Efficient and Flexible Geocasting for Opportunistic Networks

Aydin Rajaei

Foundation of Software Systems
Department of Informatics
University of Sussex

A thesis submitted, on September 2016, in partial fulfilment of the requirements for the degree of Doctor of Philosophy (PhD) in the School of Engineering and Informatics of the University of Sussex.

Dedicated to my lovely wife

and

my dear parents

Abstract

With the proliferation of smartphones and their advanced connectivity capabilities, opportunistic networks have gained a lot of traction during the past years; they are suitable for increasing network capacity and sharing ephemeral, localised content. They can also offload traffic from cellular networks to device-to-device ones, when cellular networks are heavily stressed. Opportunistic networks can play a crucial role in communication scenarios where the network infrastructure is inaccessible due to natural disasters, large scale terrorist attacks or government censorship. Geocasting, where messages are destined to specific locations (casts) instead of explicitly identified devices, has a large potential in real world opportunistic networks, however it has attracted little attention in the context of opportunistic networking. In this thesis, we propose Geocasting Spray And Flood (GSAF), a simple but efficient and flexible geocasting protocol for opportunistic, delay tolerant networks. GSAF follows a simple but elegant and flexible approach where messages take random walks towards the destination cast. Messages that follow directions away from the cast are extinct when the device buffer gets full, freeing space for new messages to be delivered. In GSAF, casts do not have to be pre-defined; instead users can route messages to arbitrarily defined casts. Also, the addressed cast is flexible in comparison to other approaches and can take complex shapes in the network. DA-GSAF as the direction aware version of the GSAF is proposed as well which use location information to aid routing decisions in the GSAF. Extensive evaluation shows that GSAF and DA-GSAF are significantly more efficient than existing solutions, in terms of message delivery ratio and latency as well as network overhead.

Acknowledgments

I would like to thank my first supervisor, Prof. Ian Wakeman for his invaluable support, guidance and constructive criticism during my studies in the Sussex university over the last four years. His knowledge and support is always encouraging and inspiring.

Special thanks to my second supervisor, Dr. George Parisis for his great support, endless guidance and supervision which always inspired me to achieve the impossible. I will never forget his endless support during my studies in Sussex university.

I would like to thank my former supervisor, Dr. Dan Chalmers, who was a true mentor that supported, guided and helped me throughout my studies in Sussex university. Without his trust, this research would have never been achieved and completed.

Many thanks to my examiners, Dr. Ian Mackie and Dr. Iain Phillips who positively criticised the downsides of my research and approved the strong aspects of my work during the examination procedure. Their suggestions and constructive criticism would be a courage for my future research path.

Moreover, I would like to extend my thanks to the members of the "Foundation of Software Systems" research group for their outstanding support and friendship throughout the past four years. Finally, many thanks to the personnel and members of the Department of Informatics who always been supportive and helpful during my studies in Sussex university.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

If you want to invest for a year, plant some wheat. If you want to invest for a decade, plant a tree. And if you want to invest for a century, educate and train a number of people.
Amir Kabir
Prime Minister of IRAN (1807 - 1852)

The proliferation of smartphones and their long- and short-range connectivity capabilities have made the deployment of opportunistic, delay-tolerant networks (DTNs) [1] [2] reality [3] [4]. Wireless technologies, such as LTE, WiFi, WiFi-Direct and Bluetooth, allow smartphones to access the Internet as well as communicate with devices within their range, in an ad-hoc, peer-to-peer fashion [3] [5]. Opportunistic networks have gained a lot of traction during the past years; they can increase network capacity [6] [7] and be used for sharing ephemeral, localised content [8]. They are also suitable for offloading traffic from cellular networks to device-to-device (D2D) ones, whose formation is assisted by cellular providers [9] [10], who have strong incentives to do so when their networks are heavily stressed [11] [12]. Equally importantly, opportunistic networks can play a crucial role in communication scenarios where the network infrastructure is (partially or fully) inaccessible due to natural disasters, large-scale terrorist attacks or government censorship. They can also be the means for (localised) communication when the network infrastructure is not trusted. For example, FireChat [13] has been extensively used during the recent protests in Hong Kong (500,000 downloaded the application in Hong Kong alone during the first two weeks of the protests).

In most of the scenarios described above, communication and content dissemination is geographically confined (e.g. within a city or a region where a natural disaster took place or a part of the city where protesters demonstrate). Apart from being able to send messages to a specific device (unicasting), it is also crucial to be able to route messages to geographical locations (*geocasting*) covered by the network. Effective geocasting has a large potential in the real world use of opportunistic networks: (1) geographical notification for emergency situations, such as fire and natural disasters; (2) location targeted advertising where a large volume of users (more than hundred thousands) is concentrated at specific locations (e.g. open festival venues or large stadiums) to attend festivals, sports events or to participate in a demonstration; (3) geographically restricted service discovery. These geographical locations (*casts*) may be pre-defined, even before a network is deployed, or specified by the sender for each message. The temporal aspect is also relevant to geocasting, apart from the spatial one; destination nodes must receive a message before it expires; e.g. for a notification that becomes invalid in a natural disaster scenario.

Unicasting has been extensively studied in the context of DTNs [14], but existing protocols cannot support geocasting, given that unicast protocols route messages to specific devices, which are explicitly identified by unique endpoint identifiers. However, little attention has been paid to geocasting, which has mostly been studied in the context of Mobile Ad Hoc Networks (MANETs) [15]. MANETs present radically different properties compared to opportunistic networks. In MANETs, connectivity (as well as the overall network topology) among mobile nodes is rather stable; no such assumptions can be made for DTNs, where mobility is high and connectivity very intermittent. As a result, no end-to-end paths among all nodes exist at all times and no node knows the network topology, which constantly changes. Hence, existing geocasting protocols for MANETs are not suitable for opportunistic networks.

A number of geocast routing protocols for opportunistic networks have been recently proposed, however they all come with significant limitations that the proposed research aims at overcoming.

In geocasting protocols like the ones proposed in [16, 17], upon encountering each other, mobile devices exchange explicit information about their location. In other protocols, such as the ones proposed in [18] and [19], devices exchange aggregate location information (in the form of cast visiting probabilities) or information that is used to collaboratively build mobility maps, respectively. Exchanging any kind of location information upon any node encounter (especially in crowded environments) requires network resources that are rather scarce in opportunistic networks. Frequently exchanging data also affects devices' battery lifetime.

Exchanging location information (e.g. current location, or future destination(s)) raises significant privacy concerns as in many scenarios users would very much prefer to not disclose any information about their whereabouts (e.g. in scenarios where trust cannot be assumed among network users). Indeed, according to the studies [20], around 40% of users deny to share their network connection due to the privacy concerns.

In [18, 19], after exchanging location information with an encountered device, a device would undergo expensive computations which could drain the battery very quickly.

In [16, 17] the network is partitioned into two layers, requiring either a third party to perform the partitioning or a distributed consensus protocol for electing nodes to be in each of these layers (consuming bandwidth especially under high node churn). Such an approach would be impractical in scenarios where a network is spontaneously created without the intervention of a service provider.

The way that the destination of a message (a geographic area we refer to as the cast) is defined is important with respect to the efficiency of a protocol and its privacy preserving characteristics. Some protocols [16–18] can only operate on top of predefined non-overlapped cells; i.e. they use pre-defined cells to divide the network into smaller areas to perform routing decisions. In such protocols it is unclear who is responsible for defining the casts and therefore they are not practical for completely distributed communication scenarios. Most geocasting protocols [8, 16–18] use circular casts. For example, in [8], the centre of the cast is the device that creates and disseminates the message in the network. There are two problems with such an approach. The granularity of the defined destination area is coarse, therefore expanding the destination area (e.g. to cover a number of users residing in a specific building) can only happen through increasing the radius of the defined circle. This inevitably leads to significantly increasing the destination area and as a result a much larger number of devices may get involved with routing although they were never meant to be potential recipients of a geocast message (essentially because the message is flooded within the coarsely defined destination area). This is very inefficient in terms of computational and network overhead, as discussed above. Apart from performance related ramifications, coarse-grained cast definition may result in privacy leaks in scenarios where messages should be directed to specific users in specific areas. Finally, in emergency scenarios, it is important to direct messages to users residing in specific areas to avoid spreading panic.

1.1 Contribution

1.1.1 Analysing assumptions and design challenges

The main goal of this research is to find a comprehensive and efficient solution for geocasting in opportunistic networks that overcomes the limitations identified above. The design of the proposed geocasting protocols has been driven by an extensive analysis of the nature and topological characteristics of opportunistic networks as well as the associated challenges and trade-offs with respect to privacy, energy consumption, CPU and memory limitations as well as user mobility.

1.1.2 Geocast routing protocols

In this thesis we present **Geocasting Spray And Flood (GSAF)**[1], a simple but efficient and flexible geocasting protocol for opportunistic, delay-tolerant networks, which overcomes limitations of existing approaches. Contrary to protocols where casts must be pre-defined [8, 19] or defined as circles (by defining a centre point and a radius) [8, 16–18] or network should be divided into pre-defined non-overlapped equal cells [16–18], GSAF allows for flexible definition of arbitrary casts based on a set of coordinates. The sender defines the cast, the cast definition is carried in the routed message and other nodes only check whether they reside within the defined cast. This flexibility is required by many communication scenarios where fine-grained specification of destination casts is crucial (e.g. fine-grained emergency notifications to avoid widespread panic). Moreover, in our approach (in contrast to [8]) a device can send a message in a cast even if it does not reside in it.

With GSAF, devices do not exchange any location-related information, thus preserving users' privacy, and take routing decisions autonomously. By following a spray-and-flood approach, we can efficiently geocast messages with significantly improving the delivery ratios and reducing the delivery latency compared to existing approaches (see Chapter 5). GSAF follows a simple but elegant approach where messages take random walks towards the destination cast. Messages that follow directions away from the cast are extinct when the device buffer gets full, freeing space for new messages to be delivered. In brief, message dissemination is as follows: a node receiving a message only has to check whether it is a destination node (i.e. it resides within the destination cast, defined in the message) or not. This requires devices' location services and presents a well-known trade-off with respect to the accuracy of the reported location (which, in turn, affects the granularity of cast definition) and battery consumption[2]. In the latter case, a device carries and forwards the message to other nodes based on a ticketing mechanism, inspired from [21]. When a message reaches a cast, it is disseminated through controlled flooding and can never exit the borders of the destination cast. Expired messages are immediately deleted.

Moreover, we introduce **Direction Aware - Geocasting Spray And Flood (DA-GSAF)**[3], an extended version of GSAF, which exploits information about users' direction to assist GSAF's routing decisions. With DA-GSAF, a node checks whether its current direction is towards the destination cast of a message and only transfers messages to encountering devices only if it is heading off the destination cast and the encountered nodes is heading towards it. This is examined in a privacy preserving fashion. The source

[1] See § 3.2.2.A.

[2] In practice, in urban areas, a location accuracy of 20 - 50 meters, which is fine for many geocasting scenarios, can be achieved without GPS. If the network infrastructure is inaccessible, then one has to rely on GPS or collaborative localisation approaches, but our approach makes very light use of location services.

[3] See § 3.2.2.B.

device will only share the destination of a message with an encountered node, which in turn checks its own direction and responds accordingly; no historical data or future destinations are shared between the encountered devices.

1.1.3 Definition of geographical area

In our approach, cast definition is fine-grained. Casts are polygons (see § 3.2.1) which are defined as a set of pairs of two-dimensional coordinates. This provides more efficient usage of network and computational resources. Geocasting requires checking whether a device resides in or out of a cast whenever it receives a new message. This check must be computationally efficient. Our approach is only marginally computationally heavier compared to testing whether a device is within the boundaries defined by a circle, by incorporating the "crossing test" that is widely used in computer graphics (as described in Chapter 3).

1.1.4 The ONE simulator software

The Opportunistic Network Environment (ONE) simulator is an open source and widely-used simulator. It is considered as the best tool to simulate the performance of opportunistic networks. It was developed by Aalto University (Finland) and being maintained in cooperation between Aalto University and Technische Universität München [22]. ONE is merely able to measure the performance of unicast opportunistic routing protocols. However, due to the requirements, we modified the core of the ONE simulator software to enable the simulation of geocasting scenarios on it. Details are discussed and explained in § 4.3 .

1.1.5 Extensive performance evaluation on all related work

Enabling ONE to simulate geocasting scenarios in opportunistic networks, provided a unique chance to study and compare the performance of all existing geocasting protocols. We have designed and simulated couple of scenarios (see Chapter 5) to investigate the performance of related work against our solution in details. This provided a full picture of studies beyond this subject which have never done previously.

1.1.6 Study on the impact of buffer scheduling policies

In the quest to find more performance in the network, we studied the impact of buffer scheduling policies on the geocasting routing protocols. In this research, we investigate this impact on the various parameters of the network which led to discover an interesting result that is discussed in § 5.4 .

1.1.7 Study on the impact of cast size to the network performance

The size of cast may vary based on the various applications or the users' need. It is important to study the impact of the cast size to discover if it is better to use smaller cast or a larger cast and how it may affect the overall performance. The result is presented in § 5.6 .

1.2 Thesis Structure

Following is the overall structure of this thesis:

Chapter 2: Background & Literature Review

In chapter two, first a background of the research is presented and discussed to identify the existing challenges in opportunistic networks. Then, state of the art of the research beyond the geocasting addressing methodology is presented and discussed. This chapter presents an overall picture to discuss, where and why our research stands for.

Chapter 3: Design

In this chapter we discuss the assumptions and remaining challenges of geocasting in opportunistic networks, and provide a detailed explanation of the designed solution. Moreover, GSAF, DA-GSAF and their design is explained in detail.

Chapter 4: Implementation

Chapter four provides an introduction about the ONE simulator, its abilities and its deficits. We discuss the requirements and provide details about the implantation of extra functionalities, routing protocols and measurement techniques. The implementation phase consists of more than ∼10000 lines of code.

Chapter 5: Evaluation & Result Analysis

In chapter five we evaluate the performance of GSAF and DA-GSAF against the existing related work. This chapter presents a wide range of simulation results and their respective data analysis. We deeply discuss the causes of the outcome as well.

Chapter 6: Conclusion

Finally, chapter six concludes the research and presents a path for future work.

1.3 Published and Presented Work

Published and presented work during the evolution of this research:

Publication

Aydin Rajaei, Dan Chalmers, Ian Wakeman, and George Parisis. "GSAF: Efficient and flexible geocasting for opportunistic networks". in Proceedings of the *2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pages 1-9. IEEE, 2016.

Presentation

1. "Geocasting in Opportunistic Networks" - in *Foundation of Software Systems*, University of Sussex, Brighton, UK - June 2016.

2. "GSAF: Efficient and flexible geocasting for opportunistic networks" - in *Next Generation Networking and Multi-Service Networks workshop*, Cosners' House, Abingdon, Oxford, UK - July 2016.

Poster Presentation

"Geocasting in Delay Tolerant Networks" - Honourable Mention Award for the best postgraduate poster, In *School of Engineering and Informatics*, University of Sussex, Brighton, UK - July 2014.

Chapter 2

Background & Literature Review

There are four tasks to consider by the wise people: learning science, preserving it, teaching it and utilising it.
Muhammad Rasul Allah
The Prophet of Islam (570 - 632)

This chapter presents a background on how Opportunistic, Delay Tolerant Networks have developed and took shape, during the last decade. Also it provides details about how DTNs work and how routing of messages takes place in the network. In addition, the literature and the latest trending research subjects are discussed and reviewed later in this chapter.

2.1 Mobile Ad hoc Networks (MANETs)

In mobile ad hoc networks (MANETs), mobile devices connect to each other wirelessly to form a decentralised network without the use of any fixed infrastructure (e.g. wireless routers or access points). Nodes (mobile devices) help each other to route and forward packets in the network. They establish a connection between each other to find a path to transfer packets from the sender to the recipient node. Dynamic network topology is the main characteristic of MANETs, since nodes can move freely in the network. Each node is connected directly to its neighbours that are located within its communication range using one or more short-range wireless technologies (i.e. Bluetooth or WiFi-Direct). The node connects to other users in order to access data in the network and may get disconnected if it gets out of the connectivity range. As shown in Figure 2.1, nodes A and C are not located in each other's communication range and therefore cannot communicate directly. However, the connection between them can be established via node B in the network. This is the multi-hop communication scheme which is used in MANETs.

In MANETs, it is assumed that there is an end-to-end path between any node in the network at all times. Thus, routing protocols continuously discover end-to-end paths among

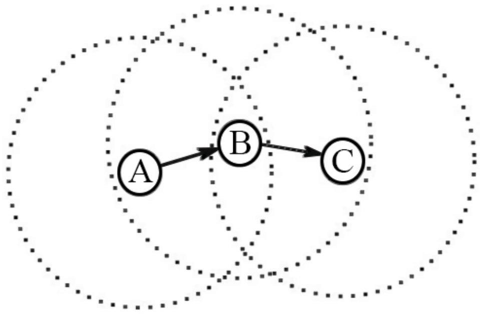


Figure 2.1: Multi-hop connection in MANETs [28]

communicating devices to forward packets to their recipients in the network. According to Conti and Giordano [23], MANET routing protocols are divided into two different categories: *proactive* and *reactive*. Proactive protocols, like OLSR (Optimized Link State Routing) [24], collect information from all paths between nodes and try to update the existing routes' chart before routing packets in the network. On the other hand, reactive protocols such as DSR (Dynamic Source Routing) [25] and AODV (Ad hoc On-Demand Distance Vector routing) [26], try to find a route between two nodes whenever this is required. These routing protocols work well when end-to-end paths from source to destination nodes are always available [27]. However, if a node in the "delivery path" gets disconnected from the network, the path ceases to exist and as a result packets that were following this path are lost or queued extensively until a new path is established. There are cases where end-to-end connection between nodes cannot be guaranteed in real world MANETs, and this is their main limitation.

Soon after the introduction of MANETs, it has been realised that they could not work properly under challenging conditions (e.g. frequent disconnection between the nodes and long delays in the order of seconds or minutes) because of their nature. Kevin Fall [1] has discussed four main "challenged environments" where MANETs and their existing routing protocols are not appropriate:

Exotic Media Networks:

Near Earth satellite communications and space connections, with long delays in the order of minutes, are examples of exotic communication media networks. Predictable connectivity interruptions (due to orbiting patterns, weather and environmental conditions) and high delays are the main characteristics of this kind of networks which affect the overall performance.

Terrestrial Mobile Networks:

These infrastructure-less networks may be get partitioned in a predictable manner, or due to nodes' mobility, or suffer by unexpected signal fluctuations. They should communicate under conditions which is well equipped against delays and disconnections.

Military Ad Hoc Networks:
Military networks are used in situations where fast data transmission and reliable communication are very important. They should be set up very fast in infrastructure-less environments. They may operate in hostile environments where intentional jamming may cause severe connectivity issues. As they transmit important data within an unstable environment, they should be able to operate under mentioned extreme conditions.

Sensor Networks:
Nodes in sensor networks are characterised by limited memory, power and process capabilities. Also, redundancy is high as there may be thousands of nodes in a network. As each node has a limited amount of power, it cannot continuously communicate with other nodes. Moreover, the amount of transferred data is small that can be transmitted in a limited given time. Hence it is more efficient to perform short window communication every once in a while, in a predicted manner. Mentioned factors may result in high delays due to traffic, packet loss and nodes' disconnection.

As discussed, MANET routing solutions are not suitable for environments where delay is high and disconnections are frequent. Efforts to overcome the challenges in these kinds of environments during the last decade led to the development of Opportunistic Networks.

2.2 Opportunistic Networks

Initial efforts in developing Interplanetary Networks (IPNs) [29] in 1998 resulted in the emergence of Delay Tolerant Networks (DTNs). The IPN project aimed to establish an Internet addressable connection between Earth and the other planets of our solar system. Deep space communication has special characteristics such as frequent disconnections and high communication delays, where standard Internet protocols fail to perform under the mentioned conditions. Soon, it has been realised that development in this area could benefit other kinds of networks such as deep water communication or terrestrial networks with similar characteristics (high delay, no end-to-end connection between the users most of the time, and unknown topology of the network). All of these networks are classified in a broader group named Opportunistic Networks (also called Intermittently Connected Networks - ICNs) [30]. In this thesis we refer to these networks as "Opportunistic Networks".

As mentioned in § 2.1, MANETs present radically different properties compared to the opportunistic networks. In MANETs, connectivity (as well as the overall network topology) among mobile nodes is rather stable; no such assumptions can be made for opportunistic

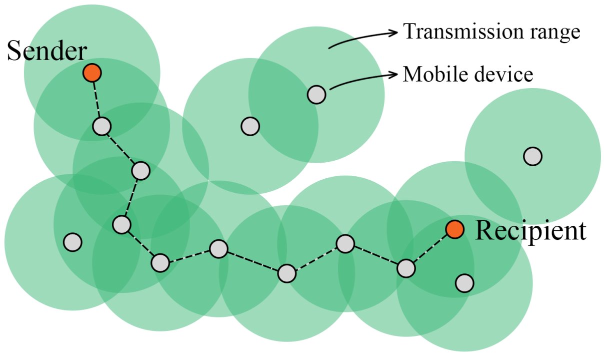


Figure 2.2: An example of Opportunistic Network

networks, where mobility is high and connectivity is very intermittent. As a result, no end-to-end paths among all nodes exist at all times and no node knows the network topology, which constantly changes. None of the existing protocols for MANETs are suitable for opportunistic networks, due to the mentioned difference between these two approaches [28]. MANETs use a *receive and forward* scheme to transfer data inside the network (packets would be dropped if connectivity between two nodes ceased to exist). However, opportunistic networks use the *store - carry - forward* scheme to overcome the problem. A user node stores the received message in its buffer, whenever there is no connection to the next node in the network, and it carries the message until it establishes a new connection (with another node) to forward it to. Figure 2.2 shows an example of an opportunistic network. Mobile devices use the connection window to exchange messages when they come across each other, trying to eventually deliver data to the recipient.

As discussed, opportunistic networks follow an unusual approach (*store - carry - forward*) to overcome intermittent connectivity in the network. They pack the data payload with extra flags and identifiers to carry and forward them through the network. Each pack is called "Bundle" or "Message". A bundle has an Endpoint Identifier (EID), which specifies the ID of the addressed recipient in the network. Since the global topology is unknown to the network devices, no device knows the existence and the address of all other users in the network. Hence, whenever a user receives a new bundle, it compares its own ID with the recipient EID of the bundle and identifies if the bundle is addressed to itself. Addressing and routing requires the implementation of extra functionality in contrast to the standard TCP/IP stack in conventional networks. The extra functionality is implemented at the bundle layer [30]. The DTN protocol stack is shown in Figure 2.3; it consists of a bundle layer above the transport layer [30]. The DTN architecture (with the help of the bundle layer) supports interoperability between different networks and provides ways to route and forward messages between them based on the *store-carry-forward* scheme. Following is a short explanation of the architecture of opportunistic networks [1, 2].

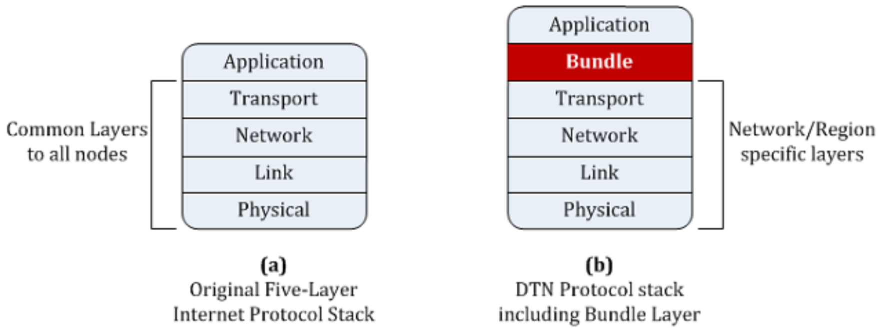


Figure 2.3: DTN protocol stack [30]

2.2.1 Naming and Addressing

Naming is a way to identify the nodes of the network and addressing is key to efficient routing in it. In opportunistic networks, nodes have identifiers that are used in the context of the bundle protocol that provides the basic message delivery service [2]. As mentioned, network nodes compare a message EID with their own ID to find out whether they are the recipient of a received message or not.

2.2.2 Bundles

The size of DTN messages are larger than regular TCP/IP packets in order to enhance the routing procedure and improve data delivery (i.e. around 512 – 1024 KB). This is an important key point in opportunistic networks which have long round-trip time, due to their nature. The actual data payload is packed with usual IP header information, such as source, destination, flags, length, etc. In addition, it includes fields special to the DTN bundle protocol, such as the custodian EID, timestamp, lifetime, etc [2]. This information provides support for various functionality and controls the transfer process in opportunistic networks. For instance, nodes can easily identify and discard messages queued for a long time in their buffer based on the "lifetime" field, in the respective bundle header. Also, priorities could be applied to bundles by setting flags in the header. The structure of the DTN bundle header is explained in details in [2].

2.2.3 Routing

Opportunistic networks are designed for environments where communication between users is prone to large delays and communication disruption most of the time. Hence, there may be no connection between a sender and a receiver most of the time. Senders use any opportunity to forward messages to a neighbour node; each connection opportunity is being characterised based on start and end time, direction, speed and capacity. All of these factors are expected to influence algorithms and routing protocol of the network. The research community has been looking at several challenging problems in this area:

1. Determination of the existence and predictability of contacts between users.
2. Efficient routing of messages to their destination, based on the context of each node.

3. Efficient use of each encounter (connection window between two neighbours).
4. Buffer management.

2.2.4 Custody Transfer and Reliability

A disconnection between a sender and a receiver in the network during data transfer is likely due to the nature of opportunistic networks. Messages in transit during the disconnection should be potentially retransmitted to the same or other nodes so that they reach their destination. The source of the packet is responsible to resend data to the destination in the standard TCP/IP network architecture. However, the source of the message is not available most of the time in opportunistic networks, e.g. due to node mobility. In this case, messages should be sent and delivered following a hop by hop strategy; whenever a message gets lost in the transfer process, the previous node is responsible to resend it. The sender should keep the message in its buffer until the delivery is being acknowledged by the next hop in the path. The custody transfer refers to this kind of hop by hop message delivery confirmation in opportunistic networks. As mentioned, custody transfer is delegating responsibility to each node to make a reliable message transfer to the next node, and the idea is implemented to counter unreliable delivery and high loss rates. In other words, custody transfer has replaced the end-to-end delivery scheme with the more reliable hop-by-hop transfer approach for opportunistic networks.

2.2.5 Congestion and Flow Control

Congestion occurs when buffer becomes full in nodes of the network due to high traffic or low available memory. There are several options for nodes who experience congestion, such as stopping to accept traffic from other nodes, moving bundles to other nodes or even dropping unexpired bundles. Nodes could apply one or many of these options whenever they are facing memory shortage. Flow and congestion control could be a key point in memory and traffic management in opportunistic networks. Limiting the flow rate could cause high queuing time in message transfer, while limitless flow rate could fill up nodes' buffers quickly. Buffer scheduling and dropping policies are examples of the introduced solutions in term of flow and congestion control in this kind of environments.

2.2.6 Security

The Bundle layer in opportunistic networks sits on top of the standard Transport layer (see Figure 2.3) and inherits standard security provided by lower layers. However, opportunistic networks are vulnerable to Denial-of-Service (DoS) attacks. In this kind of attacks, a user node can disrupt the provided service by flooding unwanted messages as well as consuming the available network resources. The latter case may be critical in this type of networks due to the high cost of the network resources. Security concerns in opportunistic networks is an open research subject (e.g. see [14, 30]). However, it is beyond the scope of our work in this thesis.

2.3 Routing in Opportunistic Networks

The particular characteristics of opportunistic networks and the requirement to operate in infrastructure-less environments has led to special attention and interest from the research community during the last decade. Several approaches have been proposed to maximise the performance of opportunistic networks [14, 30]. Research on opportunistic networks have focused on various areas, such as application layer design, convergence layer design, routing, congestion control, flow control and security. The main focus of this thesis is on the addressing methodology and routing algorithms in opportunistic networks. First, we describe the two different classes of routing schemes for opportunistic networks; the *Deterministic* and *Dynamic* routing [31]:

Deterministic Routing:

In deterministic routing, it is assumed that the future movement of nodes and connections between them are completely known to all nodes in the network. All paths and connections are modelled based on the known information in the network. For instance in [32], the proposed routing protocol selects routes and delivers messages based on the available information about the mobility patterns of the nodes. Nodes are recording their motion behaviour and exchange respective data with each other. The routing protocol is building a known connection model and routes messages based on this model. Other works, such as [33] and [34], have proposed routing protocols based on similar assumptions about predictability of nodes' movement.

Such routing protocols only work in networks where mobility patterns of all nodes in the network are completely predictable and repeatable, such as in communication between space satellites. Satellites are spinning around the earth in a completely known orbit and therefore their location is constantly known. As a result, communication opportunities with other satellites can be precisely predicted. As mentioned earlier, deterministic routing protocols could be useful in this kind of environments. Nodes in many different types of opportunistic networks move dynamically and is therefore difficult to model their movements into repeatable patterns. our research focuses on dynamic routing.

Dynamic (Stochastic) Routing:

Dynamic routing protocols are used when movement of nodes is random, not predictable or not following any repeatable pattern in the network. Dynamic opportunistic routing protocols have been classified and surveyed in a number of

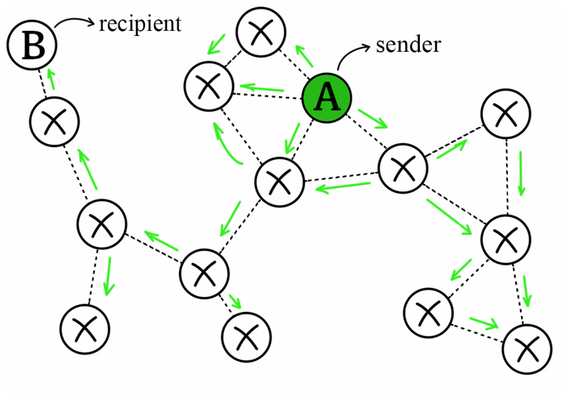


Figure 2.4: Epidemic forwarding in Opportunistic Networks

articles during the last decade [14, 30, 31, 35]. However, in this thesis we will be following the taxonomy recently introduced by Yue Cao and Zhili Sun [14]. The authors have classified researched works based on the addressing methodologies: *Unicast*, *Multicast* and *Anycast*.

2.3.1 Unicasting

In unicasting, packets are being transferred from a single sender to a single receiver. Unicast routing in opportunistic networks could be categorised into five subsets: *Epidemic*, *Estimation*, *Controlled Movements*, *Model* and *Coding* based approaches.

A. Epidemic based approaches

In epidemic routing there are no strategies to select a particular node to forward a message. Also no information about the shape of the network is assumed. In this case, a node will forward a message to all other nodes that are located within its communication range. This is an effective way to maximise the chances that a message will be delivered to its destination. For example, in Figure 2.4 node A wants to send a message to node B and it uses epidemic routing protocol. It is an effective way to find a path for delivering a message to node B. However, the message would be sent to every node of the network. This approach is not efficient with respect to network resources' utilisation.

The first epidemic approach for disconnected networks was proposed by Vahdat and Becker [27]. As mentioned, in this approach a sender would find the destination node by sending the message to every node in the network. When two nodes are within communication range, they exchange all buffered messages; this is how messages find their way to their destination. Since all messages are copied and exchanged, they quickly fill up all available buffer space in the network. This could cause serious buffer problems for the nodes that have limited amount of memory, and the networks that have high traffic load or low transmission speed. In addition, this cause high energy consumption on the mobile devices as they are usually try to blindly exchange all of their buffered messages.

The other approach in epidemic routing limits the message flow to the destination node's direction. The spraying protocol [36] routes messages based on the last known location of the destination node. This narrows down message flow, to the direction of the last known location of the recipient node. It is assumed that the destination node may be around the last known place.

The Spray and Wait protocol [21], clones messages to a limited (e.g. 3 to 5) number of nodes and assumes that one of these nodes would encounter the destination node. According to [14], Spray and Wait distributes the allowed number of copies to the encountering neighbour nodes, immediately, and waits until one of them (with a copy in hand) meets the recipient node. There are some other related work that use the epidemic forwarding approach such as [37] and [38]. Note that, spray-based routing [21] made the epidemic scheme far efficient than the basic one.

B. Estimation based approaches

In the estimation based approach, instead of blindly forwarding messages to the other users; nodes estimate the delivery chance of each neighbour and route messages to neighbours with higher delivery probabilities. If there is no node with high delivery possibility in the neighbourhood, the sender stores the packet and waits for future opportunities. Some estimation based routing protocols measure the delivery probability only based on the information about the next hop, while in other approaches, it is done based on the end-to-end path information.

PROPHET (Probabilistic Routing Protocol using History of Encounters and Transitivity) [39] is one of the main protocols that follows the estimation approach. PROPHET first estimates the delivery probability of each neighbour node based on its own algorithm, then it forwards the message to the node with the higher delivery probability among its current neighbours. Simulation results show that there is an improvement with PROPHET over the basic Epidemic [27] routing protocol.

There are other works on estimation based approach such as [40], [41] and [42]. All of these routing protocols operate based on the delivery probability of each next-hop node; each one of them has a different strategy to measure this probability. Generally in this approach, there is a performance increase over the regular epidemic routing protocol in terms of memory usage and delivery time; this is because these approaches use available information to measure the delivery probability for different paths, instead of blindly forwarding messages to all routes. With such an approach the number of message copies in the network is also decreased. Obviously, this results in better buffer management. On the other hand, such approaches require more computational resources for calculating delivery probabilities for each potential next-hop node for each message.

The estimation based approach is efficient and PROPHET is one of the benchmark routing protocols in opportunistic networks. However, there are some privacy concerns in this approach. Nodes exchange their context to calculate the delivery estimation rate in each encounter, while users may not interested to share their private information, such as current location or future destinations. Hence, using this kind of routing protocols rises privacy concerns for the users of the network.

C. Controlled Movements based approaches

This approach is very different from the other routing approaches in opportunistic networks. In all other approaches, routing and forwarding of messages does not influence the movement of nodes in the network. Users are moving freely and messages try to find their way by taking advantage of nodes' movement. If a node does not go towards the (potentially disconnected) destination, then the message will not be delivered to the recipient node. On the other hand, in controlled movements based approaches the movement of some nodes in the network is controlled so that upon request they move towards the destination of one or more messages. Messages can be delivered to disconnected areas by instructing a specific node to move towards them. For example, as shown in Figure 2.5, regions 1 and 2 are not connected to each other. Node A moves towards region 2, whenever it wants to deliver a message from region 1 to region 2 or vice versa. In other words, Node A is playing the role of the messenger, delivering messages between these two regions. This approach is also known as Message Ferrying (MF).

The NIMF (Node-Initiated Message Ferrying) and FMIF (Ferry-Initiated Message Ferrying) routing schemes [43] are examples of this type. In NIMF, a ferry node moves around

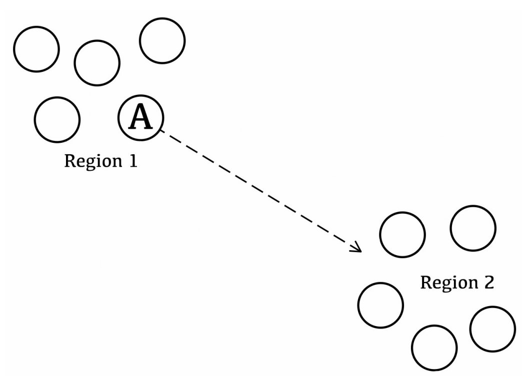


Figure 2.5: Controlled Movements based approach in Opportunistic Networks

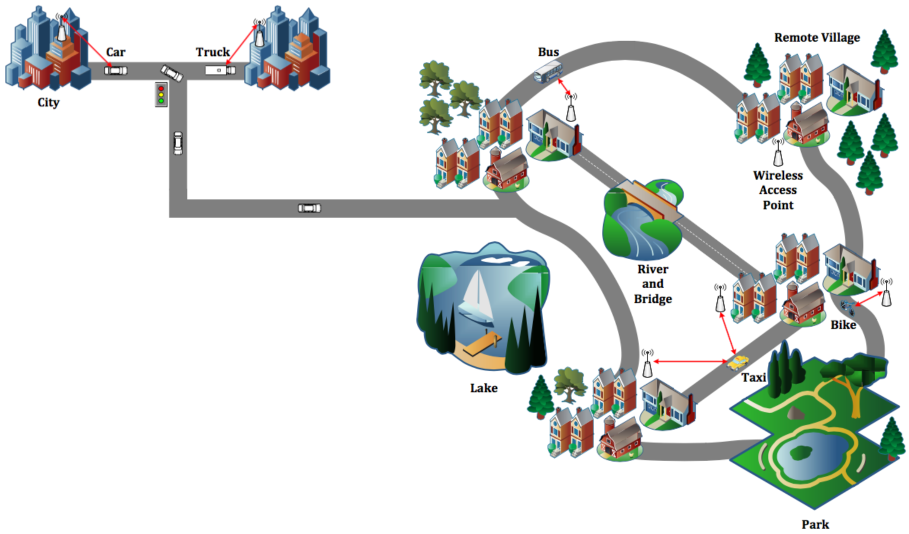


Figure 2.6: DakNet [30]

the network with known patterns. Any time it enters the communication range of sender nodes, it receives their messages. The ferry node will then hand the message to the destination based on its predefined movement path. It follows its known movement pattern to aid the routing procedure. Senders hand messages to the ferry when their destination is the same as the destination of the ferry. In FMIF, ferry nodes do not move based on predefined patterns. They move in response to node requests.

