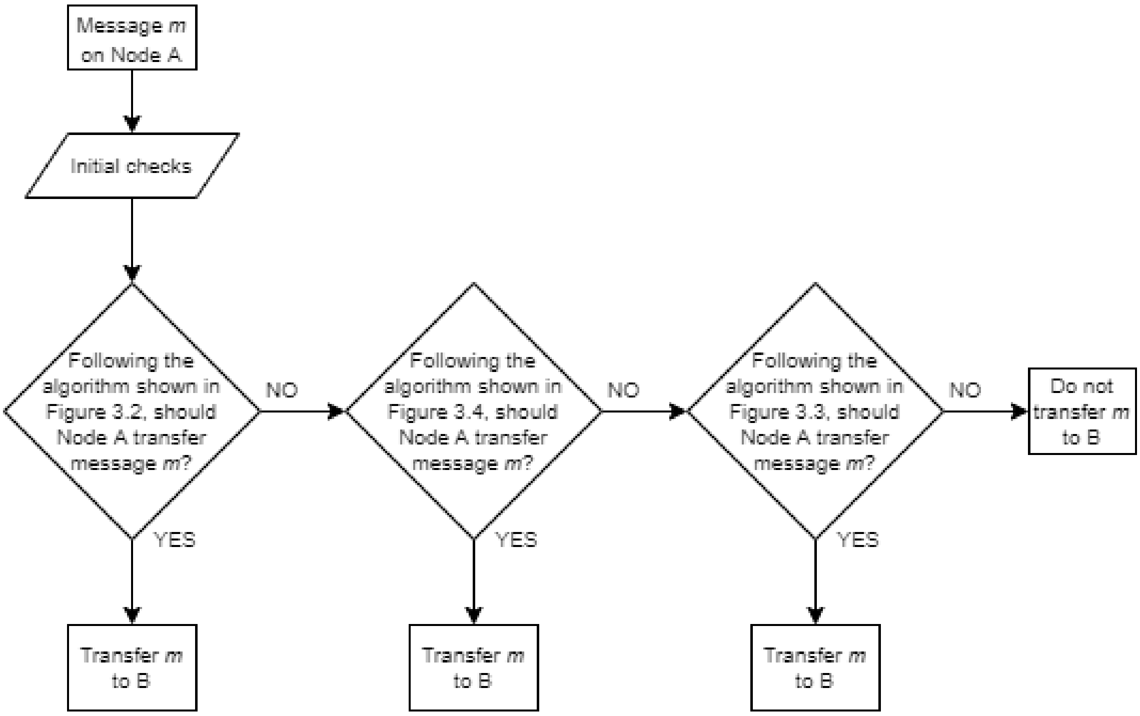

Figure 4.1: Another iteration of the SocialGeoProtocol, with the metrics being used in a sequential order.

4.1.1 Simulating with the cascading metrics

The simulation settings used in this chapter were the same as the ones detailed in section 3.3, with a few exceptions explained below. The number of nodes belonging to the first group, which act as pedestrians and use the WDM model, was changed, as well as the number of offices and the number of meeting spots. The goal was to provide a wide range of scenarios where the nodes could move between their home, office and meeting spots and analyze the impact these settings had on the overall performance of the protocol. In particular, the number of nodes in group 1 oscillated between 150 and 950, the number of offices oscillated between 30 and 60 and the number of meeting locations was set to 5, 10 or 20. The combination of these values is presented in table 4.1.

Table 4.1: Tests using the cascading metrics on scenarios with different number of WDM nodes, meeting locations and offices.

