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

João Francisco Ribeiro Silva

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. However, simulation results show that the chosen approach did not improve the algorithm's performance.

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

Introduction

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 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). According to [4], in 2018 there were 7 billion active IoT devices worldwide, in 2019 there were 26.66 billion and during 2020 31 billion new devices are estimated to be installed. 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.

State of the Art

Overview

Delay-Tolerant Networks 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) [5] 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 [6]. Figure 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.

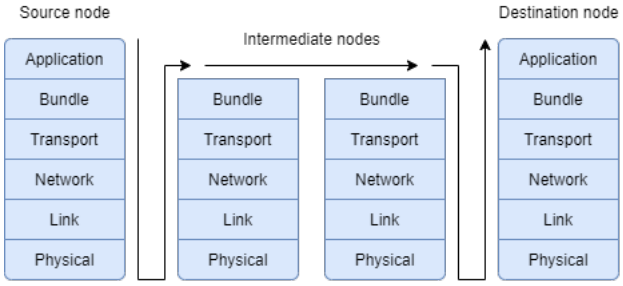

Figure 1: Architecture of a DTN Network (adapted from [1])

Routing in DTNs

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. In contrast, 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 [7].

Since routing is done using the Store-Carry-Forward paradigm, the difference between routing protocols relies on the forwarding part. When deciding if a message should be relayed to another node, protocols can take into consideration a broad range of criteria.

Traditional DTN Routing Protocols

These protocols implement the basic functionalities required for a DTN to operate.

Spray and Wait The Spray and Wait Protocol [8] 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 [9] 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 This protocol is called Probabilistic Routing Protocol using History of Encounters and Transitivity (PRoPHET) and is defined in [10]. 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 reducing. Although this approach of assuming that nodes that have met often in the past will meet again in the future is not perfect [11], it does improve the delivery rates, network overhead and delays.

Geographical DTN 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.

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) [12]. 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 [13] 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 [14].

Social DTN Routing Protocols

Unlike the geographic protocols, where the location and direction of movement are intrinsic properties of each node, social properties need to be defined. Surveys [15], [16], [17] and [18] 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. This master's thesis will focus on using Community and Centrality to categorize social relations between nodes.

A **Community** can be defined as a group of people living in the same place or having a particular characteristic in common [19]. 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.

Centrality 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 [20]. 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 [21]. 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.

BubbleRap Protocol The BubbleRap Protocol, defined in [22], 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 [23]. 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 [24], 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 [24], 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 [23].

FinalComm Protocol The FinalComm Protocol described in [23] 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 will look into the other node's storage to check if there is more recent informant 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.

The ONE Simulator

The Opportunistic Networking Environment (ONE) Simulator, proposed in [25], 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 [26], it allows the integration of complementary functionalities developed by any user.

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 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 Shortest Path Map-Based Movement Movement (SPMBM), Route Map-Based Movement (RMBM) and Working Day Movement (WDM). Nodes assigned the SPMBM 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. Nodes use pre-determined routes to follow. This is used to simulate the movements of public transports, such as trains and buses. The WDM model is used by most of the nodes in this work. Each of these nodes acts as a person with special features: each has a randomly generated home and is assigned an office from predefined list. 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.

Social and Geographical Metrics Evaluation

Merging Socially and Geographically Aware Protocols

The starting points to develop the SocialGeoProtocol were FinalComm as the Social protocol and Motion Vector (MoVe) 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.

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.

To illustrate the operation of the 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.

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 B already received message *m* A will not need to forward or replicate it. After these initial checks, each metric under analysis is examined with its specific algorithm.

Centrality 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.

Community 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 **??**. This is how nodes become familiar with each other and was already implemented and studied in great detail in [23], 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*.

Geographic 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 2, 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 represent the distance towards the destination. Using simple trigonometry described in equation 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.

$$\alpha = \arccos \frac{\langle vA, AD \rangle}{\|vA\|.\|AD\|} \tag{1}$$

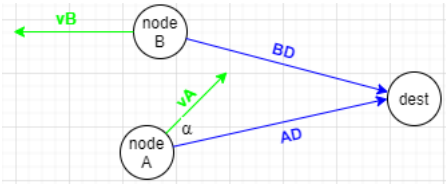

Figure 2: Demonstration of the angleMove routine, used within the geographic part of the SocialGeoProtocol

In order to combine these three algorithms, 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 2, and the result of that calculation was the one that ultimately made the routing decision. In equation 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.

$$metricsCombined = (commF \times commWeight) + (centrF \times centrWeight) + (geoF \times geoWeight) \tag{2}$$

Simulation Settings

A total of 223 nodes were created in The ONE Simulator, 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 WDM model. 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.

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.

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.

Initial Tests

The first tests were made using the weights in equation 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 1 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.

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

Table 1: Initial tests designed to assert the performance of each metric when using them with a standalone approach and combined with others

The results of these 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.

Social-Geographic Routing Protocol

Cascading the metrics

The results of the initial tests showed that a linear combination of the metrics did not prove to be the most useful. Therefore, equation 2 was disregarded and a new iteration of the protocol was designed by cascading the metrics.