Solutions for connectivity in undeveloped regions, are based on this approach. Buses and taxis play the role of the ferry in this kind of networks. As shown in Figure 2.6, buses receive the messages from nodes in the city centre and carry them in their route to disconnected regions and vice versa. DakNet [44] is such an approach. There are other works based on node movement control scheme such as [45], [46] and [47]. Unfortunately, this approach is applicable in special types of opportunistic networks. Ferry nodes may not exist in common opportunistic networks and nodes' movement cannot be controlled in common scenarios.

D. Model based approaches

In the real world, people, cars, buses and trains (carrying network nodes) move. All these have specific movement patterns and head towards specific destinations; e.g. when walking along a street or travelling between cities. Thus their movement is not random and their mobility pattern could be estimated by examining their movements. Model-based approaches [30] try to route messages by understanding mobility patterns in real world environments.

MBR (Model-Based Routing) [48] uses the location of nodes to model their movement into patterns. In [48] the authors do not provide details about how MBR routing protocol works and their assumptions are not overly realistic. The work in [49] is similar to this approach.

The FVR routing protocol [50] uses the nodes' location information that is extracted from GPS to route messages inside the network. It classifies neighbour nodes' delivery probability based on their moving direction. HVR [51] is an extended version of FVR with an improved algorithm. Considering the routing strategy of recent two protocols, they could be successful in real world opportunistic deployments.

Model based routing approaches could be an effective solution to routing messages in real opportunistic networks. There are some other routing protocols that use location data of the nodes as a part of their routing solution such as [36]. Mobility pattern of the nodes could benefit the routing performance, while it could raise privacy concerns (due to shared movement pattern and location information). Nowadays, most of the mobile devices have integrated "location services" and this could enhance opportunistic networks to optimise their routing protocols by the information based on the location of nodes. Additionally, inferring mobility patterns may require extensive computational and network resources which are scarce in opportunistic networks.

E. Coding based approaches

The basic idea behind coding based approaches [52–56] is to use network coding to improve the chances that messages are reliably transferred from one node to other. However, message transfer is costly in opportunistic networks and message delivery could not be guaranteed when packet loss is excessive. Additionally, network coding requires extensive computational resources for encoding and decoding messages which could, in turn, result in higher latency and more energy consumption.

2.3.2 Multicasting

Multicasting in computer networks is about delivering data to a group of users, instead of just one user. In MANETs, destination groups can be easily found due to the fact that the topology is known. On the other hand, in opportunistic networks it is challenging to define and find each user inside the network as the topology is not wholly known to a single network node. The work in [57] is one of the first efforts to introduce multicasting in opportunistic networks. Multicasting in opportunistic networks can be categorised into two subsets [14]: Tree-based and Unicast-based multicasting.

A. Tree based approaches

In Tree-based approaches, the network is treated as a tree and each node is responsible to forward messages to its available next hop. For example in Figure 2.7, if we would like to multicast a message from A to nodes D, E, F and G, node B is responsible to forward

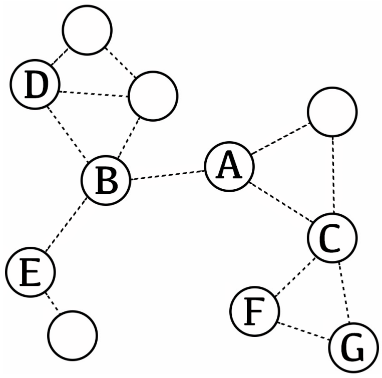


Figure 2.7: Tree based multicast approach in Opportunistic Networks

the message to nodes D and E. Also, node C is responsible to forward to F and G. The works in [58–61] can be categorised as tree based approaches. However, it is challenging to maintain the multicast tree in opportunistic networks due to the unknown topology and node mobility.

B. Unicast-based approaches

Another solution to multicasting in opportunistic networks is to break down the group to single nodes and address the same message to each destination (multi-unicasting). This is a feasible approach for opportunistic multicasting and it needs further modification in comparison to the regular opportunistic unicasting routing protocols. All works in this area, such as [62–67], used unicasting approaches and modified them for multicasting in opportunistic networks.

2.3.3 Anycasting

Anycasting refers to the method where the source node tries to deliver a message to a member of the specific group. Anycasting in opportunistic networks is a topic which needs further investigation. There are a number of papers in this area, such as [68–70], that suggest an application use case for anycasting; request for service discovery from a group of nodes in the network, where mobile nodes can connect to a node who belongs to the group to collect the requested information. This can help the dissemination of information that resides within a group of nodes that are located in the various places across the network. A node can anycast a request to the known group of users who are a part of a distributed system and the nearest node in the group can deliver data based on the request.

2.4 Geocasting in Opportunistic Networks

Three addressing methodologies *(unicast, multicast and anycast)* have been investigated in the context of opportunistic networks. However, in many application scenarios location is an important factor. Nowadays, mobile devices utilise “location services” and they are able to specify their own location coordinates with reasonable accuracy. This creates the potential to investigate the implementation of a *fourth group* in the context of opportunistic networks; *geocasting*. In geocasting, messages are delivered to a group of nodes that are located in the same geographical area. The implementation of an effective geocast addressing methodology has a significant potential in the real world use of opportunistic networks:

1. Geographical notification for emergency situations such as fire alarms, natural disasters, earthquake rescue and awareness by police stations.
2. Location targeted advertising where a large volume of users is concentrated at specific locations (e.g. open festival venues or large stadiums) to attend music festivals, sports events or to participate in a demonstration.
3. Geographically restricted service discovery.

One of the important applications of opportunistic networks is in disaster scenarios, where 3G and 4G mobile networks are inaccessible due to e.g. an earthquake or tsunami and there is no connection between mobile devices. An opportunistic network could play an important role in this kind of scenarios by establishing a network connection between mobile devices to deliver crucial information to the end users. With the current addressing methodologies in opportunistic networks, information should be sent individually to each node or broadcast to all nodes in the network. However rescue notifications may vary between different regions. With an effective implementation, each area could be grouped based on the location and each user would get notifications specific only to its own area. Also, this would reduce the network traffic due to the unnecessary message broadcasting to all unwanted users associated with broadcasting.

The second main application of geocasting in opportunistic networks is for location targeted advertisements. There are scenarios where large amount of users gather in small areas for events, such as music festivals. Cellular carriers could not support load of hundred thousands of users in a small area and the quality of service reduces noticeably. In addition, when fans gather in stadiums they experience communication problems due to the characteristics of the environment [71] which prevents mobile reception [4, 72]. Opportunistic networks could be widely used in such scenarios to offload network traffic from the service providers. Moreover, geocasting could be used in such scenarios for location-based “advertisements” or “news” for network users.

The third possible application is geographically restricted service discovery, where mobile devices request services, such as context based data subscription, in opportunistic networks (e.g. custom-tailored advertisement and news). Subscriptions could be restricted to a geographical area, such as the workplace in [8] to reduce unwanted network traffic. A successful implementation of geocasting services in opportunistic networks, would

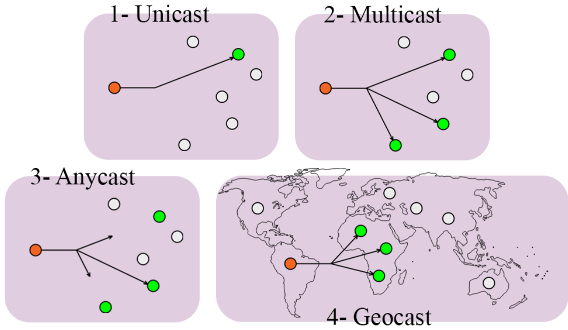


Figure 2.8: Various addressing methodologies

help network users to remotely (if they are located out of the region) request to receive geographical restricted services in opportunistic networks.

Geocasting has similarities to the previously discussed multicasting scheme. However, in geocasting the destination group of nodes is defined by a specific location in which they reside during a specific time period (i.e. the lifetime of a geocast message). Geocasting in opportunistic networks is more challenging in comparison to multicasting, because identification of nodes inside the group is not completely clear. Imagine a number of users that are located into a cast (defined geographical area) and they are the recipients of the cast based on their coordination. If a user goes out of the specified location, it is not a member of that group anymore and it should not receive the targeted geocast messages. Thus there is a *temporal*, as well as *spatial*, aspect to the definition of cast membership; it is a challenging problem to define a cast with a dynamic user membership. Figure 2.8 shows the differences between various addressing methodologies inside a network.

2.5 State of The Art

As discussed earlier, unicasting has been extensively studied in the context of opportunistic networks [14], but none of the existing protocols can support geocasting, given that unicast protocols route messages to specific devices, which are explicitly identified by unique endpoint identifiers. However, little attention has been paid to geocasting, which has mostly been studied in the context of Mobile Ad Hoc Networks (MANETs) [15, 73]. However, MANETs present radically different properties compared to opportunistic networks. In MANETs, connectivity (as well as the overall network topology) among mobile nodes is rather stable; no such assumptions can be made for DTNs, where mobility is high (mobility is actually exploited so that messages are physically carried in the devices towards their final destinations) and connectivity is very intermittent. As a result, no end-to-end paths among all nodes exist at all times and no node knows the network topology, which constantly changes. Hence, none of the existing geocasting protocols for MANETs are suitable for opportunistic networks. Following is a review of existing research on geocasting in opportunistic networks.

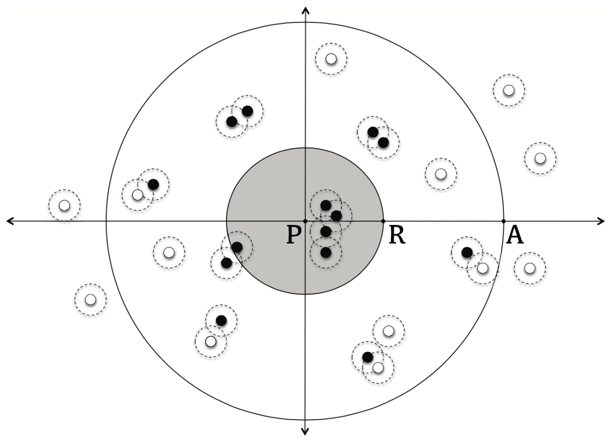


Figure 2.9: Floating Content with two circle strategy [8]

2.5.1 Floating Content

In [8] the authors describe a fully distributed, content sharing service for opportunistic networks. They introduce a geographically limited content sharing scheme that depends on the availability of nodes inside the area where the message is created; a larger number of users in the area result in higher sustainability of the shared content. In this work, the focus is on the privacy of shared content and prevention of any potential data leaks. The content sharing area is defined as a circle (centre point and radius). In this approach, authors define two circles for each area. As shown in Figure 2.9, the main circle $\{P(x,y), R\}$ is the area that the content is aimed to float inside. Any user that is located inside this area is sharing the content in a flooding fashion. In addition, there is a larger circle $\{P(x,y), A\}$ which is designed to give a chance to retrieve back the outgoing content to the main circle. Anyone who resides between the main and the larger circle ($R < r < A$) may keep the content in its buffer based on priorities[1]; to pass it to the users who are going towards the main circle. This helps to retrieve outgoing content back into the main area. Finally, users who are going out of the larger circle ($r > A$) must delete the content to free up the buffer and help to reduce the chance to leak data outside the defined area.

It is assumed that content can only be created and distributed locally (within the region), which is a limitation for the distributor. This, however, decreases security and privacy concerns by not allowing content to go out of the specified region. Although, this could cause limited application for floating content and there is a need to employ a better approach to allow users to distribute information remotely. In addition, floating content defines the geographical area based on a circle which is not flexible enough to pinpoint demanded regions accurately, and may result in sharing the content with unwanted nodes that cannot be excluded because of the way the area is defined.

[1]Priorities are calculated based on the available buffer and the amount of carried data.

2.5.2 GEOOPP

The authors in [18] propose the GEOOPP geocasting algorithm for opportunistic networks. They break down the routing procedure into two separate phases: 1- Forwarding messages to the recipient area. 2- Flooding it into the region using Epidemic [27] routing. However, they focus on the first phase which is all about delivering messages to a user inside the addressed geographical area, and ignore the second phase in their evaluation. In this work, authors try to measure the probability of *geographic progress* a node can make to carry a message to its destination (using Chebyshev's inequality) and take the probability into account whenever a message is about to be forwarded to its next hop. GEOOPP divides the network into small non-overlapping cells; and calculates the rate of reaching *to the destination cell* via *each cell* and forwards the message to a neighbour node if its future destination has a higher chance of delivery (i.e. higher calculated rate) in comparison to the current location.

In GEOOPP, during an encounter, the node with higher P_d rate for each message is selected to be the next carrying hop. P_d is calculated as follow:

$$P_d \;=\; max\ (P_{cell\ 1},\ P_{cell\ 2},\ ...,\ P_{cell\ n}) \tag{2.1}$$

It equals to the maximum chance of the progress a node can make to carry the message towards its destination through each cell. $P_{cell\ i}$ is calculated based on equation 2.2:

$$P_{cell\ i} \;=\; P_{cell\ i}(m) \ \times\ P_{cell\ i}(v) \ \times\ P_{cell\ i}(c) \tag{2.2}$$

The progress a node can make to deliver the message through the *cell i* is calculated by three factors [18]:

1. $P_{cell\ i}(m)$ is the progress a node can make carrying a message towards its destination by visiting *cell i* which is calculated in equation 2.3 .

$$P_{cell\ i}(m) \;=\; \frac{\parallel C, D \parallel_2 - \parallel I, D \parallel_2}{\parallel C, D \parallel_2} \tag{2.3}$$

∗ *C: current location; D: centre of the destination cast; I: centre of the cell i;* $\parallel C, D \parallel_2$ *is the euclidean distance between point C and point D, which calculates as follows:*
$\parallel C, D \parallel_2 = \sqrt{(x_C - x_D)^2 \ +\ (y_C - y_D)^2}$

2. $P_{cell\ i}(v)$ is the rate that calculates the chance whether a node can visit *cell i* before the message expiration time.

$$P_{cell\ i}(v) \ \geq \ 1 - \frac{\sigma^2}{(t_{exp} - t_v - \mu)^2} \tag{2.4}$$

* *μ and σ^2 are the mean and variance of the inter-visiting times of the node to cell i, t_{exp} is the expiration time of message and t_v is the most recent visiting time of the node to cell i.*

3. $P_{cell\ i}(c)$ is the rate of contact availability for the node during each visit to the *cell i*.

$$P_{cell\ i}(c) \ \geq \ 1 - \frac{\sigma^2}{(1 - \mu)^2} \tag{2.5}$$

* *μ and σ^2 are the mean and variance of the historical contact availability for the node during each visit to cell i.*

As mentioned, the node with higher P_d will be elected to carry the message towards its destination.

The presented algorithm is complex and could lead to longer processing time and higher energy consumption. The proposed approach also raises privacy concerns because mobile device users should share their geographical locations, contact history as well as future destinations with other users. It is unclear how network cells should be pre-defined, who is managing them and how all network nodes become aware of them. The paper [18] does not mention the method of cast definition nor how the delivery time interval is being applied. Also, they assume that the delivery ratio rate is same as in the unicasting methodology, while each geocasting message has more than one recipient inside the addressed cast instead of just one (methodology to measure the delivery ratio will be discussed later in § 4.3.5.A.); i.e. the delivery ratio must take into account the number of nodes in a specific cast that receive a message and not just the fact that a message is delivered in one device in the cast.

2.5.3 EVR

In the work that is presented by Y. Ma and A. Jamalipour [16], same as GEOOPP [18], the authors only focus on routing messages to the destination location instead of also to all devices inside that location (cast). The destination cast is pre-defined as a circle based on the $\{(x_m, y_m), R\}$, the coordinates of the centre and the radius of the circle (same as in [8]). Defining casts as circles suffers from the limitations described above. Routing is based on the Expected Visiting Rate (EVR), which is the probability of each user visiting the destination cast to deliver messages to their recipients. EVR is calculated based on the historical information of visited areas in the network. The exchange of location information raises privacy concerns as described above.

In EVR, network is divided into a number of non-overlapping cells same as in [18]. Briefly, the expected visiting rate is calculated as follows [16]:

$$\text{EVR } rate \ : \ \lambda_{i,m} \ \approx \sum_{cell_i(k) \in ((x_m, y_m),\ R)} \lambda_{cell_i(k)} \tag{2.6}$$

∗ *$\lambda_{i,m}$ is the expected visiting rate of cell i which resides in the destination of message m.*

In other words, EVR rate is the summation of $\lambda_{cell_i(k)}$ of cells that reside inside the destination cast. Where $\lambda_{cell_i(k)}$ can be extracted from equation 2.7:

$$\lambda_{cell_i(k)} \ = \ 1 \ / \ mean \ (\tau_{cell_i(k)}(1), \ \tau_{cell_i(k)}(2), \ ..., \ \tau_{cell_i(k)}(n)) \tag{2.7}$$

$$\tau_{cell_i(k)}(n) \ = \ t_{cell_i(k)}(n+1) \ - \ t_{cell_i(k)}(n) \tag{2.8}$$

∗ *$t_{cell_i(k)}(n)$ is the time when current node visits the cell i for n_{th} time.*

During an encounter, the node with highest EVR rate will be elected as the next carrying hop.

In this research [16], the evaluation suffers from the same limitations as the work in [18]; i.e. evaluation metrics are not reflecting the right results, and there are no investigation about the temporal membership of nodes inside the addressed cast. In the other paper that is presented by same authors [17], the authors extended the work [16] described above. They optimised the EVR algorithm to achieve better results. However, mentioned problems remain unchanged.

2.5.4 Geocasting in Mobile Partitioned Networks

The author in [19] has proposed a geocasting protocol for mobile partitioned networks. It is assumed that users in partitioned subregions are connected to each other constantly, and each node which belongs to a subregion will remain inside that particular subregion; i.e. they cannot move between subregions. Hence, the assumption is different to the scheme of geocasting in opportunistic networks as users move freely based on their own will. In this approach, users start to exchange *hello* messages, until they get a full picture of the network and infer other users' mobility pattern. This could cause high network overhead and higher energy consumption. Moreover, the mobility pattern of users may change even before other users are being informed about it; due to the complexity of opportunistic networks. Next, when the mobility pattern has extracted[2], messages are being pushed towards the regions with more stable mobility patterns to get higher chances of delivery to their destination.

[2]In this phase, it is assumed the mobility pattern is known for all users.

The explained protocols come with limitations and shortcomings in order to implement geocasting opportunistic scenarios. As discussed, related studies ignore the fact that the geocast membership definition is different to the usual unicasting and multicasting in opportunistic networks. Membership is temporal in geocasting and depends on the current location of nodes which may change through time. However, this is static in unicasting and multicasting, and the destination of a message is assigned to a node with its EID which remains unchanged. A successful solution should modify the current architecture of opportunistic networks in order to solve this problem. In Chapter 3, we discuss about the assumptions and challenges of geocasting in opportunistic networks. Then, a new solution is presented in § 3.2 that overcomes the limitations of existing approaches presented in the related work section.

Chapter 3

Design

Anyone who claims to have achieved the outrance of science, has expressed his extreme ignorance.
Ali ibn Abi Talib
Muslim Caliphate (600 - 661)

Geocasting in opportunistic networks and its applications have been discussed in the previous chapter. Also, the state of the art on opportunistic networks has been reviewed. In this chapter we present our approach for efficient and flexible geocasting in opportunistic networks. We first discuss challenges to routing and delivering geocast messages. Then, we describe our approach in detail.

3.1 Assumptions & Challenges

Before proceeding with the detailed description of the proposed geocasting protocol, we briefly discuss challenges that are relevant to geocasting in opportunistic networks; challenges that influenced our work. Geocasting in opportunistic networks is a challenging task that entails both spatial and temporal aspects and requires taking into account constraints with respect to the usage of network and device resources.

3.1.1 User Characteristics

In opportunistic networks, all devices are mobile (e.g. smartphones and tablets), they move around freely carried by their users and do not have any information about the topology of the network. End-to-end paths among mobile devices do not exist and connectivity is intermittent. Devices can only discover neighbouring nodes in their vicinity and send/receive information through one of their short-range wireless interfaces (e.g. Bluetooth and WiFi-Direct). The network is, in principle, infrastructure-less although devices could be connected to a cellular or WiFi network. Devices support location services, which may vary in the supported accuracy (and the associated battery consumption). Access to GPS for outdoors scenarios is ideal, although in most cases, coarser-grained estimations are fine. For indoors scenarios, relevant localisation approaches [74] can be used.

3.1.2 Objectives

In geocasting, the goal is to successfully deliver a message to all users (or to as many as possible) inside a specific geographical area within a specified time interval; i.e. it is not only necessary for a message to reach a cast, but it must also be efficiently disseminated within the cast. The temporal aspect is important because in many communication scenarios, messages should be invalidated and deleted from the network, either because the information they carry expires or just because the network cannot cache a message forever. The protocol should narrow down the potential recipients by defining a delivery time interval, and only the nodes that are present in the cast within that *particular time interval* should *receive* the message. Our approach for the membership model of the *recipients* follows the *"Current-Member Delivery"* model defined in [57]. As the model suggests final recipients are the nodes that are partially- (through time) or fully-present at the destination cast during the interval of the creation and expiration time of the message.

3.1.3 Requirements

Messages in geocasting are destined to a specific location, therefore some kind of destination information must be included in the message, as the *Endpoint Identifier (EID)*. For example, if casts are pre-defined at deployment time and known to all devices, a message may carry a cast identifier; otherwise, the cast definition (e.g. centre/radius pair or coordinates of a polygon, as in our approach) must be in the message. Whenever a node receives a message, it compares its own location with the EID of the message. This device is a recipient of the message if it currently resides within the cast defined by the EID (and the message has not expired yet). According to the DTN architecture specification [2], EIDs can be obtained based on the IP address of the mobile device to retain their uniqueness.

3.1.4 Memory usage

Opportunistic networks employ *store-carry-forward* based mechanisms for message routing (including geocasting), therefore mobile devices must be able to temporarily store and carry messages before they forward them to other devices. Conceptually, the network can be seen as a buffer of finite size, which is made of all available devices' buffer, collectively. Although devices' memory has grown over the past years, one would not expect to be able to utilise more than some tens of MBs of memory in each device, given that other applications and background services require access to ever increasing chunks of memory. This has implications in the way data is forwarded. For example, unconstrained message flooding would result to quickly filling up devices' buffers, resulting in the quick extinction of a large number of messages.

3.1.5 Constraints

Increasing the size of available buffers in each device does not simply solve the problem described above. Network bandwidth is limited but most importantly the time period that two devices can exchange messages is short given that users move. As a result, if very large buffers were used, only a very small portion of the carried messages could be forwarded from device to device.

Forwarding messages also comes with a cost in terms of battery consumption. Control-messages exchanged among devices (e.g. to build mobility maps [19]) as well as local computations of metrics that influence how routing is done (e.g. in [18]) may result in quick draining of the device batteries. Exchanging location-related information among devices also has privacy implications that must be taken into consideration. One of the major concerns of opportunistic network users is to loose their confidential information (e.g. their mobility pattern) by participating to the device-to-device communications. In fact as the work in [20] suggests, this issue has a noticeable negative impact on the participation of users in the routing procedure. Indeed, users whom wish to not share their own confidential data, do not participate and contribute in the routing procedure. This is one of the major concerns in real-life device-to-device communications [20].

It is worth pointing out that the network density in terms of connected mobile devices may vary significantly in different opportunistic communication scenarios. For instance, flooding the network may work well in very sparse scenarios, although the network overhead would be significantly increased as the number of users increases. Accordingly, a protocol that forwards packets very selectively (e.g. by calculating cast visiting probabilities [18]) may result in low network overhead in dense scenarios but very low message delivery ratio in sparse scenarios. In any case, a geocasting protocol should be as less sensitive to network density as possible.

3.2 Detailed Design

The challenges in designing an efficient geocasting protocol have been presented in the previous section. The proposed solution should be able to overcome the discussed challenges. **GSAF (Geocasting Spray and Flood)** and **DA-GSAF (Direction Aware - GSAF)** are two efficient and flexible solutions for geocasting in opportunistic networks. They combine a simple and effective cast definition approach with an efficient routing protocol to geocast messages and deliver them to their recipients. In addition, the proposed cast definition enables users to effectively define the exact intended delivery area in the network, without sacrificing unnecessarily network resources for delivering messages as accurately as possible in space. If the intended delivery area is not accurate enough, it can

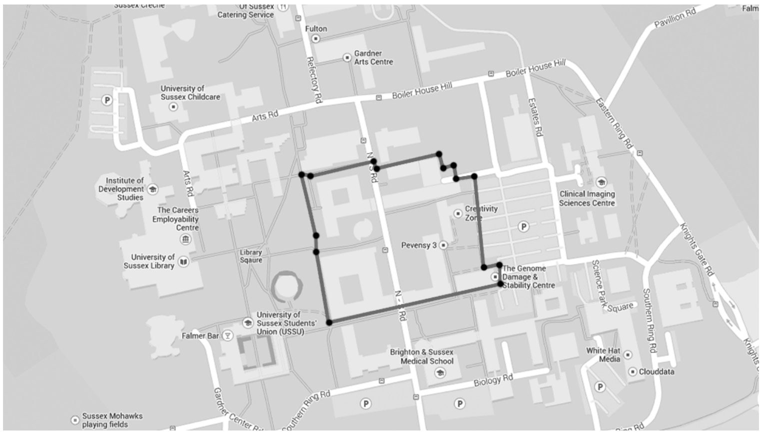


Figure 3.1: An example of a geographic cast defined by use of location coordinates at the vertices

result in wasted network bandwidth to deliver the message to unwanted recipients which also raises privacy concerns (e.g. as in [16] and [18]).

3.2.1 Cast Definition

Geographical casts are defined as the geographical areas which can be covered by our network. Any user of the network that is located inside a defined cast is a potential recipient. In geocasting, messages are addressed to a cast and every (or as many as possible) user that is located in that cast receives the message. It is necessary to define[1] the cast (or casts) inside the network; and messages should be addressed to the defined locations (casts). Then, every node which is located inside that area should receive that message.

First, we investigate how to define a geographical area (cast) in opportunistic networks. Geographical casts effectively define a group of users that reside within the same region which can be addressed as the recipient of a message. It should be defined inside the map and may take any needed shape in the network. In the proposed solution, a cast is defined as a set of coordinates in the two-dimensional space (the network). Figure 3.1 depicts an example of a cast inside a map; the area is defined by a set of coordinates (black points) as the vertices of the polygon. With this approach, users can send messages to arbitrarily defined casts. This provides great flexibility, potentially minimising the number of devices that are receiving unwanted messages compared to other geocasting approaches that define casts as circles (i.e. as centre/radius pairs). With approaches mentioned in Chapter 2 (e.g. [16], [18] and [8]), if a specific region, which is far from the centre of the circle, needs to be reached, the radius has to be increased, resulting in wasted network bandwidth for messages that reach devices for which the message is useless. In our approach, users

[1] Note that casts not needed to be pre-defined. Instead, a sender can define a cast to send a message to, on-the-fly.

can draw casts on their mobile phones where their messages will be destined. Messages carry the defined cast information (the set of coordinates), which effectively is the delivery address, therefore there is no requirement to pre-define any cast.

In opportunistic networks, the topology is unknown and each node is not necessarily aware of the existence of all other nodes. There is no way of sharing information except through exchanging data, on a hop by hop basis (as discussed in § 2.1 and § 2.2). Thus, each message should carry the information of its own recipient (a cast).

Next, the routing protocol should be able to route messages based on the destination cast. The protocol must identify if a node (user) is located inside the cast or not. Given that this check is performed for every received message, the algorithm must be very efficient. Real-life opportunistic networks can be mapped onto a two-dimensional space and user devices are represented as two-dimensional coordinates in it. In our approach we incorporate the "Point in Polygon" approach that is widely used in computer graphics and image processing [75]. "Point in Polygon" strategies are used to identify whether a point (with a known coordinate) is inside a polygon or not. According to Haines [75], there are three main strategies to solve this problem: *Crossing test*, *Angle summation test* and *Triangle test*.

Crossing test

As seen in Figure 3.2, a straight line needs to be drawn from the examined point to any fixed direction. The number of intersections between the ray and the edges (boundaries) of polygon is key to the identification. If the number is even, the point is located outside of the polygon, otherwise it is located inside.

Angle summation test

An angle is drawn by a ray being "a line from the examined point towards a vertex of the polygon" and another ray being "a line from the point towards the next neighbour vertex". If the total summation of all neighbour angles in the polygon equals to 360°, then the point is inside otherwise it is not (Figure 3.3).

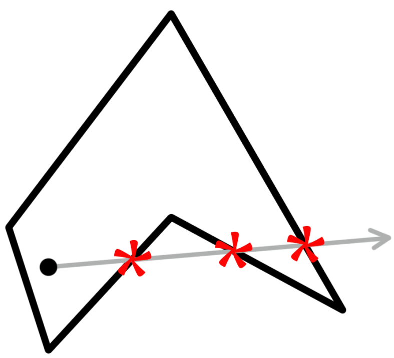


Figure 3.2: Crossing test: "Point in Polygon" strategies

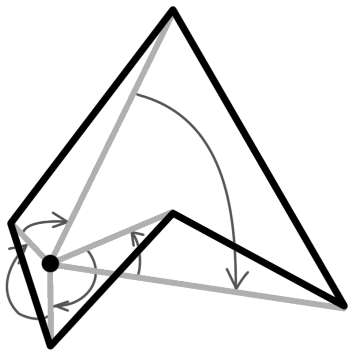


Figure 3.3:
Angle summation test [75]

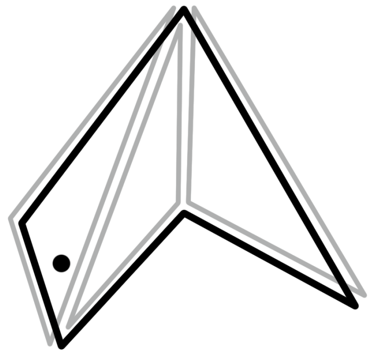


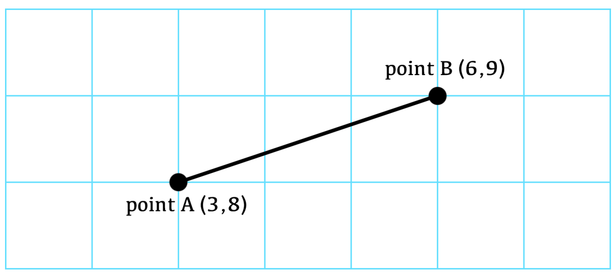
Figure 3.4:
Triangle test [75]

Triangle test

The polygon is divided to a number of triangles. The point is inside the polygon if it is inside the one of triangles, if not it is outside (Figure 3.4).

In [75], Haines claims that the "crossing test" algorithm is the fastest, requiring less processing power in comparison to the other approaches. Thus, in our approach we have used the crossing test algorithm to identify if a node is inside a cast or outside of it. Whenever a routed message is received by a node in the network, the protocol utilises the crossing test algorithm to find out if the current node is inside the defined recipient cast or not (i.e. whether a node is a recipient or not).

In our approach, at cast definition time, the linear equation that defines the line between the two neighbour points is extracted. Also, the range of the equation is limited by the



Linear equation :

$$y - y_A = \frac{y_B - y_A}{x_B - x_A}\,(x - x_A) \quad x_{range} : [\ x_A, x_B\ ] \ \Rightarrow$$
$$y - 8 = \frac{9-8}{6-3}\,(x - 3) \quad x_{range} : [\ 3, 6\ ] \ \Rightarrow$$
$$y = \frac{1}{3}\,x + 7 \quad x_{range} : [\ 3, 6\ ]$$

Figure 3.5: An example of calculating the linear equation for each neighbour points (cast definition time)

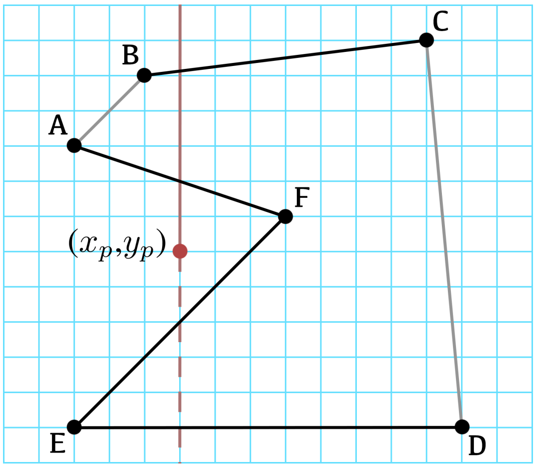


Figure 3.6: An example of implemented crossing test algorithm

$[x_A, x_B]$ in the map. In Figure 3.5 we present an example of such calculation. This is done for every defined neighbouring vertices in the polygon (cast). The extracted linear equations are used in the crossing test algorithm. Whenever there is a node that needs to be identified, a vertical line would be drawn from the node to support the crossing test. An example of the crossing test is illustrated in Figure 3.6. Initially, a vertical (to the x-axis) line that crosses the point (with coordinates (x_p, y_p)) that needs to be checked is drawn. The point (x_p, y_p) is the initial point that decomposes this line into two rays (half-lines). The number of intersections of one of the rays (e.g. the solid red line in Figure 3.6) with the sides of the polygon is used to check whether the point is in the polygon or not; if the number is even, the point is located outside the polygon, otherwise the point is inside. For each pair of neighbouring vertices in the polygon, we calculate the parameters of the line equation that defines the line that connects these two points, as shown in Figure 3.5. The x_{range} defines the projection of each side of the polygon to the x-axis. In order to calculate the intersections of the vertical line with the given polygon, we simply test whether x_p is within the x_{range} for each side of the polygon. For example, in Figure 3.6, sides BC, DE, EF and FA intersect with the vertical line, because x_p is inside the x_{range} of them.

Next, we calculate the y coordinates of the intersection points by solving the line equations (that define each side of the polygon) using x_p. Finally, we count the number of y coordinates that are larger than y_p (i.e. looking at the ray illustrated with the solid red line). Based on this number we decide whether the point (x_p, y_p) is inside (an odd number) or outside (an even number) the polygon (cast). Algorithm 3.1 in the routing protocol, applies the crossing test and checks whether a node is located inside the addressed cast or not.

Algorithm 3.1 Design of applied crossing test in our research

```
for each received message at a specific host do
    get the host location (x, y)
    calculate the vertical line in the map with the x-coordinate of the host's location
    find the list of intersections between the vertical line and the destination cast in the map
    for each intersection point in the list do
        if y-coord of intersection point is less than y-coord of receiver's location then
            exclude it from the list
        end if
    end for
    if the number of remaining intersections are an odd number then
        return True (the current host is a recipient of the message)
    else
        return False (the host is outside of cast and it is not a recipient)
    end if
end for
```

3.2.2 Geo-Routing protocol

The goal of our protocol is to successfully deliver geocasting messages to all devices that are present inside the addressed cast within a specific time interval. As mentioned in § 3.1.2, the time interval is important, due to the fact that users are mobile and they are coming into or going out of the geographic cast (i.e. cast membership is dynamic). The protocol should narrow down the recipients' list by defining a delivering time interval, and only the nodes that are present at the location between that *particular time interval* should *receive* that message. Unicast routing protocols for opportunistic networks do not work like this, since they try to deliver a message to a recipient node only based on its ID. However, the cast membership is based on the users' coordinates which is dynamic due to their mobility. A device should get the "addressed cast" and the "time interval" information from the message and route it successfully to its recipients.