Test number	#meeting spots	#offices	#nodes	Delivery Rate	Overhead	Avg Latency [s]	Avg Hopcount
1	5	30	150	69.1% $\pm$ 1.4	197 $\pm$ 23	40677 $\pm$ 1859	4.28 $\pm$ 0.03
2	5	60	150	71.5% $\pm$ 1.4	169 $\pm$ 22	40923 $\pm$ 1878	4.18 $\pm$ 0.03
3	5	30	950	65.8% $\pm$ 2.3	172 $\pm$ 26	11998 $\pm$ 706	3.32 $\pm$ 0.08
4	5	60	950	64.2% $\pm$ 0.9	160 $\pm$ 20	11848 $\pm$ 1106	3.76 $\pm$ 0.03
5	10	30	150	63.3% $\pm$ 2.1	149 $\pm$ 25	38451 $\pm$ 1180	3.89 $\pm$ 0.02
6	10	60	150	63.1% $\pm$ 1.9	157 $\pm$ 29	40539 $\pm$ 1850	4.03 $\pm$ 0.03
7	10	30	950	61.4% $\pm$ 1.7	151 $\pm$ 25	10745 $\pm$ 1134	3.27 $\pm$ 0.04
8	10	60	950	62.7% $\pm$ 1.3	146 $\pm$ 31	11084 $\pm$ 1578	3.01 $\pm$ 0.03
9	20	30	150	61.7% $\pm$ 1.5	114 $\pm$ 24	37351 $\pm$ 1452	2.47 $\pm$ 0.04
10	20	60	150	59.3% $\pm$ 2.3	118 $\pm$ 27	38655 $\pm$ 1194	2.94 $\pm$ 0.04
11	20	30	950	63.4% $\pm$ 1.7	105 $\pm$ 29	11150 $\pm$ 1668	2.39 $\pm$ 0.02
12	20	60	950	58.8% $\pm$ 1.8	104 $\pm$ 27	11240 $\pm$ 1095	2.79 $\pm$ 0.03

Initial tests with the cascading metrics did not output results indicating this approach would be significantly better than the ones obtained in chapter 3. In order to find out if what was causing the lack of better results was connected to how the WDM model was used, three distinct parameters on which the WDM model relies were tested. Each test displayed on table 4.1 was performed five times with five different seeds for each combination and in all cases a 95% confidence interval is presented. The goal was to study how the protocol responded to a more crowded network (i.e., more nodes in the network), while also studying the effect of having more locations where the DTN nodes could gather and exchange messages, namely the offices and the meeting locations.

The overall results showed a slight improvement in the protocol's performance when compared to the ones obtained earlier, as the delivery ratios increased to a certain extent, albeit very limited. The following paragraphs analyze in depth the impact that the simulation parameters had on the four parameters that measure the protocol's performance.

The first parameter under analysis is the Delivery Rate, which seemed unaffected by the number of assigned offices. Even numbered tests had 30 offices and odd numbered tests had 60. By comparing tests where this is the only setting that was changed, the difference between their respective delivery ratios averages 2%. In contrast, increasing the number of nodes and the number of meeting locations had a negative impact on the delivery rate, as shown in table 4.1 where the results for the delivery rate degrade consistently as these two parameters increased in value. Similar remarks can be made regarding the network overhead, although in this case the obtained results seem to improve as the number of meeting spots increased. The number of offices and number of nodes did not seem to directly relate to this result. When considering the average latency, the most relevant parameter was the number of nodes in the network. Tests that used 150 nodes averaged 39432 seconds (approximately 11 hours) latency, while the ones that used 950 nodes averaged 11344 seconds (approximately 3 hours and 10 minutes) latency. Regarding the average hopcount, the parameter that stands out is again the number of meeting locations, as the results obtained while simulating with 20 meeting spots had lower average hopcount than with 10 and 5 meeting spots.

After analyzing the impact of each setting, it is clear that the number of meeting locations was the crucial parameter in these tests, while the number of offices seemed to have a residual impact in the outputs. Specifically, the difference between 30 and 60 offices had marginal impact on all the metrics except the average hopcount. In this case, tests with fewer offices output slightly lower hopcount. As an example, this can be seen when comparing tests 9 and 10 from table 4.1. The number of nodes in the network influenced the average latency, as in a sparse network there are less contact opportunities which can contribute to a higher delay.

Even though the results obtained were still not enough to consider the SocialGeoProtocol a good solution for the socially and geographical aware routing algorithm, there are still some conclusions that can be drawn. Considering the whole simulation environment with most nodes using the Working Day Movement model and the results obtained in this section, as nodes leave their designated home and spend their day in an office, communities are created that correspond to each work environment. A good real life comparison would be colleagues in a company's office. What this means for the operation of the SocialGeoProtocol is that nodes in an office surpass this community's threshold by a large margin, making each individual node become a very good candidate to deliver messages destined to this community. As the working day ends and nodes move either to their homes or to the meeting locations, they start encountering nodes from other communities, possibly creating new ones in the meeting locations. These meeting spots are prime locations for messages to reach the destination's community and ultimately reach their destination.

