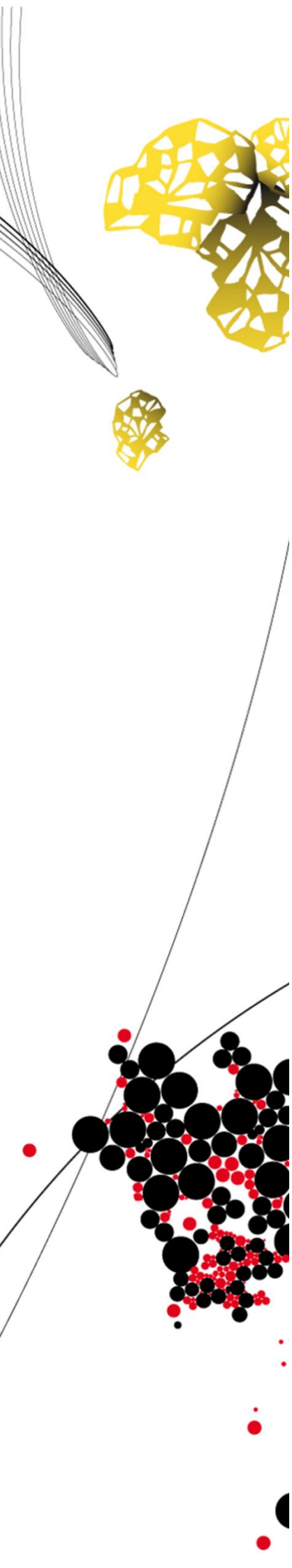


UNIVERSITY OF TWENTE.

Faculty of Electrical Engineering,
Mathematics & Computer Science

Design and evaluation of a link-state routing protocol for Internet-Wide Geocasting

Jon Larizgoitia Burgaña
M.Sc. Thesis

May 2017

Supervisors:

dr. ir. Geert Heijenk

dr. ir. Pieter-Tjerk de Boer

Bernd Meijerink

Design and Analysis of Communication Systems Group

Faculty of Electrical Engineering,

Mathematics and Computer Science

University of Twente

Enschede, The Netherlands

Abstract

Geocasting is a forwarding mechanism that is used to transport relevant data to a group of routers in a network by using their geographical location. The main objective of this project is to design a routing protocol that will be able to perform the routing of a packet based on the geographical addressing scheme for the Internet.

The proposed protocol is based on the link-state algorithm and it fulfils the requirements of geocasting in fixed infrastructure networks. The key concept used in the designed protocol is that every router in the network has full network knowledge. Therefore, every router knows the coverage area of the other routers and can calculate the best path to reach them.

In order to validate the design of the routing protocol, it has been implemented in a representative simulation model of the network in Python. The implementation consists of a node-link diagram where we can analyse and visualise how predefined packets are delivered to destination routers.

This tool permits to evaluate and compare the performance of the routing protocol in different network scenarios. The measured metrics used in this evaluation are link usage, number of path calculations and accuracy.

Keywords

Geocasting, routing protocol, link-state protocol, wired networks, simulation.

Acknowledgements

I cannot finish this project without acknowledging the people who have supported me during this process. Without them, this project would not have been possible.

I would like to express my immense gratitude to my thesis supervisor Dr. ir. Geert Heijenk, for giving me the opportunity to participate in this project and for additionally extending the deadline time. Thanks for his valuable suggestions and directions to accomplish the master thesis project.

I am also greatly indebted to my thesis supervisor Bernd Meijerink for his advice, support and guidance during this project. Providing me with constant encouragement from the beginning of the work and moral support in many ways during my stay at the University of Twente.

As well, I would like to express my sincere thanks to Dr. ir. Pieter-Tjerk de Boer, for providing me suggestions and comments during the project.

Similarly, my thankfulness to the people of Design and Analysis of Communication Systems (DACS) group of the UT and university colleagues, for providing advice and support during my whole study period in the Netherlands.

Also, I would like to express my thanks to my parents, brother and girlfriend, as they have been the ones who have always been there, giving me their support at any time, both in positive and negative moments. Not forgetting my grandmother, as she is always there.

Finally, thanks to my family and friends for their patience and encouragement. Not forgetting my close classmates; different people from the University of Twente and Deusto; and the people known in my Erasmus, for making my university period one of the most amazing life experiences.

Table of Contents

List of Figures

Acronyms

IP	Internet Protocol
ITS	Intelligent Transportation Systems
V2V	Vehicle to vehicle communication
V2I	Vehicle to infrastructure communication
GPS	Global Positioning System
EGP	External Gateway Protocol
IGP	Internal Gateway Protocol
BGP	Border Gateway Protocol
ISP	Internet Service Provider
RIP	Routing Information Protocol
EIGRP	Enhanced Interior Gateway Routing Protocol
LSP / LSA	Link State Packet / Link State Advertisement
OSPF	Open Shortest Path First
AS	Autonomous System
IS-IS	Intermediate System to Intermediate System
SPF	Shortest Path First
TTL	Time to Live
GML	Geography Markup Language
JSON	JavaScript Object Notation

1. Introduction

The availability of information has become essential in our society and the vehicles around us are not an exception. In the last years, deployment of infrastructures and development of communications have lead us to Intelligent Transportation Systems (ITS), which aim to provide services relating to traffic management.

In vehicular networks, communication between vehicles could bring additional value to drivers and passengers. Improving the actual transport networks by providing access to relevant data. Hence, if a vehicle or driver receives relevant information at the right moment, it could make a difference regarding traffic flow and safety. For this reasons, an application where geocasting is used to deliver relevant messages is very promising in vehicular networks.

Geocasting is a mechanism where the destination of a packet containing information is specified by an area, instead of by an address. In which nodes belonging to a delimited region are the destination of the message.

In geographical networking, routing is performed based on the location of nodes, rather than on their IP (Internet Protocol) address. For instance, warning messages are typically sent to all vehicles in a particular area or road segment.

However, Georouting is still subject to research and there are plenty of issues to address. The objective of this M.Sc. project is to research, design and validate an effective routing and forwarding method for the Internet-wide Geocasting.

It is expected that the emergence of communicating vehicles will allow safer and more efficient methods of transport in the future. Allowing vehicles to communicate with each other in an effort to warn drivers about potential risks of accidents and as a result of this prevent crashes.

1.1. Motivation and Problem Statement

The development of a routing protocol capable of sending packets to certain locations would improve efficient location-dependent services by implementing more signalling from the client.

In fact, if a vehicle has access to relevant data that is available on the Internet using wireless communication with infrastructure, the driver or the vehicle could make decisions depending on the traffic conditions.

There are several proposals for geocasting in wireless communications for vehicles, but there is no precedent of geocasting implementation supported by fixed infrastructures. Keep in mind that the Internet is a logical network that only knows about nodes and connected links, with no relation to the underlying geography. So, this is a huge challenge and it needs to take into account the design of scalable Internet-Wide Geocasting.

Geocasting requires a routing protocol that permits to send messages from a source to a target geographical region. In this field, it is a challenge to deploy a routing protocol capable of

managing geographical addressing scheme to route geocasting packets. Because of the lack of routing and forwarding mechanism to send geocasting packets using fixed infrastructure networks is evident.

One of the main encountered problems is that all the destinations nodes that are part of a geographical region need to be reached, regardless their position inside the network. This is a problem because actual routers do not care about physical regions.

Therefore, one of the most important challenges is the availability of an appropriate addressing mechanism to transmit information to vehicles in a target area. For this purpose, a routing protocol is needed.

1.2. Research questions

As mentioned in the previous section, the concept of integrating geocasting into the existing fixed infrastructure of the Internet is a significant development in ITS research field. This thesis focuses on the design of a routing protocol as a solution for the routing and forwarding of data packets through the wired network, based on the Efficient Geographical Addressing Scheme for the Internet [1].

This statement allows us to formulate our main research question.

- Can we design a link-state routing algorithm for Internet-Wide Geocasting scheme?

This main research question lead us to formulate some sub-research questions. These sub-research questions will guide us in the design goals that want to be achieved in this project.

- How do routers communicate with each other?
- How can we reach two or more routers that are in the same geographical region if they are not directly connected?
- How does the size of the network impact the routing performance in multiple scenarios?

1.3. Approach and Contribution

In this project, a background study has been done to identify the main issues related to the sending of geocasting packets through the wired network. This will allow us to identify problems that will be considered during the design process.

In this thesis, we have designed and evaluated a link-state routing and forwarding protocol for the Internet-Wide Geocasting. This has resulted in the following contributions:

- This thesis proposes an adaptation of a link-state routing protocol for the fixed network infrastructure as a solution of reliable sending of Geocasting packets through the network.
- Design of the routing protocol and the update mechanism to maintain the network working while topology changes occur.
- Implementation of a simulation environment to validate the design of the proposed protocol for Internet-Wide Geocasting.
- The practical application of the recently proposed system to two different real world scenarios.

This simulation environment counts with a simulation application which is capable of displaying the complete operation of the system in a real network scenario. Thanks to this platform, it can be seen how the routing algorithm performs facing certain events and incidents.

1.4. Outline

This thesis document is organised as follows. In the second chapter, we present a background study of geocasting and the actual routing protocols. Then main challenges, design considerations and assumptions are discussed in the approach of the project (Chapter 3).

In Chapter 4, we explain the details concerning the design of the routing protocol; discussing important aspects and encountered design issues during the project. Once the design is concluded, in Chapter 5, we present the implementation of the proposed protocol. We will also state the details that have been considered to develop the simulation environment.

In Chapter 6, we will describe the test data samples used to represent different scenarios and the results obtained from the simulation sessions. This way we aim to evaluate the performance of the proposed design in various network topologies. Finally, conclusions and future work are explained in Chapter 7.

2. Background

In order to find answers to our research questions, it is necessary to discuss the context of the research first. In this chapter, we describe the current situation of the Geocasting functionality in vehicular networks. Also, it discusses basic concepts regarding routing protocols in wired networks, necessary to understand later design decisions.

2.1. Vehicular Networks

In the past decade, the number of vehicular networks has grown significantly [2]. Most car manufacturers, governments and researchers have confirmed the importance and potential impact of vehicular networks by the increase of investments to develop new technologies in this field. Their main motivation is to improve road safety and traffic flow by implementing new ways of Internet access and deployment of entertainment services in vehicles [3].

A vehicular network can be divided into two categories: Ad-hoc networks and Fixed infrastructure networks. On the one hand, Ad-hoc networks group all the communications that take place between vehicles and between vehicle and infrastructures [4]. On the contrary, Fixed infrastructure networks consist of wired networks that use the actual IP infrastructure to send information from one point to another of the world.

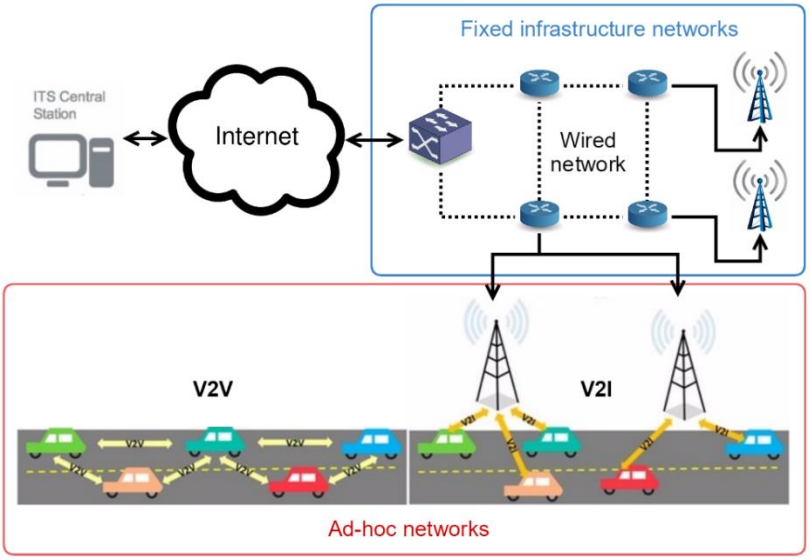


Figure 1. Vehicular networks: Ad-hoc networks and Fixed infrastructure networks [5]

Inside an Ad-hoc network, we can find two types of wireless communications: Vehicle to vehicle communication (V2V) and Vehicle to Infrastructure communication (V2I). These communications involve vehicles exchanging data with each other and with the infrastructure [6]. The V2V communication exchange data between two vehicles, without involving any node

in the infrastructure. Whereas V2I communication, share information from the vehicle to infrastructure and vice versa [7]. These enabling technologies have proved to improve traffic safety and increase the efficiency of transportation systems.