The first challenge is the implementation of a time interval into the routing functionality. It is assumed that the time interval to deliver a message to its addressed cast is the time period defined by the creation time of message and its lifetime [2] (TTL):

$$Time\ interval : [\ Creation\ time,\ TTL\ ] \tag{3.1}$$

Hence, all nodes that are located inside the cast at any given time between the creation time and the TTL of message, are recipients and they should receive a copy of that geocasting message.

Next, our approach to design a geocasting routing scheme can be broken into two separate phases:

1. Carrying the message to its destination cast.
2. Delivering it to all devices inside that area.

[2] It is assumed that the TTL is measured based on the remaining time to the expiration.

Separating these two phases helps reduce complexity. The two phase model has been discussed in [15] as URAD (Unicast Routing with Area Delivery) methodology to deliver geocasting packets in *MANETs*. In the following we present the two variants of the proposed approach, *GSAF and DA-GSAF*.

A. GSAF

The two phases of GSAF are described below:

Phase 1- Forwarding (and carrying) the message to the destination cast: In the first part, the message should be delivered to its addressed cast as fast as possible. For the first phase, GSAF follows a multi-copy spraying approach (inspired by [21]), which is fast in terms of reaching the destination cast as well as efficient in terms of message delivery ratio and network overhead. Algorithm 3.2 describes how GSAF is designed to route geocast messages inside the opportunistic networks. GSAF supports a copy ticket number (T) that allows a node to distribute T copies of the message inside the network. Upon creation, a number of tickets T is "assigned" to the message (represented as a ticket counter which is included in the message). T denotes the number of times a carried message can be forwarded to encountered devices. In other words, T specifies the number of message duplications for the current node, but not for all nodes in the network. Each time a message is copied and forwarded to another node, T is decreased by 1 in both messages (*Lines* 11 *and* 27 *of Algorithm* 3.2). In other words, whenever a source sends a copy to its neighbour node, both messages get "$T-1$" tickets for the next distribution. When T gets to zero, the message cannot be forwarded any further; i.e. it will be deleted when the local buffer gets full or when the message expires.

Phase 2- Delivering the message to all nodes inside the cast: In the second phase, the message is disseminated to all nodes inside the cast by following an intelligent flooding approach. GSAF floods the message to nodes inside the cast by handing a copy to them (*Lines* 14 *to* 16 *and* 21 *to* 24 *of Algorithm* 3.2). If a copy of the message goes out of the destination cast, it can only be forwarded back to nodes that are located inside the cast. The message will sit in the device's buffer until it expires. At that point it will be deleted (*Line* 4 *of Algorithm* 3.2). This way GSAF prevents unnecessary message exchanges outside the cast, which, given their flooding nature, would increase the network overhead significantly and could fill up the buffers of nearby devices quickly.

Figure 3.7 illustrates an example of how messages are disseminated in our approach. The sender (shown in green) creates a new message and initialises T to 3. It encounters two nodes (the one after the other) and for each such encounter, it decreases the value of T in the message and forwards a copy of the message to the remote node. As shown in the figure, T is first decreased to 2 (which is also the value of T in the message received by the

Algorithm 3.2 GSAF Router

```
1: for each encounter between two nodes in the network do
2:     one takes the sender role (N1) and the other takes the receiver role (N2)
3:     if the node is the sender (N1) then
4:         drop expired messages from buffer
5:         apply buffer scheduling policy
6:         for each message in the buffer do
7:             if the message is already existed in the N2's buffer then
8:                 pass to the next message in the for loop
9:             end if
10:            if (T > 0) then
11:                T ← T − 1
12:                forward a copy to N2
13:            else if (T = 0) then
14:                if N1 is inside the destination cast (Algorithm 3.1) then
15:                    forward a copy to N2
16:                end if
17:            end if
18:        end for
19:    else if the node is receiver (N2) then
20:        for each received message do
21:            if N2 is inside the destination cast (Algorithm 3.1) then
22:                deliver to Application Layer
23:                store a copy into buffer
24:                T ← 0
25:            else if N2 is outside the destination cast (Algorithm 3.1) then
26:                store message into buffer
27:                T ← T − 1
28:            end if
29:        end for
30:    end if
31:    change roles (sender [N1] & receiver [N2]) and go back to line 3 just for one time
32: end for
```

node above the source node) and, then, to 1 (the value of T in the message received by the node below the source node). The same takes place when these two nodes encounter other nodes in the opportunistic network. At the end of the illustrated example, a number of nodes roam outside the cast carrying a message with a T value of *zero*. These nodes will not forward the message any further. The node that resides inside the cast has also received the message (T is *zero*) but the message will keep being forwarded to recipients inside the cast, as phase 2 suggests. More generally, a message can end up inside the destination cast either after it was exchanged between a node outside and a node inside the cast or because it was physically carried by a node to inside the cast. In both cases, T can have any value (equal or greater than 0). T will be set to 0 at the beginning of the second phase (*Line* 24 *of Algorithm* 3.2).

The value of T can be pre-specified for specific network deployments (e.g. for communication within a city) based on e.g. the expected node density and mobility patterns. In § 5.5, a sensitivity analysis for the initial value of T is presented, which indicates that values close to the optimal one (with respect to message delivery ratio and latency), also result to very good performance. One could therefore dynamically set T's initial value e.g.

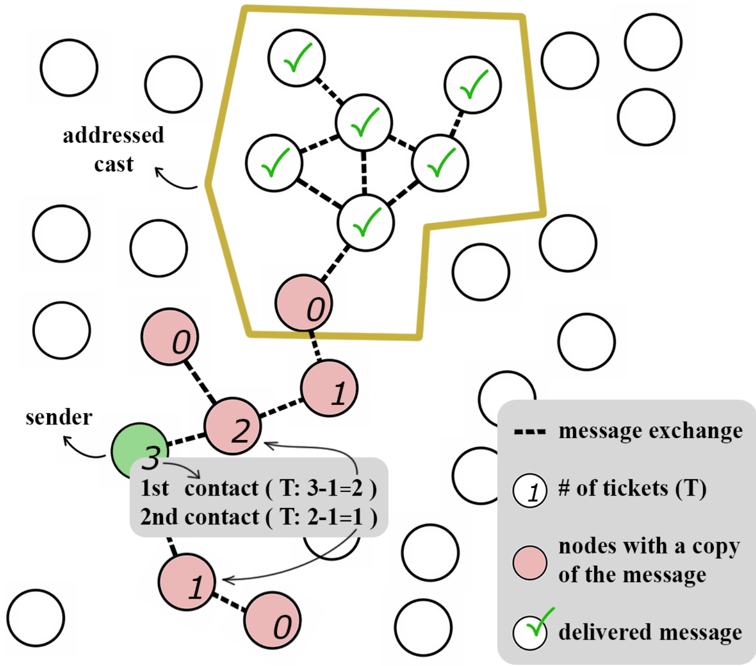


Figure 3.7: GSAF Routing scheme

by estimating the density of mobile devices, as in [76].

B. DA-GSAF

In our approach, the location context of nodes is already available and being used to realise if a node resides inside a geographical cast. To exploit its full potential, we decided to utilise the location information to aid routing decisions of GSAF without compromising users' privacy. The challenge is to be able to use these information without sharing any location or historical or future information of the users in the network. As a result, we present **DA-GSAF (Direction Aware - Geocasting Spray and Flood)**, which extends GSAF and use location information to aid routing decisions with a special design that does not share the nodes' context among the other network users (The only shared information between two encountered node is whether the node is going towards the message destination or not. However, it does not share the actual moving direction of the node.). This design does not raise any privacy concerns (as explained below), in contrast to the state of the art that is reviewed in Chapter 2.

DA-GSAF follows a two-phased routing procedure, similar to GSAF. In the first phase, when a device encounters another one, instead of just checking the remaining copy tickets (as in GSAF), DA-GSAF checks the following three conditions:

1. If there is a copy ticket left to distribute another copy.
2. If the current node is not going towards the destination cast.
3. If the encountered node is going towards the destination cast.

If all of these conditions are met when two devices encounter each other, DA-GSAF allows a copy to get distributed to the next node. This allows DA-GSAF to distribute copies of a message to the devices that have more potential to be an effective carrier node.

Algorithm 3.3 presents the design for DA-GSAF. First, similar to GSAF, DA-GSAF checks if a copy ticket left to distibute more copies in the network. This allows our design to manage the overall network overhead by controlling the most important bottleneck (i.e. network overhead) in the opportunistic networks. This condition is being investigated in *Line* 13.

In the second condition, DA-GSAF investigates if the current carrying node is an appropriate and high qualified carrier for the message. This is possible by checking if the current node is going towards the same direction as the message's destination; i.e. towards its addressed cast. Algorithm 3.4 checks the moving direction for each node in the network. Each node records its own previous coordinates and compares it with its current location by using Algorithm 3.4 to infer its moving direction. Moreover, the direction towards the required cast (message's addressed cast) is being calculated based on the combination of Algorithm 3.4 and Algorithm 3.5 . This procedure is explained through *Lines* 6 *to* 8 *and* 15 *to* 17 of Algorithm 3.3 . If the current node is going towards the same destination as the recipient cast, it already is a qualified carrier node. As a result, there is no need to distribute more copies inside the network.

Third, if the current node is not considered as a qualified carrier, we check if the encountered node is a better choice. This is possible by checking if the encountered node is going towards the message's destination cast. However, the challenge is not to share the users' location context. This issue is being addressed by sending the destination of message to the encountered node and asking the device to perform the direction test remotely, rather than requesting for its context and performing the test locally in the source node. This solution addresses the privacy concern and the amount of shared context of node is almost zero (i.e. With this solution, the encountered node can only understand if the current node is not going towards the destination of message). As a result, in the third condition, current node sends the destination cast of its message to the encountered node and requests if the encountered node is a good choice for carrying it (*Lines* 17 *to* 23 *of Algorithm* 3.3). Next, the encountered node receives the information (message's destination cast) and checks if it goes towards the same direction and responds positively or negatively (*Lines* 31 *to* 40 *of Algorithm* 3.3). If the response is positive, the source node gives a copy to the encountered node.

The second phase of routing in DA-GSAF remains unchanged to the one in GSAF which is described in Page 48 . Briefly, whenever a copy arrives into its destination cast, DA-GSAF starts to intelligently flood the message to the all devices that are located inside

Algorithm 3.3 DA-GSAF Router

1: **for each** encounter between two nodes in the network **do**
2: one takes the sender role (N_1) and the other takes the receiver role (N_2)
3: **if** the node is the sender (N_1) **then**
4: drop expired messages from buffer
5: apply buffer scheduling policy
6: $P \leftarrow N_1$ *'s current coordinates*
7: $Q \leftarrow N_1$ *'s previous coordinates*
8: $N_1 Direction \leftarrow function$ DIRECTION (Q, P) $_{Algorithm\ 3.4}$
9: **for each** message in the buffer **do**
10: **if** the message is already existed in the N_2's buffer **then**
11: pass to the next message in the for loop
12: **end if**
13: **if** $(T > 0)$ **then**
14: $T \leftarrow T - 1$
15: $R \leftarrow function$ CASTCENTERPOINT (*message recipient cast*) $_{Algorithm\ 3.5}$
16: $MessageDirection \leftarrow function$ DIRECTION (P, R) $_{Algorithm\ 3.4}$
17: **if** $(N_1 Direction \neq MessageDirection)$ **then**
18: send the $MessageDirection$ to N_2 and wait for the response
19: $Res \leftarrow response$
20: **if** $(Res = Yes)$ **then**
21: forward a copy to N_2
22: **end if**
23: **end if**
24: **else if** $(T = 0)$ **then**
25: **if** N_1 is inside the destination cast $_{(Algorithm\ 3.1)}$ **then**
26: forward a copy to N_2
27: **end if**
28: **end if**
29: **end for**
30: **else if** the node is receiver (N_2) **then**
31: $P \leftarrow N_2$ *'s current coordinates*
32: $Q \leftarrow N_2$ *'s previous coordinates*
33: $N_2 Direction \leftarrow function$ DIRECTION (Q, P) $_{Algorithm\ 3.4}$
34: **for each** received $MessageDirection$ **do**
35: **if** $(N_2 Direction = MessageDirection)$ **then**
36: **return** *response* to $N_1 \leftarrow$ Yes
37: **else**
38: **return** *response* to $N_1 \leftarrow$ No
39: **end if**
40: **end for**
41: **for each** received message **do**
42: **if** N_2 is inside the destination cast $_{(Algorithm\ 3.1)}$ **then**
43: deliver to *Application Layer*
44: store a copy into buffer
45: $T \leftarrow 0$
46: **else if** N_2 is outside the destination cast $_{(Algorithm\ 3.1)}$ **then**
47: store message into buffer
48: $T \leftarrow T - 1$
49: **end if**
50: **end for**
51: **end if**
52: change roles (sender [N_1] & receiver [N_2]) and go back to line 3 just for one time
53: **end for**

Algorithm 3.4 Direction Calculator

Require:
Two points with x and y coordination.

```
function Direction(Point A(x, y), Point B(x, y))
    if ((x_a = x_b) ∧ (y_a = y_b)) then
        return Still
    else if (y_b >= y_a) then
        if (x_b < x_a) then
            return NorthWest
        else
            return NorthEast
        end if
    else if (y_b < y_a) then
        if (x_b < x_a) then
            return SouthWest
        else
            return SouthEast
        end if
    end if
end function
```

Algorithm 3.5 Cast Center

Require:
Cast Object

```
function CastCenterPoint(Cast x)
    for all defined vertices of Cast x do
        x_m ← mean(x_vertex 1, x_vertex 2, ..., x_vertex n)
        y_m ← mean(y_vertex 1, y_vertex 2, ..., y_vertex n)
    end for
    return CenterPoint(x_m, y_m)
end function
```

the cast. Flooding happens due to the fact that everyone inside the cast are recipients and they must receive a copy of message. In addition, *inteligent* flooding prevents messages to go out of the cast once they have arrived inside, and it stops any potential data leak or extra resource consumption.

3.2.3 Geocast Message Bundle

Messages in our approach include the cast definition (as a set of two dimensional coordinates), a pair of epoch times that define the time period during which the message is valid, the number of tickets (T) and the actual payload. Note that in our approach messages can become valid after their creation and initial forwarding. One could therefore account for the cast delivery latency and make the messages valid in the near future, when it is anticipated that they will reach the destination cast. This can also come handy in scenarios where messages are created a priori. For instance, in a geographically targeted advertisement scenario it could prepare several messages that can become valid at specific times during the day. Mobile devices are assumed to be loosely synchronised, a fair assumption for today's smartphones.

3.2.4 Buffer Scheduling Policy

Devices store messages in their buffer to carry and propagate them inside the network. Buffer management plays an important role in terms of keeping and delivering messages, as opportunistic networks follow a *store-carry-forward* approach to overcome the intermittent connectivity in the network. Each node stores every arriving message in its buffer and tries to forward them when it encounters other nodes, according to a routing protocol. Buffer behaves like an array list and it keeps messages in a specific order. The order defines which message should be processed and transferred first. On the other hand, the transmission rate of the node specifies the number of processed messages during an encounter. Faster transmission rates enable faster message transfer; as a result a larger number of messages could be exchanged during an encounter (transfer window). If messages fill-up the buffer, by default, the protocol would delete the oldest message to make room for new messages.

In low transmission speed scenarios (i.e. with Bluetooth connectivity), encounters usually suffer from insufficient bandwidth, therefore nodes cannot exchange all their messages. Due to this fact, it is important to manage the order, messages are forwarded. The buffer scheduling policy defines the placing order of messages inside the buffer of a device. In Chapter 5, we study the effect of four buffer scheduling policies in the overall performance of the network.

- **FIFO (First In First Out):** In this policy, messages are forwarded to the remote device in the order they have been received.
- **LIFO (Last In First Out):** Messages are forwarded to remote devices in the reverse order compared to the one they were received.
- **HTFO (Highest TTL First Out):** The message with the longest lifetime is forwarded first.
- **LTFO (Lowest TTL First Out):** The message with the shortest lifetime is forwarded first.

This chapter described the proposed approach for Geocasting in opportunistic networks. The definition of a geographical cast and how this is implemented was also discussed. The crossing test algorithm was introduced to check the presence of a node in a cast. In addition, we have described the design details of a novel routing scheme (*GSAF and DA-GSAF*) for the geocasting service in opportunistic networks. In the next chapter we will present details about the implementation of our approach in the ONE [77] simulator.

Chapter 4

Implementation

Always remember, the last remaining key in your hand may be the key to success.
Cyrus the Great
King of IRAN (c. 590 - 529 BC)

This chapter describes the implementation of the proposed geocasting protocol in the ONE Simulator. ONE [77] supports a set of powerful tools to evaluate the performance and feasibility of GSAF in opportunistic networks. Below we describe the features supported by the ONE simulator as well as the implementation of a "geocasting service" and the "GSAF and DA-GSAF" schemes.

4.1 The ONE simulator

The ONE (Opportunistic Networking Environment) simulator is an open source, agent-based simulation engine which can be used to simulate opportunistic networks. It focuses on the modelling of users' movement, how they contact each other, how they exchange information as well as the routing decisions and message handling at the mobile nodes. In addition, it supports a powerful tool to collect information for the result analysis. "The ONE simulator is a simulation environment that is capable of: (1) Generating node movement using different movement models. (2) Routing messages between nodes with various DTN routing algorithms and sender and receiver types. (3) Visualising both mobility and message passing in real time in its graphical user interface." [77]. It is specifically designed to simulate the performance of opportunistic, delay tolerant networks and it is being used by the research community as the primary simulator software for the performance evaluation of opportunistic networks' routing protocols.

ONE focuses on the simulation of the *store-carry-forward* routing scheme, as well as the bundle layer which is different to the regular network stack (see Figure 2.3). It simulates the lower layers of the network, based on specific given assumptions (e.g. transmission rate and transmission range), instead of fully modelling them in the network. An extended description of how the ONE simulator works is described in [78] by Keränen et al. However, it is useful to outline the main capabilities and the work flow of the ONE simulator, as the proposed schemes are implemented and evaluated in the ONE simulation environment.

The ONE simulator is an agent based event-driven simulation environment that uses a scenario to simulate a DTN network. Scenarios specify the simulation environment, hosts' abilities, network characteristics, network load etc. The simulator engine simulates the users' movement and connections to each other. It tracks down each message and its progress in the network as well as its delivery status to the final recipient. The delivery performance of messages to their destination, the generated traffic volume and the delivery delay determine overall performance of an opportunistic network in the ONE simulator.

4.1.1 Simulation Environment

In the ONE simulator, all simulated actions take place inside a two-dimensional board (the world) with a predefined size. Nodes are located inside the world and can move freely to emulate a real world environment. In addition, maps and routes can be imported into the "simulation world" based on the "WKT" file format; Well Known Text (WKT) is a text markup language which is designed to represent the geometric routes and objects in geographical maps. It contains a set of coordinates that define points, lines and shapes in the maps. As mentioned, routes could be imported inside the simulation world. The simulation engine uses various mobility models to move nodes in the simulated network.

4.1.2 Nodes and their Capabilities

In the ONE simulator, mobile nodes simulate real world user devices. Each user is able to move freely and contact other nodes by sending or receiving messages in the network. The moving speed and pattern define the type of the node; i.e. a pedestrian, a cyclist, a car, a bus, etc. For example, to simulate a bus in the network, three things should be set for the nodes that will act as buses in the simulation scenario: (1) bus movement speed as the speed of the node; (2) bus routes in the map as the routes for the node; (3) bus stops and the bus mobility pattern [1]. All of the mentioned characteristics can be set in the ONE simulator for a group of nodes. Mixed groups of people, cars, buses and trains could be supported simultaneously, providing a detailed and realistic simulation of an opportunistic network.

In addition, each node supports a network interface through which communication with other devices is simulated. Various communication technologies, such as Bluetooth and

[1] Buses behave like a device or network user (i.e. Daknet discussed in the page 30 and [44]); the ONE simulator cannot simulate people that use the bus to commute.

WiFi-Direct, are supported. The connection interface characteristics are set for each group of users; these contain the transmission range and the transmission speed.

4.1.3 Messages

A message is a bundle of information that is being transferred from a sender to a recipient node. It contains a set of headers (e.g. message ID, size, sender ID, endpoint ID, etc.) and the actual payload. Every single message is reported and tracked down during a simulation. Messages are being generated by a message generator in the ONE simulator. Their characteristics (e.g. size) can be pre-set for each simulation scenario. The message generator sets a unique sender and unique receiver for each message, randomly, and releases them into the network (by handing it to its source node) at a specified time. Next step, nodes' router takes the routing decisions for each message based on its routing algorithm. Each message is being forwarded based on these decisions. The sending process starts by forwarding the message *hop-by-hop* until the recipient node is reached. Due to the nature of opportunistic networks, a message may or may not get delivered to its final destination.

4.1.4 Router

There is a router object inside each node in the ONE simulator that takes all routing decisions for all messages inside the device's buffer. It processes each message that is located in the buffer and queues it based on a pre-set scheduling policy. Upon an encounter with another node, the router object processes each message in the queue and determines whether the neighbour is an appropriate next hop for the message or not. Then it transfers the message to the encountered node and waits for a delivery response. The neighbouring node receives the message and stores it into its own buffer and it sends a delivery response to the sender. When the router object of the sending node receives the delivery response, it marks the message delivery and deletes it from its own buffer (i.e. this is the custody transfer scheme discussed in § 2.2.4). The same sequence of actions is followed for each message in the sending node's buffer. After the first node sent all its messages, the encountered node becomes the sender and the same steps are followed. Nodes stop exchanging messages either when all messages are exchanged or when the connection between the two devices gets terminated (i.e. when the nodes are not anymore within the transmission range supported by the used wireless interface).

The router object delivers messages based on their recipient Endpoint Identifier (EID). Whenever a node receives a message, it compares it's own EID with the recipient EID that is carried with the message. If these two are the same, then the current node is the recipient of the message. Subsequently, the router object delivers the message to the application layer and deletes it from its own buffer. Note that the ONE simulator is only capable of simulating unicasting communication scenarios; multicasting or anycasting scenarios are not supported by the current version of the ONE simulator [78]. Furthermore, as

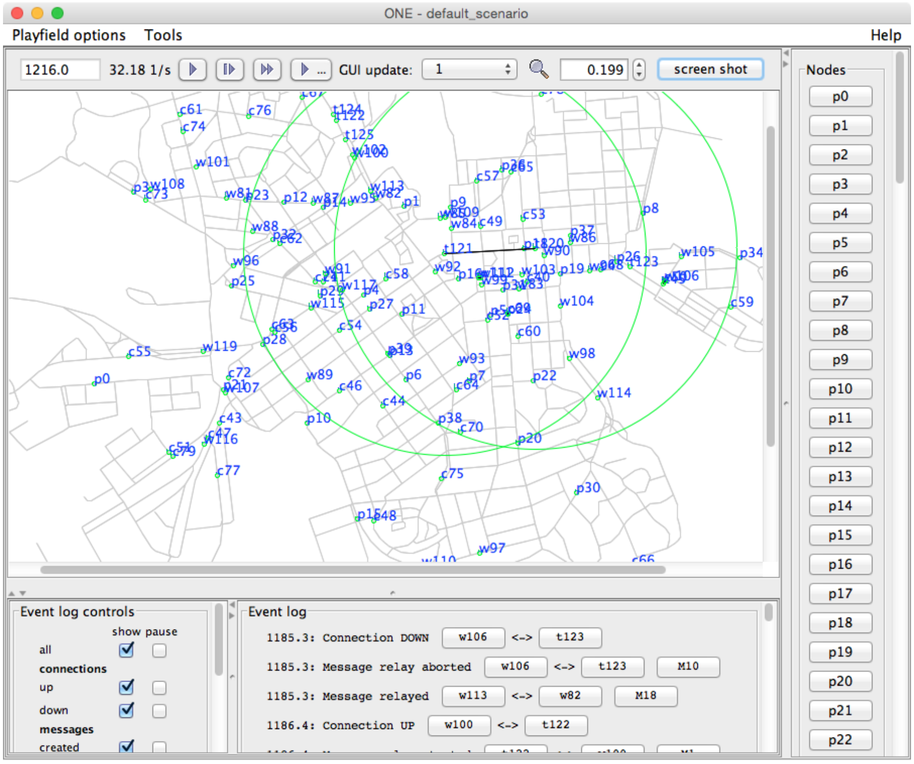


Figure 4.1: Screenshot of the ONE simulator's GUI

the router object in each simulated devices identifies recipients based on their EIDs, it is impossible to route messages based on a specific geographical location (i.e. as required by geocasting). In order to evaluate our research, we had to implement not only the proposed geocasting protocols but also extensive support for routing to multiple recipients and specific geographic locations.

The ONE simulator supports widely studied routing protocols for DTN networks, such as Epidemic [27], Prophet [39], Direct Delivery[2] and SAW (Spray And Wait) [21] routing.

4.1.5 Listeners and Reports

Listeners are used in various parts of the ONE simulator. They track and update information. All listeners are connected to simulation reports and every event, such as node movement, encounter, transferred message, is being saved in a relevant report through the usage of a listener.

4.1.6 Graphical User Interface

The ONE simulator supports a graphical user interface that visually presents the simulation and various events in real time. It shows the simulation environment (e.g. map,

[2]In the Direct Delivery approach, source nodes do not create any extra copies into the network and wait to visit the recipient itself.

routes, nodes and their movements) as well as a real-time report of message transfers between network devices. A screenshot of the GUI is shown in Figure 4.1. The GUI helps users to have a detailed understanding about the simulation that is being run, also providing support for tracking down specific groups of nodes and their behaviour in the network. Alternatively to the provided GUI, it is possible to run the simulator in “batch mode” without the graphical user interface.

4.2 Requirements

As mentioned above, the ONE simulator is restricted to the simulation of unicast communication scenarios; other types of routing (i.e. multicast, anycast and geocast) are not currently supported, therefore, in the context of this thesis, we had to modify the core of the simulator to enable support for geocasting to implement the proposed geocasting protocols.

An outline of the required changes is presented below:

Recipient Geographical Casts and Geocasting Messages:
We implemented functionality so that cast definition is supported and casts can be used as destinations of messages.

Geocasting Router and Host:
We implemented GSAF and DA-GSAF as Geocasting router objects so that their performance could be evaluated. Router objects reside within a host in the ONE simulator, therefore we had to implement support for Geocasting at the hosts.

Listeners and Reports:
We implemented Listeners to transfer all geocasting-relevant events to relevant reports. Additionally, we implemented reports so that the performance of the proposed protocols could be recorded (discussed in § 4.3.5.A.).

Graphical User Interface:
We implemented functionality for visualising Geocasting-related information at the GUI.

4.3 Implemented Modifications

Below, we present a detailed description of all required modifications we implemented in the ONE simulator.

SussexUni.wkt

```
LINESTRING (164.31616315039554 -45.17160711675103, 172.82310190526422 -80.79441315276362)

LINESTRING (171.75973456090563 -44.108239772392444, 179.1709498032085 -80.30611716061406)

LINESTRING (229.7132548284485 -86.11124987455655, 250.19675094068847 -86.64193589548606)

LINESTRING (228.04333530673878 -118.4570829544022, 210.0409589578147 -108.44196410608683,
191.8745842495791 -119.20722319244867, 181.3300406601329 -125.45584161582418)

LINESTRING (210.0409589578147 -108.44196410608683, 210.572642629994 -90.89640292417018)

LINESTRING (191.8745842495791 -119.20722319244867, 185.58351003756724 -94.08650495724594)

LINESTRING (164.31616315039554 -45.17160711675103, 158.99932642860261 -23.904260229579332)

LINESTRING (158.46764275642332 -16.99237249124853, 167.5062651834713 -15.397321474710655,
168.0379488556506 -13.270586785993485, 163.78447947821624 9.060127445536796)

LINESTRING (154.214173378989 7.465076428998919, 163.78447947821624 9.060127445536796,
169.1013162000917 10.655178462074673, 167.02260417830811 28.769317471875343)

LINESTRING (271.07914613842945 -189.4721659760732, 254.68538305514247 -108.74510311867644,
250.19675094068847 -86.64193589548606, 243.7813941409454 8 -55.05108582681214,
240.36310368248303 -38.21855339949772, 238.75187725549648 -30.284464295730842,
235.20779701155732 -15.00825634771716, 228.18040395562892 15.282230962319,
224.39641810665557 37.239362071039295)

LINESTRING (235.20779701155732 -15.00825634771716, 243.0053466329308 -12.738903113814192,
240.34692827203435 -0.510178653690467, 228.18040395562892 15.282230962319)
```

Figure 4.2: A map based on the “WKT” format

4.3.1 Geographical Casts as Message Destinations

Geographical *casts* are defined based on a list of coordinates. *Routes* and *maps* in the ONE simulator are defined by a list of coordinates as well. The simulator engine imports and uses maps and routes based on the WKT format, which is a text mark up language to represent the geographical objects based on a list of coordinates. Figure 4.2 shows a map definition based on the WKT format. In this format, each route is represented as a collection of coordinates. Any map and routes within, can be transformed into the WKT format. Then, the file should be located inside the data folder of the simulator. At run time, the simulation engine reads the WKT files and imports them into the *SimScenario* class, located in the core package. Then, various parts of the core can use the map and routes as Java objects. Also, the GUI transforms the map and route objects to the shapes that are shown to the user. Movement models use these objects to move nodes on the defined routes during the simulation. Routes (if they are available) are assumed as the only paths between two points in the simulation environment and nodes can only use these routes to find a path to the destination location.

We implemented a similar approach for defining and importing geographical casts into the simulator. First, a cast should be drawn on a map and transformed to the WKT format. We use OpenJump [79] to draw casts on maps based on the WKT format. “OpenJump is an open source Geographic Information System (GIS) written in the Java programming language. It is developed and maintained by a group of volunteers from around the globe” [79]. Figure 4.3 shows a screenshot of the OpenJump software environment.

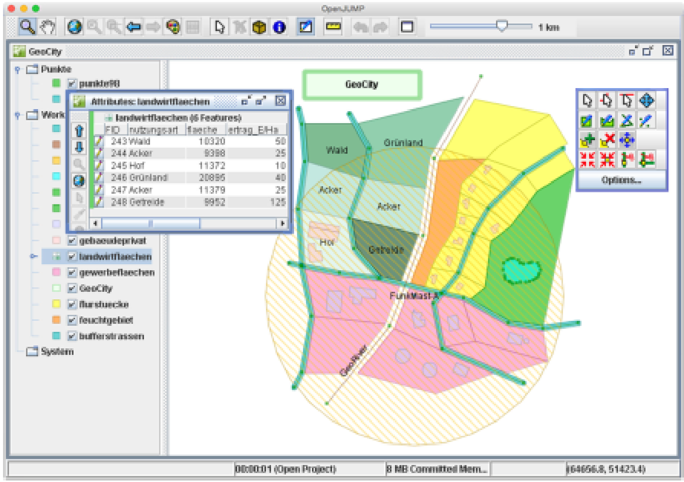


Figure 4.3: Screenshot of the OpenJump software

Each cast is defined based on a set of coordinates. Three classes have been implemented and added to the simulator's core package to add the ability to define the casts: (1) *"WktCastReader"*, (2) *"CastSim"*, (3) *"Cast"*. The *"WktCastReader"* class reads and imports each cast as a list of coordinates. The *"CastSim"* class then creates a *"Cast"* object for each imported cast. A Java object is created for each cast and the list of cast objects is imported into the simulation scenario. Cast objects can be used in different parts of a simulation (e.g. in geocast message generator, geocast message, router, GUI).

We have implemented a mechanism that gets a cast object and the location of a node (as a pair of coordinates in the two-dimensional space) and determines if that node is located inside or outside the given cast. As explained in the § 3.2.1, the "point in polygon" approach has been implemented to provide support for the functionality described above. We have implemented the crossing test (Algorithm 3.1) as it is the most efficient algorithm (as explained in page 44) into the ONE simulator.

In overall, during the simulation time, the list of cast objects is imported into the *"SimScenario"* class and various classes of the core package use these objects. For example, the *"GeoMessageEventGenerator"* uses the information to create destination casts for generated messages and the router object uses the information to route the messages as well as identify if they are located inside the cast or not. Also, GUI uses the cast information to draw the casts.

4.3.2 GeoMessage and GeoMessage Generator

Messages are generated by the message event generator in the ONE simulator. Message properties (e.g. payload size and creation interval) could be set in the scenario settings. Throughout the simulation, the message generator generates messages and gives each generated message a random recipient address (a cast) in the network. Then, it hands the message to a randomly selected sender node to start the routing process. The ONE simulator only supports unicast routing scenarios, hence, each message has an EID of one node as its recipient address. In our implementation, both unicasting and geocasting can be supported simultaneously. To do so, we added the *"GeoMessageEventGenerator"* engine (which generates geocasting messages) to the standard *"MessageEventGenerator"*. The GeoMessage generator creates geocasting messages and choose a random cast, from the list of existing casts in the map, as the recipient address for each one of them. It then hands each message to a random source node just as how the standard Message generator does.

There is a message class in the source code which creates message objects during a simulation. We implemented a *"GeoMessage"* class to deal with messages that are geocast in the network. A GeoMessage object contains the cast information (as a set of coordinates in the two-dimensional space) as its recipient address.

4.3.3 Geocasting Router

The Router object plays an important role in the ONE simulator. Buffer management, message routing and forwarding are managed by the router object at each simulated device in the network. There are three groups of tasks that are managed by three different Java classes:

MessageRouter class
: As shown in Figure 4.4, the *MessageRouter* class is the superclass that is responsible for: (1) buffer management and buffer scheduling, (2) adding incoming messages or newly created messages in the buffer, (3) deleting outgoing messages from the buffer.

ActiveRouter class
: It is the subclass of *MessageRouter* and is responsible for connection establishment, connection management and forwarding messages upon encountering other nodes.

Router class
: This third class takes care of all routing decisions and is the subclass of the *ActiveRouter* class. The actual routing protocol is implemented in this class.

As mentioned earlier, existing router objects only support unicast routing based on EIDs as message destinations. Extending this functionality so that geocasting can be supported, requires dealing with two problems. First, support for multiple recipients for each geocasting message must be added. Hence the router should not delete and eliminate messages from the network whenever a message is delivered to a recipient; instead, it should hand

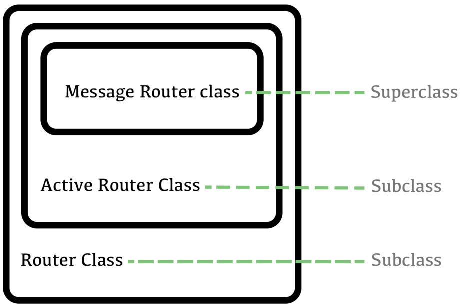


Figure 4.4: Hierarchy of three tier router classes in the ONE simulator

the geocasting message to the application layer but also keep a copy in the buffer for forwarding to other nodes in the future. Second, in the current version a router discovers final recipients by matching the EID of a message to the EID of the node. Although a geocasting message carries the cast information instead of EID in its header, the router object should so be changed so that cast membership is tested, as mentioned above. Both of the aforementioned issues are related to the first two Java classes (*MessageRouter* and *ActiveRouter*) in the ONE simulator. We have changed both classes accordingly so that both unicast and geocast routing are supported at the same time even for the same simulation.

A. GeoEpidemic Router

All existing routing protocols are subclasses of the *ActiveRouter* class in the ONE simulator. The *ActiveRouter* class itself is a subclass of the *MessageRouter* class. The epidemic routing protocol [27] could work for geocasting scenarios with minimal changes in its design. The Epidemic protocol could be adapted to the geocasting environment because of its flooding (broadcasting) nature. In epidemic routing, every message is flooded to all nodes in the network to find the addressed destination. Thus it could be adapted to a geocasting scenario to deliver messages to a number of nodes instead of just one. In this approach, every node receives the message although it is not necessarily a recipient. This could be a method to deliver geocasting messages; it would however waste a huge amount of buffer space in the network and incur significant network overhead. By changing the *MessageRouter* and *ActiveRouter* classes to support geocasting (as discussed above), only minor tweaks were required to adapt the Epidemic router to support geocasting. To avoid any confusion with epidemic routing for unicast routing scenarios, from now on, we will be using the term *"GeoEpidemic"* to refer to the geocasting-capable version of the Epidemic router.