As mentioned earlier, the amount of meeting spots was the parameter that had the most impact in the test results in this section. Higher values for this setting meant less network overhead, a smaller average hopcount and a smaller delivery rate. The smaller delivery rate can be explained as nodes have more options of places to visit after work, making it harder to find good candidates to deliver messages as all nodes are more scattered. As for the overhead and average hopcount, bearing in mind that nodes spend their working days inside an office creating and consolidating a community, when they meet nodes from other communities in the meeting spots, a message is only relayed a small amount of times. This happens because nodes in the same community are very likely to all have a high community metric, not relaying the message to other nodes in the same community and delivering it directly to the destination.

The average latency was severely impacted by the number of nodes in the network, as it is natural that a sparser network (150 nodes) has a lower contact frequency, resulting in a higher delay when compared with a denser network (950 nodes).

4.2 Adding an initial spray phase

As mentioned in 2.2.1, the Spray and Wait protocol has two separate phases. In the first one messages are delivered, with no criteria, to the first nodes encountered. This idea was adapted into the SocialGeoProtocol, defining a Spray phase with a value of 6 for the *L* parameter. As node A creates any message *m*, starting with *L* copies of it, and then encounters node B, it delivers half of the copies of *m* it still has in its possession. Since this is node A's first encounter, it will forward three copies and keep three to itself. Once the Spray Phase is over, i.e. a node only has a single copy of a message *m*, the SocialGeoProtocol behaves the same way as described earlier in section 4.1. The flowchart for this iteration of the algorithm is displayed in figure 4.2.

The addition of an initial spray phase had already proven to increase the performance of the geographical protocol described in [38]. The goal behind it was to increase the number of copies of the same message in the network so it would reach an increased number of nodes in the spray phase and, as the Wait phase began, apply the socially and geographically aware algorithms described in chapter 3. By defining a value of 6 for the *L* parameter, there would be six nodes carrying a copy of a message *m*, thus making it more likely that at least one of them would move closer to the destination or to find another node that belongs to the message's destination community.

4.2.1 Simulating with the initial spray phase

To test this hypothesis, the simulation settings used were the same as described in section 4.1.1, except for the number of meeting spots, which was set to 5 and 20, as displayed in table 4.2 along with the other relevant settings.

Table 4.2: Tests using the initial spray phase on scenarios with different number of WDM nodes, meeting locations and offices.

Test number	#meeting spots	#offices	#nodes	Delivery Rate	Overhead	Avg Latency	Avg Hopcount
1	5	30	150	70.37% ± 1.62	419.5 ± 29.6	41838 ± 1878	4.79 ± 0.04
2	5	60	150	69.36% ± 1.32	341.1 ± 33.9	40937 ± 1577	4.61 ± 0.03
3	5	30	950	66.4% ± 2.39	323.7 ± 29.1	12345 ± 675	3.86 ± 0.07
4	5	60	950	64.27% ± 1.44	319.7 ± 34.3	11940 ± 1153	4.63 ± 0.03
5	20	30	150	64.12% ± 1.94	218.7 ± 17.9	37848 ± 1810	2.86 ± 0.04
6	20	60	150	62.58% ± 1.97	220.1 ± 29	38710 ± 1685	3.79 ± 0.03
7	20	30	950	61.37% ± 2.42	206.8 ± 37.2	11627 ± 1906	3.08 ± 0.03
8	20	60	950	64.06% ± 1.76	228.7 ± 31.6	11682 ± 1214	3.27 ± 0.02

The outcome of these tests showed that adding the initial spray phase did not improve the protocol's performance significantly. A closer analysis on the results found in table 4.2 highlights the marginally bet-