This project takes place in a Fixed infrastructure networks scenario. One of the objectives is to design a routing protocol to route the geocasting packets using the existing wired network infrastructure.

2.2. Geocasting

In this segment, we are going to do an overview of geocasting concepts. We will introduce some of the use cases of geocasting and the problems associated with the past proposals. Finally, we will explain the Geoaddressing scheme [1] on which is based on this project.

Geocast is defined as a forwarding mechanism that is used to transport data from a single node to all nodes within a target area, which is specified by geographical regions [8]. We consider that some nodes are in the same group when they are inside the boundaries of a determined geographic area.

There are several methods to define a particular geographic area. In this project, it has been decided to use the method described in section 2.2.3 based on GPS coordinates to define those geographic regions.

This means that the number of devices found in a target area will only depend on the nodes that belong in that instant to the geographic region. For example, if some vehicles enter a target area they will pass to be part of the group of that geographic region.

In the following lines, we introduce and compare three kinds of communication forwarding schemes. On the one hand, unicast corresponds to one to one communication and transmits data from a source to a single destination. On the other hand, multicast is a forwarding mechanism that allows a source node to send the same data packet to multiple destinations, where all the members of the same group share the same multicast address [9].

As we have commented before, the main aim of Geocasting is to deliver the data to a group of nodes situated within a specified geographical area [9]. We consider that the node will change its geographic area if it moves or it joins another coverage area. This way it supports the delivery of packets to a certain geographical area, addressing a geographical area instead of a common IP address.

The concept of Geocasting is similar to multicasting. The main difference between them is that in multicasting nodes can join or leave a group as desired, whereas in geocasting nodes join or leave the group by entering or leaving the coverage region. This coverage region is determined by the physical location of the node or by the router itself.

In a first approach, it was suggested to adapt multicast mechanism to geocasting routing. However, the tracking of the routers is a complex task because new nodes will enter or leave a certain geographic region. Keep in mind that the Internet is a logical network that only knows about its nodes and connecting links and has no relation to the underlying geography. This mainly is the reason why we cannot use multicasting functionalities in a geocasting environment.

2.2.1. Applications

The implementation of the geocasting mechanism will allow enabling the creation of location dependent services. Particularly in the vehicular networks field, geocasting will permit the distribution of relevant information to a certain destination region and all the nodes that are inside that region. Those regions could be roads, highways or even a city and its surroundings.

The implementation of the geocasting mechanism will enable to reach the vehicles in a specific area and provide them with relevant information in order to decrease the probability of traffic accidents. For instance, some of the use cases in this field are the delivery of traffic information, such as emergency alerts, road conditions, weather information and delivery of entertainment data. [3]

It must, however, be pointed out that there exist similar use cases apart from the ones seen in the vehicular field, where a source node in the Internet geocasts several packets to a destination area.

2.2.2. Past proposals

In the last decade, there have been some proposals of geocast based routing protocols in Ad-hoc networks. However, papers describing geocasting systems for large fixed IP networks are uncommon. In this section, several past proposals in the geocasting field are briefly described.

- **Georouting Overlay**

In this approach, Navas and Imielinski proposed to add an overlay network on top of the existing IP infrastructure. The network is formed by routers with special characteristics and modification of sending and receiving nodes. Each of those special routers will be responsible for a delimited area, where regions are defined by addressing based on GPS coordinates [10].

The main problem with this proposal is the introduction of new routers with special needs to the network to enable geocasting functionalities.

- **Full network knowledge georouting**

The previous authors have also proposed a routing algorithm that uses full network knowledge to route geographic packets [11]. The problem with this approach is the significant amount of routing tables used to route a geocasting packet through the network. The destination areas need to be simplified in forwarding routers to reduce routing time.

- **GeoTORA**

This proposal is based on an anycast modification of TORA (Temporally Ordered Routing Algorithm) [12]. GeoTORA maintains for each geocast group a directed graph including all network nodes, which shows routing direction to the destination region [13]. Remind that this protocol is aim to operate in Ad hoc networks. Therefore, as the nodes are supposed to change their position, it will be impossible to keep the network graph stable.

- **GeoHash**

GeoHash is a method for efficiently indexing geographical coordinates with arbitrary precision.

This approach splits the world map into halves based on the GPS coordinates. The author wanted to geocode specific geographical points as a binary string in which each character

indicates alternating divisions of the global longitude-latitude rectangle [14]. However, GeoHash has limitations in some geographic regions; the addressing does not work correctly near Greenwich meridian.

- **GeoGRID**

GeoGRID approach separates the network into logical grids and elects a single gateway in each partition. The border gateways will exchange messages, but flooding is used to propagate messages inside each partition [13].

- **GPS based multicast**

In this proposal, each section is assigned a multicast address. It has the advantage that this protocol can work in any IP multicast network. However, troubles appear when a sender needs to know the exact destination area because the target regions are predefined [15]. In busy environments, each router would need to keep track of a large number of multicast groups.

- **eDNS**

Extended DNS (eDNS) consist of a modified DNS system that allows the implementation of a layer geocast solution. When a device wants to perform geocast to a destination area, the device executes a DNS lookup. This DNS lookup will return IP addresses belonging to geographical regions. The source can then use this addresses as destination address for the geocasting of the packets [16].

This proposal establishes the need to update periodically in a central database the location of each device involved in the communication.

2.2.3. Efficient Geographical Addressing Scheme [1]

This project is based on the Efficient Geographical addressing scheme. The idea behind this addressing scheme is to deliver geocasting messages using the bandwidth of the wired network in an efficient way.

The geographic regions are represented by polygons. Notice that the fundamental polygon used to delimitate the areas is a rectangle. There are other approaches that propose other types of polygons that will lead to a more accurate form to select a geographic region, but there has not been an exact solution yet that considers a different polygon apart from rectangles.

This addressing scheme is based on the GPS coordinates, latitude and longitude (WGS 84).

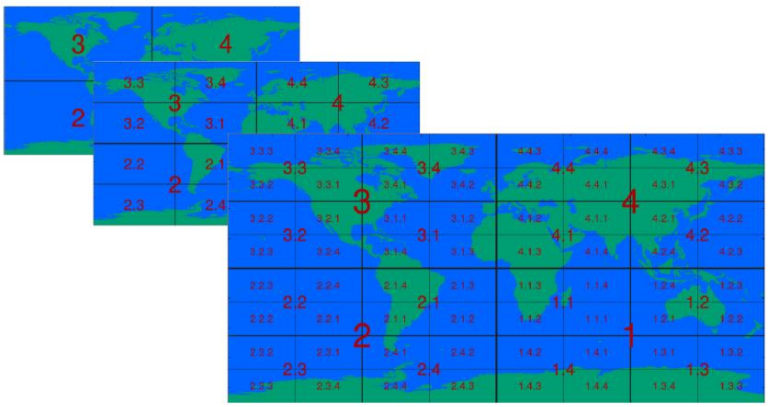


Figure 2. Geographical addressing scheme until level 3 [1]

So that this addressing system that has been introduced permits to combine adjacent rectangles into a single area. The system allows splitting the world map in four rectangles (level 1) based on WGS 84 coordinate system (90 to -90 degrees Latitude and -180 to 180 degrees Longitude).

In each extra level, the map splits into four sub-rectangles. As it is shown in Figure 2, it combines neighbouring rectangles into a single area to address a geographic region. If we get a closer look at Figure 2, we notice that neighbouring sub-rectangles that are on the same level and from different parents share numbers. With this system, we can establish relation adjacencies between neighbour rectangles and, if necessary, combine them into a single area.

The main achievement of this addressing scheme is the integration of physical location into an IPv6 address. This addressing method has a binary representation that can be used to perform route lookups based on prefix matching with an OR operation, avoiding to do costly calculations. Each level is represented as a 4 bit block (1 = 1000, 2 = 0100, 3 = 0010, 4 = 0001). For instance, imagine we want to address “Hengelostrat road” in Enschede. This road corresponds to 4.4.2.3.2.1.1.2.4.2.[1,2].4.[2,3] address.

4.4.2.3.2.1.1.2.4.2. [1,2].4. [2,3] =
0001.0001.0100.0010.0100.1000.1000.0100.0001.0100.1100.0001.0110

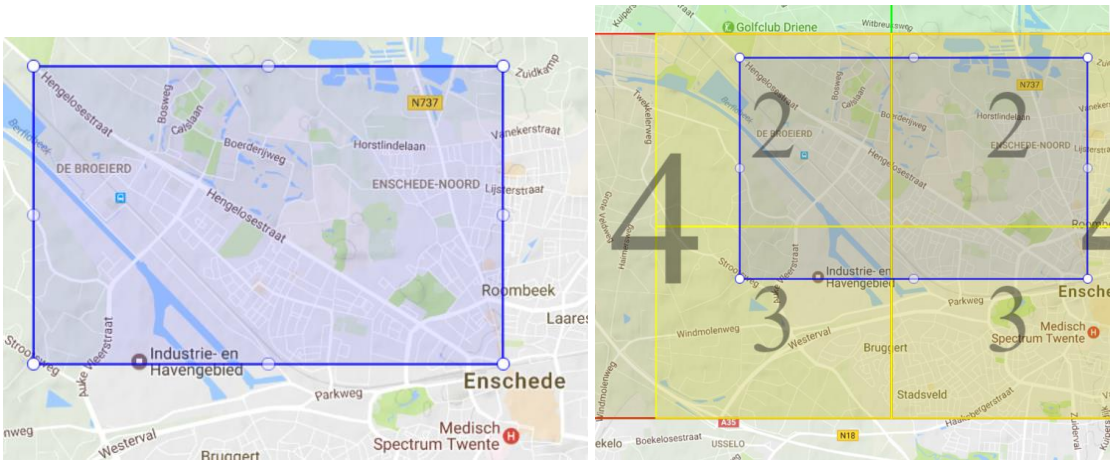


Figure 3. Addressing of Hengelosestraat road

The geoaddress is the IPv6 address associated with a specific region on the world map. This addresses are predefined and can be calculated with the latitude and longitude coordinates. They represent a coverage region of the word map. So, routers will have a coverage area that refers to a specific address that delimits a specific world map region.

A geocasting mechanism based on the introduced addressing scheme could allow distributing location-based information in a vehicular network scenario.

The goal of this project is to work in the design of routing and forwarding mechanism that could use Geographical addressing scheme to geocast messages in Fixed infrastructure networks.

2.3. Routing protocols

Routing is how packets get from one point of the network to another. A routing protocol is a set of processes, algorithms and messages that are used to exchange routing information and populate the routing tables. This information is crucial for the routing protocol to choose the best path for each message [17].

The purpose of a routing protocol includes maintain update routing information, identify other nodes and determine the best route to each destination inside the network. In case the current path is no longer available, it has the ability to find a new alternative route [17]. Therefore, we can say that routing protocols are the mean of information propagation between routers in the network.

Routing protocols are based on various algorithms which depend on metrics to find the best path for sending data through the Internet. Those metrics are stored in routing tables which are continuously updated by dynamic routing protocols [18].

In this section, we present a brief overview of dynamic routing protocols and make a comparison between different types of common routing protocols.

Dynamic routing protocols are divided into Interior and Exterior Gateway protocols. Interior Gateway Protocols (IGP) are used to route Internet communications within a local area network; while Exterior Gateway Protocols (EGP) are used to exchange routing information between different networks, generally on the Internet backbone [19].