B. GSAF Router

GSAF's design has been explained in § 3.2.2.A. GSAF consists of two phases: (1) The message is first forwarded to the destination cast and (2) is flooded to all nodes in the destination cast, in a controlled fashion where messages are not allowed to exit the cast. Algorithm 3.2 illustrates the design of GSAF and its implementation in the ONE simulator. The router object applies the algorithm for each message that needs to be relayed to encountered devices. At the time of connection between two hosts, each of them (as a sender [*Lines* 3 *to* 18] or receiver [*Lines* 19 *to* 30]) has a separate to-do list that is specified inside the algorithm. Note that both of the routing phases (one and two) are embedded inside the algorithm and it can identify the routing phases spontaneously.

GSAF is implemented in the router class as a subclass of the *GeoActiveRouter* class (the geocasting-enabled *ActiveRouter* class) and *GeoMessageRouter* (the geocasting-enabled *MessageRouter* class). Each host has a object of router inside, in the way that it can perform routing decisions as well as membership control for each message in its buffer, separately, during an encounter.

C. DA-GSAF Router

As discussed earlier in § 3.2.2.B., the implementation of DA-GSAF is similar to GSAF but the conditions for the first phase. Two additional algorithms are needed to implement the DA-GSAF router, as it utilises direction-related information to enhance forwarding decisions. Algorithm 3.4 shows how a node can calculate its own direction in the ONE simulator. We added a functionality which records the previous coordinates of the node and updates it in each time interval. This enables each node to calculate its own direction during an encounter.

Moreover, the *direction* towards the *destination cast* is calculated with the combination of Algorithms 3.5 and 3.4 described in Algorithm 3.3 (*Lines* 15 *and* 16). The implementation of DA-GSAF is also described in Algorithm 3.3 . *Line* 13 evaluates condition 1, *Line* 17 evaluates condition 2 and *Line* 20 evaluates condition 3 (as enumerated in Page 50). Similar to GSAF, DA-GSAF is implemented as a subclass for *"GeoActiveRouter"*.

D. EVR and GEOOPP Routers

In the context of this thesis, we have compared the proposed schemes with prior work in geocasting in opportunistic networks; i.e. GeoEpidemic, EVR and GEOOPP routing protocols. We had to implement EVR and GEOOPP from scratch as there was no available implementation for the ONE simulator. Extensive description of both protocols are available in § 2.5 as well as [16] and [18]. Both of the EVR and GEOOPP protocols are required to divide the network into non-overlapped equal cells to perform and apply their routing decisions. We implemented this functionality into the ONE simulator engine, especially for the use of these protocols as we would like them to operate exactly in the same described conditions. The latter case has improved the accuracy of comparison that is presented in Chapter 5 .

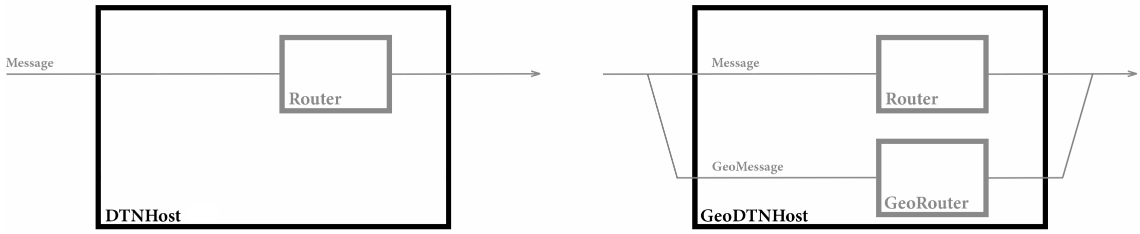


Figure 4.5: Diagram of the DTN Host with one router and the Geo DTN Host with two routers

4.3.4 Enabling ONE Users for Geocasting

In the ONE simulator, the *"DTNHost"* class is responsible for simulating network nodes. A DTNHost object keeps the reference to a router object, which is responsible for all message-related activities.

In our implementation, each one of the implemented routing protocols can operate in parallel to the unicast routing protocols supported by the ONE simulator. This is achieved by having two separate Router subclasses that are instantiated and operate in parallel; one for unicast and the other for geocast routing. Both of them use the same network interfaces that are supported by each network device to send and receive messages. When the connection gets into idle mode by one of the routers, the other one is allowed to use it to forward its own messages. All this functionality is wrapped under a an extended ONE *host* implementation, called *"GeoDTNHost"*. This design enables the handling of unicast and geocast messages, simultaneously, during the same communication window in the ONE simulator.

For each received message, the Bundle layer identifies whether it is a unicast or geocast message based on its recipient address; i.e. an EID or a cast and hands it to the respective router object. As shown in Figure 4.5, a *GeoDTNHost* object contains references to two separate protocols, for unicasting and geocasting messages.

4.3.5 Listeners and Reports

Listeners and Reports are vital to collect the simulated data. We have implemented and/or modified fourteen report classes, to support reporting of all activities relevant to geocasting and enable effective processing of the simulation output.

A. Metrics and Measurements

Apart from collecting data, reporting classes produce various reports (e.g. delivery probability, delivery latency and number of relayed messages) based on events that are captured during the simulation. We use these reports to evaluate the proposed geocasting schemes through three metrics, namely the *message delivery ratio*, *message delivery latency* and *number of relayed messages*.

■ Message delivery ratio

In geocasting, messages are not addressed to specific network devices but to geographical areas where multiple devices may reside during the lifetime of a message. Contrary to unicast protocols where the delivery status of a message can have two values (*delivered* or *undelivered*), in geocasting both the spatial and temporal aspects should be taken into account. The message delivery ratio for a unicast protocol would be calculated as follows:

$$unicast\ delivery\ ratio = \frac{number\ of\ delivered\ messages}{number\ of\ created\ messages} \tag{4.1}$$

In geocasting, each message has a delivery ratio itself, instead of a mere delivery status (*delivered* or *undelivered*). The per-message delivery ratio (*pmdr*), which is the fraction of the number of devices that were located in the cast throughout the lifetime of the message and received the message (*actual number of recipients*) to the total number of devices that should have received the message (*total number of recipients*), is calculated as follows:

$$pmdr = \frac{actual\ number\ of\ recipients}{total\ number\ of\ recipients} \tag{4.2}$$

Equation 4.2 calculates the delivery ratio of a single message. The overall delivery ratio (*odr*) of the geocasting protocol is measured based on the delivery ratio of all created messages, as shown in Equation 4.3.

$$odr = \frac{\sum_{i=1}^{n} pmdr\ i}{number\ of\ created\ messages} \tag{4.3}$$

■ Message delivery latency

The same rationale is followed when measuring the message delivery latency. We measure the time it takes for a message to reach a recipient node (i.e. a node in the destination cast) and calculate the per-message delivery latency. The overall delivery latency (*odl*) is the average for all created messages.

■ Number of relayed messages

The number of all relayed messages in the network provides an indication of the efficiency of a geocasting protocol with respect to the amount of generated traffic. This also provides a hint on the energy efficiency of the protocol, as the number of stored and transmitted messages affects the battery life of mobile devices in the network.

Algorithm 4.1 Implementation of overall delivery ratio in the ONE simulator

$n \leftarrow$ the overall number of generated geocasting messages
for each generated geocasting message in the network **do**
  create a list of recipients
  monitor the recipient cast of the message during the simulation runtime
  if a node visited the cast in the delivery time interval **then**
    put its ID into the list of recipients
  end if
  for each node in the recipient list **do**
    check if the node has received the message
  end for
  $x \leftarrow$ number of nodes who have received the message
  $y \leftarrow$ total number of nodes in the recipients list
  $pmdr$ (*per message delivery ratio*) $\leftarrow x \;/\; y$ (*equation* 4.2)
end for
return odr (*overall delivery ratio*) $\leftarrow \left(\sum_{i=1}^{n} pmdr\ i \right) / \; n$ (*equation* 4.3)

B. Implementation of Geocast Delivery Ratio

Calculating the *message delivery latency* and the *number of relayed messages* for geocasting scenarios is identical to unicasting. However, it is not the same for the *message delivery ratio* due to the mentioned differences. The calculation of *pmdr* (equation 4.2) and *odr* (equation 4.3) is implemented in the ONE simulator as shown in Algorithm 4.1 . For each message, listeners and report classes observe the network to find the list of recipient nodes based on the delivery time interval (described in § 3.2.2). The implemented

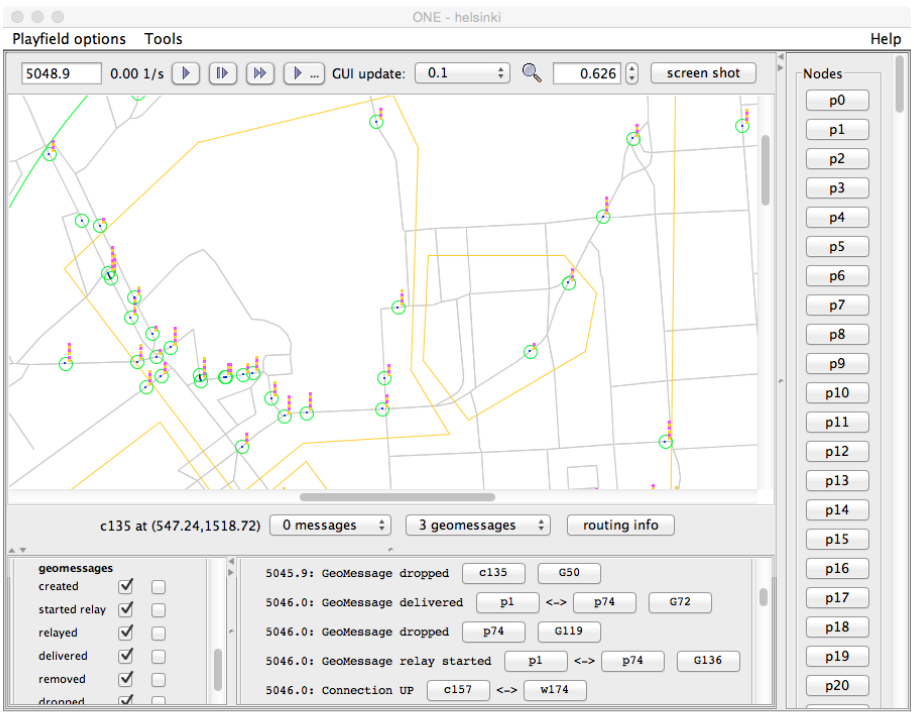


Figure 4.6: Screenshot of the modified version of the ONE simulator
(∗ Amber lines show the boundaries of imported geographical casts)

"GMStatProReport" class in the ONE simulator is responsible for calculating all aforementioned metrics for geocasting based on Algorithm 4.1 .

4.3.6 Graphical User Interface

As mentioned in § 4.1.6, ONE's GUI shows real-time information about the simulation scenario while the simulation runs. We have implemented functionality so that cast- and geocasting-relevant information is shown in the GUI. Imported geographical casts are added to the map and the user can see and calibrate their exact location through adjusting the *"cast offset setting"* in the scenario settings. Geocasting messages and their recipient casts, details about the unicast messages and geocast messages at each node are available at run-time through ONE's GUI. Figure 4.6 shows the GUI in our modified version of the ONE simulator.

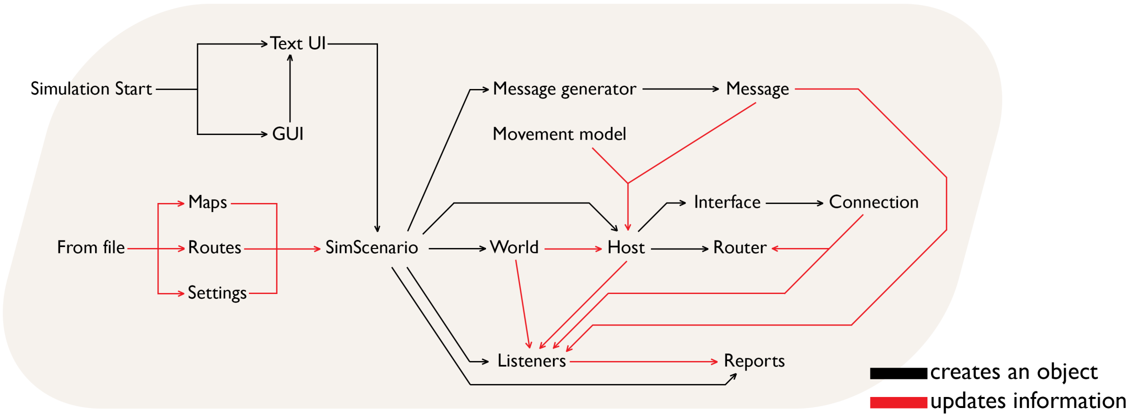


Figure 4.7: The overall interaction of objects in the ONE simulator

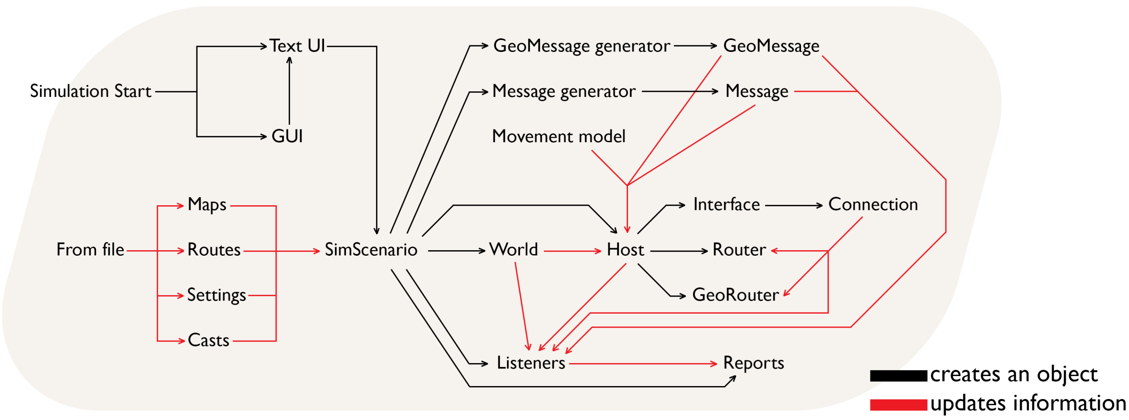


Figure 4.8: The overall interaction of objects in the ONE simulator with the Geo ONE package

4.4 The Geo ONE package

All implemented functionality is packed to the *Geo ONE* package which could enable researchers experiment with existing and future geocasting protocols in the ONE simulator. *Geo ONE* consists of various parts including cast definition and import, geocast message and message generator, geocast enabled routers (i.e. GSAF, DA-GSAF, GeoEpidemic, EVR and GEOOPP), extra listeners and reports and a long list of minor changes. *Geo ONE* consists of extra ∼10000 lines of code which has increased the ONE source code by ∼1/3 overall. Figure 4.7 and 4.8 illustrate the main parts of the ONE simulator engine and the added *Geo ONE* package.

Chapter 5

Evaluation & Result Analysis

The knowledge of anything, since all things have causes, is not acquired or complete unless it is known by its causes.
Avicenna
Iranian Polymath (980 - 1037)

In this chapter we present the results of the extensive experimental evaluation of GSAF and DA-GSAF, in comparison with existing geocasting protocols. We have designed a number of realistic simulation scenarios which we run using the ONE simulator (more specifically the Geo ONE package we described in § 4.4). We are interested in the metrics described in § 4.3.5.A., namely the message delivery ratio, message latency and the induced network traffic.

First, we examine the performance of the proposed protocols in a confined geographical area, such as the University of Sussex campus (§ 5.1). Second, we investigate the behaviour of GSAF and DA-GSAF when operating in larger environments like the city of Helsinki (§ 5.2). We then look at how different mobility models affect the performance and efficiency of our protocols in § 5.3 . We evaluate the impact of different buffer scheduling policies, the *copy ticket value* and the *size of casts* in § 5.4 , § 5.5 and § 5.6 , respectively. Then, we investigate how GSAF and DA-GSAF behave when the number of user devices increases (§ 5.7). Finally, we summarise the results of the performance evaluation in § 5.8 .

5.1 Sussex University Campus

The first set of experiments is to evaluate the behaviour of the proposed protocols in scenarios where the network is within a confined geographical area. This type of opportunistic networking is very common, for example when a network is set up across a festival venue or a public demonstration. As mentioned above, for this set of experiment we simulate networks that operate across the campus of the University of Sussex.

Figure 5.1: Simulation map of Sussex university with its pre-defined casts[1]

Table 5.1: The movement model of pedestrians

	Speed	Mobility Model
Pedestrian	0.5 - 1.5 (m/s)	Shortest Path Map Based Movement[3]

5.1.1 Simulation Scenario

In the first scenario, we simulate the performance of GSAF and DA-GSAF in the Sussex university campus (which covers an area of 1150×1450 meters). Although GSAF does not rely on static, pre-defined casts, for evaluation purposes only, we have created 11 casts[2] for the Sussex university, as illustrated in Figure 5.1 . All messages are destined to one of these casts. In this scenario, we use only pedestrian users (due to the nature of the simulated scenario). We range the number of users that are employed with devices that support opportunistic networking from 40 to 320 . We use the default mobility model for the pedestrians in the ONE simulator, which is summarised in Table 5.1 . Unless otherwise stated, the default total number of users in our simulations is 120 in this scenario.

Using the same simulation scenario, we have experimented with different sizes of device buffers (5, 10, 15, 20, 25 and 30 MB; the default being 10 MB) and different message lifetimes (30, 60, 90, 120, 150, 180, 210 and 240 minutes; the default being 120). We have simulated our protocols with two wireless interfaces; (1) WiFi 802.11ac with a transmission speed of 433 Mbps and a range of 20 meters, and (2) Bluetooth 802.15 v4.0 with a

[1]This map is created with the help of Google maps and the OpenJump [79] software.

[2]Note that non-overlapping casts, as shown in Figure 5.1, is not a design restriction for the proposed protocols.

[3]With this mobility model a user is assigned with a shortest path route from its current location in the map to a randomly selected destination point. A new such path is calculated by the simulator every time the user reaches the previously calculated destination.

transmission speed of 2 Mbps and a range of 10 meters. The simulated time for all simulations is 16 hours; the warm-up and cool-down periods[4] for each simulation were 2 hours, therefore our results are drawn from 12 hours of simulated time for each run. We repeat each simulation 5 times with different seeds for the mobility model. We schedule messages as follows: a sender and a destination cast are selected uniformly at random from the set of devices present in the network and the set of pre-defined casts, respectively; the message size is fixed (500KB) and a new message is scheduled every 25 to 35 seconds (a values selected uniformly at random from this time range; the ONE simulator default). Also, the buffer scheduling policy is the random one in this scenario (as we will evaluate various scheduling policies in § 5.4), which means that messages are randomly selected from the device buffer when a device encounters another device in the network. Finally, we compare GSAF and DA-GSAF against the GeoEpidemic, EVR and GEOOPP routing protocols[5].

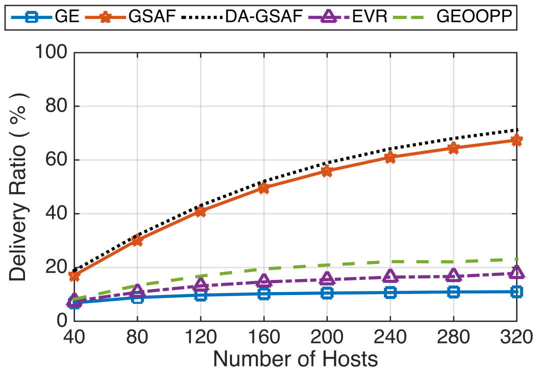
5.1.2 Influence of user density

For this set of simulations, we used the default values for the *buffer size* and *message lifetime* and we varied the *number of users* in the map. Our objective is to examine the impact of user density to the delivery ratio and latency of messages and the induced network overhead. Figure 5.2 shows the results. We observe that the increase in user density improves the message delivery ratio, whereas the average latency is not significantly affected. GSAF and DA-GSAF deliver significantly more messages compared to the other protocols while maintaining similar levels of delivery latency (although in the Bluetooth scenario GSAF performs much better). The diagrams from Figure 5.2 (e and f) show the amount of generated network traffic in terms of relayed message copies in the network. It is evident that GSAF and DA-GSAF not only outperform GeoEpidemic, EVR and GEOOPP but also do so by inducing significantly lower network overhead. Also note that the observed delivery ratio for GeoEpidemic decreases when the supported transmission rate is higher (i.e. in the WiFi case). This counter-intuitive observation is because when more messages are flooded in the network (as in GeoEpidemic), then new messages quickly flood the network overriding older and mostly undelivered messages from devices' buffers.

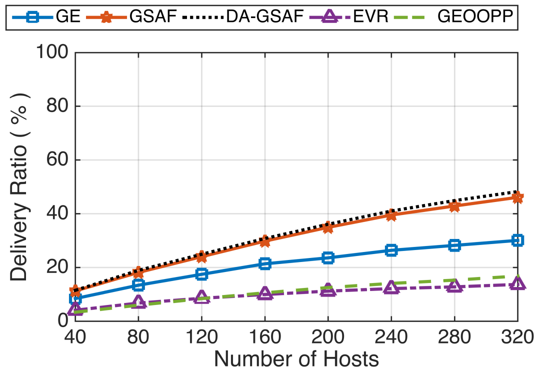
The results presented in Figure 5.2 refer to data drawn from all casts in the network, and are indicative of the overall performance of the protocols. However, the very nature of opportunistic networks is such that some casts are less popular than others in terms of how often users visit them. In Figure 5.3 we present the results for the upper 10% of the most visited casts (2 casts out of 16) in the Sussex campus (where bars indicate

[4]The warm-up period is necessary to fully deploy the network and fill the buffers with messages as well as to establish the movement patterns of the nodes. Moreover, applying the cool-down period can exclude newly created messages which did not have the time to get delivered in the remaining (simulated) time.

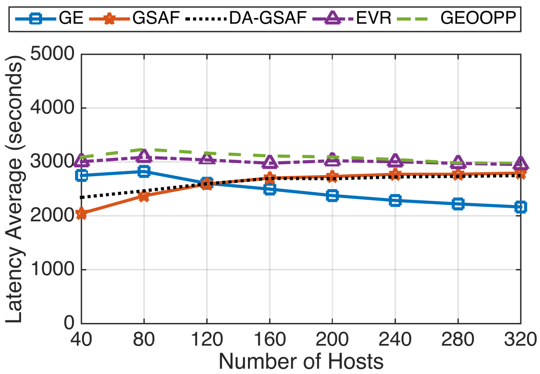
[5]Throughout the whole evaluation chapter we have used the same cast definition methodology and the same casts (as described in the simulation scenarios) for all protocols to keep the comparison fair. Since EVR and GEOOPP require the centre of a circular cast for their calculations, we make sure both can obtain the centre point of the polygonal-shaped casts based on Algorithm 3.5 .



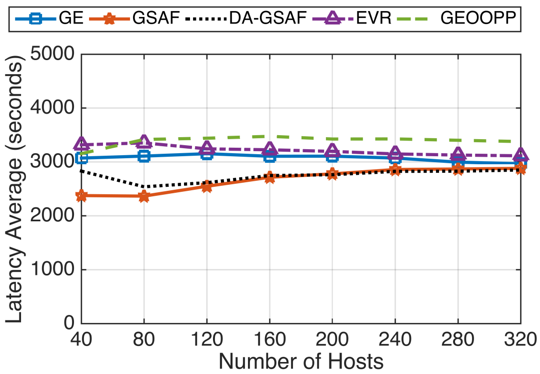
(a) Delivery Performance (WiFi)



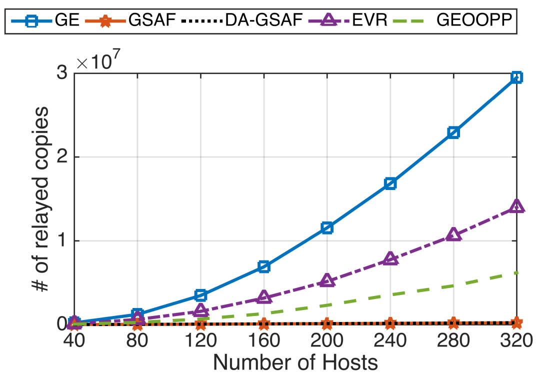
(b) Delivery Performance (Bluetooth)



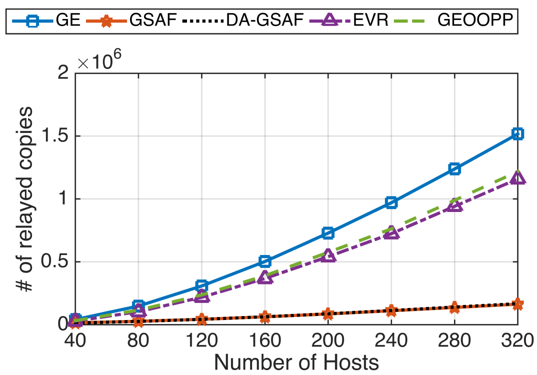
(c) Average Latency (WiFi)



(d) Average Latency (Bluetooth)

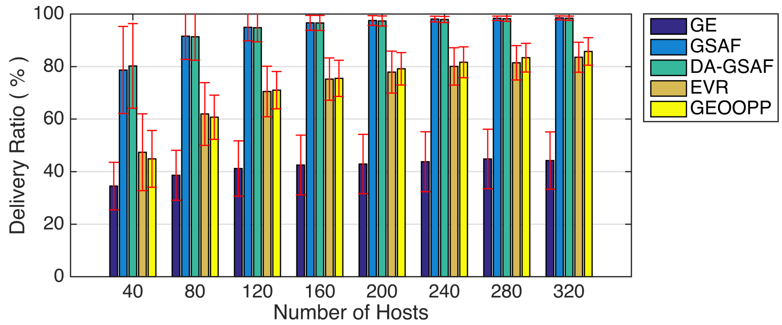


(e) Network Traffic (WiFi)

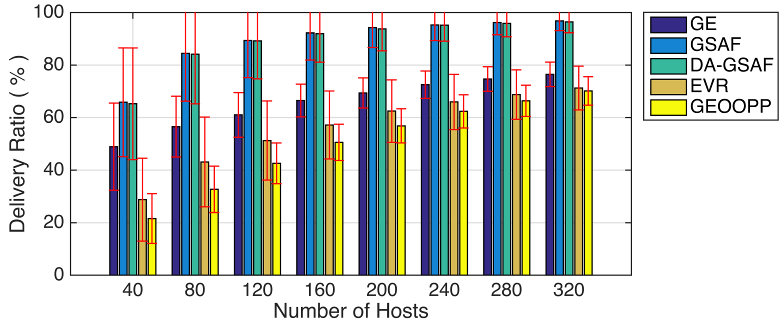


(f) Network Traffic (Bluetooth)

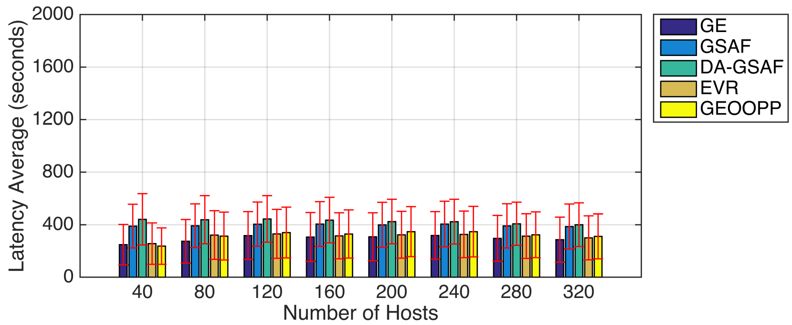
Figure 5.2: Influence of user density on the performance of geocasting protocols (Sussex University)



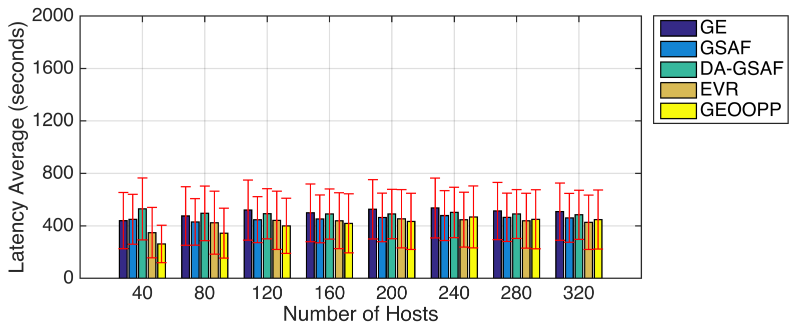
(a) Delivery Performance (WiFi)



(b) Delivery Performance (Bluetooth)



(c) Average Latency (WiFi)



(d) Average Latency (Bluetooth)

Figure 5.3: Influence of user density on the performance of geocasting protocols; Upper 10% of most visited casts (Sussex University)

the standard deviation). As depicted in the figure, the delivery ratio for both GSAF and DA-GSAF is significantly higher compared to all other protocols. Note that for crowded scenarios our protocols lead to delivery ratios close to 100%. Given that GSAF and DA-GSAF perform marginally worse compared to other protocols with respect to the delivery latency, although the standard deviation for these measurements is high relatively to the average values. We consider this as a negligible penalty we pay for dramatically reducing the network overhead and increasing the delivery ratio.

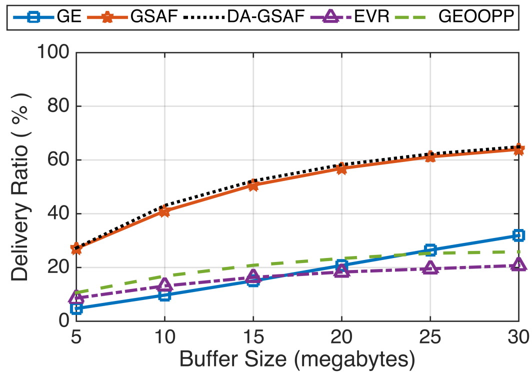
5.1.3 Influence of buffer capacity

For the second series of experiments, we keep the *user density* and *message lifetime* constant, in order to observe how the *buffer size* affects the performance of the proposed protocols. In Figures 5.4.a and 5.4.b the delivery ratio for GSAF and DA-GSAF is higher compared to the rest of the protocols. When WiFi is used, the message delivery ratio increases with the buffer availability because a device can exchange all its currently stored messages with other devices when they encounter each other. However, the situation is different when Bluetooth is used; the delivery ratio reaches a plateau ($\sim$35%) when the buffer size is 20 MBs. As the buffer size increases, a larger number of messages can be carried by each device in the network. However, in opportunistic networks there is not always enough time to exchange all buffered messages given the limited bandwidth and, more importantly, the mobility of users. As a result, increasing the size of a buffer does not necessarily mean that the performance of a protocol is increased. Note that, as in the previous experiment, GSAF and DA-GSAF keep the induced network overhead to a minimum (Figures 5.4 [e and f]). The average latency (Figures 5.4 [c and d]) follows a similar pattern to the results presented in the previous experiment. Note that the latency increases along with the buffer size for all geocasting protocols due to the reason mentioned above. A buffer capacity of 20 to 25 MBs is adequate for GSAF and DA-GSAF (as well as for all other protocols) to handle all network traffic.

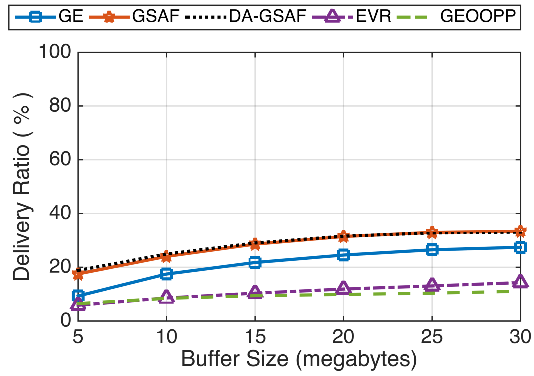
In Figures 5.5.a and 5.5.b we observe that the proposed protocols perform exceptionally well with respect to the delivery ratio of messages in the upper 10% of the most visited casts. Note that although the increase in performance for GSAF and DA-GSAF in Figure 5.4.b (Bluetooth) is not as significant as for the WiFi case (Figure 5.4.a), this drastically changes for the delivery ratio for the most visited casts, where for both the Bluetooth and WiFi scenarios the increase is significant. This indicates that GSAF and DA-GSAF can perform very well in crowded environments even when used with lower bandwidth wireless technologies, such as Bluetooth.

5.1.4 Influence of message lifetime

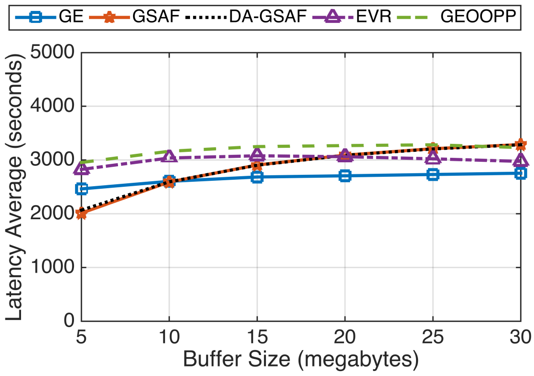
The third set of simulations are designed to study the impact of message lifetime on the performance of the proposed geocasting protocols. We define the message lifetime as



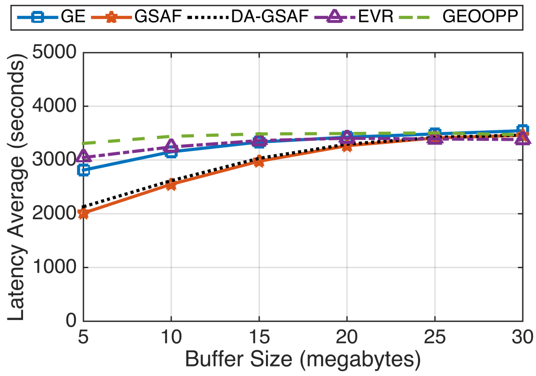
(a) Delivery Performance (WiFi)



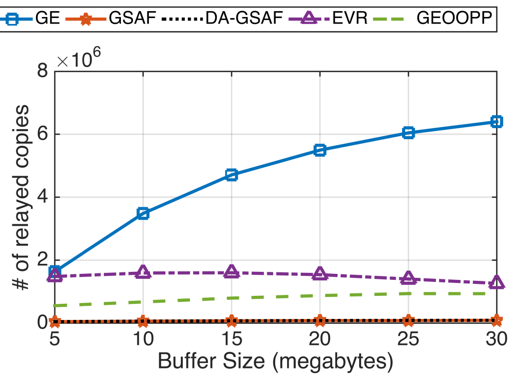
(b) Delivery Performance (Bluetooth)



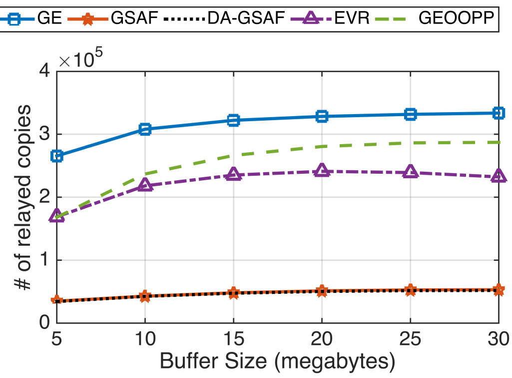
(c) Average Latency (WiFi)



(d) Average Latency (Bluetooth)

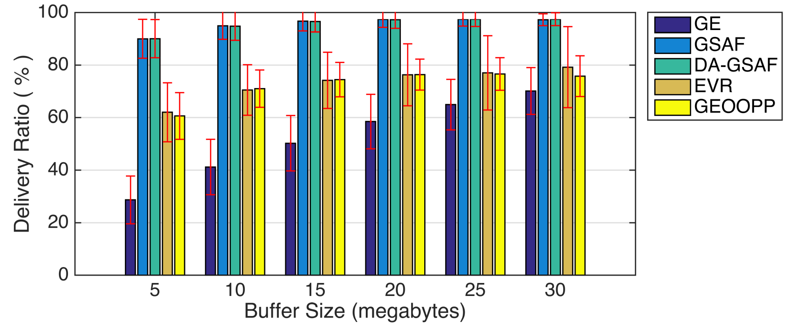


(e) Network Traffic (WiFi)

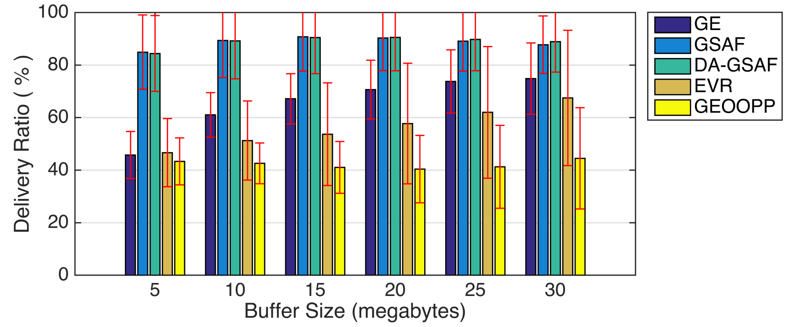


(f) Network Traffic (Bluetooth)

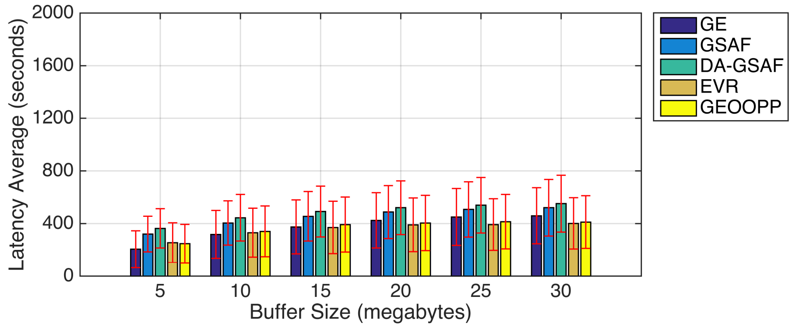
Figure 5.4: Influence of buffer capacity on the performance of geocasting protocols (Sussex University)



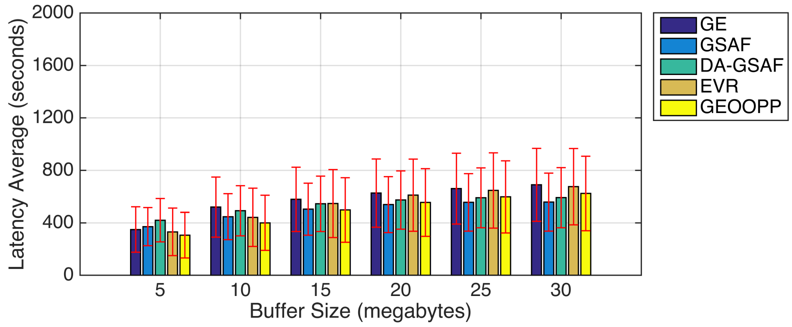
(a) Delivery Performance (Wifi)



(b) Delivery Performance (Bluetooth)



(c) Average Latency (Wifi)



(d) Average Latency (Bluetooth)

Figure 5.5: Influence of buffer capacity on the performance of geocasting protocols; Upper 10% of most visited casts (Sussex University)

the "delivery interval" during which a message is valid. This interval effectively defines the recipients located in a geographical cast. As shown in Figures 5.6.a and 5.6.b, the delivery ratio decreases as the message lifetime increases. This is counter-intuitive and highlights the rather complex nature of opportunistic networks; one would expect that as the lifetime of a message increases, then there would be more time to deliver it in its destination cast. However, longer lifetimes imply the need for larger buffers to store (and carry) the messages and higher bandwidth to exchange them. Given the finite (and rather limited) nature of both of them the delivery ratio actually decreases as the message lifetime increases, because messages are being extinct due to the lack of buffer availability. Also note that longer message lifetimes mean larger number of recipients (that resided in the cast within the message lifetime [see § 3.2.2]) which through time may have moved out of the cast and never received the message.