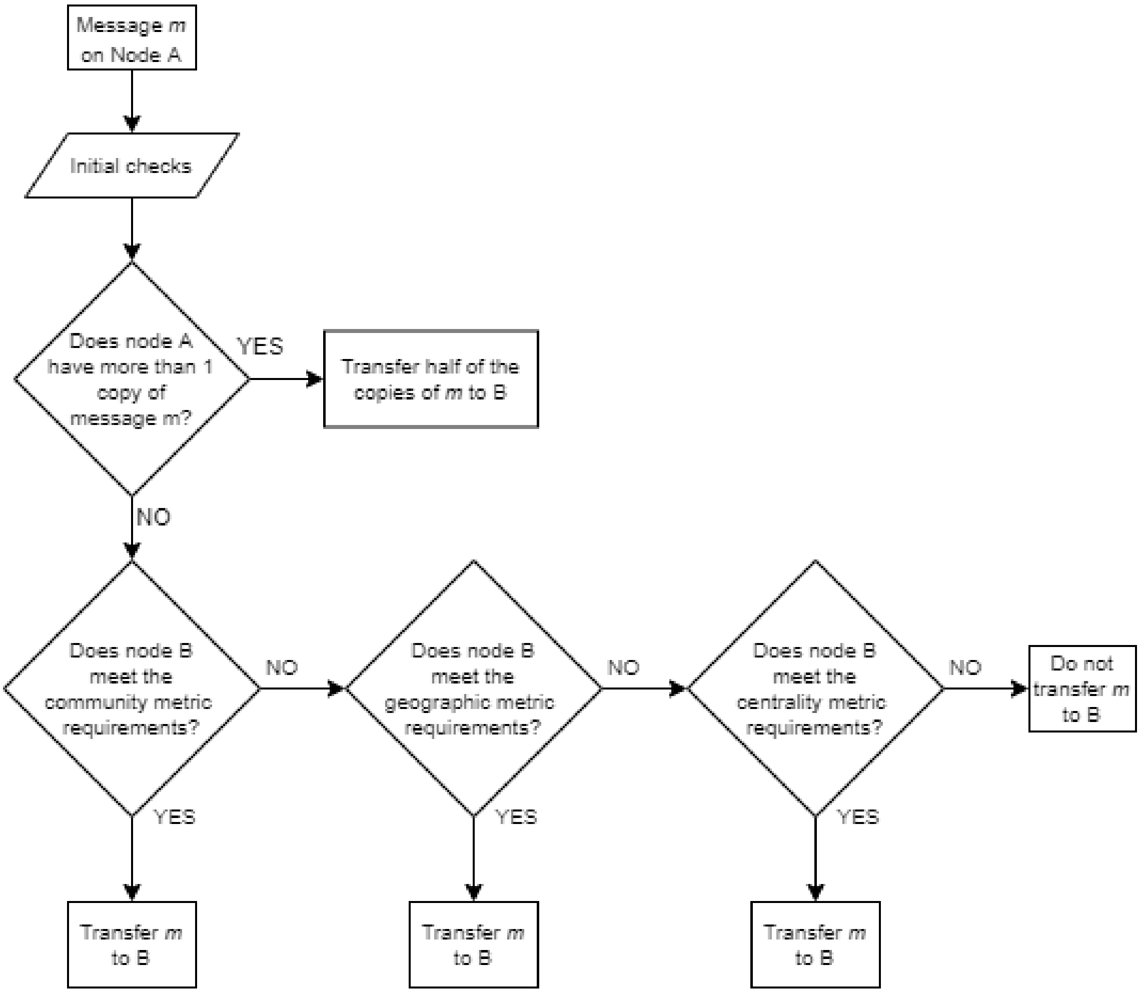

Figure 4.2: Flowchart with the added Spray and Wait phases.

ter delivery ratios, while the network overhead increased to much higher values. A possible explanation relies on the core functionality behind the spray phase. While carrying more than one copy of a message, a node will forward half of the remaining copies of that message to any node it encounters. Since nodes have to spend eight hours inside an office, as soon as a message is created and the nodes are in this environment, copies of the message are quickly spread through the network. However, messages only leave the office area when the working day ends. This causes the spray phase to only disseminate copies of the message inside the same community.

Chapter 5

Conclusions

This chapter is divided into three sections. The first one summarizes the work done in this master thesis, the second one outlines the most significant conclusions drawn from this study and the third one presents considerations and suggestions for future work to be developed in this research area.

5.1 Summary of work done

The main objective of this master's thesis was to develop a DTN routing protocol called SocialGeoProtocol, which used information about the geographical location of the nodes in a network, as well as the social relations between them to achieve better message delivery ratios than protocols that focus on using just one of these types of information. This section summarizes the chapters written to implement and describe the SocialGeoProtocol.

In Chapter 2, the fundamentals of DTNs were presented. Starting with an overview on DTNs and focusing on routing protocols that have been studied before, categorizing them in three different groups: Traditional, Socially Aware and Geographically Aware. The ONE Simulator, used to test the SocialGeoProtocol proposed in this master's thesis, was also presented in this chapter.