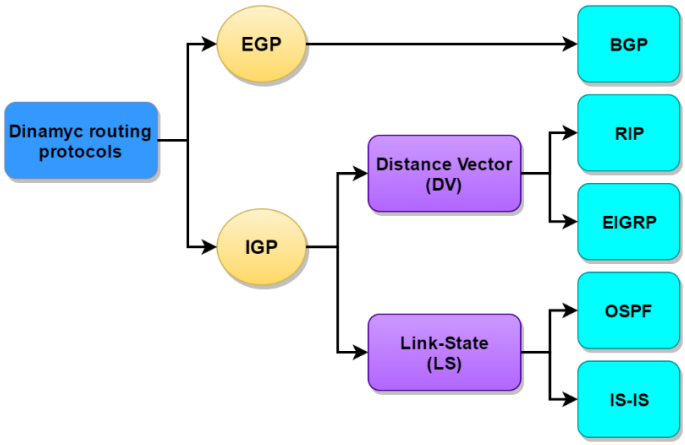


Figure 4: Types of routing protocols

As shown in Figure 4, the most used EGP protocol is BGP (Border Gateway Protocol). BGP is widely used to exchange routing information on the Internet between Internet Service Providers (ISP) [20].

Since in this project we are going to analyse the operation of a given network in a geocasting environment, in the following subsections we are going to focus on the key features of Interior Gateway Protocols.

2.3.1. Flooding

Flooding is a simple network routing algorithm in which incoming packets of a router are sent through every outgoing link, except the neighbour node from which the packet was received.

Multiple copies of the same packet may be received by a router, because the same packet may be forwarded by different links each time.

In Uncontrolled Flooding, there is no control over duplicate packets. This creates nodes to broadcast packets indefinitely. As a consequence of this phenomenon, Controlled Flooding or Sequence Number Controlled Flooding (SNCF) is introduced to solve loop problems.

In controlled flooding, every sender node puts its address and sequence number in the broadcast packet. Each node that receives this broadcast packet keeps a cache of this information. If the router receives the packet with the same sequence number again, it will immediately discard it. Otherwise, it will resend it.

As we have seen, flooding mechanism is the most basic forwarding mechanism. It makes possible to route a packet to any or every router without knowing anything about the connectivity or the topology of the network.

The main problem with this procedure is that a huge number of redundant copies of the messages are needed to deliver a packet. Due to this reason, flooding is inefficient over vast networks of the Internet. Therefore, other routing protocols are required to use the available bandwidth efficiently.

2.3.2. Distance vector routing protocols

Distance vector protocols use distance to decide the best route to a destination node. This distance is represented by a total metric, such as hop count, and the vector that refers to the next hop.

This distance or cost is used to calculate the next hop with the least cost to reach certain destination node.

Routing information is only exchanged between directly connected neighbours. This means a router knows from which neighbour it learned a route, but it does not know where that neighbour learned the route.

Moreover, each node maintains a vector table of minimum distance to every node. After building their routing tables, they transmit the entire routing table to directly connected routers.

When a node receives some information about directly connected neighbours, it performs the hop count calculations and then distributes the results back to its neighbours. Therefore, we can say that the distance vector protocols are iterative, asynchronous and distributed.

The most used distance vector routing protocols are the following:

- **RIP**

Routing Information Protocol (RIP) is an established distance vector protocol that measures the distance of a route by hop count [21]. The term hop count refers to the number of nodes that a packet passes through from its source to its destination. On the whole, we can assume that passing through each link has a cost of one by default.

The maximum cost of the path is limited to 15 hops to prevent infinite loops, which is considered a limitation for large networks [22].

In RIP, routing updates are exchanged periodically between neighbours, approximately every 30 seconds. As we have mentioned before, each router maintains a routing table with the list of router destinations and the distance to each of those routers. This causes routers to broadcast their entire current routing database periodically.

RIP has little overhead regarding used bandwidth, configuration time and management time. Also, it is very easy to implement, especially in relation to other IGPs.

- **EIGRP**

Enhanced Interior Gateway Routing Protocol (EIGRP) is a Cisco proprietary routing protocol that combines the advantages of link-state protocols with the benefits of distance vector protocols. It is a distance vector protocol that allows routers to share information to the adjacent routers which are inside the same area [23].

The main difference with RIP protocol is that instead of sending the entire routing table to the neighbouring routers, EIGRP only shares the information that is needed. EIGRP only sends routing updates about paths that have changed. This reduces the workload and the amount of data that needs to be transmitted because only Hello packets need to be sent periodically.

In addition, this Cisco protocol considers hop count, delay and available bandwidth to form a metric value. As we mentioned, this metric value is used to select the optimal route to the reachable nodes inside an area.

2.3.3. Link-state routing protocols

The link-state protocol calculates the minimum cost path between a source and a destination node using full network knowledge. This means that network connectivity (network´s topology) and link state information of all links are available to nodes for making routing decisions. In the next paragraphs, it is explained how a link-state protocol obtains that information.

First of all, each router establishes communication with every neighbour node using hello packets. We consider as neighbour node all adjacent nodes directly connected to the source router and reachable by using one link.

Then each router creates its own link-state packet (LSP), also known as a link-state advertisement (LSA), that includes information about neighbours and the link to reach them.

Once the LSP is created, the node floods it to all its neighbours. Every time a neighbour receives an advertisement, it will save the information and forward the packet to its own adjacent neighbours.

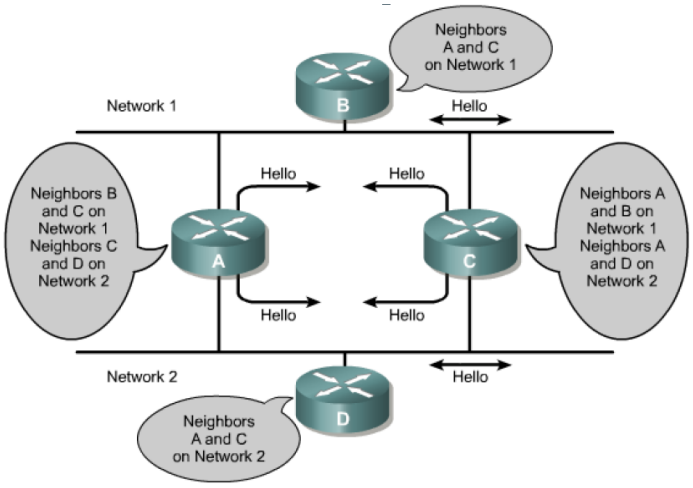


Figure 5. Link-state protocol: Discover and maintain neighbour adjacencies [24]

When all routers have received the LSPs, they create a map that describes the topology of the network called topological database or link-state database. Keep in mind that when the distribution of these advertisement packets has concluded, the databases in all routers should be identical. In consequence, when the network is stable (no link failure occurs), the view of network topology by each router should be synchronised.

Once the link state database is completed, any router is able to access full network knowledge information. Based on this information, each router calculates the shortest path to any router using the Dijkstra's algorithm (explained in Chapter 2.4.1) and inserts this information into the routing table, so that the routing table contains a list of known paths and interfaces.

We have seen that link-state protocols track the status and connection type of each link and produce a calculated metric based on link state (up or down), bandwidth and congestion. Next, based on this information calculates the total cost of the path to reach the desired destination nodes. Actually, a link-state protocol will choose a path which has more hops, but that uses a faster medium over a path using a slower medium with fewer hops.

Link-state routing protocols use broadcast to distribute link-state information used to calculate optimal paths. Concretely, controlled flooding is used to broadcast link state advertisements.

The most popular link-state routing protocols are the following:

- **OSPF**

Open Shortest Path First (OSPF) is a link-state routing protocol developed for IP networks as an IGP. The key features of this protocol are its availability for public use and that it uses Dijkstra shortest path first algorithm, which will be explained in Section 2.4.1.

OSPF routing protocol operates in a hierarchical structure. This means that a general network is divided into smaller networks. In other words, a collection of networks under a common administration entity is divided into areas. We consider an area a group of adjacent nodes that can communicate with each other.

A topological database is a map of node distribution in an area. It is created using the LSAs received from all routers in the same area. This means that nodes inside Area 1 are invisible for routers outside this area.

There are two different types of OSPF routing, depending on whether the source and destination nodes are in the same area or not. Intra-area routing occurs when the source and the destination nodes are in the same are; while Inter-area routing happens when routers are located in different areas.

For each area, Area Border Routers (ABR) are elected to achieve the communication between nodes in different areas. On the one hand, these border nodes have multiarea configurations, which means that they need to maintain multiple topology databases to connect various areas of the domain. On the other hand, routers inside each area have single area configuration.

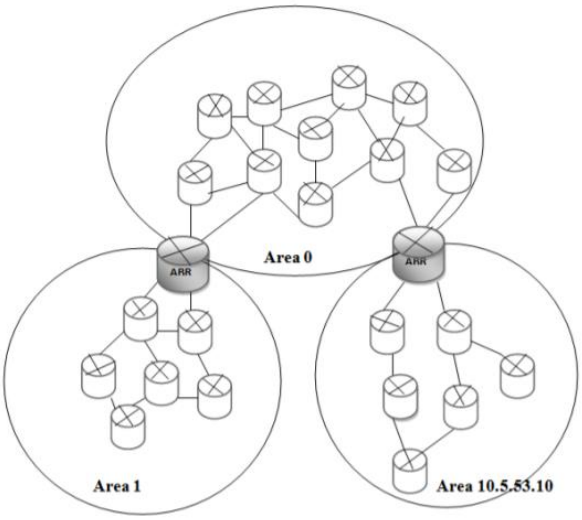


Figure 6. OSPF implementation [25]

In Figure 6, we can see an OSPF network divided into three areas. Nodes that belong to different Intra-area networks are not visible to each other because they are only aware of the topology information of their own area. The area 0, also known as Backbone area, is responsible for distributing routing information between areas and it is connected with the rest of area networks using border gateways. Therefore, all the Area Border Routers need to use the same algorithms to maintain routing information within the backbone area.

Area border routers running OSPF learn about exterior routers through an EGP protocol or configuration.

- **IS-IS**

Intermediate system to Intermediate system (IS-IS) is a link-state routing protocol used by network devices to determine the best way to deliver a packet [26]. The protocol uses controlled flooding to broadcast link-state information through a network of routers. Based on the received link-state packets (LSP), every router builds a database of the network's topology [27].

IS-IS organises the routing domain in two-level hierarchical routing. As we can see in Figure 7, IS-IS routers are named as Level 1 (intra-area, L1), Level 2 (interarea, L2) or Level 1-2 (both). Routing information is exchanged between L1 routers and other L1 routers of the same area. In the same manner, L2 routers can only exchange information with other L2 routers. So Level 1-2 routers share information with both levels and are used to connect the inter-area routers with the intra-area routers [28].

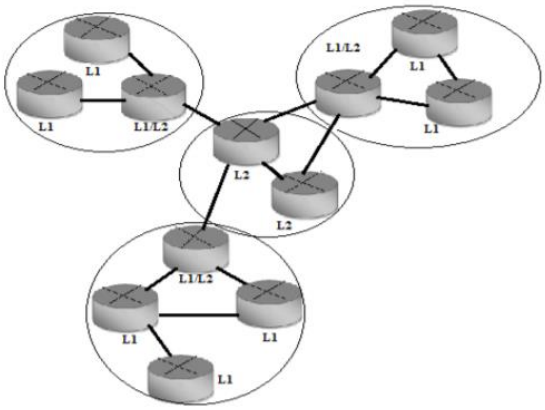


Figure 7. IS-IS implementation [25]

As we have seen, IS-IS and OSPF protocols use similar mechanisms to achieve full network knowledge. Besides, they both use Dijkstra's algorithm (Section 2.4.1) to calculate the best route through the network for a given packet.

Some of the differences between OSPF and IS-IS are based on different engineering points of views. Such as OSPF packet is transmitted with an IP data packet header (works on Network layer, OSI layer 3); whereas an IS-IS packet is transmitted directly on top of the data link layer (OSI layer 2) [29].

We can conclude that OSPF is a more optimised protocol, while IS-IS uses fewer calculations between intra-area routers [29].

2.3.4. Comparison between Distance vector and Link-state

In Figure 4, we can see an overview of the classification of the dynamic routing protocols. As it is mention, there are two different families of routing protocols in IGP protocols: Distance vector and Link-state routing protocols. Distance vector uses Bellman-Ford algorithm, while link-state uses the shortest path first algorithm.

Link-state routing protocols essentially know the entire network and every router has the same view of the network. In contrast, Distance vector only knows information about its directly connected neighbours and the cost of networks advertised by those.