The results for the delivery latency (Figures 5.6 [c and d]) and network traffic (Figures 5.6 [e and f]) show that GSAF and DA-GSAF perform better than all other geocasting protocols. Note that the number of relayed messages hits a plateau when the TTL is 90 minutes. The bottleneck here is the size of the available buffer in the network devices (i.e the default value of 10 MBs). The results in Figure 5.7 are similar to the ones presented in the previous experiment.

5.1.5 Per-message delivery

In order to get a better idea about how values of message delivery ratio are distributed for all different messages across all casts in the Sussex campus, we present Figures 5.8 (a to e), which depict scatter plots of the per-message delivery ratios (for 1400 messages being generated during each simulation) for all routing protocols. These results are extracted by running a simulation with the default values, as described in § 5.1.1 . As shown in the figures, GSAF and DA-GSAF result in a much larger of messages with delivery ratios that are higher than 50% compared to all other protocols. Note that the number of messages (out of 1400) that are never delivered to their destination cast is 213 for GSAF, 145 for DA-GSAF, 848 for GeoEpidemic, 1027 for EVR and 938 for GEOOPP.

Overall, these results show that GSAF and DA-GSAF perform significantly better when messages are disseminated in geographically confined areas (such as a University campus), especially for crowded casts (where in most cases messages would be destined to in real-world scenarios). GEOOPP performs marginally better compared to EVR, while EVR performs slightly better in more crowded areas (upper 10% of most visited casts). The flooding nature of GeoEpidemic means that it induces a tremendous network overhead which has ramifications in terms of battery consumption.

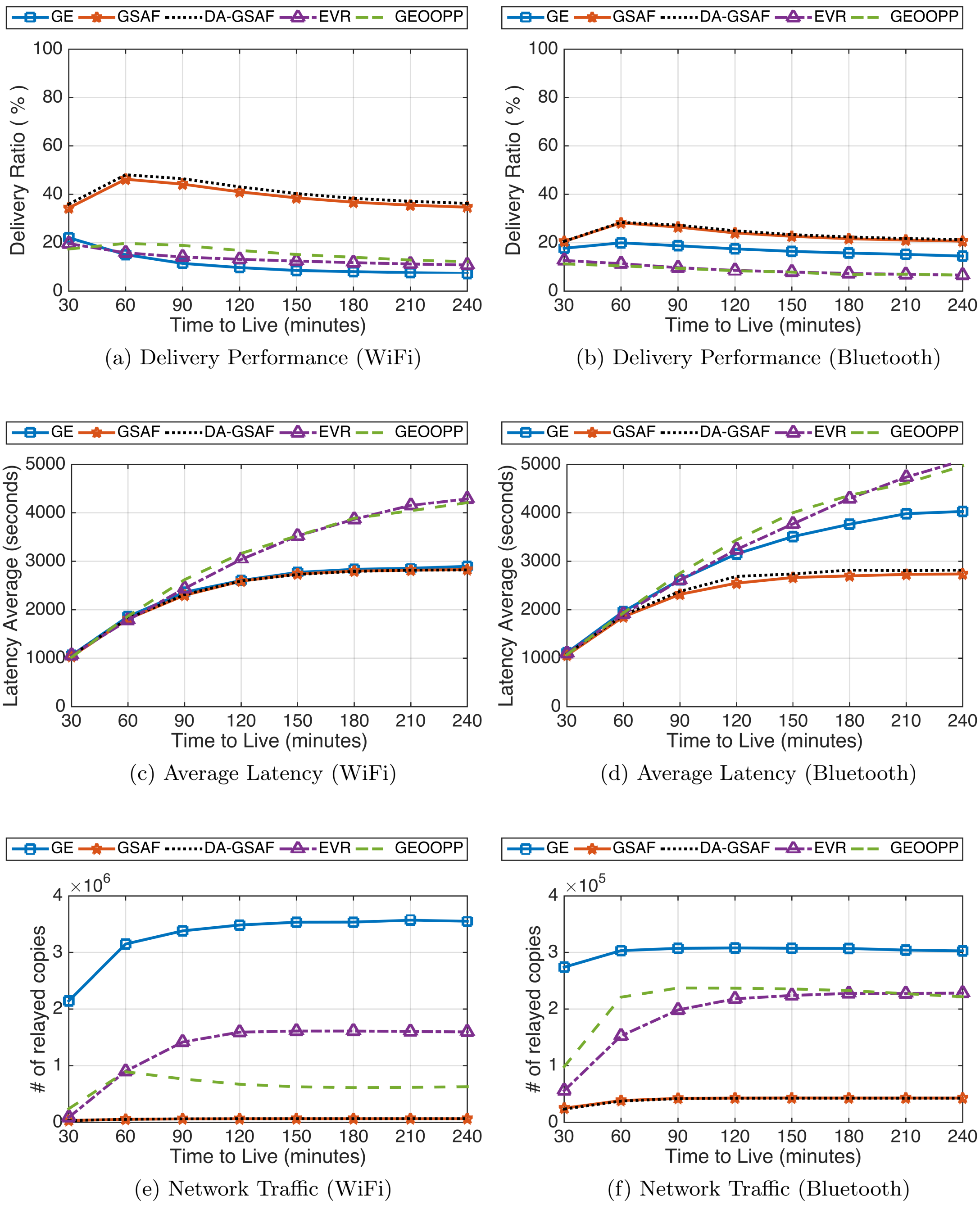
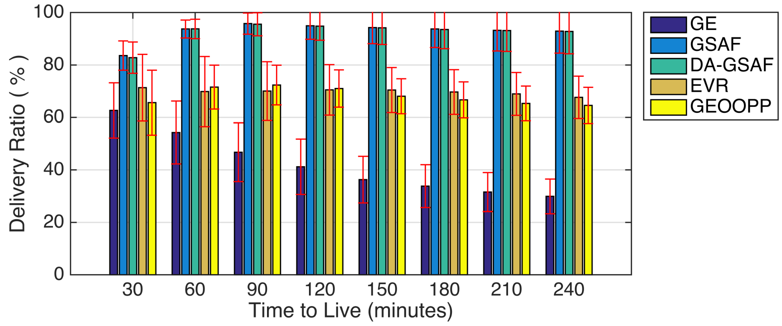
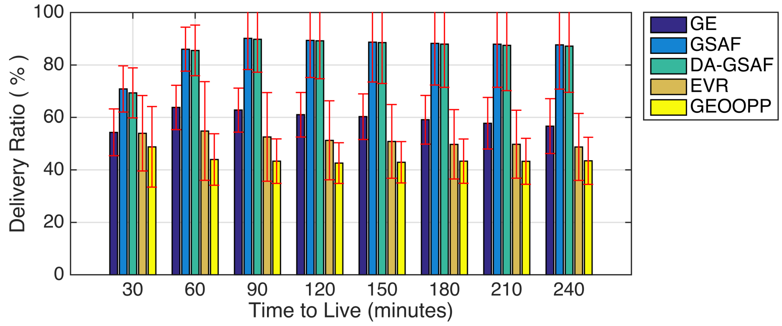


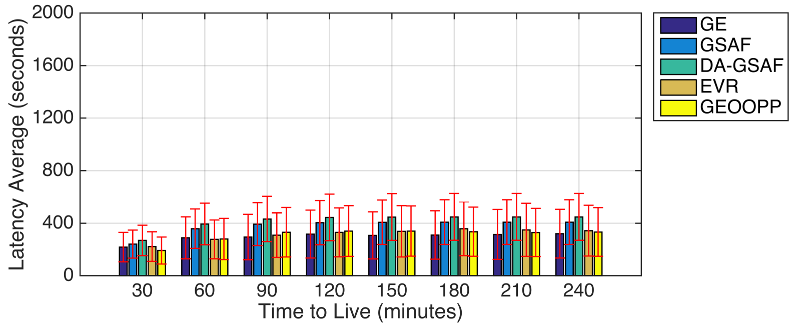
Figure 5.6: Influence of message lifetime on the performance of geocasting protocols (Sussex University)



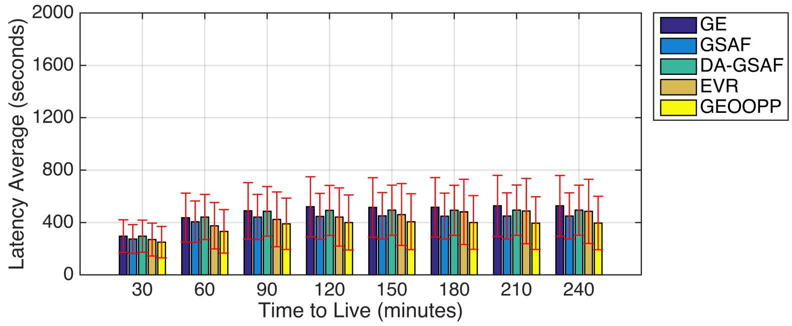
(a) Delivery Performance (WiFi)



(b) Delivery Performance (Bluetooth)

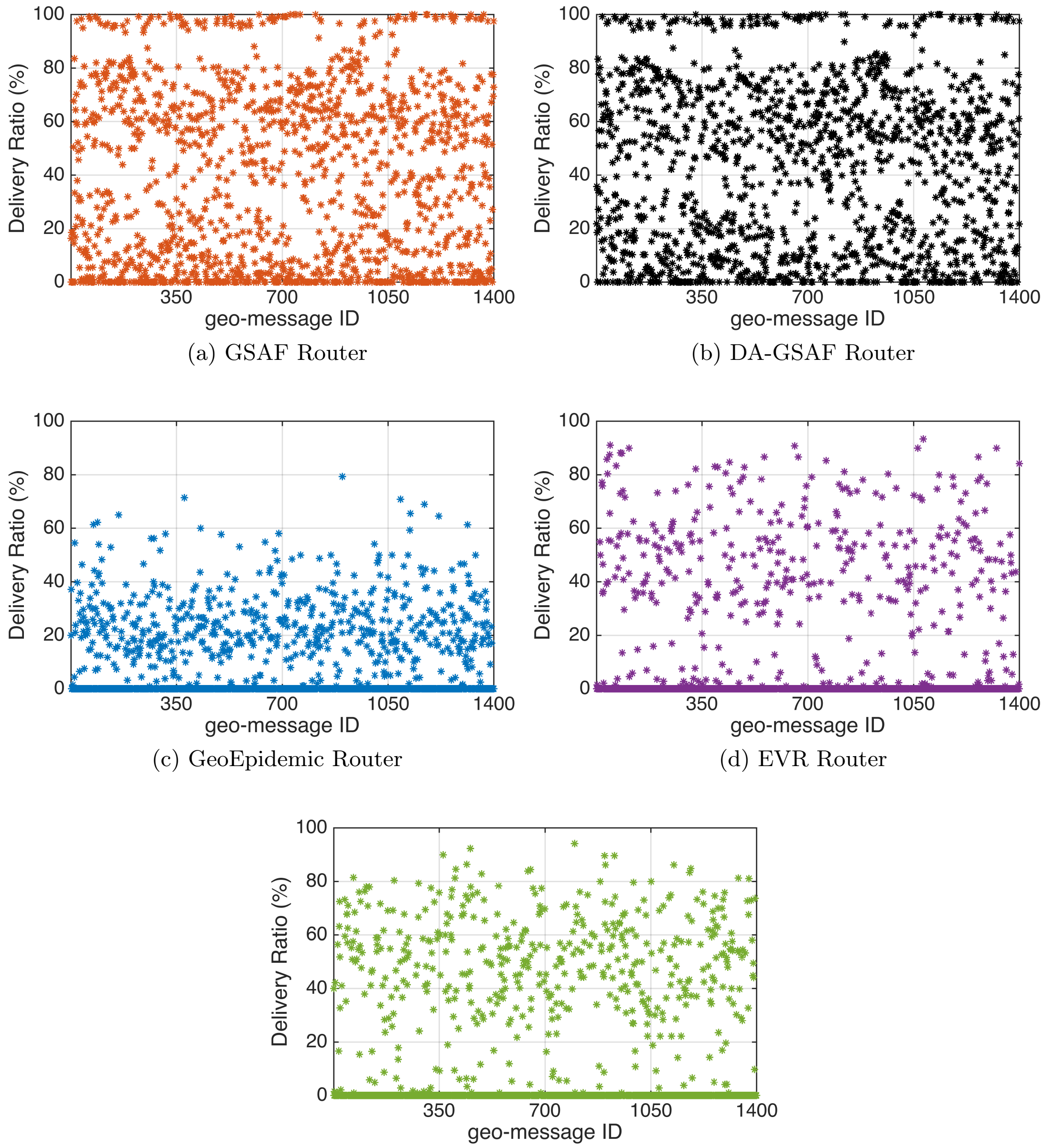


(c) Average Latency (WiFi)



(d) Average Latency (Bluetooth)

Figure 5.7: Influence of message lifetime on the performance of geocasting protocols; Upper 10% most visited casts (Sussex University)



(a) GSAF Router

(b) DA-GSAF Router

(c) GeoEpidemic Router

(d) EVR Router

(e) GEOOPP Router

Figure 5.8: Per-Message Delivery Ratio (Sussex University)

Figure 5.9: Simulation map of Helsinki city centre
with its predefined casts

5.2 Helsinki City Centre

Apart from performing well in geographically confined areas, such as a University campus or an open-festival venue, routing protocols, and geocasting ones in particular, must perform equally well in larger scale communication scenarios, such as in large cities. As described in the motivation of this thesis, there are use cases where the formation of opportunistic networks in large cities could be very important. In the second set of simulations, we investigate how GSAF and DA-GSAF perform in larger scale scenarios. We have used the ONE simulator to simulate all implemented protocols in the Geo ONE package in the city centre of Helsinki (an area of 4500×3400 meters).

5.2.1 Simulation Scenario

We have defined 16 casts in the map of Helsinki, as illustrated in Figure 5.9 . In this set of simulations, we have experimented with 8 different levels of user density (65, 130, 195, 260, 325, 390, 455 and 520; the default being 195 unless otherwise stated). All simulations involve three types of users; Pedestrians, Cars and Buses. The detailed number of users for each type of user is shown in Table 5.2 . Also, the mobility models of all three simulated types of users are shown in Table 5.3 . All the other parameters remain as described in § 5.1.1 .

There are five different bus routes in the map including a number of bus stops, as illustrated in Figure 5.10 . Buses are distributed equally and serve a single route (round trip) throughout the duration of a simulation.

In this scenario, we evaluated the performance of GSAF, DA-GSAF, GeoEpidemic and EVR. GEOOPP [18] involves a complex algorithm, for deciding when to route a message,

Table 5.2: Detailed number of users

Total #	65	130	195	260	325	390	455	520
Pedestrian	40	80	120	160	200	240	280	320
Car	20	40	60	80	100	120	140	160
Bus	5	10	15	20	25	30	35	40

Table 5.3: Users' mobility models

	Speed	Movement Model
Pedestrian	0.5 - 1.5 (m/s)	Shortest Path Map Based Movement
Car	2.7 - 13.9 (m/s)	Shortest Path Map Based Movement
Bus	7 - 10 (m/s)	Bus Movement

that is computationally heavy. Simulating GEOOPP requires significantly more time compared to all other protocols (around two times more than EVR and four times more than GSAF) as well as more memory. Even using the Sussex HPC infrastructure, it would take around three months to simulate GEOOPP for the scenarios presented below; we therefore decided to exclude GEOOPP from the rest of the experimental evaluation[6].

5.2.2 Influence of user density

Similar to § 5.1.2, in this set of simulations, we keep the *buffer size* and *TTL* constant, while varying the *number of user devices* that geocast the messages in the city of Helsinki. Figure 5.11 shows the results which confirm that GSAF and DA-GSAF perform significantly better than the rest of the protocols. DA-GSAF performs slightly worse compared to GSAF in larger scenarios in terms of message latency. Although the difference is not significant, the reason for that is the fact, the DA-GSAF is more selective when it comes to routing messages to encountered nodes. As in the previous simulation scenario (Sussex university campus), GeoEpidemic's performance does not increase with the number of user devices (as it does with all other protocols). This is because of its flooding nature which results in the fast extinction of messages in the network (as also indicated in § 5.1.2). This argument is also strengthened by the fact that in Figure 5.11.b GeoEpidemic performs better than EVR (and GeoEpidemic with WiFi); this is because more messages can survive the flooding, in contrast to the WiFi scenario, as the network bandwidth is limited when Bluetooth is used.

Figure 5.12 illustrates the performance of all studied protocols for the most visited casts (upper 10%) in the network. All results are consistent with the ones presented in the

[6]It is worth pointing out that, as also shown in the previous scenario, GEOOPP performed similarly to EVR; its performance being much poorer compared to GSAF and DA-GSAF. We strongly believe that this trend would be confirmed for all simulation scenarios where GEOOPP is excluded.

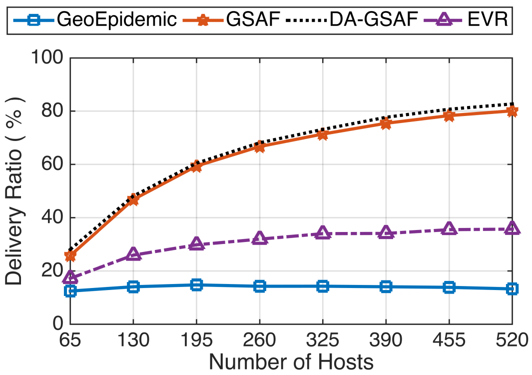
Figure 5.10: Bus stops and their routes in the Helsinki city centre

previous section. Note that, as depicted in Figure 5.12.a (and in Figure 5.11.a), DA-GSAF presents the highest delivery ratios in scenarios with a small number of devices (e.g. just 65). The reason for that is the fact that DA-GSAF takes into account the direction of both nodes during an encounter in order to decide which one is best for carrying it to its destination.

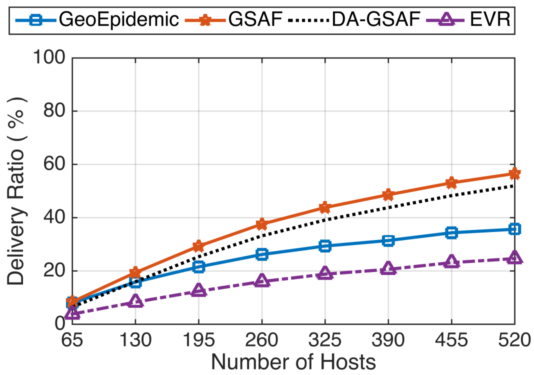
5.2.3 Influence of buffer capacity

Figure 5.13 illustrates the behaviour of the geocasting protocols in the city of Helsinki for varying sizes of devices' buffer. Overall, GSAF is the best in terms of delivery and latency performance while DA-GSAF is more efficient in terms of the induced network overhead (i.e. the total number of relayed messages). Note that in the Bluetooth scenario, as the available bandwidth is low, the delivery ratio for all protocols quickly reaches a plateau and does not increase with the size of the buffer.

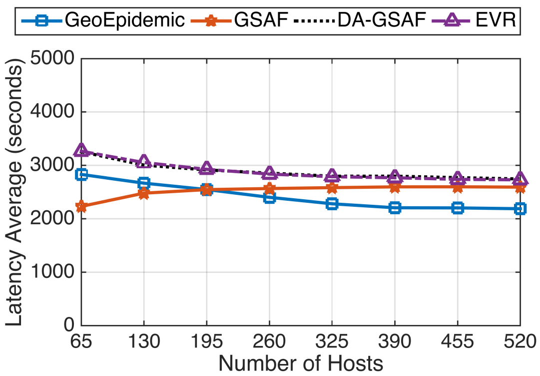
For the most popular casts (Figure 5.14), EVR's performance is much closer to GSAF compared to the average values for all casts. Nevertheless, GSAF and DA-GSAF are still better in terms of delivery ratio and message latency in popular casts.



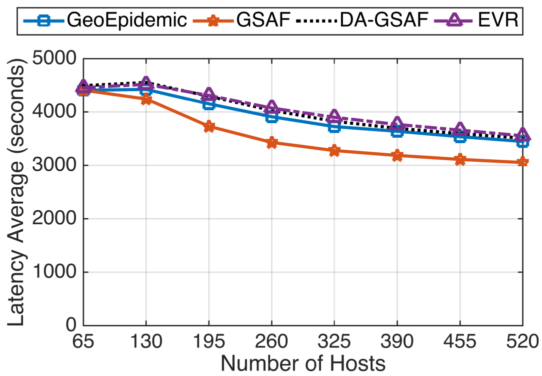
(a) Delivery Performance (WiFi)



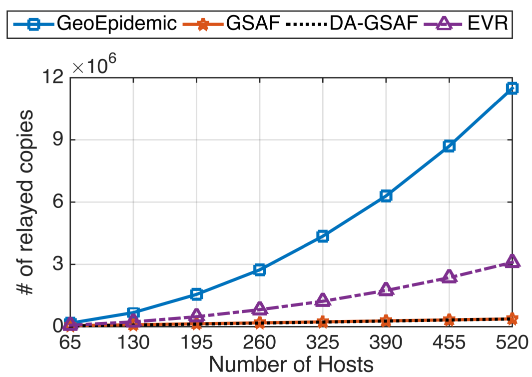
(b) Delivery Performance (Bluetooth)



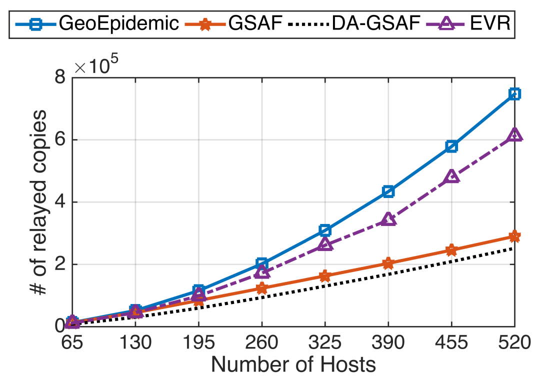
(c) Average Latency (WiFi)



(d) Average Latency (Bluetooth)

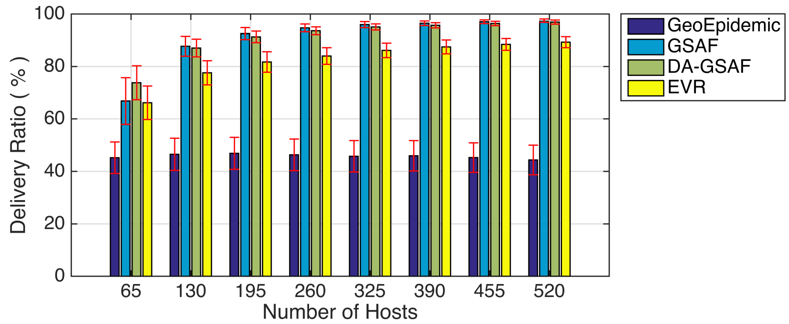


(e) Network Traffic (WiFi)

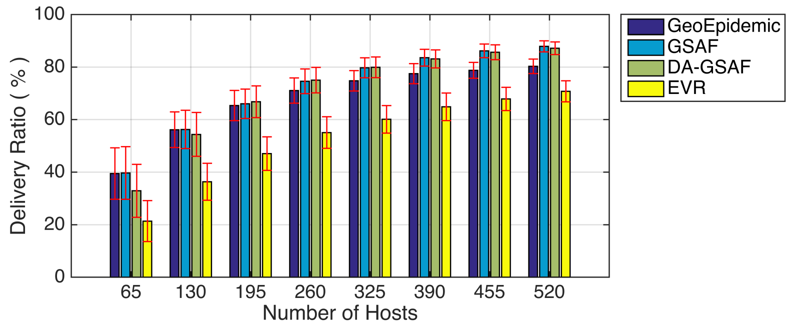


(f) Network Traffic (Bluetooth)

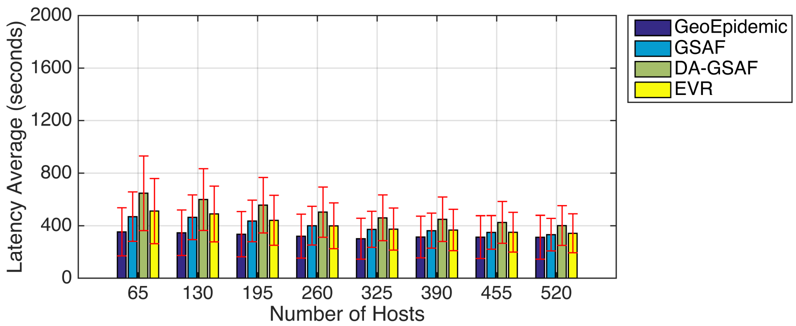
Figure 5.11: Influence of user density on the performance of geocasting protocols (Helsinki city centre)



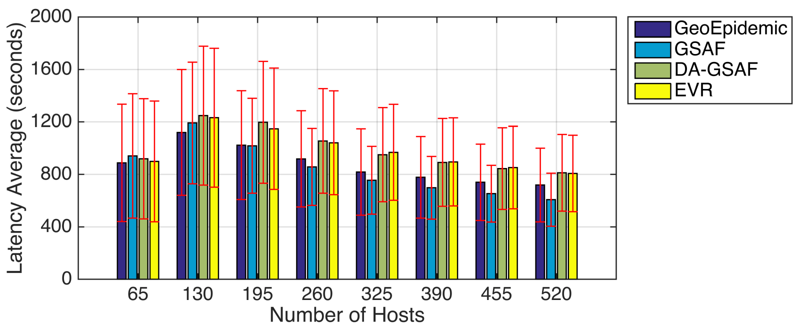
(a) Delivery Performance (WiFi)



(b) Delivery Performance (Bluetooth)

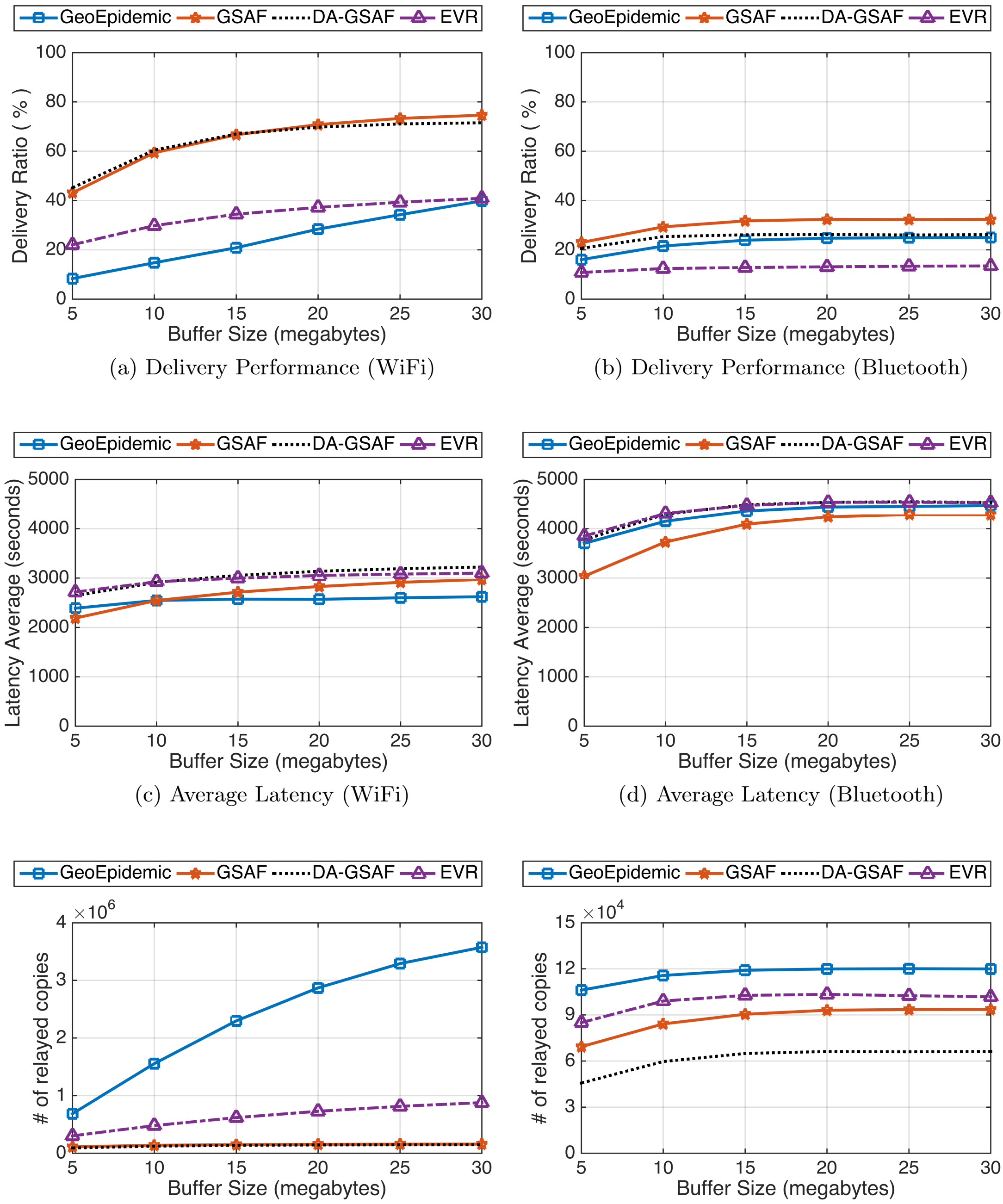


(c) Average Latency (WiFi)



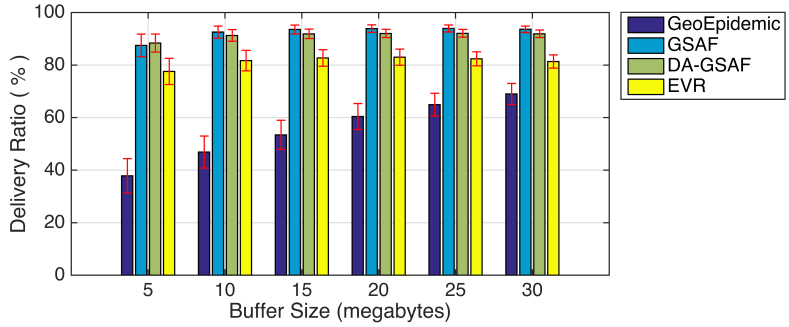
(d) Average Latency (Bluetooth)

Figure 5.12: Influence of user density on the performance of geocasting protocols; Upper 10% most visited casts (Helsinki city centre)

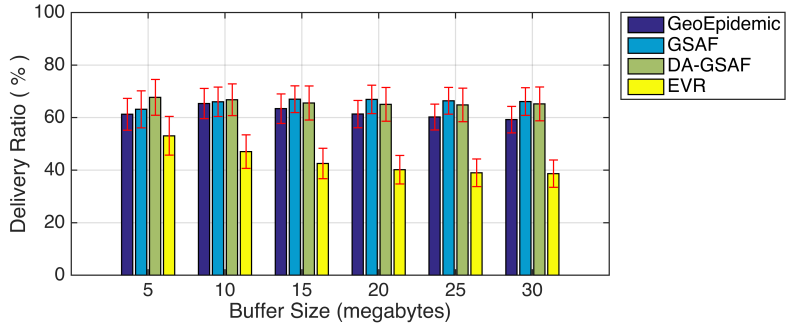


(a) Delivery Performance (WiFi)
(b) Delivery Performance (Bluetooth)
(c) Average Latency (WiFi)
(d) Average Latency (Bluetooth)
(e) Network Traffic (WiFi)
(f) Network Traffic (Bluetooth)

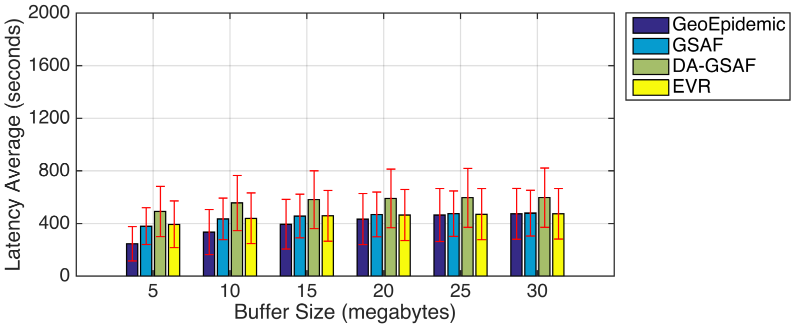
Figure 5.13: Influence of buffer capacity on the performance of geocasting protocols (Helsinki city centre)



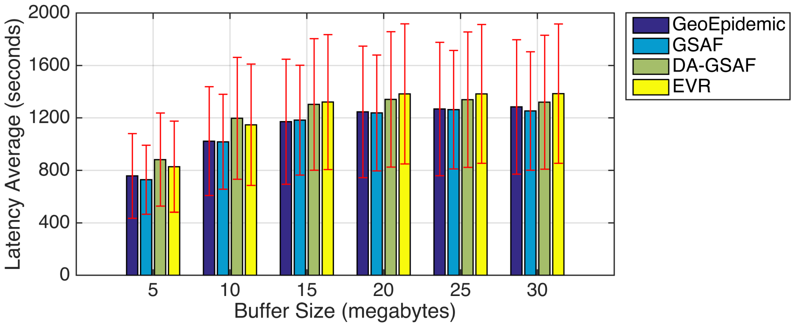
(a) Delivery Performance (WiFi)