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. The order in which the algorithm evaluates each metric was defined based on the simulation results obtained earlier. Since the community metric was the one that performed better when considered alone, it was selected to be the first to be analyzed, followed by the geographical one and only then by the centrality metric.

To test this new version of the SocialGeoProtocol, the simulation settings remained the same as before, with a few exceptions: 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 2.

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

Table 2: Tests using the cascading metrics on scenarios with different number of WDM nodes, meeting locations and offices.

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. 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).

Initial Spray phase

The addition of an initial spray phase had already proven to increase the performance of the geographical protocol described in [27]. 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 as described earlier. 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.

To test this hypothesis, the simulation settings used were the same as described in the previous section, except for the number of meeting spots, which was set to 5 and 20, as displayed in table 3 along with the other relevant settings.

Test number	#meet 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

Table 3: Tests using the initial spray phase on scenarios with different number of WDM nodes, meeting locations and offices.

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 3 highlights the marginally better 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.

Conclusions and 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.

Moving away from a linear combination of the metrics and cascading them started to improve the protocol's performance. For future work, it would bring even more insight to change the order in which the metrics were evaluated, while keeping track of how many times each one is used. 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.

References

[1] Warthman, Forrest: *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*. ISSN 1913-8989. URL `https://www.nasa.gov/content/dtn` Accessed: 2020-08-18.

[3] Burleigh, S, A Hooke, L Torgerson, R Durst, K Scott, K Fall, and H Weiss: *Delaty-Tolerant Networking Architecture. RFC 4838*, 2007. `https://ietf.org/rfc/rfc4838.html`.

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

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

[7] Jain, Sushant, 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, ISBN 1581138628. `http://portal.acm.org/citation.cfm?doid=1015467.1015484`.

[8] Spyropoulos, Thrasyvoulos, 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, ISBN 1595930264. `http://portal.acm.org/citation.cfm?doid=1080139.1080143`.

[9] Spyropoulos, T., 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, February 2008, ISSN 1063-6692. `http://ieeexplore.ieee.org/document/4430784/`.

[10] Lindgren, Anders, Avri Doria, and Olov Schelén: *Probabilistic routing in intermittently connected networks*. ACM SIGMOBILE Mobile Computing and Communications Review, 7(3):19, July 2003, ISSN 15591662. `http://portal.acm.org/citation.cfm?doid=961268.961272`.

[11] Nguyen, Hoang Anh, 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, June 2007, ISBN 978-1-4244-0992-1. http://ieeexplore.ieee.org/document/4351696/.

[12] Karp, Brad 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, ISBN 1581131976. http://portal.acm.org/citation.cfm?doid=345910.345953.

[13] LeBrun, Jason, 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, ISBN 0-7803-8887-9. http://ieeexplore.ieee.org/document/1543743/.

[14] Gong, Jiayu, 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. http://ieeexplore.ieee.org/document/4278994/.

[15] 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, 2014, ISSN 1553-877X. http://ieeexplore.ieee.org/document/6512845/.

[16] Zhu, Ying, 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, 2013, ISSN 1553-877X. http://ieeexplore.ieee.org/document/6177189/.

[17] Wei, Kaimin, 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, June 2014, ISSN 1045-9219. http://ieeexplore.ieee.org/document/6636888/.

[18] Xia, Feng, Li Liu, Jie Li, Jianhua Ma, and Athanasios V. Vasilakos: *Socially Aware Networking: A Survey*. IEEE Systems Journal, 9(3):904–921, September 2015, ISSN 1932-8184. http://ieeexplore.ieee.org/document/6619433/.

[19] Stevenson, Angus.: *Oxford dictionary of English*. Oxford University Press, 2010, ISBN 9780191727665. https://www.oxfordreference.com/view/10.1093/acref/9780199571123.001.0001/acref-9780199571123.

[20] Freeman, Linton C.: *Centrality in social networks conceptual clarification*. Social Networks, 1(3):215–239, January 1978, ISSN 0378-8733. https://www.sciencedirect.com/science/article/abs/pii/0378873378900217.

[21] Magaia, Naércio, Alexandre P. Francisco, Paulo Pereira, and Miguel Correia: *Betweenness centrality in Delay Tolerant Networks: A survey*. Ad Hoc Networks, 33:284–305, October 2015, ISSN 15708705. https://linkinghub.elsevier.com/retrieve/pii/S157087051500092X.

[22] Hui, Pan, Jon Crowcroft, and Eiko Yoneki: *BUBBLE Rap: Social-Based Forwarding in Delay-Tolerant Networks*. IEEE Transactions on Mobile Computing, 10(11):1576–1589, November 2011, ISSN 1536-1233. http://ieeexplore.ieee.org/document/5677535/.

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

[24] Moreira, Waldir, 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, June 2012, ISBN 978-1-4673-1239-4. http://ieeexplore.ieee.org/document/6263749/.

[25] Keränen, Ari, 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, ISBN 978-963-9799-45-5. http://eudl.eu/doi/10.4108/ICST.SIMUTOOLS2009.5674.

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

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