Chapter 3 introduced the concept behind merging geographically and socially aware protocols. Looking over the research detailed in Chapter 2, two protocols were selected and merged. The FinalComm protocol was the chosen one for the social algorithm, as it has been thoroughly described and, when compared to other well known social protocols, it performed better. For the geographical algorithm, the MoVe protocol was selected. This choice was made based on the simple way this algorithm locates nodes, which meant that the additional information that nodes had to store and share among each other would not impact the overall performance of the final protocol. Three distinct metrics were analyzed and an algorithm for each of them is presented. Afterwards, a linear combination of the three metrics was proposed and the first tests were made, using The ONE Simulator.

The tests performed in chapter 3 showed that a linear combination of the metrics would not be enough to achieve a good performance of the SocialGeoProtocol. However, the test results also highlighted which metrics performed better when considered alone. With that in mind, chapter 4 focused on combining the metrics in a nonlinear fashion, cascading one after the other. The order in which the metrics were analyzed was founded on the results obtained earlier and more simulations were performed to assess the performance of this new version of the protocol. The results turned out to be slightly better than the previous ones, albeit not as much as intended. This led to another iteration of the SocialGeoProtocol, with the addition of an initial spray phase. This would prove to be very ineffective, consuming many resources while having a marginal impact in the overall performance of the protocol.

Finally, the conclusions of this work are presented in the next section, followed by some considerations of future work that can be done to research this topic further.

5.2 Our conclusions

Results of the simulations performed in Chapter 3 provided the first insight into the problem of merging the two algorithms. When used independently from each other, the metric with the highest delivery ratio was the community one, followed by the geographical one and only then by the centrality one. This chapter focused on a linear combination of the metrics, which also showed that combining them in pairs resulted in the community and centrality metrics having the best performance, albeit not as good as good as the community one by itself. Additionally, by pairing the community and geographical metrics, which as mentioned were the ones that performed better when considered individually, although the performance improved when compared to the geographical metric's performance, it still did not output the same delivery rate as the community metric alone. The final conclusion from this chapter was that the linear combination of the three metrics did not achieve a better performance than when using the best metric (the community one) when it was used alone.

This outcome led to a new approach to the way the protocol should be developed. Taking into account the results from the previous chapter, instead of combining the metrics linearly, the SocialGeoProtocol was refactored in order to analyze each metric in a cascading fashion. To do so, the community metric, which had the best delivery rate in those tests, was picked to be analyzed first, followed by the geographical one, as it had the second best performance. Finally, the centrality metric was the last to be considered. Chapter 4 focused on using the Working Day Movement model to simulate a real world scenario where nodes would move as if leaving their homes in the morning, spend their day inside an office and having a chance to visit a social location before going back to their home. Tests were performed with different values for the number of offices and the number of meeting locations, as well as using a sparse network and a dense one. These resulted in a slight improvement on the overall performance of the SocialGeoProtocol.

The impact of these parameters was evident in the simulation results. For instance, increasing the number of meeting spots allowed the nodes to move to more locations, which led to a smaller network overhead and messages required less hops to reach their destination. However, this also meant that when nodes moved after work, they were more scattered, which impacted the delivery rate in a negative way, as messages expired before being relayed. Regarding the average latency, sparse networks experience higher delay as there are less contact opportunities than in a denser network.

As the socially aware part of the algorithm seemed to be having a positive impact on its performance, the focus shifted to the geographical part. Adding an initial spray phase to the MoVe protocol had already proven to provide better results. However, as this change was introduced in the SocialGeoProtocol and more tests were performed, the output did not improve as intended. This change brought a penalty to the network overhead, as there was an initial spray phase where messages were being created. Since nodes spent a lot of time in range of communication of the same other nodes (i.e., when they are in their offices), the impact of the spray phase was diminished, as messages did not spread to another community and only moved away from their original starting location by the end of the working day.

5.3 Future Work

This master's thesis started with the premise that both Socially and Geographically Aware DTN Routing Protocols perform better than traditional ones and thus there could be a way of combining both these concepts in order to achieve even better results. Even though the simulation results obtained may indicate that this combination does not fulfill the initial goal, further research can be done, using the results obtained here as a new starting point.

Regarding the tests made with the cascading metrics, the order in which they were evaluated was chosen based on the results obtained in chapter 2. Nevertheless, changing this order could be a way to confirm if these assumptions were correct. Moreover, one could also study how many times each metric was used in these scenarios with the cascading algorithms in order to assert how important the second and third metrics become as they are shuffled in the cascading order.