Distance vector routing protocols are more simple than link-state routing protocols. They require routers to inform periodically about its neighbour’s topology changes. As a result, they have less computational complexity, but poor convergence properties. We use convergence to refer to the process of routers updating their routing tables and deciding which are the optimal routes. Due to this Distance vector require less management.

However, a significant problem to address in Distance vector protocols is the creation of loops. If they do not have a predefined maximum hop count, the algorithm will keep counting. This problem is known as count to infinity problem. Because of this maximum hop count, it is said that they do not scale well and, consequently, it is said that distance vector protocols are efficient for small networks.

In contrast, routers using link-state protocols require a router to inform all the nodes in a network of topology changes and need to carry out shortest path calculations. The convergence time is

faster in Link-state protocols and that is why link-state algorithms are suitable for large routing domains. As a result, they require more processing power and memory. That is the reason why Link-state needs higher infrastructure requirements.

We can easily configure distance vector, but the configuration of link-state is advanced and complex. However, link-state protocols have better performance in large networks.

The world has pretty much converged on link-state protocols as the modern IGP architecture for large IP infrastructure networks are divided into network areas.

2.4. Algorithms

In the most general sense, an algorithm is a system that is capable of solving a problem. In this case, the problem consists of finding the shortest path from a single source to multiple destinations.

The search for the optimal path from a source point to a target point is a common problem discussed in graph theory. A graph is composed of a set of nodes and edges, where a circle represents a node and an edge is represented by a black line (Figure 8). The graph must be connected; this means each edge connects two or more nodes creating a network.

We need to keep in mind that in this project one graph represents a network topology where links interconnect the routers.

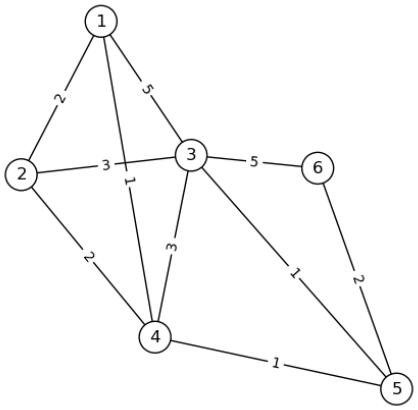


Figure 8. Graph formed by nodes and edges

The objective of the following algorithms is to find the best available route to perform routing and consequently be able to send messages through the network.

2.4.1. Dijkstra algorithm

Dijkstra's algorithm permits to find the shortest paths from a single source to multiple destinations.

This algorithm uses accumulated costs along each path, from source to destination, to determine the total cost of a route [30]. So, it can choose the least cost path value in each iteration.

For instance, consider the graph shown in Figure 8 and let's calculate the shortest path from source node 1 to other nodes of the network. The results of the Dijkstra algorithm are presented in Figure 9.

```
Number of nodes: 6
Number of edges: 10

> Dijkstra algorithm:

 src_node = 1
    Node id     Shortest path       Cost
    2           [1, 2]              1
    3           [1, 4, 5, 3]        3
    4           [1, 4]              1
    5           [1, 4, 5]           2
    6           [1, 4, 5, 6]        3
```

Figure 9. Dijkstra algorithm results

First, the algorithm calculates all available paths to a certain destination. Remember that each alternative path has a total cost, calculated by summarising all the costs between nodes through an iterative process until the destination node is reached.

Finally, the Dijkstra algorithm compares the total cost of all the alternative paths and determines the most optimal path from source node to destination node. Using this algorithm, we can figure out the route to any node within the network.

2.4.2. Steiner tree

The Steiner problem in graphs consists of finding a set of edges with a minimum cost which connects a given subset of points in a weighted graph.

The Steiner tree algorithm manages to calculate the subtree that connects a root node to every node set as a destination in an edge-weighted graph. Hence the solution to the Steiner problem is always a tree, where its root is the source node [31].

The Steiner tree problem in graphs includes the combination of the shortest path problem and the minimum spanning tree problem. The difference between Steiner tree and spanning tree is that in the first one we are allowed to select intermediate connection nodes to reduce the cost of the tree, while in the second we can only select final node destinations as subtree nodes.

For instance, let's consider the graph shown in Figure 8 and calculate the Steiner tree for the node 1. The Steiner tree will span through the nodes, looking for the destination ones and passing through intermediate nodes [32].

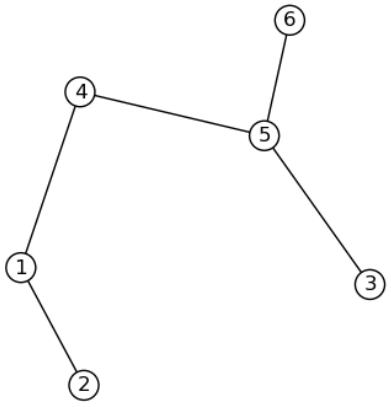


Figure 10. Steiner tree for node 1

In summary, the Steiner tree algorithm can be utilised in any situation where the objective is to minimise the cost of connections among the graph to reach some destination nodes.

We must mention that there exist other heuristic approximation algorithms, which even if they do not calculate the best shortest path to a target, they get an approximate and less time-consuming solution of the shortest path using fewer calculations. Due to optimisation of the processing of the shortest path is not in our scope, we are going to skip them.

3. Approach

In the previous chapter, we have reviewed the context in which this project takes place. Here we present the approach of the project, where we mention challenges related to routing protocols in geocasting environment, design considerations and assumptions.

3.1. Challenges

In this section, we bring up the main challenges encountered in the geocasting routing protocol field based on previous proposals (Section 2.2.2).

The main objective of the project is to set the basis to design a scalable Internet-Wide Geocasting. In order to achieve this goal, first following challenges need to be solved:

- Efficiently address any geographic area in the world.
- Design a Geocast routing and forwarding solution that requires minimum changes to existing routing infrastructures.
- Accuracy to reach all the routers that belong to the same geographical region.
- Scalability to deploy an Internet-wide geocasting solution.

This overview will help us to identify the main issues faced by routing protocols in the geocasting domain. In the next section, we will consider the described challenges to make the decisions concerning the design of the routing protocol.

3.2. Design considerations

In the previous section, we have reviewed the main difficulties of geocasting routing protocols. Next, we will discuss the design considerations in relation to those problems.

- Efficiently address any geographic area in the world.

This project is based on the Efficient Geographical addressing scheme (seen in Chapter 2.2.3). This addressing method can be used to efficiently address arbitrary areas with low computational cost.

Routers that belong to the same geographical area or coverage area can be determined by a single address, which allows using a destination address for a packet and as a route entry for routers [1].

With this system, we can create an IPv6 address that corresponds to a specific geographic area. This way, geographical information is implicit in the addressing scheme and packets can be forwarded to a target geographical area. It is worth to mention that it is compatible with the actual IP network because the binary representation of the resulting address fits in the IPv6 header.

This method allows addressing of any device in a specific region with enough precision in any location in the word. Since we are on a fixed infrastructure in this project, all devices are supposed to be routers (or nodes).

Another advantage of this addressing solution is that it can operate without a predefined set of possible destination areas, avoiding the subscription of devices to particular groups.

- Design a Geocast routing and forwarding solution that requires minimum changes to existing routing infrastructures.

One of the premises of this project is to introduce minimum changes to existing routing infrastructures. The idea is to take advantage of the IP based infrastructure to deploy a solution that enables geocasting capabilities in the network.

Although we have reviewed other approaches of geocasting systems in the background (Section 2.2.2), most of them introduce changes in the IP layer or add extensions to existing infrastructure protocols.

Based on those premises, the goal is to design a distributed routing and forwarding mechanism that is able to deliver geocasting packets through the network. This protocol needs to respond to the following issues regarding geocasting aspects, which are presented below.

The primary issue is to figure out how two different routers can know that a packet had been delivered to both nodes. We propose three possible solutions: pruning, tracking or predefine the path in each packet.

In the first solution, if a node receives a duplicate packet it will prune it. The main problem here is that the networks need to be flooded with a large number of packets to assure the delivery of the messages.

The second approach consists of tracking the packets through the network using a global controller. This implies an enormous amount of data exchanged in the network for signalling purposes. A notable disadvantage of this solution is the lack of feedback mechanism to the source node.

The last solution refers to the approach of precalculating the optimal route that the packet needs to follow in order to reach all the destination nodes in the network. The biggest problem is that if the path is too large, it will not be possible to carry relevant data to destination nodes. Nevertheless, we have decided to work on this approach during this project.

Also, it is necessary to establish a method to perform lookup and translation of target area to IP address. Route lookup can be carried out based on the prefix matching explained on the Efficient Geocasting Addressing scheme [1].

Nodes will need to know which geo address belongs to each node. Hence, we need to think about a method to keep this relevant information available to each node. In addition, an exchange mechanism will be required too, to exchange destination area status information.

In a network with these characteristics, loops are considered a major problem because they can cause protocol malfunctions. In order to avoid the formation of loops, we are going to develop a routing solution with full network knowledge.

Full network knowledge uses the known position of destination nodes and intermediate nodes in the network to enable routing and forwarding of packets over the networks. Therefore, the possibility that routing loops occur disappears in link-state routing, while each router maintains a complete and synchronised view of the network's topology.

Another critical aspect is to design a system that rapidly adapts to link failures and node disconnections. This could be achieved with a link-state routing protocol that only reacts when needed. However, to maintain full network knowledge, it will need to exchange update information between nodes and this demands greater processing requirements.

For those reasons and after comparing link-state and distance vector protocols (Section 2.3.4), it has been decided to develop an adaptation of the link-state protocol. Concretely, we are using the OSPF protocol as a basis to develop a link-state routing protocol for Geocasting.

- Accuracy to reach all the routers that belong to the same geographical region.

Geocast routing protocol needs to deliver the packet to all vehicles within a geographic area. Therefore, it is important that all nodes in a particular area receive a copy of the packet.

One of the main problems is that all the destination nodes that are part of a geographical area need to be reached, regardless their position inside the network. This is a major problem because the actual routers do not care about physical regions.

In this project, we propose a link-state protocol that reaches several routers that are in the same geographical region and they do not necessarily need to be directly connected. The final goal is to introduce a geocasting routing system that assures reliable communications between nodes.

- Scalability to deploy an Internet-wide geocasting solution.

A scalability problem appears when the number of regions starts to grow. It is possible to achieve global coverage if the network is divided into small areas like OSPF. From this perspective, the protocol will be capable of supporting large traffic volumes of packets and a large number of active devices.

The point is to use area aggregation in order to deploy Internet-wide geocasting. As a consequence, it would be necessary to define how sub-networks are connected and how the election of border gateway nodes is performed.

Border gateways are responsible for communicating different networks. The election criteria of border gateways are important because they are the responsible nodes to pass one packet from one subnetwork to another.

Hence, it is important to define how networks could be connected between them and define the criteria to select the border gateway nodes of the network.

However, this part is out of the scope of the project. In this project, we are going to focus on the single area routing protocol due to time constraints. The main objective is to define how routing and forwarding mechanism function inside a geocasting network.

To summarise, the purpose of the proposed protocol is to extend the capabilities of the current Internet-wide network by adding a new geographic routing protocol to the current IP suite of protocols.

3.3. Assumptions

In this section, we are going to outline the assumptions considered during the project.

- All nodes are part of the same network.

In our network model, we will assume that all nodes are part of the same network. This means that every node is able to reach any node belonging to the network.

We assume that the network is already running some sort of OSPF type protocol. So, each node knows how to identify and reach other nodes inside the network. If the destination node is not reachable from the source node, we will consider that there does not exist a path between the source and destination nodes. Therefore, in this situation source node is not able to calculate the shortest path from source to destination.

- Destination geoaddress is included in each message.

Each message received by a node will contain a destination geoaddress in IPv6 format. This permits to address all the possible destination regions on the earth by using 18 levels. This way, if a link failure occurs during the sending process, the intermediate node can recalculate the path to the remaining destination nodes using the destination geoaddress attached to the message.

Source unicast address will be used to identify the source node from which the message is delivered. This solution is way less complicated than introducing source geoaddress data in each message.

- Full duplex links between two or more nodes.

We will assume that each link connecting two or more nodes is bidirectional. That is to say; the transmissions between two nodes works equally well in both directions. So, a packet does not need to respect the direction of its edges (undirected). This is a fundamental assumption because the existence of unidirectional links in real networks is a real challenge for routing protocols.