(b) Delivery Performance (Bluetooth)

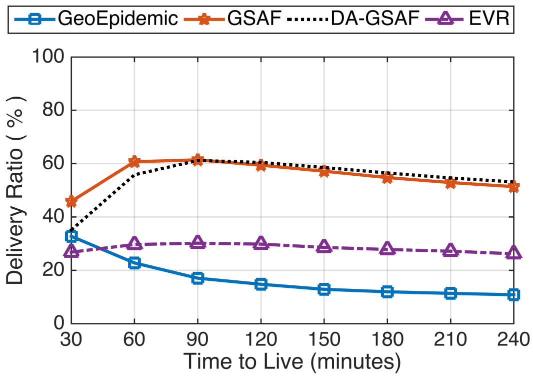


(c) Average Latency (WiFi)

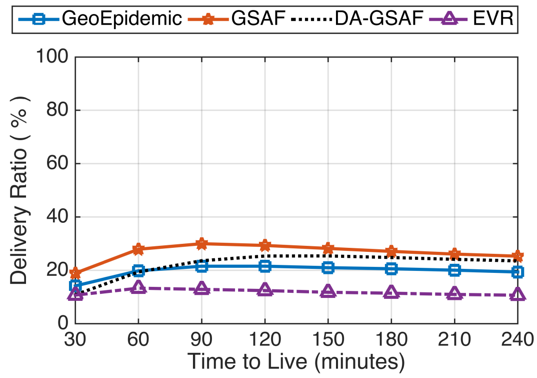


(d) Average Latency (Bluetooth)

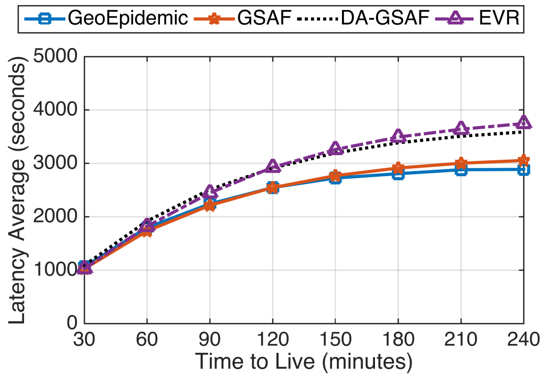
Figure 5.14: Influence of buffer capacity on the performance of geocasting protocols; Upper 10% most visited casts (Helsinki city centre)



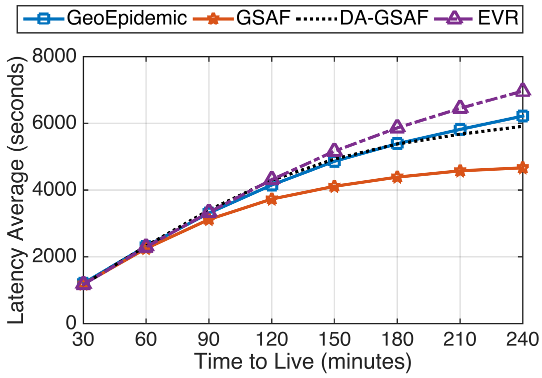
(a) Delivery Performance (WiFi)



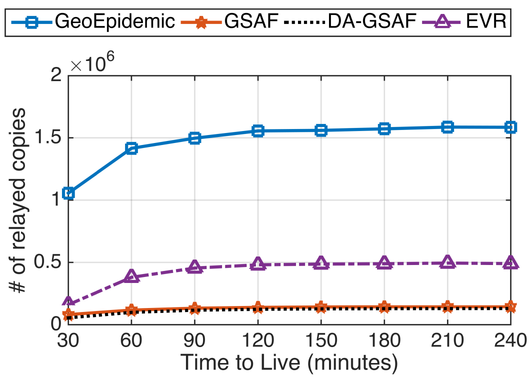
(b) Delivery Performance (Bluetooth)



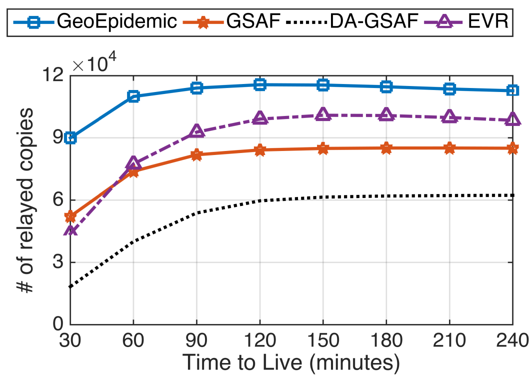
(c) Average Latency (WiFi)



(d) Average Latency (Bluetooth)

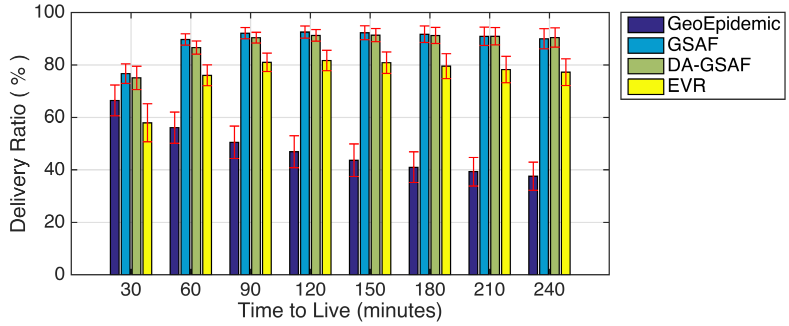


(e) Network Traffic (WiFi)

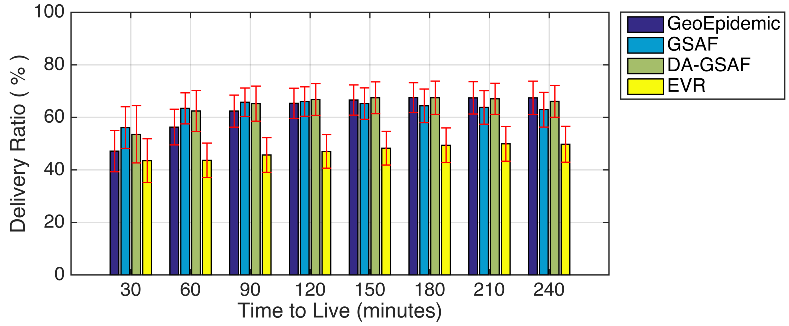


(f) Network Traffic (Bluetooth)

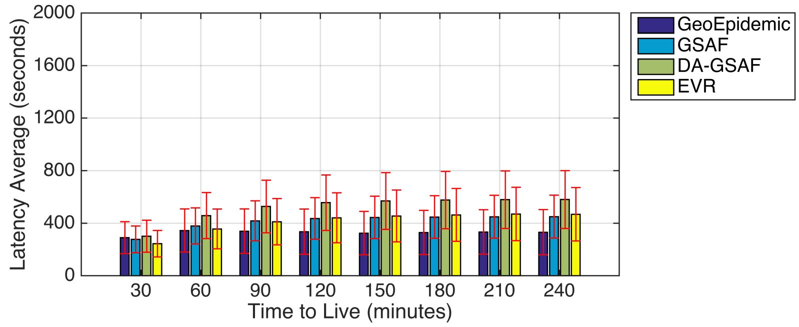
Figure 5.15: Influence of message lifetime on the performance of geocasting protocols (Helsinki city centre)



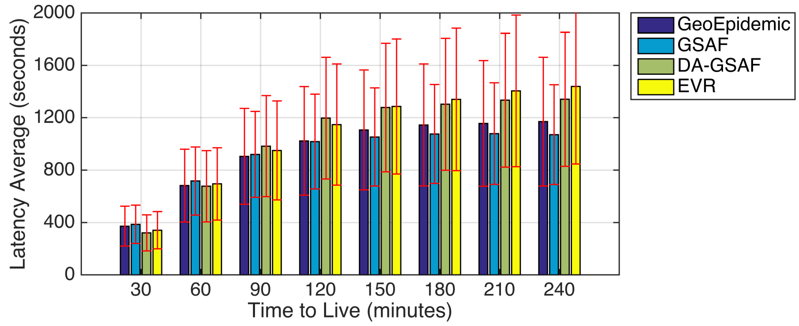
(a) Delivery Performance (WiFi)



(b) Delivery Performance (Bluetooth)



(c) Average Latency (WiFi)



(d) Average Latency (Bluetooth)

Figure 5.16: Influence of message lifetime on the performance of geocasting protocols; Upper 10% most visited casts (Helsinki city centre)

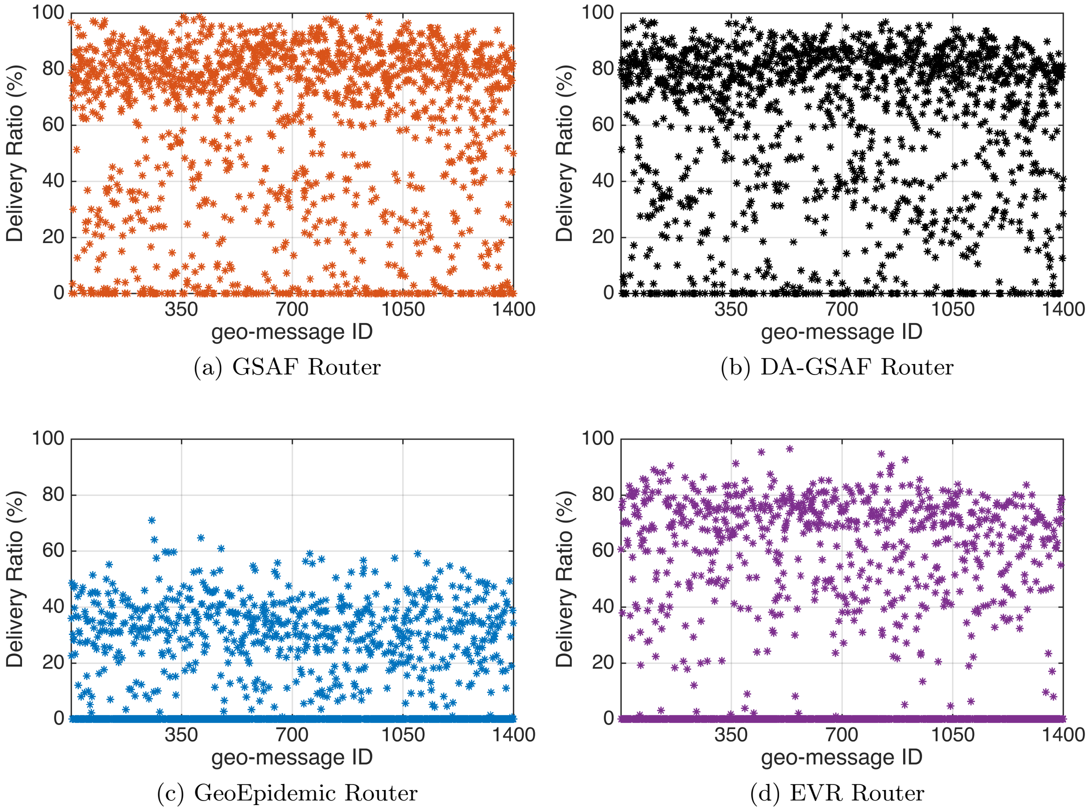
5.2.4 Influence of message lifetime

In the next set of simulations we examine how the message lifetime affects the performance of the geocasting protocols in the city of Helsinki (the *buffer size* and *number of users* are kept constant). Figure 5.15 and 5.16 illustrate the results which confirm the superiority of GSAF and DA-GSAF.

It is worth pointing out that the network overhead (the total number of relayed messages) for DA-GSAF is significantly lower compared to GSAF. This has been the case for all previous simulations, but it becomes apparent here because of the scale of the figure (which is in 10^4 messages in contrast to 10^5 or 10^6 messages in other figures). This is because DA-GSAF is more selective and, although both GSAF and DA-GSAF start with the same number of message tickets (T), DA-GSAF does not always spend all tickets before a message reaches its destination cast.

5.2.5 Per-message delivery

Finally, we present scatter plots that illustrate the per-message delivery ratio for the studied geocasting protocols. As shown in Figure 5.17, GSAF performs the best (note the high density of points in the area of delivery ratio of 85%). GeoEpidemic performs the worst



(a) GSAF Router (b) DA-GSAF Router

(c) GeoEpidemic Router (d) EVR Router

Figure 5.17: Per-Message Delivery Ratio (Helsinki city centre)

as more than half of the messages are never delivered. DA-GSAF, although performing slightly worse than GSAF, results in less undelivered messages. EVR's performance sits somewhere between DA-GSAF and GeoEpidemic.

5.3 Helsinki City Working Day

In all simulations presented so far (University of Sussex campus in § 5.1 and Helsinki city centre in § 5.2), we have used ONE's default mobility model where a destination is randomly generated and routes are the shortest paths between the current location and the generated destination. In this section we study how GSAF and DA-GSAF behave in simulated scenarios where nodes move realistically, based on the "working day" mobility pattern [80]. People go to work in the morning and sit in their office during standard working hours. Some may go to popular points of interests, such as shopping centres or parks. In addition, simulated users may be pedestrians or use a car. Undoubtedly, this a much more complex and realistic scenario where a protocol like EVR, that takes into account location- and destination-specific historical information, could potentially shine.

5.3.1 Simulation Scenario

In this scenario, we evaluated the performance of geocasting protocols using four types of users (employees, pedestrians, cars and buses) that co-exist in the simulated world. The number of users for each different type is shown in Table 5.4 . Most of the users follow the working day mobility model while a smaller portion move around the city centre. The details of the working day model settings are shown in Table 5.5 . We assume that the working day is 4 hours long and we can therefore simulate three full day cycles during a single run which "lasts" for 12 simulated hours. The working day mobility model requires information about the coordinates of residential and commercial areas in the town. Figure 5.18 shows where homes, offices and points of interests are located in Helsinki's city centre. All other simulation parameters are as described in § 5.2.1 . Below we use the same structure to present the results of the evaluation (user density, buffer capacity, message lifetime and per-message delivery).

5.3.2 Influence of user density

As shown in Figure 5.19, the overall performance of all routing protocols is worse compared to the simulation scenario where the ONE's default mobility model was used (Figure 5.11). Looking at Figures 5.19 (e and f) and 5.11 (e and f), one can see that the total number for relayed messages for both the WiFi and Bluetooth scenarios is reduced by a factor of 2 . This is because with many simulated users working in their offices, there are fewer opportunities to relay messages towards their destination cast. Employees sit in their offices during the working hours and do not contribute to the routing procedure. This significantly reduces (around 60%; see Table 5.4) the number of active nodes during

Table 5.4: Detailed number of users (working day scenario)

Total #	65	130	195	260	325	390	455	520
Employee	40	80	120	160	200	240	280	320
Pedestrian	10	20	30	40	50	60	70	80
Car	10	20	30	40	50	60	70	80
Bus	5	10	15	20	25	30	35	40

Table 5.5: Employees' mobility model on working day scenario

Working Day Mobility Model for Employees	
Mobility Model	Working Day Movement
Speed	0.8 - 1.4 (m/s)
Probability to own a Car	50%
# of Offices	50
Working Day Length	4 Hours
Probability to go shopping after Work	50%
Minimum Shopping time	1 Hour
Maximum Shopping time	2 Hours
# of Meeting Spots	10
Minimum Group size	1
Maximum Group size	3

the working hours. Moreover, in this scenario there are casts that are very rarely visited (given the distribution of the offices and shopping centres in the map) and therefore GSAF and DA-GSAF perform slightly worse with respect to delivering messages to these casts[7]. Although EVR takes a smaller hit in performance compared to GSAF and DA-GSAF, our protocols still perform better.

The results for the most visited casts are shown in Figure 5.20 . In casts that are visited frequently, the performance of all protocols, including GSAF and DA-GSAF, is similar to their performance when the default mobility model was used.

5.3.3 Influence of buffer capacity

The influence of buffer capacity on the performance of all studied protocols is illustrated in Figures 5.21 and 5.22 (for most visited casts). The results are consistent to all our previous observations. Figure 5.22.a shows DA-GSAF's and EVR's performance is inelastic to the available buffer size, whereas GSAF and especially GeoEpidemic are more sensitive

[7]This is also evident in Figures 5.25 (a and b) - see also discussion in § 5.3.5 .

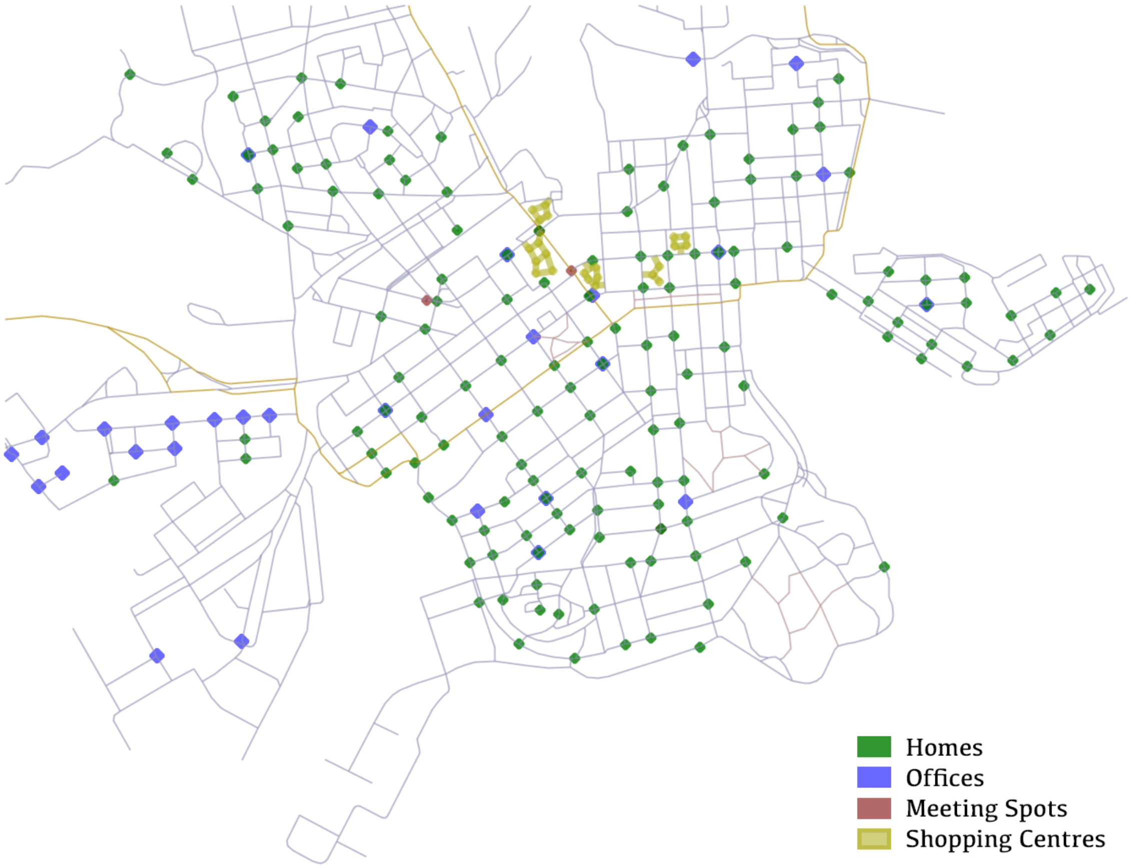


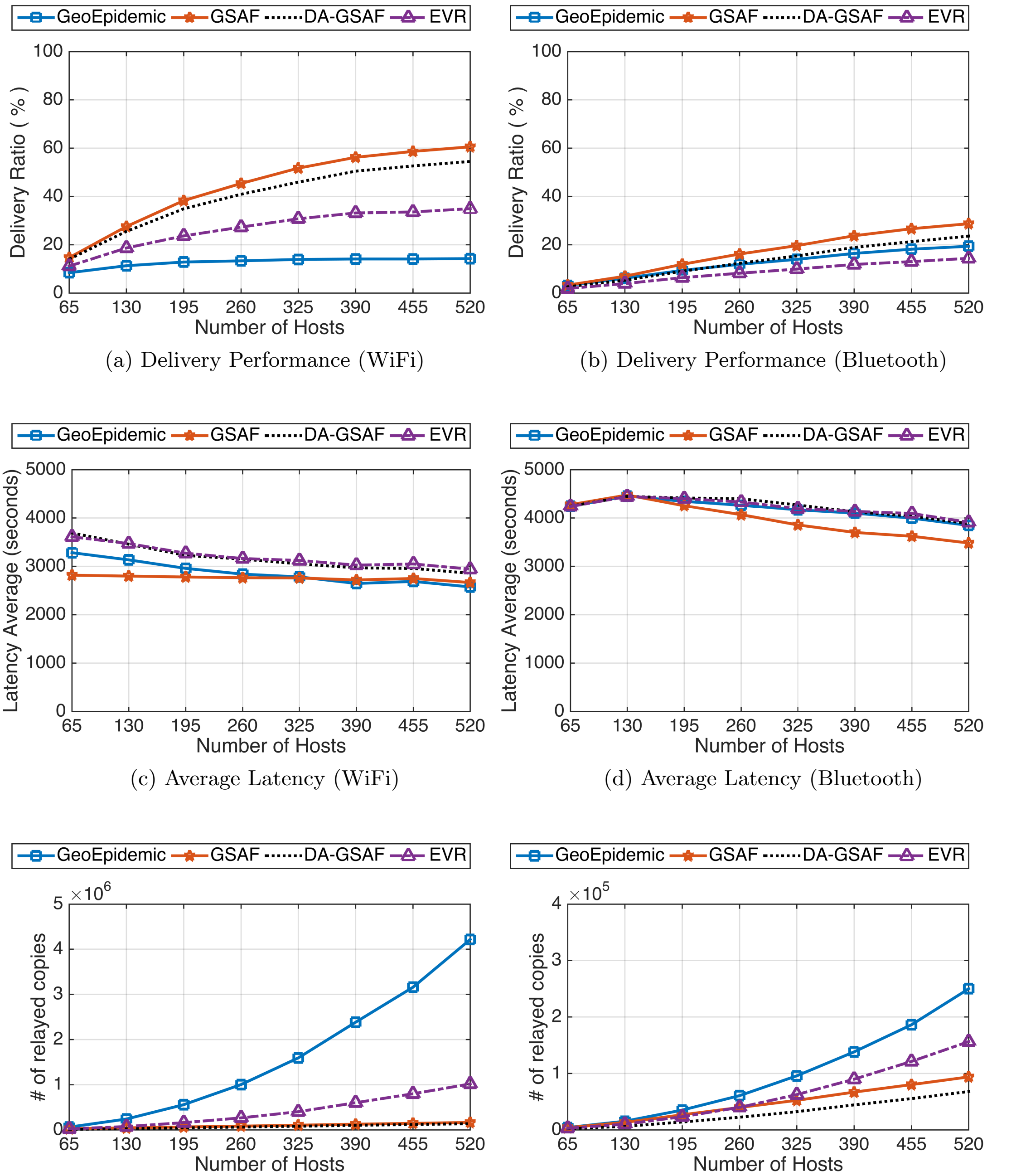
Figure 5.18: Helsinki city centre - working day map

to changes in the buffer size. DA-GSAF reaches a plateau when 15 MBs of memory are available at each device for storing and carrying messages. The reason is its selection mechanism for relaying messages, is such, the destination cast can be reached with minimum network overhead; this, in turn, means that this can be achieved with small buffers. Looking at the average latency for all and the most visited casts, we observe that the difference is significant (in some cases around five times less).

5.3.4 Influence of message lifetime

Figures 5.23 and 5.24 show the impact of message life time when the working day mobility model is simulated. In comparison to the result in § 5.2 (Figure 5.15 and 5.16) the performance pattern for all protocols is very similar. Same as in § 5.3.2 and § 5.3.3, protocols perform worse compared to the default mobility model because of the existence of casts that are very rarely visited. GSAF and DA-GSAF outperform both EVR and GeoEpidemic for all different values of the message lifetime.

As discussed earlier, the impact of message lifetime is not straightforward to understand. Longer lifetimes result in more messages being alive in the network. Memory and network resources are stressed. Additionally, if the lifetime of a message is large, then a potentially



(a) Delivery Performance (WiFi)

(b) Delivery Performance (Bluetooth)

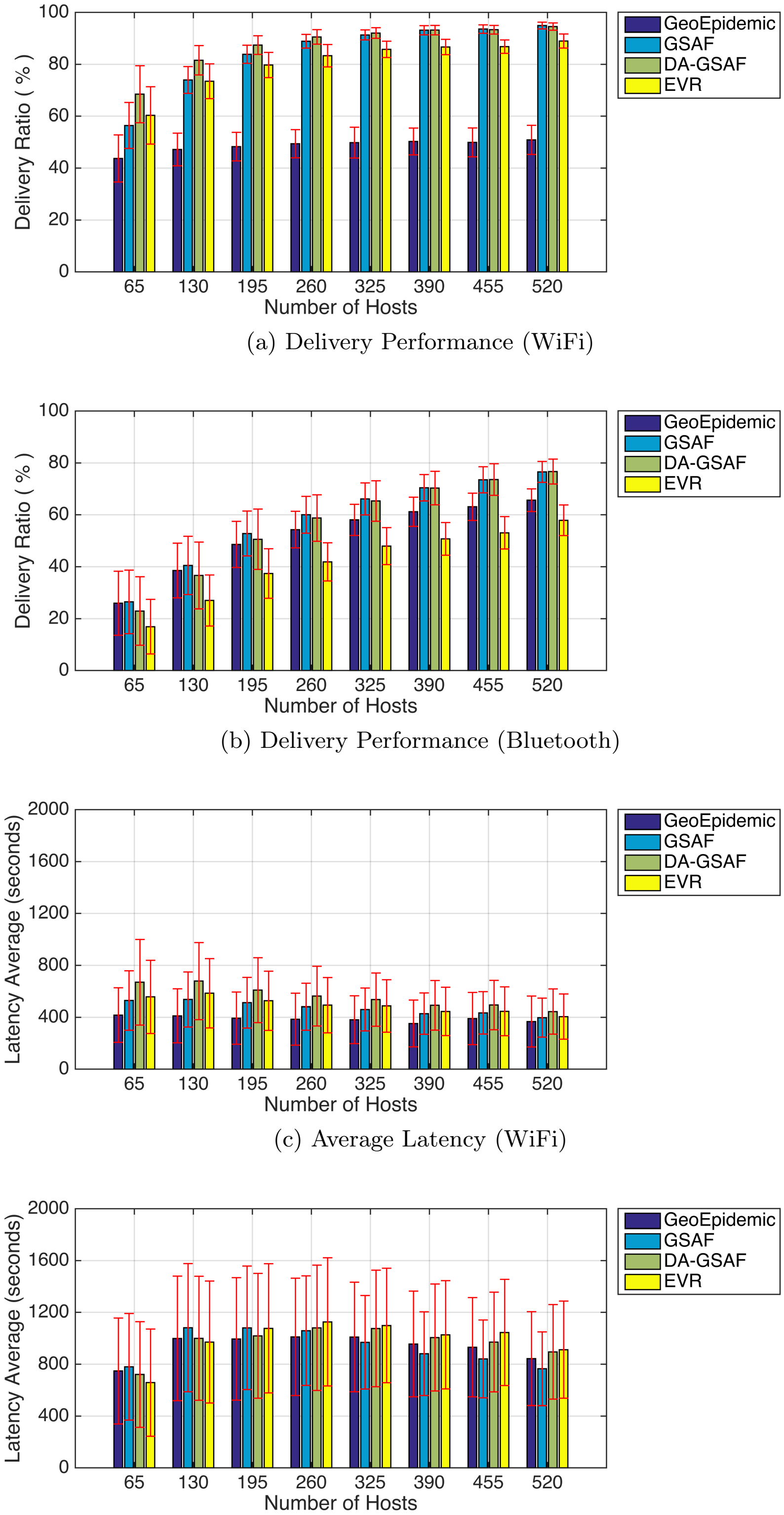
(c) Average Latency (WiFi)

(d) Average Latency (Bluetooth)

(e) Network Traffic (WiFi)

(f) Network Traffic (Bluetooth)

Figure 5.19: Influence of user density on the performance of geocasting protocols (Helsinki working day)



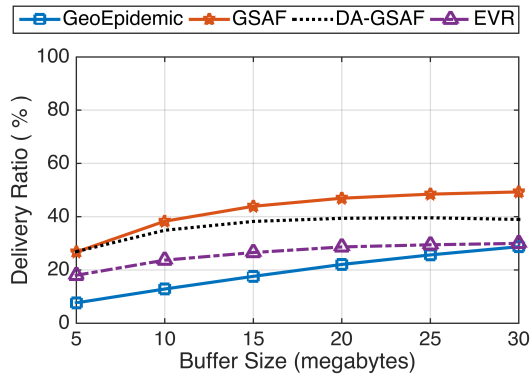
(a) Delivery Performance (WiFi)

(b) Delivery Performance (Bluetooth)

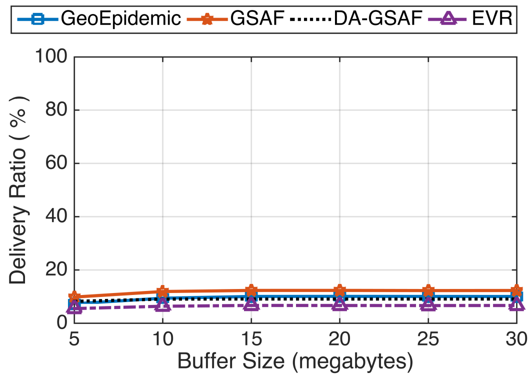
(c) Average Latency (WiFi)

(d) Average Latency (Bluetooth)

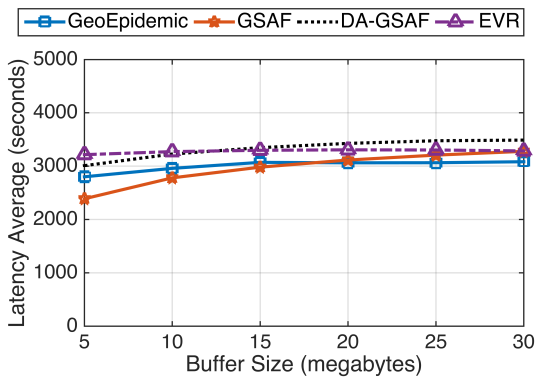
Figure 5.20: Influence of user density on the performance of geocasting protocols; Upper 10% most visited casts (Helsinki working day)



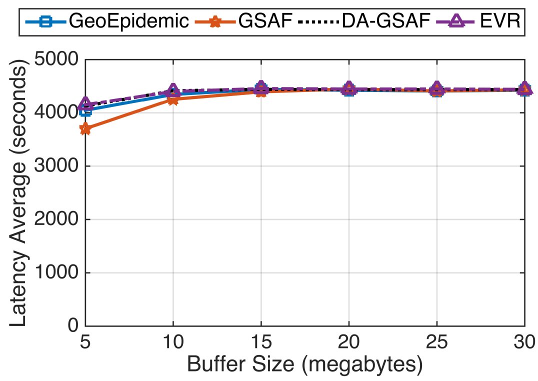
(a) Delivery Performance (WiFi)



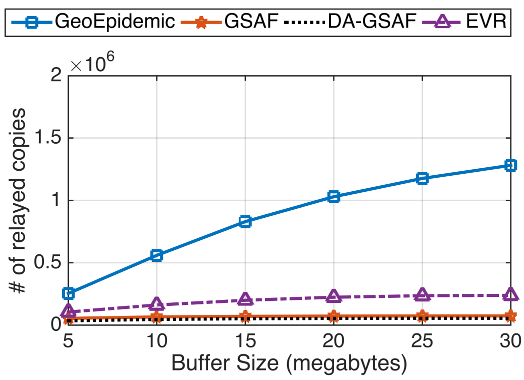
(b) Delivery Performance (Bluetooth)



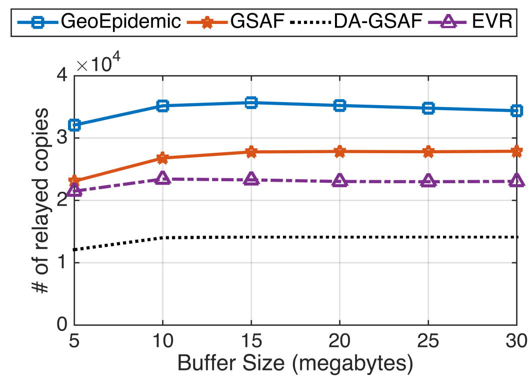
(c) Average Latency (WiFi)



(d) Average Latency (Bluetooth)

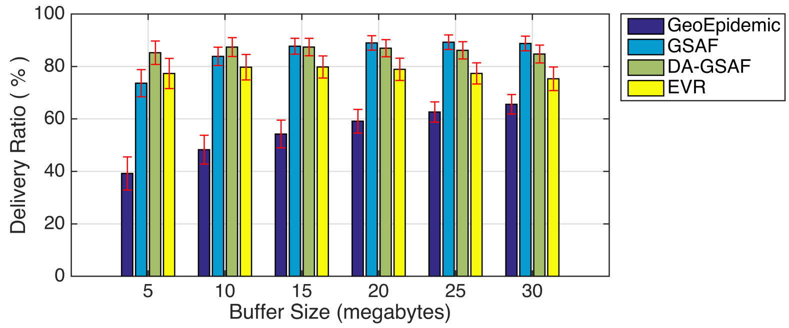


(e) Network Traffic (WiFi)

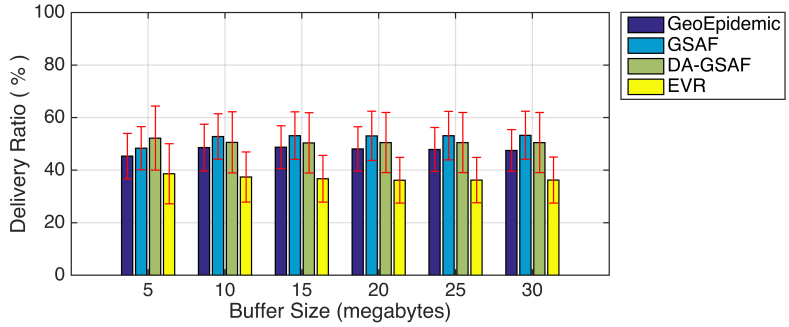


(f) Network Traffic (Bluetooth)

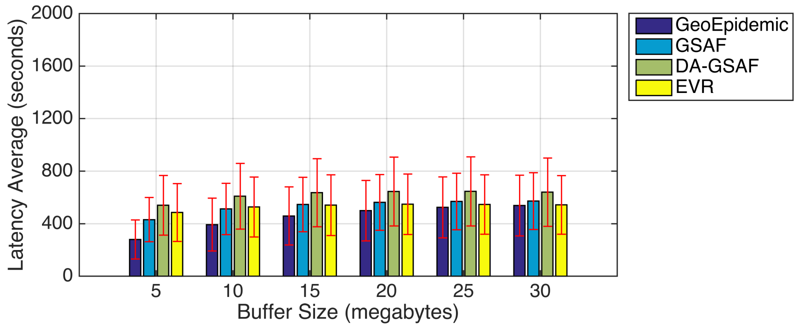
Figure 5.21: Influence of buffer capacity on the performance of geocasting protocols (Helsinki working day)



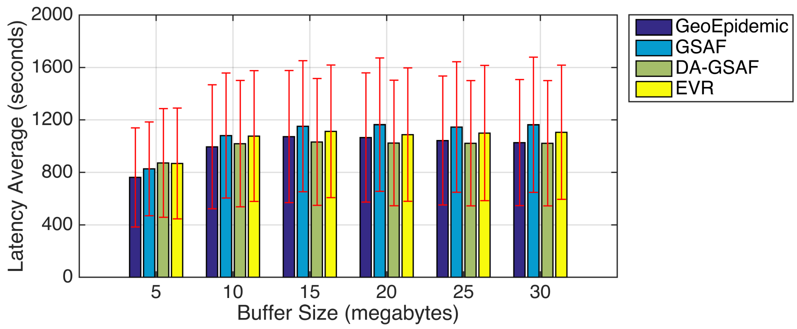
(a) Delivery Performance (WiFi)