Another aspect of the SocialGeoProtocol was the addition of an initial spray phase. This was done based on previous research where such addition improved the geographical protocol MoVe. An improvement with direct impact on this part of the protocol would be to make the spray phase more selective. Instead of replicating a message to the first nodes encountered, making it so that a node only sprays a message to nodes not in its community or when it is away from the original location where the message was created.

Finally, the last consideration for future work would be to consider using both techniques, social and geographic, in a dynamic fashion. What this means is that while nodes using the WDM model have a social advantage because they create communities easily, for the geographical part of the algorithm, their movement is very limited. Spending eight hours in an office and even more than that inside their home, the geographical approach is not very suitable. In contrast, nodes not using the WDM model move much more frequently and have less opportunities to create communities. This dynamic approach would have all nodes sharing the same social and geographical dictionaries on their contacts, while prioritizing one or the other according to their individual features when making a routing decision.

Bibliography

[1] Forrest Warthman. Delay- and Disruption-Tolerant Networks (DTNs) - A Tutorial, Version 2.0. *The InterPlaNetary (IPN) Internet Project. InterPlanetary Networking Special Interest Group (IPNSIG)*, 2012.

[2] Disruption Tolerant Networking. URL `https://www.nasa.gov/content/dtn` Accessed: 2020-08-18.

[3] S Burleigh, A Hooke, L Torgerson, R Durst, K Scott, K Fall, and H Weiss. Delaty-Tolerant Networking Architecture. RFC 4838, 2007.

[4] The IoT Rundown For 2020: Stats, Risks, and Solutions – Security Today. URL `https://securitytoday.com/Articles/2020/01/13/The-IoT-Rundown-for-2020` Accessed: 2020-06-11.

[5] Kevin Fall and Stephen Farrell. DTN: An architectural retrospective. *IEEE Journal on Selected Areas in Communications*, 26(5):828–836, 6 2008.

[6] IRTF Delay-Tolerant Networking Research Group (DTNRG) [CONCLUDED], Feb 2018. URL `https://irtf.org/concluded/dtnrg` Accessed: 2020-06-11.

[7] K Scott and S Burleigh. Bundle Protocol Specification. RFC 5050, 2007. URL `https://www.rfc-editor.org/rfc/rfc5050.txt`.

[8] Sushant Jain, Kevin Fall, and Rabin Patra. Routing in a delay tolerant network. In *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications - SIGCOMM '04*, number 4 in 34, page 145, New York, New York, USA, 2004. ACM Press.

[9] Kevin Fall. A delay-tolerant network architecture for challenged internets. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications - SIGCOMM '03*, page 27, New York, New York, USA, 2003. ACM Press.

[10] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. 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, IEEE SECON 2004*, pages 235–244. IEEE, 2004.

[11] Shyam Kapadia, Bhaskar Krishnamachari, and Lin Zhang. Data Delivery in Delay Tolerant Networks: A Survey. In *Mobile Ad-Hoc Networks: Protocol Design*. InTech, 1 2011.

[12] M. R. Schurgot, C. Comaniciu, and K. Jaffres-Runser. Beyond traditional DTN routing: social networks for opportunistic communication. *IEEE Communications Magazine*, 50(7):155–162, 7 2012.

[13] Amin Vahdat and David Becker. Epidemic routing for partially-connected ad hoc networks. Technical Report CS-2000-06, Department of Computer Science, Duke University, Durham, NC, 2000.

[14] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and wait. In *Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking - WDTN '05*, pages 252–259, New York, New York, USA, 2005. ACM Press.

[15] T. Spyropoulos, K. Psounis, and C.S. Raghavendra. Efficient Routing in Intermittently Connected Mobile Networks: The Multiple-Copy Case. *IEEE/ACM Transactions on Networking*, 16(1):77–90, 2 2008.

[16] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 7(3):19, 7 2003.

[17] Hoang Anh Nguyen, Silvia Giordano, and Alessandro 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*, pages 1–6. IEEE, 6 2007.

[18] Sardar Muhammad Bilal, Carlos Jesus Bernardos, and Carmen Guerrero. Position-based routing in vehicular networks: A survey. *Journal of Network and Computer Applications*, 36(2):685–697, 3 2013.