- Full knowledge of the network

The proposed routing protocol is based on a full knowledge network. In other words, each node is aware of where the other nodes are and can calculate the optimal path to reach those nodes.

Each node has a coverage database that works as a map of all the network. The coverage database contains data about the location of the nodes and links of the network. Also, it contains the list of nodes and the associated geoaddresses.

By using this data, each node can perform an SPF algorithm to calculate the optimal path to reach any given destination. This is fundamental to succeed in the routing process.

- Each node knows where it is and the region it covers

We assume that each node knows its current location precisely. The coverage area of each node is given in the coverage address associated with each node. With this coverage address, a router knows its location and what area it can cover.

During the project, we will consider that each of the nodes is aware of their physical location and the region it covers. This means that each node knows its GPS coordinates (latitude and longitude). With this coordinates as input data, a node is able to calculate its own geoaddress using an algorithm. Also, as we mentioned before, a node knows how to reach other nodes and furthermore knows what region it covers.

- Rectangle polygons represent geographical regions.

During this project, we are going to assume that the world map surface is divided in regular polygons; as it is explained in Section 2.2.3. Concretely, these polygons will be rectangles.

If the number of vertices in a polygon is increased, the complexity is directly affected. Hence increasing the amount of information needed to define a polygon and causing significant overhead when carrying this data in a packet [11].

- In Fixed infrastructures, the topology of the network maintains stable most of the time.

In Fixed infrastructure networks, the routers stay at their place and the topology is relatively static. It is true that wired networks are relatively static compared to the Ad-hoc environments. So, we assume that nodes do not move, they only change their links' states from time to time.

We consider that the delivery of a packet from a source to multipoint destinations using multiple hops is possible due to network characteristics. These destination nodes could be directly connected or not; it depends on the distribution of network topology.

Once we had summarised several important properties and assumptions of the project, we will explain the design of the link-state protocol in the next chapter.

3.4. Simulation tool

Another objective of the project is to evaluate the performance of the proposed protocol. In consequence, we need to choose a simulation tool that allows us to test the protocol in a wide variety of scenarios. In this section, we will describe the available network simulation tools and three alternative solutions to validate the proposed routing protocol in this project.

- Discrete event simulator

One of the alternatives is to use a discrete event simulator to implement the proposed routing protocol. Network simulators try to model the real-world networks. They allow to model a network and modify attributes to analyse how the network behaves under different configurations. This process of model modification is way cheaper than implementing a complete real network [33].

There are many network simulators that have different features in various aspects. NS2 / NS3 [33] and OMNET++ [34] are both open source C++ based discrete event network simulators that are commonly used in the research field.

NS-2 / NS-3 [33] are object-oriented discrete event simulators that are written in C++. The core of NS-3 is written in C++ and it comes with Python scripting interface. Both are designed as a set of modular libraries that can be combined with each other.

One idea to evaluate the design of the protocol will be to implement OSPF v3 with Geocasting capabilities in a NS-2 environment.

OMNet++ [34] is another important network simulator which has a very powerful graphical interface and modular core design. Due to its flexibility, a possible solution could be the implementation of OSPFv2 in OMNET++ and implement on our own IPv6 capabilities to run geocasting capabilities.

- Network emulator

Another alternative is to use a virtual network emulator, where the network is simulated in order to observe its performance or to predict the impact of possible changes, or optimisations. Two examples of network emulator are GNS3 [35] and Mininet [36].

Global Network Simulator (GNS3) [35] is a graphical network emulator that allows the design complex network topologies.

Mininet [36] deploys an instant virtual network running in an environment that simulates the sending of messages between different devices. Mininet uses process-based virtualization to run many hosts and switches on a single OS kernel.

GNS3 and Mininet do not take the place of real routers, but they are meant to be a tool for learning and testing in a lab environment. On top of that, these tools permit researchers to programme and test any algorithm. Due to network emulator characteristics, we would have to implement the adapted link-state protocol from the ground in order to run it in GNS3 or Mininet.

- Direct and abstract simulation of the algorithms

The last alternative is to use a scripting and prototyping environment to implement the routing protocol. In this approach, we could programme a simulation environment that interprets the network as a graph. Routers and links will be represented as nodes and edges respectively.

A direct and abstract simulation of the algorithm, implementing basic OSPF functionalities on a Matlab or Python script. The results obtained from the simulation will strongly depend on the codification of the simulation functions.

Matlab is a cross-platform mathematical software tool that offers an integrated development environment for all types of calculations and mathematical models. It uses its own high-level language (M language) that allows the implementation and creation of user interfaces. Matlab is a perfect software for the development and testing of algorithms, due to the great capacity of matrix calculation that it offers.

Python is an interpreted programming language that supports object orientation and functional programming. This allows Python to be a multiplatform interpreted language. The main advantage over the other options is that Python has open source and GNU license. In addition, Networkx and Matplotlib libraries add support and functions to perform a study of complex networks.

After having analysed the alternatives of simulation tools to perform the evaluation of our project the best solution is to develop a simulation environment in Python.

In this case, the decision to use the Python programming language is justified by the ability to program tasks using scripts and the fluidity of the environment when executing and visualising

the models. Due to time constraints and the learning curve of this simulator, it has been decided to implement a comprehensive simulator that simulates the behaviour of a network in an abstract view. In other words, this means that the network will be treated as a graph problem and the nodes conforming the network will represent the routers and the edges the link connecting two or more routers.

4. Design

This chapter describes the design of the link-state routing protocol for an Internet-Wide Geocasting environment.

As we have seen in previous chapters, the concept of Geocasting is defined as the delivery of a message to nodes within a geographical region [13]. In other words, Geocasting is the provision of information to a group of nodes in a geographic area, where some data packets are sent from a source node to multiple destination nodes.

The proposed protocol is based on the link-state algorithm. It fulfils the requirements of geocasting in fixed infrastructure networks. The key concept used in the protocol is that every node in the network has full network knowledge. Therefore, every node knows where the other nodes location is and can calculate the best path to reach them.

This link-state routing protocol reduces the message overhead compared to distance vector algorithms, where every node will need to recalculate the path and retransmit each message when it receives the first copy of the message.

4.1. Network infrastructure model

For this project, a network model for a fixed infrastructure network scenario has been created.

The scenario in Figure 11 shows a source node, which is in a certain location and is connected to destination routers via the Internet. The packets generated by the source node are routed through the Internet using the destination geoaddress. A routing algorithm is used to forward the packets to the destination area. Once the packets reach the destination routers, they are broadcast to all vehicles inside the destination area.

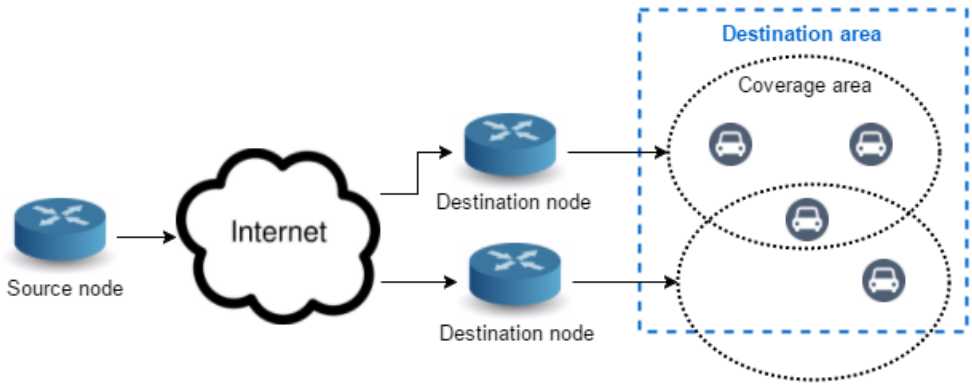


Figure 11. Internet-wide geocasting scenario

In the following subsections, there are described the elements taking part in the network infrastructure model. The network model is composed of three main elements: nodes, links and packets.

4.1.1. Nodes

A node is a router that makes conventional use of the network by interconnecting to other nodes. It is also known as router or georouter in the geocasting field.

A node has the ability to understand and manage the delivery of geocasting packets. Also, it can calculate the best routing path of a geocast packet for a given destination geoaddress.

As we are describing a link-state protocol, we assume each node has full network knowledge. This information is necessary in order to create and maintain an updated topology map of the network. This means that each node belonging to the network knows the information about the location of other nodes and how these nodes are connected between them.

With this data, a node can calculate its coverage region using latitude and longitude coordinates. Then, this information will be exchanged with neighbouring nodes. At the same time, these neighbour nodes will distribute the update information until all the nodes of the network have updated their routing tables.

As a result, a node, independently and with the available information in routing tables, can decide the best routing path for a given destination. So, a node with geocasting capabilities is able to perform the following functions:

- Calculate its coverage region.
- Calculate the best routing path for a given destination address.
- Exchange link-state information with its direct neighbours.

It is worth to mention that nodes do not have a predefined geo address. It makes no sense for a router to have a predefined geo address because it could move or there could be another router on top of it. Instead, routers have a coverage area. This coverage area is only useful for geocasting communications and it can exist together with the unicast addressing scheme. Hence, there is no unicast replacement in the network.

At this point, we need to make a design decision about where the coverage database information should be. There are two possible solutions. The first solution implies that each router needs to have a copy of coverage database regarding coverage area information. The other solution consists in some of the nodes are selected to have the coverage database information and the rest of the nodes must request on demand information every time they need to send a geocast packet.

We must take into consideration that nodes, which have full network knowledge information, are the only ones able to calculate the shortest path for the others.

The problem with the second approach is that it increases the traffic load produced by the introduction of coverage data request packets. This solution concentrates all the processing load in some routers and can lead to convergence problems when link failures occur.

Due to this reasons, the final decision is to include a copy of coverage database and topology database in all the routers. This way every node has the ability to calculate the shortest path to

each destination node and the processing load is distributed through all the nodes of the network.

In addition, we have to mention that a node can take three different roles (depending on the situation) during the routing process: source node, intermediate node or destination node. The particularities of each type will be explained in Section 4.2.2 (Routing protocol).

Once there have been reviewed the general functions of the nodes, it is time to present the attributes of a node:

- **Node id:**

Node-id represents a unique number that identifies the node in the network.

- **Latitude, Longitude:**

They are the GPS coordinates of the node. They are required for calculating the coverage address associated with each node. Also, this GPS coordinates are useful to represent the physical location of a node in a graph.

- **Coverage address / area:**

It is an address that specifies the area that routers serve, not necessarily the area the router is in. Because the router that handles the traffic of, for instance, a building could be located in another building.

Nevertheless, each node can calculate its coverage address by using the latitude and longitude as input attributes. Each node can calculate its related coverage area with this coordinates and add it to the node list. Alternatively, the coverage address can be manually configured.

These nodes are able to communicate and send messages between each other by following the routing algorithm explained in Section 4.2.2. As we said before, this routing algorithm is based on full network knowledge information.

Each node maintains some routing tables which allows it to route the packets for other destination in the network. These routing tables contain link and node status information of the network. This full network knowledge information is reflected in the following routing tables:

- **Neighbour adjacency table:**

The neighbour table stores information about neighbours that are directly connected to the node. The relation between neighbour nodes is dynamically established by the periodical exchange of hello packets (explained in Section 4.2.3).

In the neighbour table, we can find the router-id, interface, current link-state and link cost data. Through this neighbour table, the node is capable of realising when link-state changes happen.

If there is a link-state change, the node updates neighbour table information and creates a LSP with the new updated data. Later, this update packet is broadcast through the network in order to maintain full network knowledge in every router.

- **Routing table:**

The routing table contains the shortest path to each node of the table. The stored paths in this table are the optimal paths considered by the Dijkstra algorithm.

A router will have a routing table entry for each node in the network. Each entry contains information of the node identification, the shortest path to reach it and the associated cost.

These entries do not have an expiration time. They will be useful unless a link-state change happens or a node abandons the network. Therefore, if any of neighbour or topology tables are modified, affected node shortest paths are recalculated to update route information.

In that case, when the update packet is received the node will calculate the shortest path to the nodes affected by this change in order to refresh the table entries. This is not a big deal if we assume that the network would maintain most of the time stable, without nodes leaving and entering coverage regions every instant.