(b) Delivery Performance (Bluetooth)

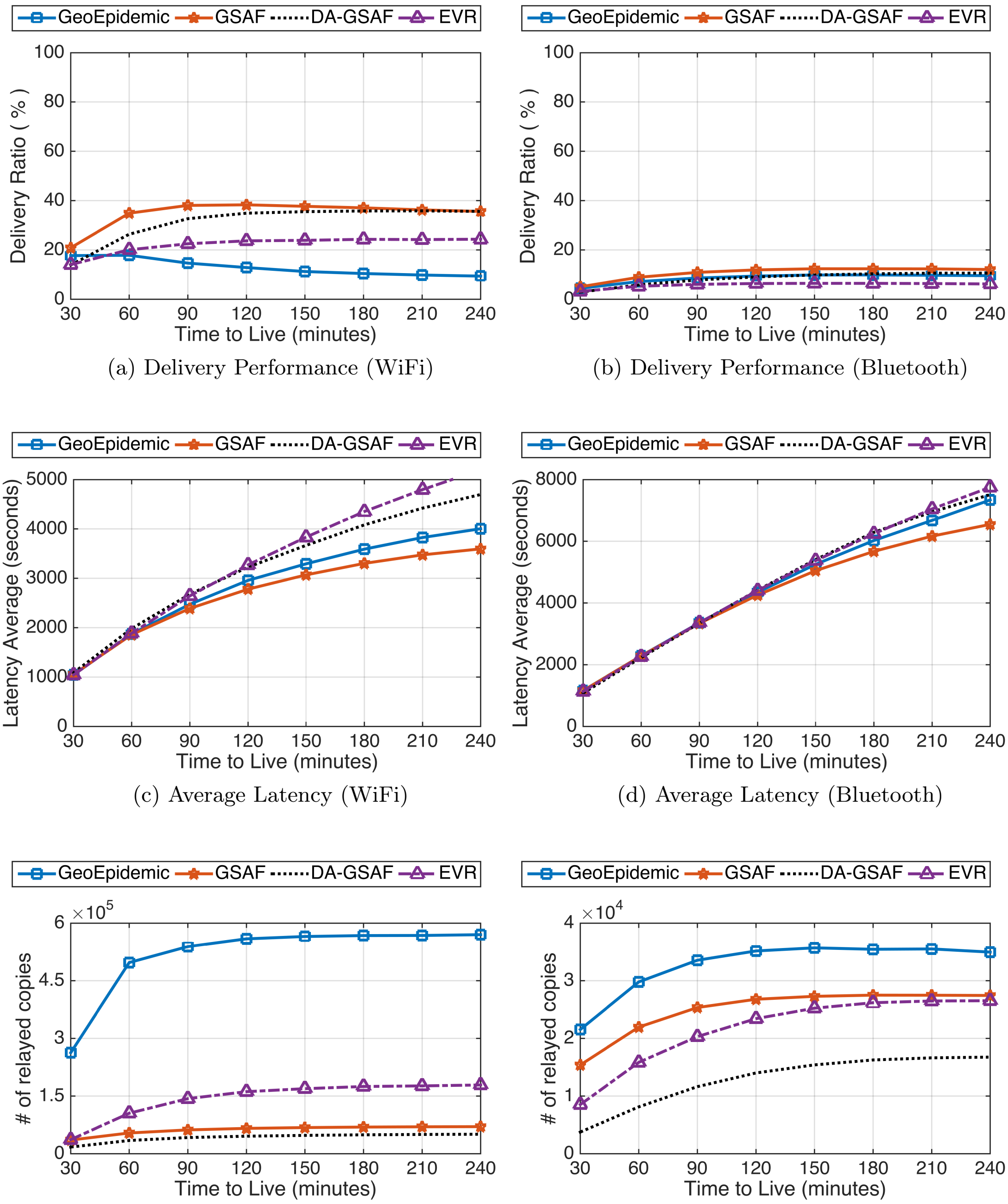


(c) Average Latency (WiFi)



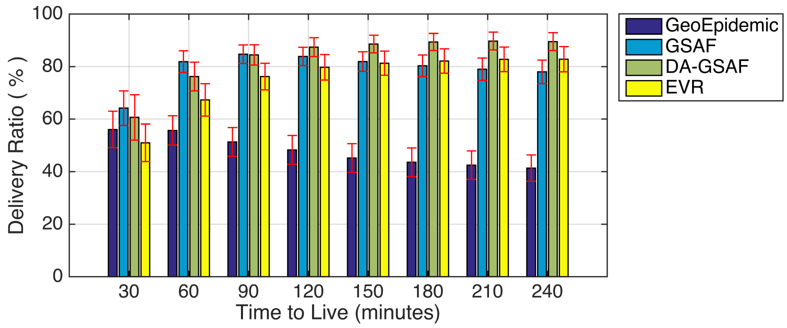
(d) Average Latency (Bluetooth)

Figure 5.22: Influence of buffer capacity on the performance of geocasting protocols; Upper 10% most visited casts (Helsinki working day)

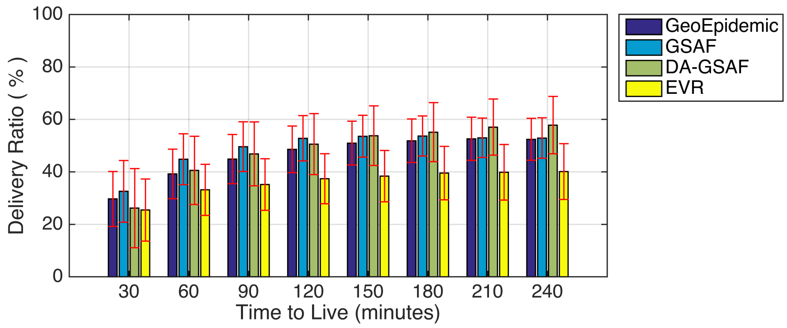


(a) Delivery Performance (WiFi)
(b) Delivery Performance (Bluetooth)
(c) Average Latency (WiFi)
(d) Average Latency (Bluetooth)
(e) Network Traffic (WiFi)
(f) Network Traffic (Bluetooth)

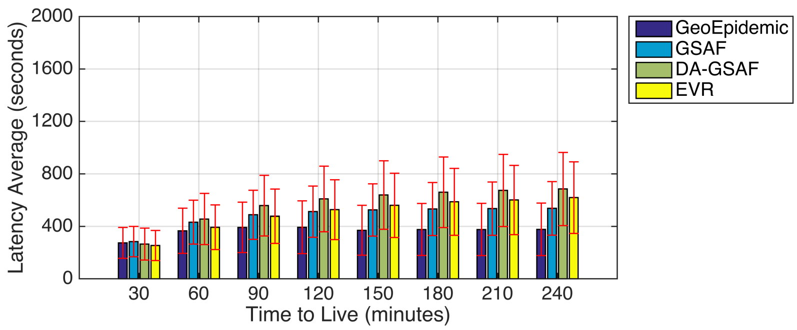
Figure 5.23: Influence of message lifetime on the performance of geocasting protocols (Helsinki working day)



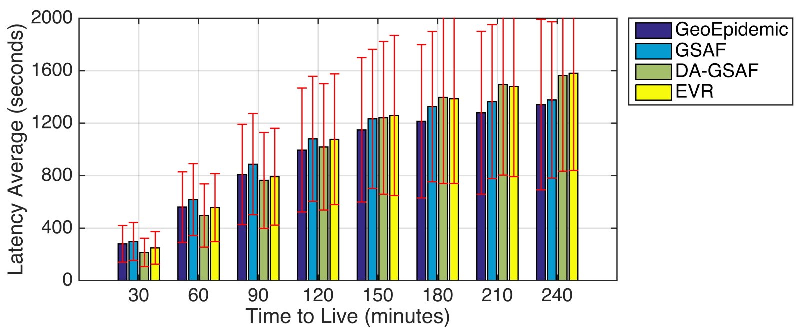
(a) Delivery Performance (WiFi)



(b) Delivery Performance (Bluetooth)

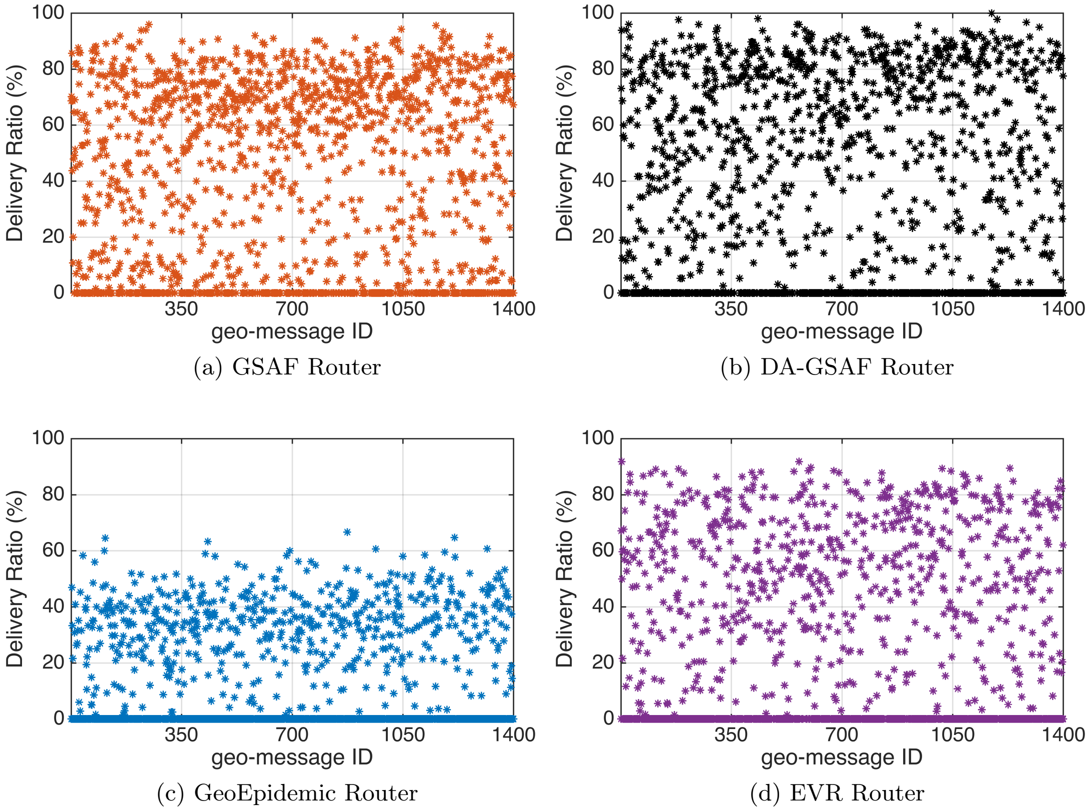


(c) Average Latency (WiFi)



(d) Average Latency (Bluetooth)

Figure 5.24: Influence of message lifetime on the performance of geocasting protocols; Upper 10% most visited casts (Helsinki working day)



(a) GSAF Router
(b) DA-GSAF Router
(c) GeoEpidemic Router
(d) EVR Router

Figure 5.25: Per-Message Delivery Ratio (Helsinki working day)

very large number of users (many of which may have stayed for a short time in the cast) that visited its destination cast comprise the set of the message recipients. Sustaining large delivery ratios in such a case is very challenging.

5.3.5 Per-message delivery

Figure 5.25 shows the per-message delivery ratio. As mentioned above, because of the employed mobility model, a number of casts are rarely visited. As a result, the number of undelivered messages (with 0% delivery ratio) increases compared to the default mobility model. More specifically, the number of undelivered messages is 350 for GSAF, 510 for DA-GSAF, 850 for GeoEpidemic and 738 for EVR (out of 1400 created messages during the simulation runtime). This confirms that GSAF is performing the best while the DA-GSAF is the second best.

5.4 The Impact of Buffer Scheduling Policy

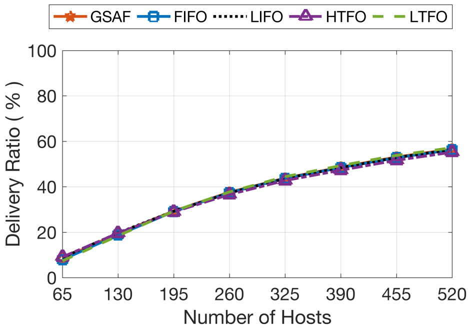
In § 3.2.4, we discussed about buffer scheduling policies and how they may affect the overall performance of geocasting in opportunistic networks. Conceptually a buffer works like a queue and the first message in front of the queue is the first one that gets transferred upon an encounter with another device. A buffer scheduling policy defines the process by which messages are ordered in the queue. In all previous simulations, we examined the performance of geocasting protocols using the default scheduling policy which randomly selects messages to be sent to encountered nodes. In this section we investigate the impact of buffer scheduling policies on the performance of GSAF. We experiment with the buffer scheduling policies described in § 3.2.4 (i.e. GSAF+FIFO, GSAF+LIFO, GSAF+HTFO and GSAF+LTFO) against the default random policy. The simulation scenario is identical to the one used in § 5.2 .

Note that, we do not present the results for different buffer scheduling policies when devices communicate through WiFi. This is because the buffer scheduling policy has no effect when a device can transfer *all* its stored messages to the encountered device; and this is the case when WiFi is used to exchange buffers up to 30MB (as in our simulations), for all types of users. However, when communication takes place through Bluetooth, the adopted buffer scheduling policy plays a role (as only a percentage of buffered messages can be exchanged) in the performance of the geocasting protocols.

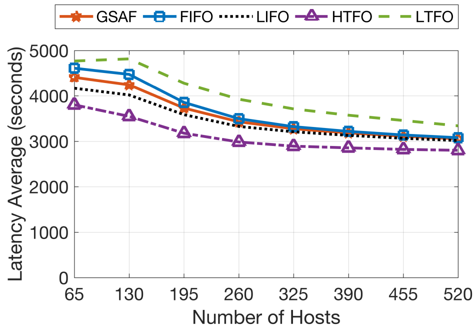
5.4.1 Influence of user density

In Figure 5.26.c (as in all relevant figures in this section) we observe that buffer scheduling policies do not influence the number of relayed messages in the network, as they only define which messages among the ones present in the buffer are transferred to encountered nodes. With respect to the delivery latency, both FIFO and LIFO policies essentially behave like the random one, given that the sequence of message exchanges is defined by how devices encounter each other. This adds randomness which results in a very similar performance of both FIFO and LIFO compared to the default random policy. On the other hand, the performance of HTFO is significantly better compared to the others, especially to the LTFO. With LTFO, the network prioritises messages that are closer to expiration (which means that their delivery latency is already high). As a result, newer messages will have to wait for the older ones, which subsequently delays their delivery too. Overall, the average delivery latency is increased for all messages. With HTFO, messages that are closer to expiration are left to expire by favouring the delivery of newer messages, therefore the observed low delivery latency. Note that the delivery ratio is not affected by the policy as the number of relayed messages (the lifetime of which is evened out by the policy) is the same.

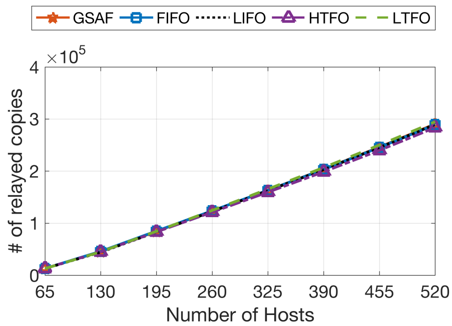
The results for most visited casts (Figure 5.27) confirm our findings above; delivery ratio is not sensitive to the policy and HTFO performs the best with respect to the delivery latency.



(a) Delivery Performance (Bluetooth)

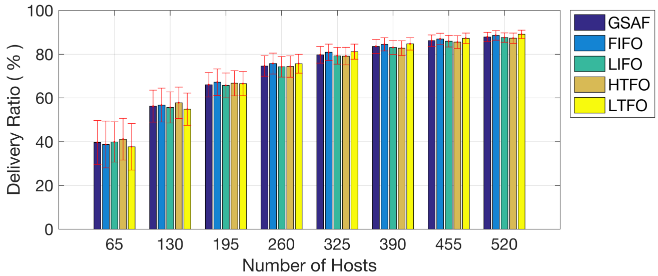


(b) Average Latency (Bluetooth)

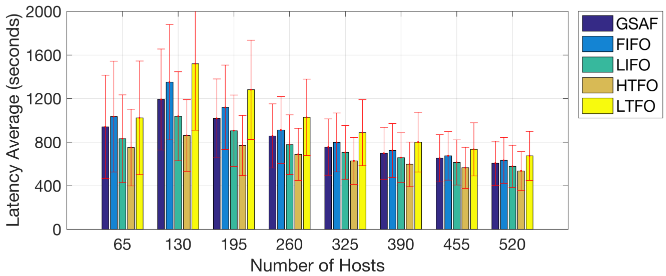


(c) Network Traffic (Bluetooth)

Figure 5.26: Influence of user density (Impact of buffer scheduling policies)

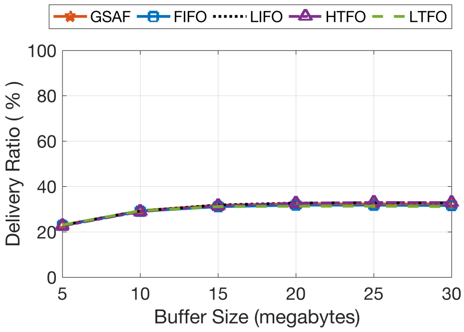


(a) Delivery Performance (Bluetooth)

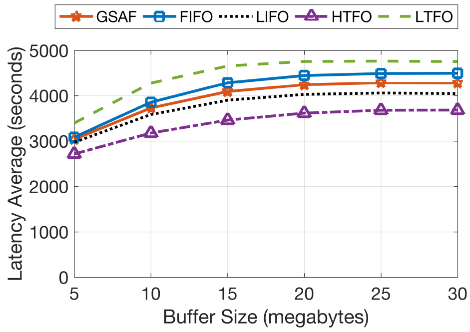


(b) Average Latency (Bluetooth)

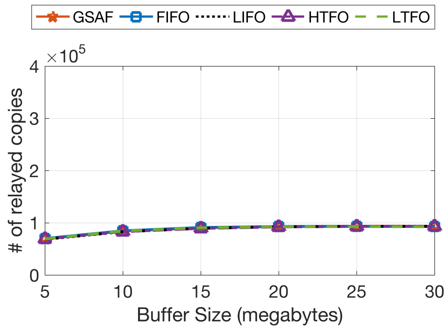
Figure 5.27: Influence of user density; Upper 10% most visited casts (Impact of buffer scheduling policies)



(a) Delivery Performance (Bluetooth)

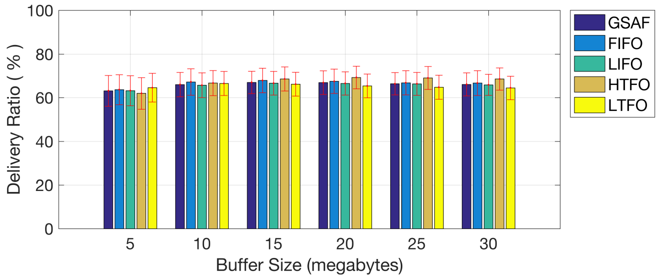


(b) Average Latency (Bluetooth)

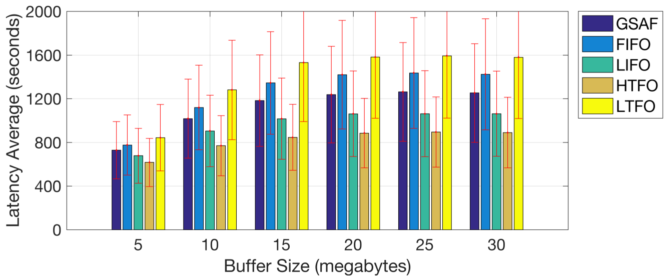


(c) Network Traffic (Bluetooth)

Figure 5.28: Influence of buffer capacity (Impact of buffer scheduling policies)

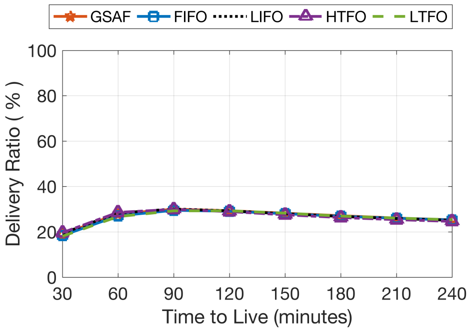


(a) Delivery Performance (Bluetooth)

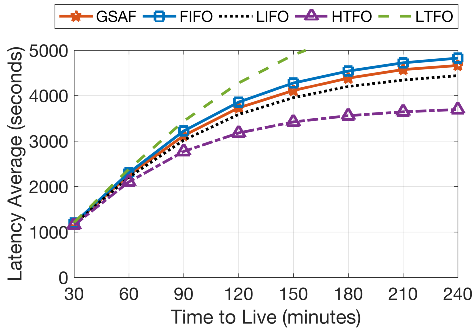


(b) Average Latency (Bluetooth)

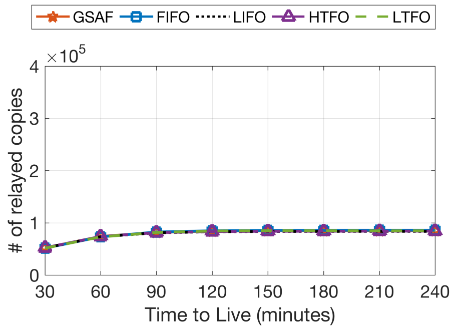
Figure 5.29: Influence of buffer capacity; Upper 10% most visited casts (Impact of buffer scheduling policies)



(a) Delivery Performance (Bluetooth)

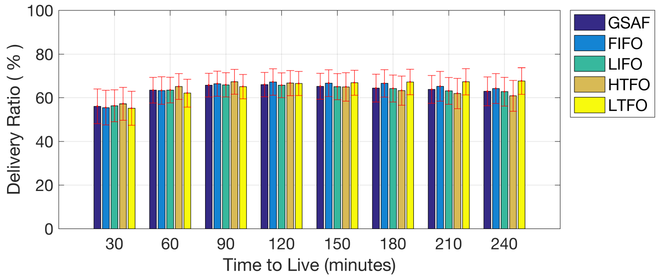


(b) Average Latency (Bluetooth)

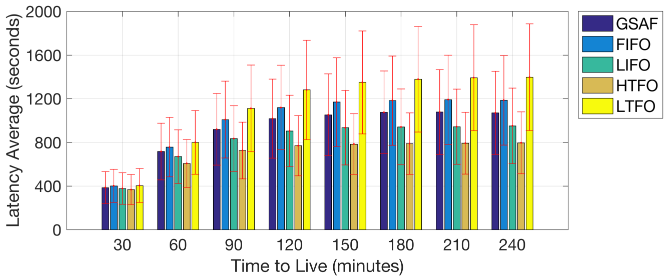


(c) Network Traffic (Bluetooth)

Figure 5.30: Influence of message lifetime (Impact of buffer scheduling policies)



(a) Delivery Performance (Bluetooth)



(b) Average Latency (Bluetooth)

Figure 5.31: Influence of message lifetime; Upper 10% most visited casts (Impact of buffer scheduling policies)

5.4.2 Influence of buffer capacity

In this set of simulations, we observe the influence of buffer capacity on the performance of the buffer scheduling policies. The results (Figures 5.28 and 5.29) for the delivery latency follow a similar pattern compared to the previous set of simulations. Note that the latency increases along with the buffer size (for all buffer scheduling policies); due to the fact more buffer result in less extinct messages and less equal delivery chance for each of them. HTFO is the best in terms of delivery latency while the rest of parameters remain relatively unchanged and equal to all of them.

5.4.3 Influence of message lifetime

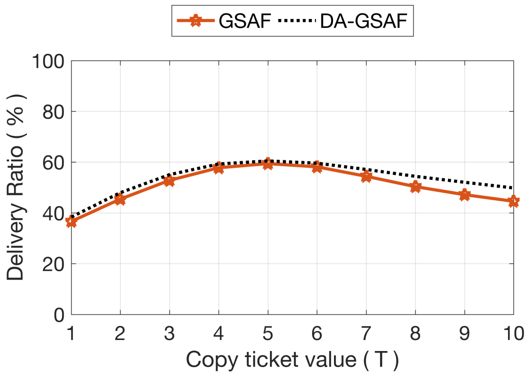
The third set of simulations investigate the impact of message lifetime in the performance of GSAF. Figures 5.30 and 5.31 follow a similar pattern to the ones observed above. The HTFO buffer scheduling policy performs the best for the same reasons as the ones described earlier. It shows the importance of buffer scheduling policies and how they could improve the average delivery latency. Based on our observation, there is no negative impact by the use of HTFO policy and it totally benefits the overall performance of our protocols.

5.5 The Impact of Copy Ticket Value

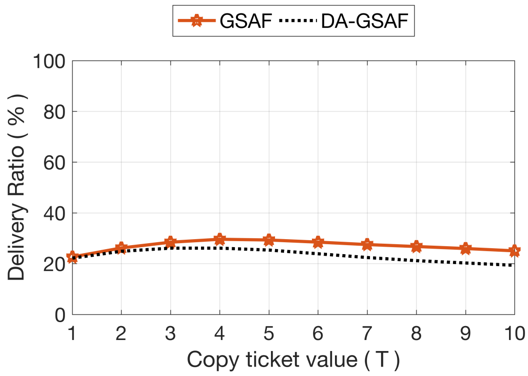
The first phase of routing in GSAF and DA-GSAF heavily relies on the number of tickets (T) a message is assigned with upon its creation. As mentioned in Page 49, the value of T could be dynamically adjusted (e.g. based on the inferred device density) to a value that provides the best performance. In this section we investigate how different values of T influence the performance of GSAF and DA-GSAF. We are interested in looking at how sensitive our approach is to T, which, in turn, means what the penalty of misconfiguring T in a dynamic approach would be.

To find the impact of copy ticket value, we design a scenario based on the default scenario in § 5.2 . We keep all parameters unchanged and examine the performance of GSAF and DA-GSAF for various copy ticket values (1 to 10). The results shown in Figure 5.32 indicate that the delivery ratio is not significantly sensitive to T (especially around the optimal value); therefore, in an approach where T is dynamically adjusted, missing the optimal value would not have a significant impact in the behaviour of the protocol (i.e. $\sim$60% for the optimal value [5] compared to $\sim$53 - 58% for values '4' and '6'). The results are similar for the message delivery latency. Finally, it is well-expected that the number of relayed messages will increase as T increases.

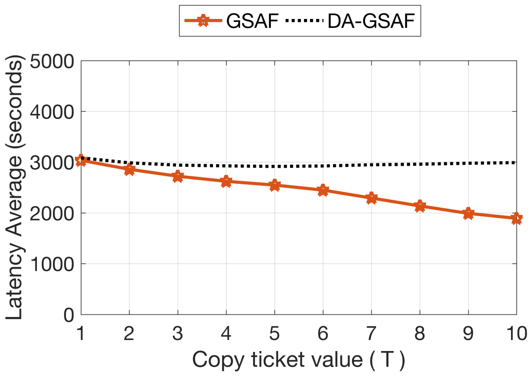
As shown in the figure, DA-GSAF is less sensitive to changes of T compared to GSAF. This is because DA-GSAF is more selective when it comes to passing messages to the



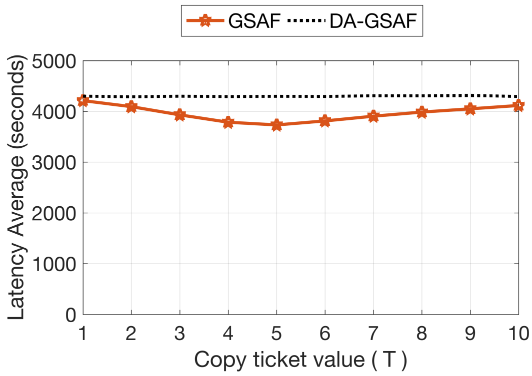
(a) Delivery Performance (WiFi)



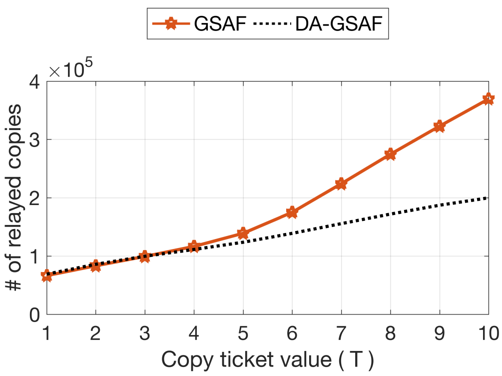
(b) Delivery Performance (Bluetooth)



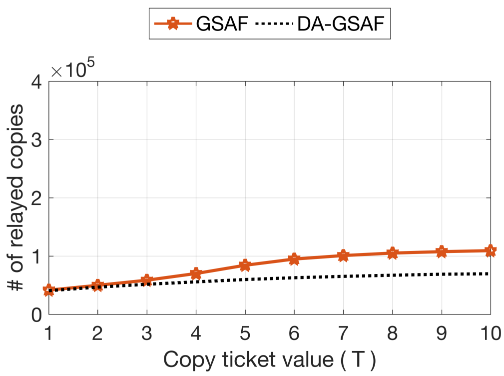
(c) Average Latency (WiFi)



(d) Average Latency (Bluetooth)



(e) Network Traffic (WiFi)



(f) Network Traffic (Bluetooth)

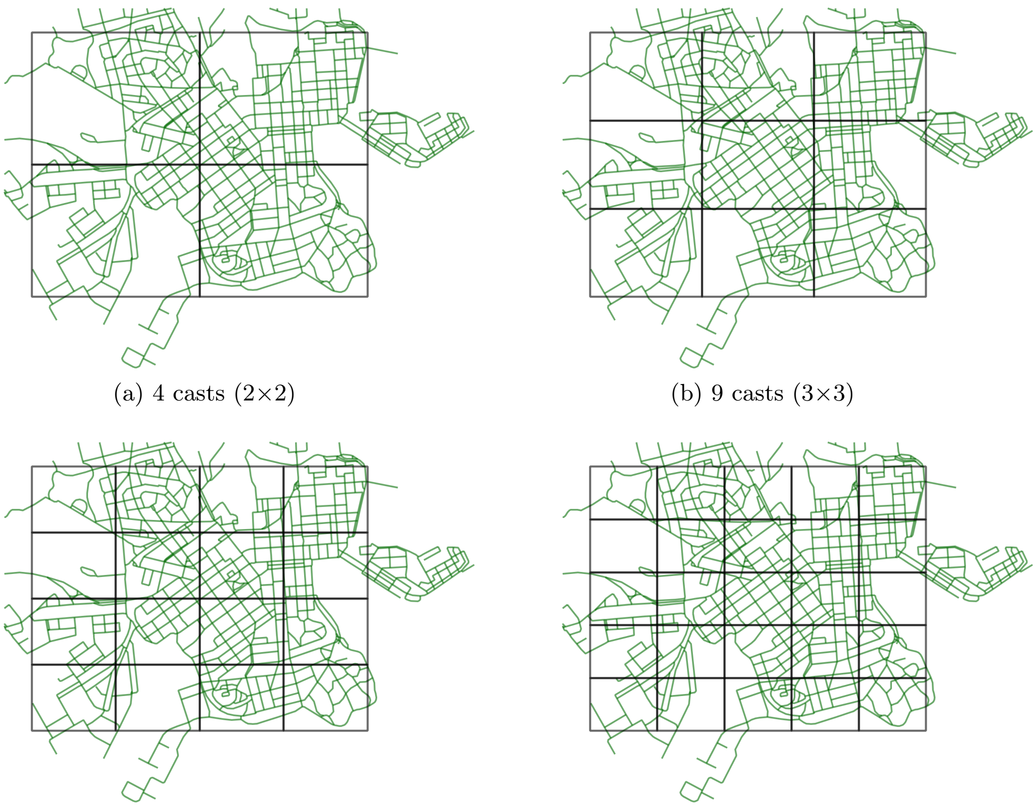
Figure 5.32: Influence of Copy Ticket Value on the performance of GSAF and DA-GSAF

encountered nodes. In many cases, with DA-GSAF not all tickets are consumed, therefore increasing T has no effect on the behaviour of the protocol.

5.6 The Impact of Cast Size

Geocasting in opportunistic networks are all about the geographical casts that are the final recipient of messages in the network. The size and shape of the cast depends on the application use case and demand of the users. According to our design, geographical casts can take any shape in the network (relatively small or large) and could be defined on the go or pre-defined based on the various application scenarios. It is useful to understand the impact of the cast size on the overall performance of protocols.

For the following simulation we divide Helsinki's city centre into a variable size of equal non-overlapping rectangular casts (4 [2×2], 9 [3×3], 16 [4×4] and 25 [5×5]) and use them as the recipients of messages in the simulated network. Figure 5.33 shows the casts on the Helsinki city centre map. The simulation scenario is the default one (see § 5.2.2) and in all presented results (Figures 5.34 and 5.35) simulated devices communicate through WiFi.