[19] Brad Karp and H. T. Kung. GPSR. In *Proceedings of the 6th annual international conference on Mobile computing and networking - MobiCom '00*, pages 243–254, New York, New York, USA, 2000. ACM Press.

[20] Jason LeBrun, Chen Nee Chuah, Dipak Ghosal, and Michael Zhang. Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks. In *IEEE Vehicular Technology Conference*, volume 61, pages 2289–2293. IEEE, 2005.

[21] Jiayu Gong, Cheng-Zhong Xu, and James Holle. Predictive Directional Greedy Routing in Vehicular Ad hoc Networks. In *27th International Conference on Distributed Computing Systems Workshops (ICDCSW'07)*, pages 2–2. IEEE, 2007.

[22] Kaimin Wei, Xiao Liang, and Ke Xu. A Survey of Social-Aware Routing Protocols in Delay Tolerant Networks: Applications, Taxonomy and Design-Related Issues. *IEEE Communications Surveys & Tutorials*, 16(1):556–578, 21 2014.

[23] Ying Zhu, Bin Xu, Xinghua Shi, and Yu Wang. A Survey of Social-Based Routing in Delay Tolerant Networks: Positive and Negative Social Effects. *IEEE Communications Surveys & Tutorials*, 15(1):387–401, 21 2013.

[24] Kaimin Wei, Deze Zeng, Song Guo, and Ke Xu. On Social Delay-Tolerant Networking: Aggregation, Tie Detection, and Routing. *IEEE Transactions on Parallel and Distributed Systems*, 25(6):1563–1573, 6 2014.

[25] Feng Xia, Li Liu, Jie Li, Jianhua Ma, and Athanasios V. Vasilakos. Socially Aware Networking: A Survey. *IEEE Systems Journal*, 9(3):904–921, 9 2015.

[26] Angus. Stevenson. *Oxford dictionary of English.* Oxford University Press, 2010.

[27] Linton C. Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1(3):215–239, 1 1978.

[28] Naércio Magaia, Alexandre P. Francisco, Paulo Pereira, and Miguel Correia. Betweenness centrality in Delay Tolerant Networks: A survey. *Ad Hoc Networks*, 33:284–305, 10 2015.

[29] Francisco Fernandes, Naercio Magaia, and Paulo Rogerio Pereira. Friendship-based Routing Protocol for Delay-Tolerant Networks. In *2018 16th International Conference on Intelligent Transportation Systems Telecommunications (ITST)*, pages 1–7. IEEE, 10 2018.

[30] Pan Hui, Jon Crowcroft, and Eiko Yoneki. BUBBLE Rap: Social-Based Forwarding in Delay-Tolerant Networks. *IEEE Transactions on Mobile Computing*, 10(11):1576–1589, 11 2011.

[31] Pedro Dinis António Cardoso Gomes. Social Routing Protocol for Delay Tolerant Networks. Master's thesis, IST/UL, September 2016.

[32] Waldir Moreira, Paulo Mendes, and Susana Sargento. Opportunistic routing based on daily routines. In *2012 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1–6. IEEE, 6 2012.

[33] Waldir Moreira, Paulo Mendes, and Susana Sargento. Social-aware opportunistic routing protocol based on user's interactions and interests. In *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, volume 129, pages 100–115, 2014.

[34] Waldir Moreira and Paulo Mendes. Impact of human behavior on social opportunistic forwarding. *Ad Hoc Networks*, 25:293–302, 2 2015.

[35] Jens Dede, Anna Forster, Enrique Hernandez-Orallo, Jorge Herrera-Tapia, Koojana Kuladinithi, Vishnupriya Kuppusamy, Pietro Manzoni, Anas bin Muslim, Asanga Udugama, and Zeynep Vatandas. Simulating Opportunistic Networks: Survey and Future Directions. *IEEE Communications Surveys & Tutorials*, 20(2):1547–1573, 22 2018.

[36] Ari Keränen, Jörg Ott, and Teemu Kärkkäinen. The ONE simulator for DTN protocol evaluation. In *Proceedings of the Second International ICST Conference on Simulation Tools and Techniques*. ICST, 2009.

[37] GitHub - akeranen/the-one: The Opportunistic Network Environment simulator. URL `https://github.com/akeranen/the-one` Accessed: 2019-06-16.

[38] Elizabete De Barros Moreira. Encaminhamento Baseado em Localização para Redes Tolerantes a Atrasos. Master's thesis, IST, 2017.