The point of having this routing table is to prevent the calculation of each shortest path to each target node every time a node needs to send a geocasting packet. Using this table the shortest path to each calculation is already computed. Therefore, when there is a topology change, only affected nodes will need to calculate the shortest path, instead of all of them.

Hence, using this routing table the lookup time to calculate the optimised route to reach nodes inside a destination region decreases and processing load is reduced.

- **Coverage database / table:**

Coverage database is included because of the need to convert the destination coverage address to actual destination nodes.

The coverage database stores the mapping between destination geo address (IPv6 address that corresponds to a geographic region) and the node it belongs to (node id). This information needs to be dynamically maintained and be up to date.

As we have mentioned before, each node has a copy of the coverage database. In the coverage database, there is an entry for each node in the network. An entry contains node identification, unicast address and the respective coverage address of the node. The coverage address is a geographical scheme address that is related to certain geographic regions.

Using a search tree data structure would allow the routing algorithm to find the desired information in a faster way. However, if a change occurs and the coverage database needs to be updated, the complexity increases because of the need to maintain a search tree data structure table updated.

When new nodes are identified or when mapping between destination geo address and the associated node change, the coverage database will be updated. We are going to explain the details of this update mechanism in Section 4.2.3.

- **Topology database / table:**

A topological database is essentially an overall picture of networks about routers [37].

The topological database contains the collection of LSPs. This LSPs are received from other nodes within the same geographical area. Because routers within the same area share the same information, they have identical topological databases.

This table can be defined as a map of the actual network topology. It contains all the possible routing information and all the paths. Topology table is used to find the best paths to each target node.

4.1.2. Links

A link is a communication path between two neighbour nodes. It is defined as the physical and logical network component used to interconnect nodes in a network.

The links that connect two or more nodes had an associated cost value. The cost of links, also known as metrics, is assumed to be known by all nodes when they run the Dijkstra's shortest path algorithm.

There are many ways to assign a cost to a link. This metric can be calculated considering some parameters such as bandwidth, traffic congestion, transmission and throughput.

These metrics are dynamic and change over time. Because of this features, nodes should send update packets when the change in the metrics happen. In this project, we will not use dynamic metrics because otherwise, it will increase the complexity of the network. So, the analysis of how the metric affects to the selection of the path is out of the scope of this project. Hence, the metric used is to assign a cost of 1 to each one of the symmetric links that connect two nodes.

In order to simplify the network model, we assume that the links between nodes are full duplex. In other words, the transmission between nodes can be carried out simultaneously in both directions.

It is worth to mention that in a link-state algorithm, each router needs to broadcast a link-state message to all other routers in the network.

4.1.3. Packets

A packet is a simple data object that represents a message that needs to be delivered from source to some destinations in a certain network. In this project scenarios, the geocast packets are generated by the source nodes.

A packet includes the following attributes:

- **Packet id:** This packet id is a unique identifier of the packet.
- **Type:** There are three types of packets:
    - *Geocasting packets* carry useful data from a source to multiple destination nodes inside a geographic region.
    - *Hello packets* are used to establish and maintain relations between adjacent nodes.
    - *LSP* describes the contents of the topological database.
- **Router id or Source unicast address:** Router id sequence or source unicast address is used to identify the source node from which the message had been created.
  The included source address is a regular unicast address, already assigned by common routing.
- **Size:** The size specifies the packet's length in bytes.
- **Data:** This attribute contains useful data that carries the packet.

From now on, there have been added some attributes that are necessary to accomplish geocasting.

- **Destination geoAddress:** The geoaddress or IPv6 destination address is carried in the packet header. It is useful for identifying all the destination nodes associated with a geographical area. So, the forwarding decision is based on the information included in this field.
  As it is determined in the Efficient Geographical addressing scheme [1], a depth of 18 levels in an IPv6 geoaddress can accurately cover most areas in the world map. A rectangle of level 18 can be encoded into 72 bits, leaving 56 bits unused for other information in the address.

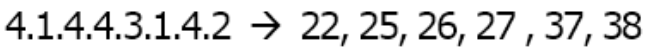


Figure 12. Conversion from destination geoaddress to nodes within that geographical area

- **Forwarding tree / Path:** Forwarding tree is a path encoded or included in the packet. This is a path calculated using the Dijkstra algorithm. It includes nodes that the packet has to pass through to reach all the destination nodes.
  A node uses IPv6 type destination geoaddress to calculate destination nodes and the shortest path to reach them. This shortest path includes both destination nodes and intermediate nodes for the routing decisions.
  Shortest path will be carried in IP header extension. At this point, we need to have in mind that standard Internet routers do not process the IP header extension of the packet.
  An alternative solution to this problem is to carry this information in the application layer header. In this case, the lookup speed of the routers will be affected.
  One way or another, the optimal route is recorded in the packet and, therefore, the size of the packet will be increased in proportion to the number of nodes that compose this shortest path.

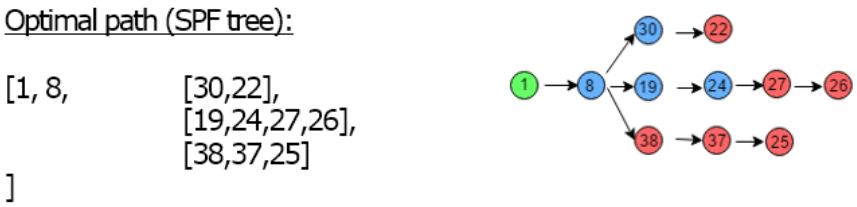


Figure 13. Representation of the calculated forwarding tree

This additional information included in the packet is necessary because if a problem occurs and a path is no longer available, any router can recalculate an alternative path to reach all the nodes in the destination area based on that extra data.

4.2. Link-state protocol

We can say that the routing protocol has two main functions; select routes involving source-destination nodes and deliver messages to their correct destination.

Link-state routing protocols are being increasingly used in modern communication networks. A notable feature of this class of routing protocols is that network connectivity and state

information of all links are available to nodes for making routing decisions. Two main components of a link-state routing protocol are the routing algorithm and the update mechanism These components must be properly designed for efficient routing [38].

One of the advantages of a link-state routing protocol is that it can operate in a distributed manner on different routing infrastructures. Being able to extract required routing information from a previously deployed unicast routing protocol.

In contrast, link-state routing requires all routers to know about the distribution of other routers in the network. To do so link-state information is flooded through all over the network area, to ensure all nodes have a synchronised copy of the area's link-state database. From this shared database, each router constructs its relative shortest-path tree for all known routes.

We assume that a node knows the state of its links with other nodes. Hence, we can conclude that a node will have enough information to calculate the best path to any destination.

The basic idea is that the nodes disseminate the information they have (the state of the links with their direct neighbours). When this information is available for each node, it is capable of building a network map and choosing the best path to each destination. This is a sufficient condition to find the best route.

Link-state protocols are based on two mechanisms to achieve efficient routing: reliable dissemination of link-state information and calculation of the shortest path based on accumulated information.

4.2.1. Initialization

First of all, every node of the network needs to have full network knowledge. Without this information, nodes are not capable of routing geocasting packets efficiently.

Full network knowledge consists of nodes having updated tables with network topology information and data related to coverage region of all nodes.

In order to achieve that, node tables need to be updated and synchronised. This means that nodes need to share node coverage area information, topology information and link-state information between them.

When the protocol starts working routing tables of each node are empty. In order to fulfil them, every node performs the following steps:

1. Calculate coverage address of a node based on GPS coordinates of the router. As the result of this calculation may not match the desired coverage area with its real geographical position, node coverage address can be manually configured to serve a specific geographical area.
2. After that, a node will establish a relation with its neighbour routers to enable the exchange of link-state information. We consider neighbour routers to nodes that are directly connected. The shared information is about link-state and the associated characteristics of the path, such as path cost.
3. Once link-state information has been exchanged between neighbour nodes, a node is able to create a LSP (Link-state Packet) that contains that information.
4. Finally, a node floods created LSP through all the network to propagate link-state information.

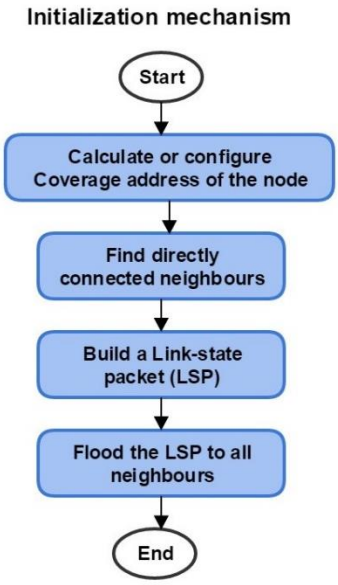


Figure 14. Initialization mechanism

If a new node joins a network, it will repeat the previous steps in order to gain full network knowledge and to advertise its presence to the rest of the network.

The neighbour nodes are responsible for providing updated information about topology of the network. They send update packets containing topology database information of the network. This process will be repeated until the new node has all the information about topology database and therefore the full network knowledge (rest of the nodes location and link states).

Once the information of the topology database is completed, the new node will be aware of the map of the network. After that, it will flood an update packet containing information about each coverage region. This way the rest of the routers would be able to incorporate the new entry to their coverage database. At the same time, the coverage database of the remainder nodes will be updated.

4.2.2. Routing protocol

In this section, it is described how routers in the same network can communicate with each other to deliver a geocasting packet from a source node to multiple target routers.

It is worth to mention that this routing mechanism is based on the distribution of nodes in the topology. Any node can act as a traffic source and perform routing and forwarding functions.

As mentioned in the previous section (Section 4.1.1, Nodes), nodes can take three different roles that are called: source node, intermediate node and destination node. Depending on the role they follow different algorithms to deliver packets to destination region routers.

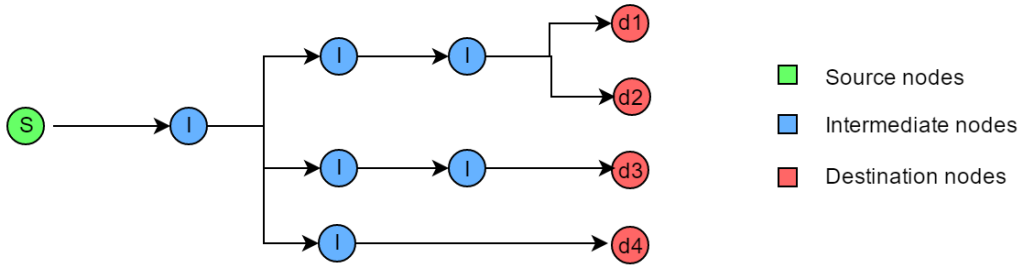


Figure 15. Types of nodes

As it is shown in Figure 15, any node regardless of its type is able to forward a copy of the packet to multiple nodes. This is specified by the predefined path that is encapsulated in the packet.

- **Source node:**

The source node is responsible for identifying nodes that belong to a certain destination, calculating the optimal shortest paths to reach those destinations and forwarding the packet to the next intermediate routers. Hence, the source node is the router where the highest processing load takes place.

By using full network knowledge, source nodes placed anywhere on the Internet can calculate the shortest path using the position of the destination node(s) and the position of intermediate node(s). With this information, the rest of routing devices (intermediate nodes and destination nodes) can make routing decisions based on the shortest path included in the packet.

In source routing, each packet carries a complete path that is used to make routing and forwarding decisions. The main advantage of this approach is that routing loops are avoided. This is an important aspect to take into account because it guarantees that a packet will reach all the destination nodes. In this ideal case, we are assuming that there is no link failure in the calculated path.

On the contrary, a significant disadvantage is that each packet needs to carry a predefined path. This requires an overhead to store the shortest path. In a large network, where the destination nodes could be “far” (a large number of hops) from the source node, the predefined path could be too large to fit in the packet.

We lead to the encapsulated path solution because the alternative of sending feedback backwards to the source node creates routing loops through the entire network. Increasing the number of signalling packets.

In fact, the encapsulated path solution allows IP packets to be sent to multiple destinations at the same time, saving network bandwidth.

The routing algorithm in the source node follows the steps illustrated in Figure 16.

First, source node receives a geocasting packet, which includes a destination geoaddress that represents the target geographical region. Destination geoaddress permits to identify the destination node(s) by looking up coverage database relations.