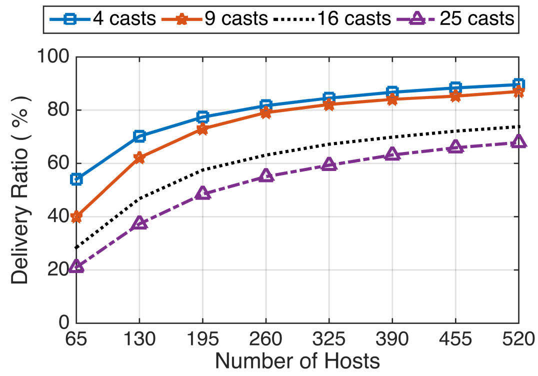
(a) 4 casts (2×2)

(b) 9 casts (3×3)

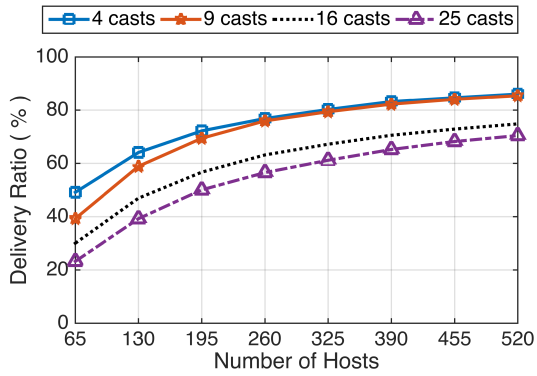
(c) 16 casts (4×4)

(d) 25 casts (5×5)

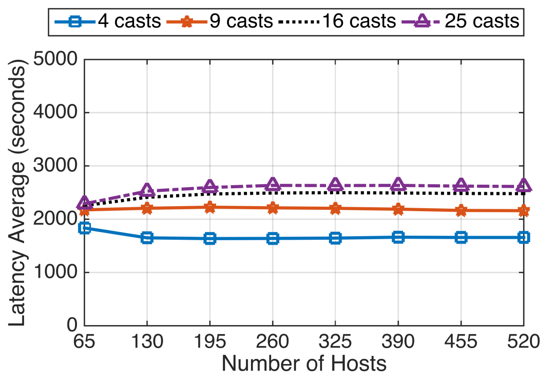
Figure 5.33: Equal non-overlapping rectangular casts in the Helsinki city centre



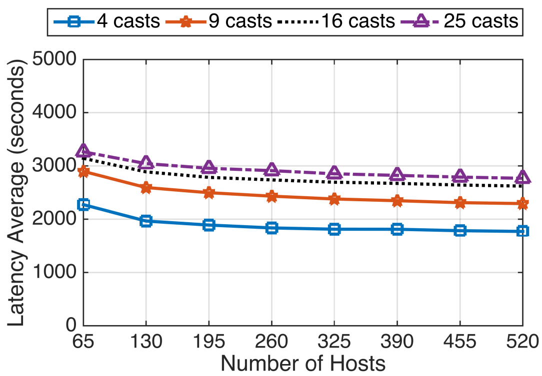
(a) Delivery Performance (GSAF)



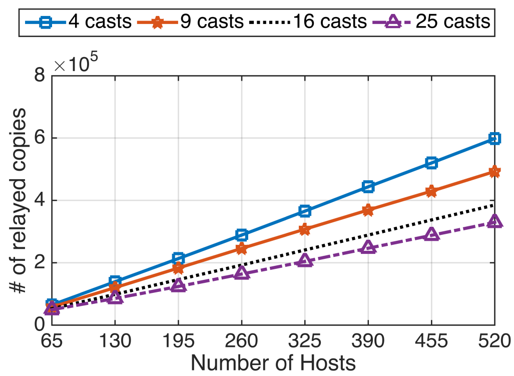
(b) Delivery Performance (DA-GSAF)



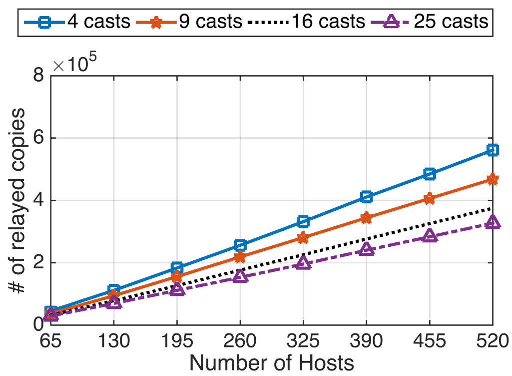
(c) Average Latency (GSAF)



(d) Average Latency (DA-GSAF)

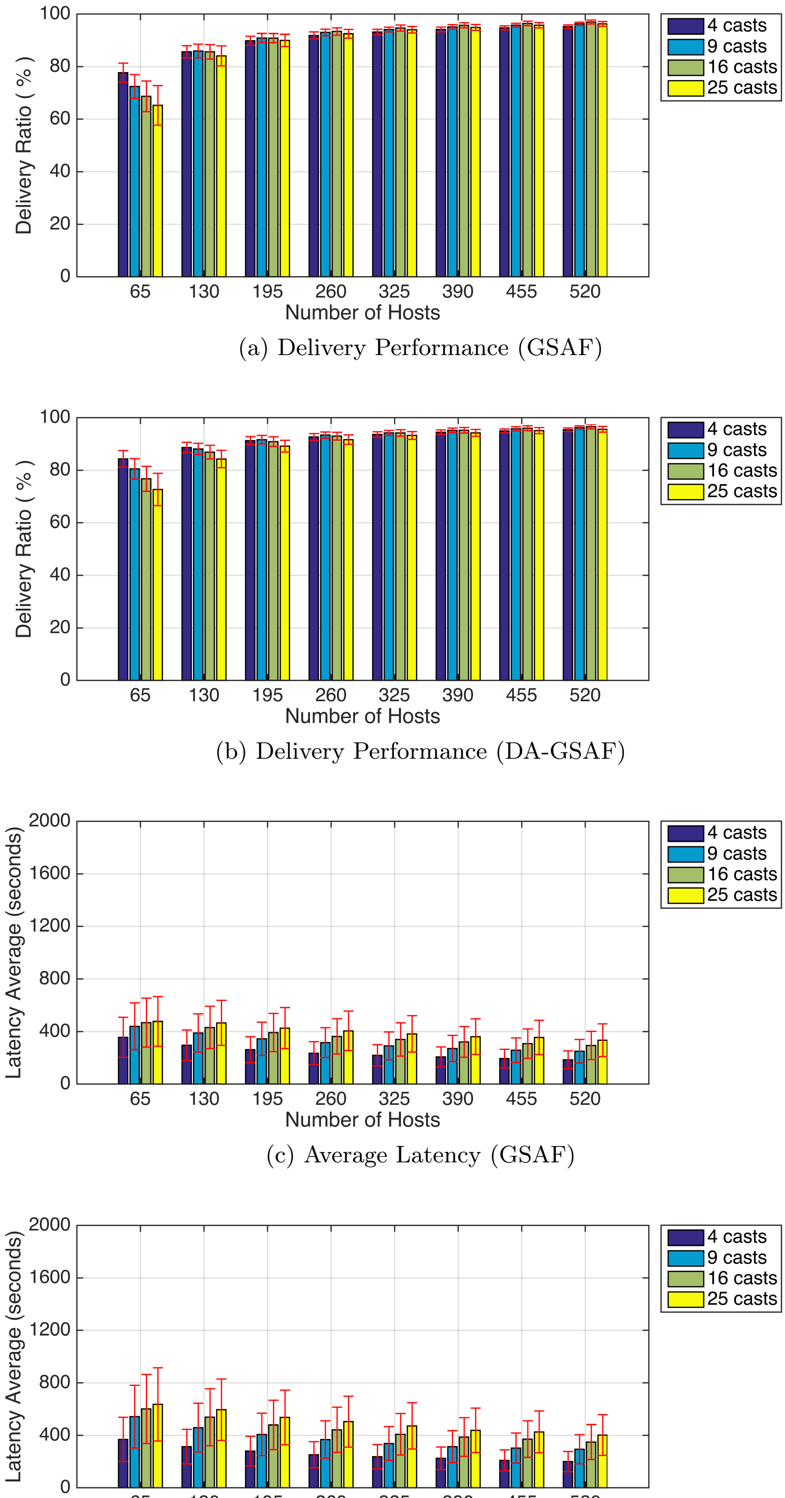


(e) Network Traffic (GSAF)



(f) Network Traffic (DA-GSAF)

Figure 5.34: Influence of user density on the performance of geocasting protocols (The impact of cast size)



(a) Delivery Performance (GSAF)

(b) Delivery Performance (DA-GSAF)

(c) Average Latency (GSAF)

(d) Average Latency (DA-GSAF)

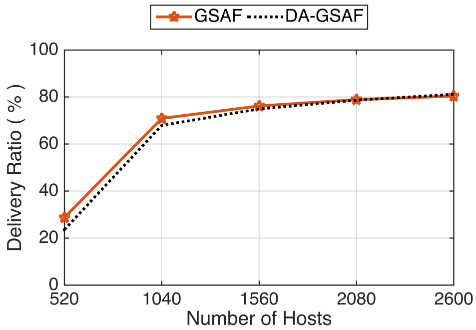
Figure 5.35: Influence of user density on the performance of geocasting protocols; Upper 10% most visited casts (The impact of cast size)

Unsurprisingly, larger (and fewer) casts have a better delivery ratio and less average latency. For large casts messages reach the destination area quickly and then flooding takes over, as indicated by the large number of the relayed copies in Figure 5.34 (e and f). Moreover, large casts mean larger covered area which provides more space for nodes to cover during their activity and it reduces the number of nodes who leave the cast without receiving a copy of the message.

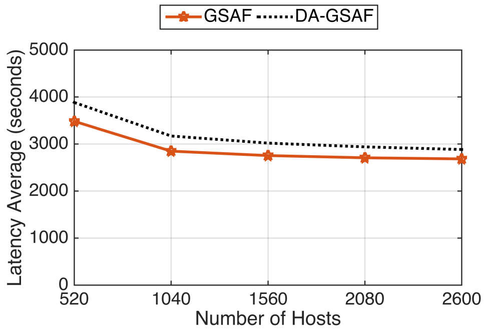
5.7 User Scalability in GSAF and DA-GSAF

In this section, we investigate the behaviour of the proposed protocols as the number of users in the network increases. Geocasting messages when a large number of users roam in the network is challenging with respect to the potentially large number of message exchanges in crowded areas due to frequent encounters between users. A geocasting protocol should be designed in a way that messages don't become extinct very quickly due to the finite buffer space, while an adequate number of copies are disseminated so that the destination cast can be reached. In our evaluation, we extend the simulation scenario in § 5.2.2 to support up to 2600 users in the network. More specifically we evaluate the performance of 520, 1040, 1560, 2080 and 2600; i.e. up to five times more users than in the simulations presented in the previous sections. In the following, we only present the results for Bluetooth connectivity. Simulating thousands of devices with WiFi connectivity was proven extremely time consuming as more messages can be transferred during each encounter compared to the Bluetooth case.

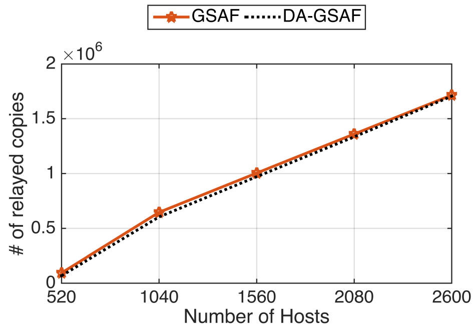
Figure 5.36 illustrates the delivery ratio, message latency and number of relayed messages for GSAF and DA-GSAF. The delivery ratio increases as more users are added in the network, reaching a plateau around ∼80%. This seems to be a close-to-optimal value given the definition of cast membership; the fact that it involves a spatial and temporal aspect. More specifically, during the time it takes for a message to reach a cast, there is a number of users that entered and left the cast (and are therefore recipients of the message) without receiving the message. It would be extremely difficult, if not impossible, to deliver the message to such users. Comparing the results for 2600 against 520 nodes in Figure 5.36.a, we observe a significant improvement in the delivery ratio as the number of users increases. There are two reasons that justify this behaviour. First, as the number of users increases there are more opportunities for both GSAF and DA-GSAF to pass messages to the encountered nodes; i.e. in previous simulations (and this one for 520 users) there were cases where not all tickets were consumed until a messaged reached the destination cast. Secondly, with more users in the network and therefore in destination casts, the dissemination of messages within the destination cast is more efficient, although more messages are relayed for that purpose (see Figure 5.36.c). Note that DA-GSAF is still slightly more selective and therefore the number of relayed messages is smaller compared to GSAF. Additionally, message sustainability within the cast increases along



(a) Delivery Performance (Bluetooth)

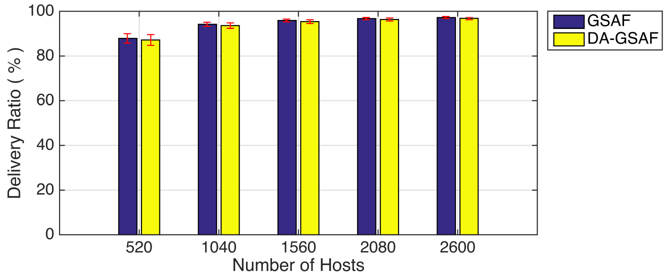


(b) Average Latency (Bluetooth)

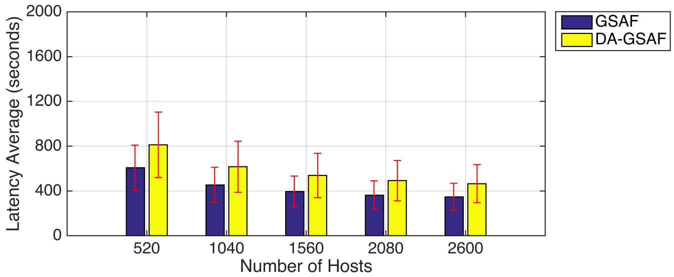


(c) Network Traffic (Bluetooth)

Figure 5.36: Influence of user density on GSAF and DA-GSAF (Crowded scenario)



(a) Delivery Performance (Bluetooth)



(b) Average Latency (Bluetooth)

Figure 5.37: Influence of high density on on GSAF and DA-GSAF;
Upper 10% most visited casts (Crowded scenario)

with the number of users in the network during the message lifetime. Moreover, Figure 5.37 presents the result for the most visited casts in the network. It depicts the impact of higher user density (with the respect to the findings in Figure 5.36 and § 5.2.2) which results in a better delivery ratio as well as less delivery latency.

5.8 Summary

In this chapter, we have presented an extensive performance evaluation of GSAF and DA-GSAF in opportunistic networks using the Geo ONE package which we implemented as an extension of the ONE simulator. We also compared our work with existing geocasting protocols for opportunistic networks. Through a wide range of simulation scenarios we investigated the impact of various parameters on the performance of the proposed protocols. The presented results is the outcome of more than ∼70,000 hours of simulation (clock time) which would not be possible without the help of Sussex university HPC (High Performance Cluster).

Experiments are framed around three major factors and their impact on the performance of routing protocols; i.e. user density, buffer capacity and message lifetime. Each one of them has an impact on the performance of geocasting protocols. Overall, the increase in user density helps a better delivery ratio due to the nature of opportunistic networks (as more nodes help better delivery procedure) as well as the higher network overhead. Even though it does not have a significant effect on the average latency, as more encounters in the routing procedure neutralise the impact of better delivery with more options at hand.

The increase in buffer capacity does not normally have any significant impact on the delivery ratio (only a slight improvement) because of the transmission speed and the encounter time (due to the mobility of nodes) which are the bottlenecks in message exchanging. However, it has a negative effect on the network traffic as larger buffers result in less extinct messages in the network and, therefore, more exchanged messages (especially in the WiFi scenarios). Moreover, the increase in message lifetime has a negative impact on the delivery ratio in general. Due to a wider delivery interval which adds more users (see § 3.2.2) to the final recipients list where some of them may have already left prior to the message arrival into the cast.

In this chapter, we present results based on two different transmission interfaces (i.e. WiFi and Bluetooth). Obviously, WiFi results in better performance due to its higher transmission speed and range. However, as indicated by the results, in the mostly crowded areas, Bluetooth is sufficient to provide an effective communication interface. In addition, we examine the behaviour of the protocols based on two different mobility models, namely random and working day. Due to its complex nature, the working day mobility model is proven challenging to provide an effective routing protocol for. In general, we observed a reduction in the performance of the geocasting protocols, compared to the random mobility model, as most of the users sit in their offices during the working hours and do not contribute in the routing procedure.

All results presented in this chapter, confirm that both GSAF and DA-GSAF are superior compared to the state of the art. According to the results, they perform exceptionally well throughout various scenarios in smaller and larger opportunistic networks. In general, both GSAF and DA-GSAF provide significantly better performance to overhead ratio, due to their copy ticket mechanism, while maintaining the same level of average latency in the network; i.e. significantly better delivery ratio and better network overhead (i.e. less relayed messages).

DA-GSAF follows slightly more advance elective mechanism in comparison to the GSAF which results in a slight difference between the performance of both. It is worth mentioning that DA-GSAF behaves the best with respect to the network overhead, which is a crucial property of a routing protocol in opportunistic networks. GSAF is marginally the second best in terms of network overhead. DA-GSAF shows its strength in very sparse scenarios as well as smaller network environments, because it can select better candidate hops to deliver messages compared to GSAF. However, GSAF is marginally better in denser scenarios. Indeed, GSAF has shown a better performance for the working day mobility model.

In addition, we investigate the impact of (T) copy ticket value on the performance of GSAF and DA-GSAF as they both influenced by its initial value during the first phase of routing. Results show that both protocols perform very well even if the initial values are misconfigured. While DA-GSAF is getting least effect by *"T"*, as it use two more conditions to consume copy tickets in the network[8]. DA-GSAF could help in the situations where it is hard to configure the value of copy ticket. Indeed, to decide among both protocols, GSAF is the *standard* in general while DA-GSAF could help more in the use cases mentioned above.

The implementation of Geo ONE package on the ONE simulator contributed in understanding the behaviour of state of the art geocasting protocols in opportunistic networks. GeoEpidemic proved to be the worst among all of the protocols as it has significantly worse performance to overhead ratio. Its flooding nature is effective but it comes with an immense overhead penalty. GEOOPP and EVR protocols outperform GeoEpidemic, while generally both delivering similar performance. Overall, GEOOP is marginally better, while EVR performed slightly better in more crowded areas. However, GEOOPP uses a complex algorithm which was proven to be computationally very heavy. In a real-life deployment

[8]This helps an efficient use of copy tickets in DA-GSAF during the routing procedure which most of the time results in the less consumption of it. In fact, this is the reason DA-GSAF is better in terms of network overhead against the GSAF.

this would result in much higher energy consumption and quick battery depletion, which could potentially render it unusable. It is worth mentioning that EVR performs far better in the more crowded and dense areas which still is not as good as the GSAF and DA-GSAF (EVR comes with significantly worse network overhead). Moreover, EVR takes less hit in terms of performance by the complexity of mobility model due to its algorithm which use more context of users during the routing procedure. However, it still performed worse in comparison to GSAF and DA-GSAF.

In this chapter, we also discussed about the impact of buffer scheduling policies and how they affect the performance of geocasting routing protocols. The results showed that out of four different scheduling policies (i.e. FIFO, LIFO, HTFO and LTFO), HTFO performs better than the rest and contributes to the reduction of average delivery latency in the network. As it constantly prioritises the messages with higher lifetime and prevents the transmission of messages which may later found to be expired.

Finally, we study the impact of the size of the cast on the GSAF and DA-GSAF. The results show that as the cast size is reduced, the delivery ratio is decreased, the average latency is increased and the network overhead is reduced. Note that the size of casts may vary based on the various application of the geocasting routing protocols.

Chapter 6

Conclusion

What I did, is the duty of researchers against their predecessors; which is observing previous work with selflessness, and modestly fixing their unwanted faults or defects, if there is any.
al-Khwarizmi
Iranian Mathematician, Astronomer & Geographer (c. 780 - 850)

In this thesis we discussed about the importance and the applications of the geocasting in opportunistic networks. The state of the art in opportunistic network geocasting (i.e. EVR, GEOOPP and GeoEpidemic) has been introduced and discussed in Chapter 2. We also extensively presented the major differences between the proposed work with unicast protocols in opportunistic networks and geocast protocols in MANETs.

GSAF (Geocasting Spray and Flood) and DA-GSAF (Direction Aware - GSAF) have been presented as two efficient and flexible protocols for geocasting in opportunistic networks. We highlighted significant challenges in geocasting in the context of opportunistic networks and described how GSAF and DA-GSAF deal with these challenges, overcoming limitations of existing approaches. During a wide range of experiments which are presented in Chapter 5, GSAF and DA-GSAF was proven to be significantly better in comparison to existing approaches. In the crowded areas, most of the times, GSAF and DA-GSAF managed to deliver to more than ∼90% of the recipients which was not possible using previous approaches. This is achieved by keeping the network overhead low through the design of a copy ticket mechanism that is presented in Chapter 3. This allowed a better performance to overhead ratio in GSAF and DA-GSAF against their predecessors.

The majority of the opportunist network protocols use the context of users to get a better understanding of environment to aid routing procedure. This raises privacy concerns for the users of the devices in the opportunistic network. According to the literature, this has a noticeable negative impact on the overall performance as a significant amount of users do not participate in the routing procedure in the real-life applications. In this research, we proposed a different approach to use the context of users without compromising their privacy, to aid routing decisions in the DA-GSAF protocol. This approach has benefited DA-GSAF as we have discussed about the various aspects of its performance in Chapter 5.

Moreover, a new approach to the way of defining geographical casts, has been proposed. This added a set of new challenges which have been solved based on our design in Chapter 3. Polygon shaped casts allow more flexibility based on a better mapping of various shapes into the network environment. This allows less unwanted traffic flow (preventing potential data leak) as well as resource consumption against the circle shaped approach in the network.

The ONE simulator environment provided a chance to understand the problems and solutions which led to the implementation of geocasting opportunistic environment with more than ∼10,000 lines of code. We spent more than hundred thousands of hours on the simulator during the course of four years to understand the potential shortcomings of previous work to shape our design to address the existing problems. Finally, we have implemented the Geo ONE package which provided a tool to simulate geocasting opportunistic networks with the ability to simulate the state of the art geocasting routing approaches; i.e. GSAF, DA-GSAF, EVR, GEOOPP and GeoEpidemic.

6.1 Future Work

Based on the results presented in Chapter 5, GSAF and DA-GSAF outperformed the state of the art geocasting routing protocols in several different scenarios. However, we did not study how close they are to the optimal performance for specific opportunistic network scenarios. Achieving higher performance may be possible with some compromises; e.g. a solution which has better performance in sparse networks would sacrifice the performance of crowded scenarios and may affect other factors (e.g. network overhead). In the research beyond this work, it would be interesting to look towards the special scenarios and a possible solution (with the introduced challenges in mind; see § 3.1) for particular use cases.

We discussed about the applications of geocasting in opportunistic networks. Three major application scenarios have been introduced in Chapter 2. Opportunistic networks are flexible to various existing challenges in the real-life device to device communications. This can lead to more adoption of our research into the other areas in the future.

Throughout the design of DA-GSAF routing protocol, we proposed a new approach to use the context of users to aid routing decisions in the network; without sharing the information with other users and compromising their privacy. This opens up a new approach to the message routing based on the users' context. Designing new ways of utilising more user related information without a potential data leak could be a subject to the future work.

The results presented in § 5.4 investigate the impact of buffer scheduling policies in the geocasting opportunistic networks. We discussed about the importance of congestion management which we proposed in the form of buffer scheduling policy. Four different policies have been proposed and tested and results show the importance of their affect on the performance of the network. More complex congestion management in opportunistic networks could be a subject for a future work.

In § 2.2.6, we mentioned security in opportunistic networks and how DoS attacks may affect their operation. Denial-of-Service attacks may affect GSAF and DA-GSAF (same as all the other routing protocols in opportunistic networks), as there are possibilities of a node intentionally reporting wrong copy-tickets or wrong movement directions in the network. Exploring such issue, was beyond the scope of the work in this thesis. Research on solutions about DoS attacks in geocasting routing protocols is a topic for future work.

Bibliography

[1] 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*, pages 27–34. ACM, 2003.

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

[3] Ian Wakeman, Stephen Naicken, Jon Rimmer, Dan Chalmers, and Ciaran Fisher. The fans united will always be connected: building a practical dtn in a football stadium. In *International Conference on Ad Hoc Networks*, pages 162–177. Springer, 2013.

[4] Tribe Hive project, DTN for football stadiums. `http://tribehive.co.uk/`.

[5] Sacha Trifunovic, Bernhard Distl, Dominik Schatzmann, and Franck Legendre. Wifi-opp: ad-hoc-less opportunistic networking. In *Proceedings of the 6th ACM workshop on Challenged networks*, pages 37–42. ACM, 2011.

[6] Matthias Grossglauser and David Tse. Mobility increases the capacity of ad-hoc wireless networks. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1360–1369. IEEE, 2001.

[7] Vladimir Vukadinović and Gunnar Karlsson. Spectral efficiency of mobility-assisted podcasting in cellular networks. In *Proceedings of the Second International Workshop on Mobile Opportunistic Networking*, pages 51–57. ACM, 2010.

[8] Jörg Ott, Esa Hyytiä, Pasi Lassila, Tobias Vaegs, and Jussi Kangasharju. Floating content: Information sharing in urban areas. In *Pervasive Computing and Communications (PerCom), 2011 IEEE International Conference on*, pages 136–146. IEEE, 2011.

[9] Bo Han, Pan Hui, VS Kumar, Madhav V Marathe, Guanhong Pei, and Aravind Srinivasan. Cellular traffic offloading through opportunistic communications: a case study. In *Proceedings of the 5th ACM workshop on Challenged networks*, pages 31–38. ACM, 2010.

[10] Haiyun Luo, Ramachandran Ramjee, Prasun Sinha, Li Erran Li, and Songwu Lu. Ucan: a unified cellular and ad-hoc network architecture. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 353–367. ACM, 2003.

[11] Jeffrey Erman and Kadangode K Ramakrishnan. Understanding the super-sized traffic of the super bowl. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 353–360. ACM, 2013.

[12] Muhammad Zubair Shafiq, Lusheng Ji, Alex X Liu, Jeffrey Pang, Shobha Venkataraman, and Jia Wang. A first look at cellular network performance during crowded events. In *ACM SIGMETRICS Performance Evaluation Review*, volume 41, pages 17–28. ACM, 2013.

[13] FireChat – the messaging app that's powering the hong kong protests. `http://tinyurl.com/ogsz75o/`.

[14] Yue Cao and Zhili Sun. Routing in delay/disruption tolerant networks: A taxonomy, survey and challenges. *Communications Surveys & Tutorials, IEEE*, 15(2):654–677, 2013.

[15] Christian Maihofer. A survey of geocast routing protocols. *Communications Surveys & Tutorials, IEEE*, 6(2):32–42, 2004.

[16] Yaozhou Ma and Abbas Jamalipour. Opportunistic geocast in disruption-tolerant networks. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1–5. IEEE, 2011.

[17] Yaozhou Ma and Abbas Jamalipour. Opportunistic geocast in large scale intermittently connected mobile ad hoc networks. In *Communications (APCC), 2011 17th Asia-Pacific Conference on*, pages 445–449. IEEE, 2011.

[18] Shanshan Lu and Yonghe Liu. Geoopp: Geocasting for opportunistic networks. In *Wireless Communications and Networking Conference (WCNC), 2014 IEEE*, pages 2582–2587. IEEE, 2014.

[19] Michał Piórkowski. Mobility-centric geocasting for mobile partitioned networks. In *Network Protocols, 2008. ICNP 2008. IEEE International Conference on*, pages 228–237. IEEE, 2008.

[20] Iain Parris and Tristan Henderson. The impact of location privacy on opportunistic networks. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, pages 1–6. IEEE, 2011.

[21] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks.

In *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 252–259. ACM, 2005.

[22] The opportunistic network environment simulator (Github page). `https://akeranen.github.io/the-one/`.

[23] Marco Conti and Silvia Giordano. Multihop ad hoc networking: The theory. *Communications Magazine, IEEE*, 45(4):78–86, 2007.

[24] Philippe Jacquet, Paul Mühlethaler, Thomas Clausen, Anis Laouiti, Amir Qayyum, and Laurent Viennot. Optimized link state routing protocol for ad hoc networks. In *Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century. Proceedings. IEEE International*, pages 62–68. IEEE, 2001.

[25] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile computing*, pages 153–181. Springer, 1996.

[26] Charles Perkins, Elizabeth Belding-Royer, and Samir Das. Ad hoc on-demand distance vector (aodv) routing. Technical report, 2003.

[27] Amin Vahdat and David Becker. Epidemic routing for partially connected ad hoc networks. Technical report, Technical Report CS-200006, Duke University, 2000.

[28] Elizabeth M Daly and Mads Haahr. The challenges of disconnected delay-tolerant manets. *Ad Hoc Networks*, 8(2):241–250, 2010.

[29] Interplanetary networking special interest group (IPNSIG). `http://www.ipnsig.org/`.

[30] Maurice J Khabbaz, Chadi M Assi, and Wissam F Fawaz. Disruption-tolerant networking: A comprehensive survey on recent developments and persisting challenges. *Communications Surveys & Tutorials, IEEE*, 14(2):607–640, 2012.

[31] Zhensheng Zhang. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges. *IEEE Communications Surveys & Tutorials*, 1(8):24–37, 2006.

[32] Radu Handorean, Christopher Gill, and Gruia-Catalin Roman. Accommodating transient connectivity in ad hoc and mobile settings. In *Pervasive Computing*, pages 305–322. Springer, 2004.

[33] Sushant Jain, Kevin Fall, and Rabin Patra. Routing in a delay tolerant network. 34(4), 2004.

[34] Shashidhar Merugu, Mostafa Hamed Ammar, and Ellen W Zegura. Routing in space and time in networks with predictable mobility. 2004.

[35] Zhensheng Zhang and Qian Zhang. Delay/disruption tolerant mobile ad hoc networks: latest developments. *Wireless Communications and Mobile Computing*, 7(10):1219–1232, 2007.

[36] Fabrice Tchakountio and Ram Ramanathan. Tracking highly mobile endpoints. In *Proceedings of the 4th ACM international workshop on Wireless mobile multimedia*, pages 83–94. ACM, 2001.

[37] Delphine Nain, Noshirwan Petigara, and Hari Balakrishnan. Integrated routing and storage for messaging applications in mobile ad hoc networks. *Mobile Networks and Applications*, 9(6):595–604, 2004.

[38] Tara Small and Zygmunt J Haas. The shared wireless infostation model: a new ad hoc networking paradigm (or where there is a whale, there is a way). In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 233–244. ACM, 2003.

[39] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE mobile computing and communications review*, 7(3):19–20, 2003.

[40] James Davis, Andrew H Fagg, Brian N Levine, et al. Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks. In *Wearable Computers, 2001. Proceedings. Fifth International Symposium on*, pages 141–148. IEEE, 2001.

[41] Evan PC Jones, Lily Li, and Paul AS Ward. Practical routing in delay-tolerant networks. In *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 237–243. ACM, 2005.

[42] Mirco Musolesi, Stephen Hailes, and Cecilia Mascolo. Adaptive routing for intermittently connected mobile ad hoc networks. In *World of wireless mobile and multimedia networks, 2005. WoWMoM 2005. Sixth IEEE International Symposium on a*, pages 183–189. IEEE, 2005.

[43] Wenrui Zhao, Mostafa Ammar, and Ellen Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, pages 187–198. ACM, 2004.

[44] Alex Sandy Pentland, Richard Fletcher, and Amir Hasson. Daknet: Rethinking connectivity in developing nations. *Computer*, 37(1):78–83, 2004.

[45] Wemi Zhao, Mostafa Ammar, and Ellen Zegura. Controlling the mobility of multiple data transport ferries in a delay-tolerant network. In *INFOCOM 2005. 24th annual joint conference of the IEEE computer and communications societies. Proceedings IEEE*, volume 2, pages 1407–1418. IEEE, 2005.

[46] Ioannis Chatzigiannakis, Sotiris Nikoletseas, and Paul Spirakis. Analysis and experimental evaluation of an innovative and efficient routing protocol for ad-hoc mobile networks. In *Algorithm Engineering*, pages 99–110. Springer, 2001.

[47] Ioannis Chatzigiannakis, Sotiris Nikoletseas, Nearchos Paspallis, Paul Spirakis, and Christos Zaroliagis. An experimental study of basic communication protocols in ad-hoc mobile networks. In *Algorithm Engineering*, pages 159–171. Springer, 2001.

[48] Christian Becker and Gregor Schiele. New mechanisms for routing in ad hoc networks through world models. In *Proceedings of the 4th CaberNet Plenary Workshop, Pisa, Italy*, pages 1–4, 2001.

[49] Zong Da Chen, HT Kung, and Dario Vlah. Ad hoc relay wireless networks over moving vehicles on highways. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 247–250. ACM, 2001.

[50] Hyunwoo Kang and Dongkyun Kim. Vector routing for delay tolerant networks. In *Vehicular Technology Conference, 2008. VTC 2008-Fall. IEEE 68th*, pages 1–5. IEEE, 2008.

[51] Hyunwoo Kang and Dongkyun Kim. Hvr: History-based vector routing for delay tolerant networks. In *Computer Communications and Networks, 2009. ICCCN 2009. Proceedings of 18th Internatonal Conference on*, pages 1–6. IEEE, 2009.

[52] Yong Wang, Sushant Jain, Margaret Martonosi, and Kevin Fall. Erasure-coding based routing for opportunistic networks. In *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 229–236. ACM, 2005.

[53] Ling-Jyh Chen, Chen-Hung Yu, Tony Sun, Yung-Chih Chen, and Hao-hua Chu. A hybrid routing approach for opportunistic networks. In *Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, pages 213–220. ACM, 2006.

[54] Badri N Vellambi, Ramanan Subramanian, Faramarz Fekri, and Mostafa Ammar. Reliable and efficient message delivery in delay tolerant networks using rateless codes. In *Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, pages 91–98. ACM, 2007.

[55] Atsushi Fujimura, Soon Y Oh, and Mario Gerla. Network coding vs. erasure coding: Reliable multicast in ad hoc networks. In *Military Communications Conference, 2008. MILCOM 2008. IEEE*, pages 1–7. IEEE, 2008.

[56] Yunfeng Lin, Ben Liang, and Baochun Li. Performance modeling of network coding in epidemic routing. In *Proceedings of the 1st international MobiSys workshop on Mobile opportunistic networking*, pages 67–74. ACM, 2007.

[57] Wenrui Zhao, Mostafa Ammar, and Ellen Zegura. Multicasting in delay tolerant networks: semantic models and routing algorithms. In *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 268–275. ACM, 2005.

[58] Qing Ye, Liang Cheng, Mooi Choo Chuah, and Brian D Davison. Os-multicast: On-demand situation-aware multicasting in disruption tolerant networks. In *Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd*, volume 1, pages 96–100. IEEE, 2006.

[59] Qing Ye, Liang Cheng, Mooi Choo Chuah, and Brian D Davison. Performance comparison of different multicast routing strategies in disruption tolerant networks. *Computer Communications*, 32(16):1731–1741, 2009.

[60] Qing Ye, Liang Cheng, Mooi Choo Chuah, and Brian D Davison. Shim: a scalable hierarchical inter-domain multicast approach for disruption tolerant networks. In *Proceedings of the 2007 international conference on Wireless communications and mobile computing*, pages 170–175. ACM, 2007.

[61] Peng Yang and Mooi Choo Chuah. Context-aware multicast routing scheme for disruption tolerant networks. In *Proceedings of the 3rd ACM international workshop on Performance evaluation of wireless ad hoc, sensor and ubiquitous networks*, pages 66–73. ACM, 2006.

[62] Janico Greifenberg and Dirk Kutscher. Efficient publish/subscribe-based multicast for opportunistic networking with self-organized resource utilization. In *Advanced Information Networking and Applications-Workshops, 2008. AINAW 2008. 22nd International Conference on*, pages 1708–1714. IEEE, 2008.

[63] Wei Gao, Qinghua Li, Bo Zhao, and Guohong Cao. Multicasting in delay tolerant networks: a social network perspective. In *Proceedings of the tenth ACM international symposium on Mobile ad hoc networking and computing*, pages 299–308. ACM, 2009.

[64] P Yang and M Chuah. Efficient interdomain multicast delivery in disruption tolerant networks. In *Mobile Ad-hoc and Sensor Networks, 2008. MSN 2008. The 4th International Conference on*, pages 81–88. IEEE, 2008.

[65] Jie Wu and Yunsheng Wang. A non-replication multicasting scheme in delay tolerant networks. In *Mobile Adhoc and Sensor Systems (MASS), 2010 IEEE 7th International Conference on*, pages 89–98. IEEE, 2010.

[66] Yunsheng Wang, Xiaoguang Li, and Jie Wu. Multicasting in delay tolerant networks: delegation forwarding. In *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pages 1–5. IEEE, 2010.

[67] Ramanathan Subramanian and Faramarz Fekri. Throughput performance of network-coded multicast in an intermittently-connected network. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on*, pages 212–221. IEEE, 2010.

[68] Yili Gong, Yongqiang Xiong, Qian Zhang, Zhensheng Zhang, Wenjie Wang, and Zhiwei Xu. Wsn12-3: Anycast routing in delay tolerant networks. In *Global Telecommunications Conference, 2006. GLOBECOM'06. IEEE*, pages 1–5. IEEE, 2006.

[69] Samuel C Nelson and Robin Kravets. Achieving anycast in dtns by enhancing existing unicast protocols. In *Proceedings of the 5th ACM workshop on Challenged networks*, pages 63–70. ACM, 2010.

[70] Ederson R Silva and Paulo R Guardieiro. An efficient genetic algorithm for anycast routing in delay/disruption tolerant networks. *Communications Letters, IEEE*, 14(4):315–317, 2010.

[71] M Zubair Shafiq, Lusheng Ji, Alex X Liu, Jeffrey Pang, Shobha Venkataraman, and Jia Wang. Characterizing and optimizing cellular network performance during crowded events. *IEEE/ACM Trans. Netw.*, 24(3):1308–1321, 2016.

[72] Ian Wakeman, Stephen Naicken, Jon Rimmer, Dan Chalmers, and Ciaran Fisher. The fans united will always be connected: building a practical dtn in a football stadium. In *Ad Hoc Networks*, pages 162–177. Springer International Publishing, 2014.

[73] Xia Jiang and Tracy Camp. A review of geocasting protocols for a mobile ad hoc network. Technical report, Colorado School of Mines, 2002.

[74] Robert Harle. A survey of indoor inertial positioning systems for pedestrians. *Communications Surveys & Tutorials, IEEE*, 15(3):1281–1293, 2013.

[75] Eric Haines. Point in polygon strategies. *Graphics gems IV*, 994:24–26, 1994.

[76] Jens Weppner and Paul Lukowicz. Bluetooth based collaborative crowd density estimation with mobile phones. In *Pervasive computing and communications (PerCom), 2013 IEEE international conference on*, pages 193–200. IEEE, 2013.

[77] The opportunistic network environment simulator. `http://www.netlab.tkk.fi/tutkimus/dtn/theone/`.

[78] Ari Keränen, Jörg Ott, and Teemu Kärkkäinen. The one simulator for dtn protocol evaluation. In *Proceedings of the 2nd international conference on simulation tools and techniques*, page 55. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009.

[79] OpenJump open source geographic information system (GIS). `http://www.openjump.org/`.

[80] Frans Ekman, Ari Keränen, Jouni Karvo, and Jörg Ott. Working day movement model. In *Proceedings of the 1st ACM SIGMOBILE workshop on Mobility models*, pages 33–40. ACM, 2008.