The source node calculates the shortest path first (using Dijkstra algorithm) from source to each destination node(s).

After the shortest path to each destination are calculated we have a list of paths that the packet needs to follow to reach all targets. The algorithm looks iteratively for same nodes in order to optimise the path the packet will follow.

The route optimisation consists of an algorithm that looks iteratively for corresponding nodes in a list of shortest paths. If there is a match, the two paths will be merged. Hence, unnecessary duplicates are avoided and the list of shortest paths is reduced. We have to consider that each entry in the list is one packet more that needs to go across the network.

After route optimisation, the path is included in the packet and the source node sends copies of the packet to the next node in the route.

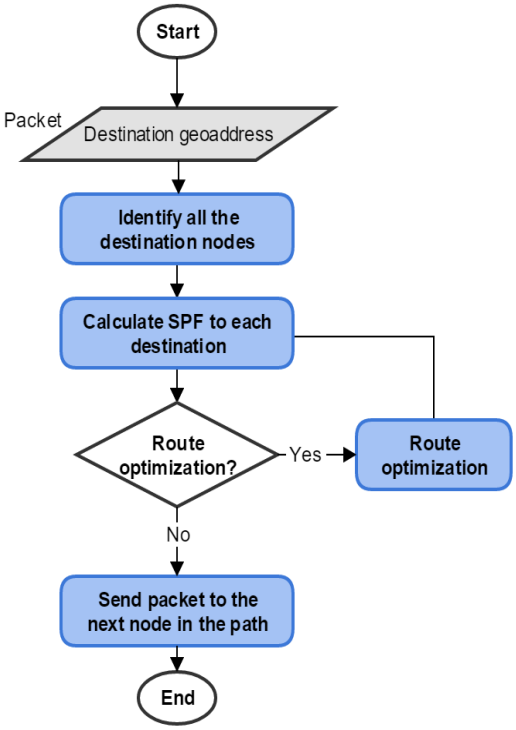


Figure 16. Source node flowchart

As it is shown in Figure 17, a graph is formed by a set of nodes and link connections. This draw is an interpretation of the path included in the geocasting packet. A green coloured node represents the source node, whereas red colour illustrates destination nodes. Intermediate nodes are nodes that the packet pass through in order to reach destination nodes and are identified by blue colour.

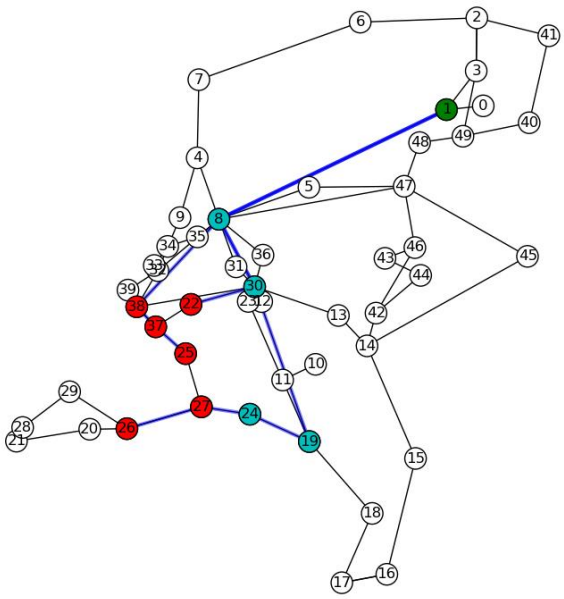


Figure 17. Calculation of shortest path to certain destination

From now on, we can say that routing and forwarding decisions are made based on the shortest path included in the geocasting packet.

- **Intermediate node:**

Intermediate nodes are the nodes in the middle between the source node and the destination nodes. These are the nodes a packet needs to pass through to reach the destination nodes.

When an intermediate router receives a packet, it reads the path attached to it. The path data includes the entire path to the destination.

At this moment, the node validates if the node is one of the desired destinations by comparing its own coverage address with the destination geoaddress of the packet.

If the comparison is positive, the node keeps the packet for himself. Otherwise, the intermediate node just reads the shortest path information that carries the packet. This route gives the information to forward the packet to the next router.

If any link failure is detected, it is important to define that any intermediate node will recalculate the shortest path to unreached destination nodes and replace the path in the packet. So, it can reach its destination(s).

Essentially, multiple routes can be used to deliver a message to a destination area, even if the precalculated path is not available anymore.

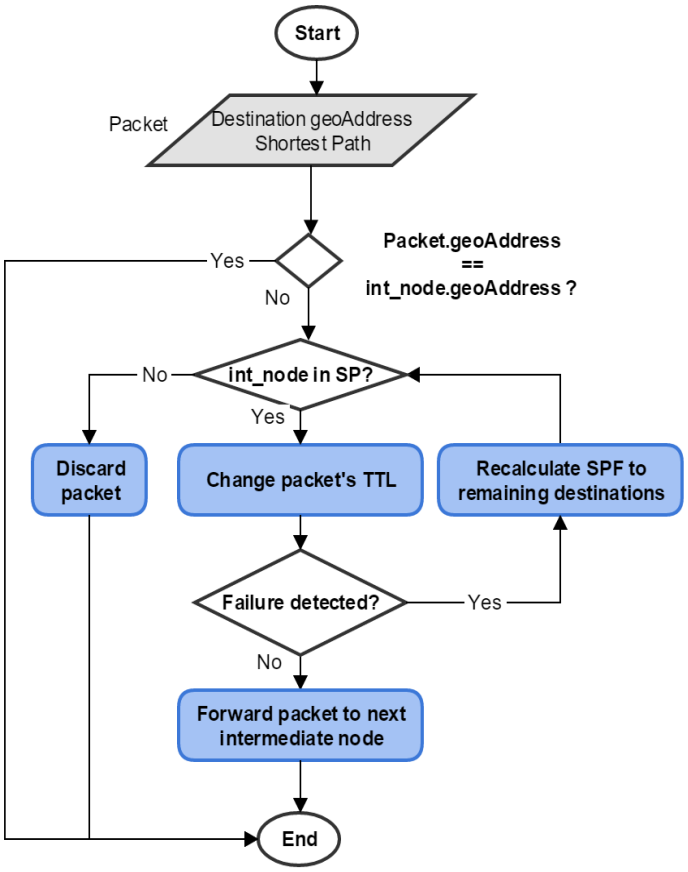


Figure 18. Intermediate node flowchart

In this architecture, the processing charge is distributed over several nodes of the network. However, the computation of a route at each router is widely based on the previous processing results made by other nodes and the predefined path sent with the packet (previously calculated by source node).

It is worth to mention that it is possible for the link-state routing algorithm to identify the origin of a route by looking to the source address of the packet. The identification of the source of the packet allows intermediate nodes to calculate alternative paths in case link failures happen. This feature ensures loop-free routes in the calculated path. Avoiding to send a message to destination nodes that previously had been reached by other copies of the packet.

After studying the behaviour of the network, we have realised that the only case in which an intermediate node can receive a packet that does not belong to itself, is if a sequence of multiple link failures happens.

It is for that reason that if an intermediate node receives a packet, it should not have received, that is to say, the identifier of the intermediate node is not in the path; the node will immediately discard the packet. Also, if necessary, it will propagate an update packet warning about multiple link failures. This way nodes are forced to exchange link-state information between them.

- **Destination node:**

Destination nodes are all the nodes belonging to a specific geographical region. As we mentioned before, this geographic area information is implicit in the coverage address assigned to each node.

In case the coverage address specifies a very concrete geographical area, it could be that only one node forms this group of destination nodes.

When a geocasting packet arrives at a destination node, it checks the path the same manner the intermediate node does. In case the forwarding tree continues, it will forward a copy of the packet to the next node. However, if the actual destination node is the end of the path; it will keep the packet for itself.

We will consider a successful communication when a packet sent from a source node reaches all the destination nodes.

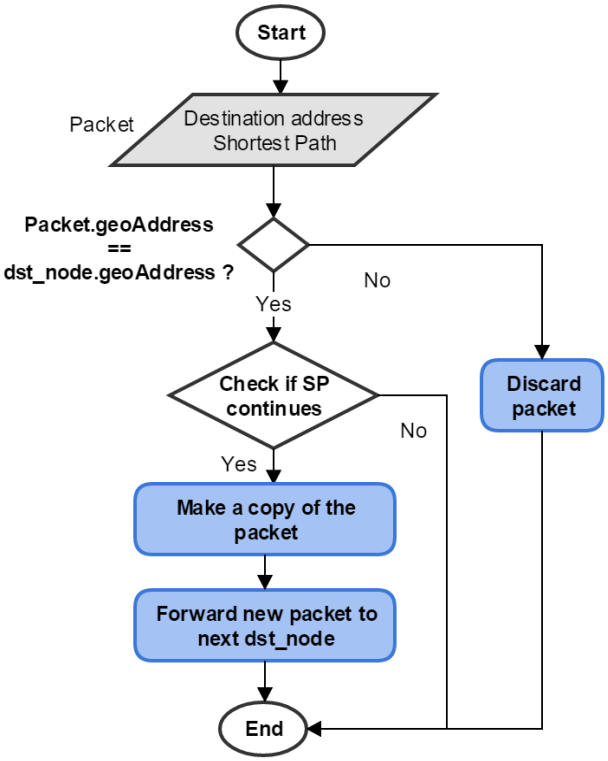


Figure 19. Destination node flowchart

With this approach, we have tried to design a routing mechanism that is capable of reaching all the nodes belonging to a certain area. Along with this section, it has been described how routers communicate with each other.

In order to prevent looping of packets when there is no path available because of multiple link failures, it has been established a fixed number of maximum recalculations of alternative paths. If this maximum number of recalculations are exceeded, the packet will be drop out.

Hence, we can conclude saying that routing is done on best effort basis. This approach avoids any feedback to the source node.

The complexity of the algorithm is in the processing and calculation of the shortest path to each destination nodes and the update mechanism to keep the routing protocol working against link dropouts.

4.2.3. Update mechanism

The update mechanism is used to maintain synchronised routing tables in order to have full network knowledge available in every node. As we point out in assumptions (Section 3.3), we assume that the network is already running a link-state routing protocol, such as OSPF.

When we apply controlled flooding mechanism, a source node sends a copy of the update packet to all its neighbours. The moment a node receives a broadcast packet, it duplicates the packet and forwards it to its directly connected routers, except for the node which it received the packet.

The network is filled with many redundant copies of the packet, due to this reason, it is guaranteed that the packet will reach all the routers.

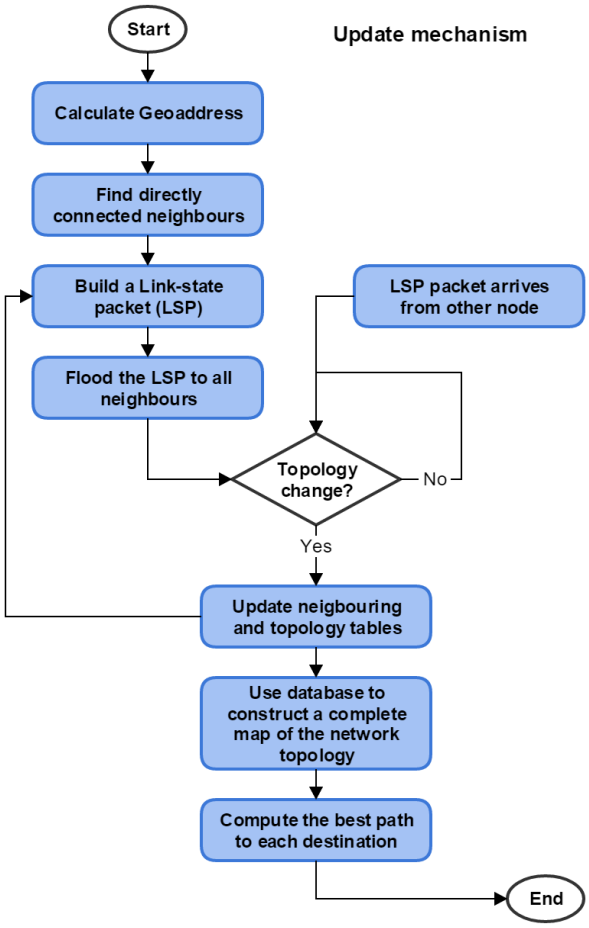


Figure 20. Update mechanism

Two main types of packets maintain the link state database aware of network updates: Hello packets and LSPs.

- **Hello packets:**

Hello packets are used to establish a relation between a node and its adjacent nodes. This relation is maintained by periodically exchanging hello messages between neighbour routers.

A Hello packet includes information about a node's neighbours and their links and maintains informed the neighbour nodes about link-state status. If a link-state change occurs or a node is disconnected, the node will immediately realise and will prepare an LSP to warn the rest of the nodes.

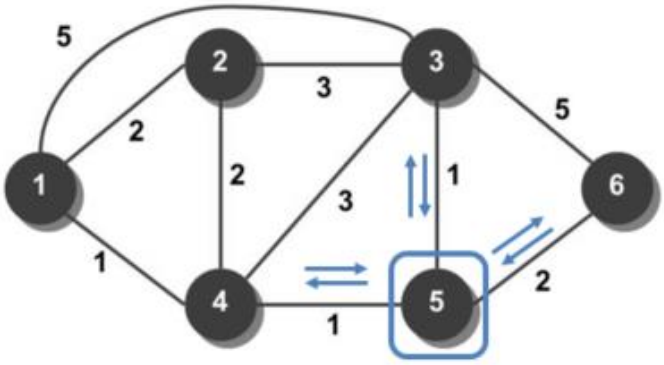


Figure 21. Hello packets exchange between node 5 and its neighbours

Regarding usable links, every link needs to be checked in both directions in order to be considered available.

- **LSPs or Update packets:**

A LSP or update packet is a link-state information packet. They are used to update topology information in topology databases of each node in the network. Essentially, the LSP describes the link-state status of the interfaces directly connected to the node.

The flooding mechanism ensures that all nodes get a copy of the link-state information. Flooding is done by forwarding the LSPs that a node receives for all its links except for the one from which it arrived. This process must continue until all information has reached all nodes in the network.

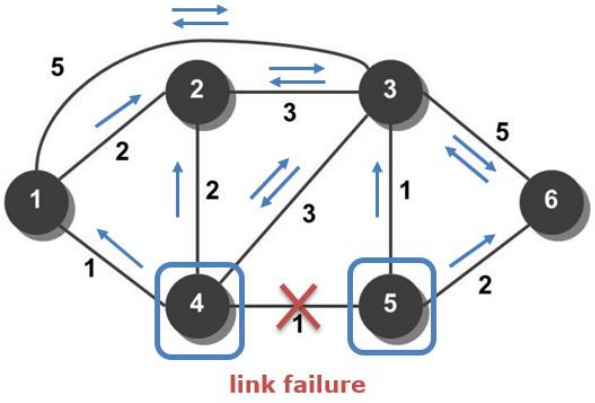


Figure 22. Flooding of LSPs when a failure happens

As we mentioned before, when a change occurs in the network, each affected node generates an LSP packet with the following information:

- *Node-id* that created the LSP.
- A *list of neighbours* (directly connected nodes) of the node, along with the *status* and *cost* of each link.
- A *sequence number* used to compare if the incoming update packet is a duplicate or not. The new update packets are stored in a cache and if a duplicate is received, it is rejected.
- *TTL* of the packet to avoid routing loops.

The first two fields are used to calculate best routes. The other two fields are used to make the flooding process reliable and ensure that all nodes receive the most recent copy of the LSP. This assures fast convergence times.

These are the fundamentals of how the mechanism works to update link-state information over all nodes. This mechanism is used in the OSPF protocol. The moment a node receives all LSP of its neighbours, it can create a network topology scheme.

4.3. Limitations

In this section, we will discuss the main limitations of the proposed routing protocol in an IPv6 environment.

In the design, we assume that the packet header contains the whole path to reach all destination nodes. The main problem here is that the packet header length would vary with the number of intermediate and destination nodes in the route. In addition, the encoding and decoding process of the packet header could originate unwanted delays and increase overall processing time.

Definitely, in large networks reaching to the point where packet paths with a large number of nodes could use a significant percentage of the overall size of the packet. For future improvements, we should keep these packet headers impact small.

When we talk about sending LSPs with network knowledge data, we are not considering that the information of this packet could exceed the maximum length of IPv6 protocol. If this happens, the packet has a considerable amount of chances to be discarded.

This is a potential limitation in this design. During the thesis, we are not considering these restrictions as an issue. Although in a real or practical implementation would affect the performance of the routing protocol.

Another possible area of improvement is the size of routing tables and the improvement of algorithms to reduce overall processing time in each router. In order to have full network knowledge in each router, we need to fill several routing tables with a list of coverage address and link-state information. There is a need to simplify destination areas to reduce routing tables entries and therefore reduce lookup time.

In our design, we have viewed the network as a collection of interconnected routers. We assume that all the routers have the same characteristics and execute the same routing algorithm to compute routing paths through the entire network. In practice, this model and its view of a homogenous set of routers is a simplified scenario.

If the number of routers increases a lot, the overhead involved in processing, storing and communicating becomes restrictive. The overhead required to broadcast link-state updates among all routers on the Internet would leave no bandwidth left for sending data packets.

We can solve this problem by dividing the network into smaller areas. By dividing the Internet-Wide into areas, we can reduce SPF graph complexity and allow route aggregation.

With the election of border gateway nodes, any network would be able to connect a network to outside networks. The border gateway routers are the nodes responsible for forwarding packets to destinations outside their own network. A border gateway would need more processing power due to it performs the calculation of SPF per area. However, this part is out of the scope of the project due to time constraints.

The approximation algorithms described in this chapter are not practical for large networks since they assume complete knowledge of the network. Such algorithms have a theoretical interest. The future study of these algorithms will contribute to a better understanding of how geocasting mechanism work.

5. Implementation

This chapter covers the implementation of this project, which design is described in Chapter 4. This implementation will result in a simulation environment that let us prove the performance of the proposed protocol with real topology networks, which motivation is to demonstrate the viability of the proposed design.

5.1. Used technologies

In this section, we mention the technical aspects concerning the development of a simulation environment to validate the proposed protocol.

- **Python 3.5**

The used programming language is Python. The main reason for choosing this has been its ability to execute it on different operating systems, as well as its high-level abstraction that allows a faster iteration through the development. Another fundamental aspect to take this decision had been developer's experience in such language.

- **NetworkX library**

NetworkX is a Python library that provides utilities for studying graphs and networks. We are using this library to represent the network and node interactions. Moreover, this library provides several data structures and algorithms for graph theory, such as Dijkstra algorithm, that are used to calculate the shortest path from source node to destination nodes.

- **Matplotlib library**

Matplotlib is a 2D and 3D plotting library for Python. It provides tools to draw figures in a variety of formats, such as general graphs, histograms and scatter diagrams. We are going to use it to draw the results of the simulation.

It must be mentioned that Numpy library for Python is going to be used along with Matplotlib. Numpy provides specific functions for matrix and vector numerical calculations.

- **GML files**

The website Topology Zoo provides us with GML files that contain information about many network topologies around the world [39].

GML (Geography Markup Language) is used to model, transport and store geographic data. In this project, GML format files found on the mentioned website contain data about routers location and links that connect them. The simulation environment uses this data as input data to create the graph that represents network topology.

```
node [
  id 0
  label "Westerbork"
  Country "Netherlands"
  Longitude 6.60833
  Internal 1
  Latitude 52.85
  type "Node"
]
```

```
node [
  id 1
  label "Dwingeloo"
  Country "Netherlands"
  Longitude 6.36944
  Internal 1
  Latitude 52.83417
  type "Node"
]
```

Figure 23. Node data in GML format

- **JSON library**

JSON (JavaScript Object Notation) is a popular independent text format used to save or read object data from text files. This format is supported in many programming languages and has a good mapping to basic data structures, such as list, dictionary, string, numerical variables and objects. Also, JSON format is fast, human readable and can be parsed in main programming languages. Due to these reasons, it is suitable for data exchange between programs.

In this project, we have used JSON format to manage (save and load) data logs generated by completed simulation sessions.

5.2. Simulation environment overview

The simulation development is based on the concepts explained in Chapter 4. Although it is a functional simulation environment; the developed software does not implement all the capabilities described in Design chapter. Notice that the results obtained from the simulation environment will strongly depend on the coded implementation features.

The implemented simulation environment is a discrete event simulator, which consists of a simulator object that executes several events in order. It has been coded following object-oriented programming structure.

The aim of this simulator is to provide an abstract view of what is happening in each iteration on the network. The main difference with other simulators is that it is turn-based instead of time-based. This means that iterations of events occur in a certain turn, while in other simulators time stamps are used to determine when an event happens.

In each turn, all nodes perform their processing and transmission tasks communicating with each other with message passing. This turn-based approach permits to develop a discrete event simulator without using multiple threads. Therefore, turns permit to implement a sequential simulator that performs the data in different phases.

The reason to follow a turn-based implementation is to keep a simple simulation environment, to reduce development time to build the simulator and to decrease the number of resources needed to run it. Hence, the implemented simulator works in a sequential mode to prevent risk and reduce the complexity of the development.

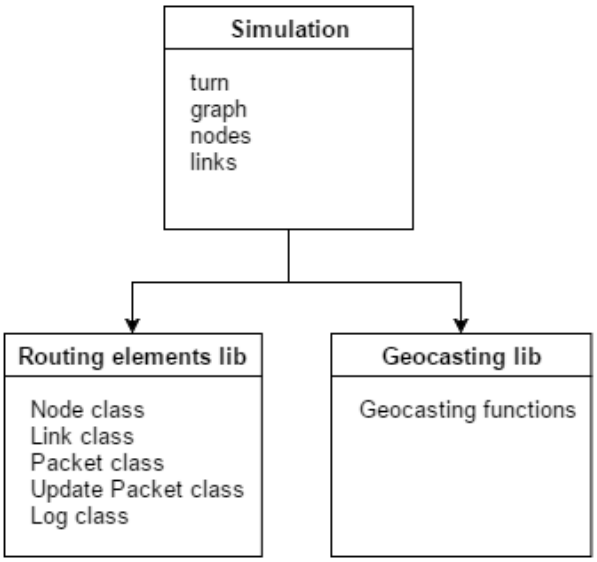


Figure 24. Simulation modules

As it is shown in Figure 24, the simulation environment is designed as a set of library modules, each of which simulates a specific part of the communication protocol during the simulation runtime. The libraries have been developed using Python version 3.5 and other libraries, that have been mentioned in Section 5.1. A simulation object makes use of other classes and functions found in “Routing_elements_lib” and “Geocasting_lib” libraries (explained in Sections 5.3 and 5.4).

The simulation environment is divided into three phases: initialization, processing and transmission. In turn zero, network topology data is imported and variables are initialized. In the following turns, nodes perform iteratively processing and transmission tasks, until all packets are delivered to destination nodes. In each turn, the nodes carry out two phases consecutively: processing and transmission.

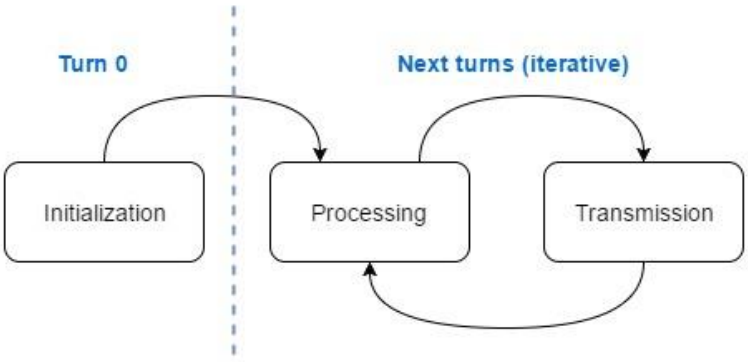


Figure 25. Simulation workflow

Let’s explain below the operations taking part in each phase.

- **Initialization phase:**

Initialization phase takes place before the simulation starts. This function initializes variables and import necessary data (such as topology graph, nodes and links) to run the simulation session.

To create a simulation session, we need to have the following attributes:

  - **Turn:** Incremental integer variable which is used to count the number of iterations.
  - **Graph:** Collection of nodes and links that form network topology, which are imported from GML files.
  - **Nodes:** List of nodes that form the network.
  - **Links:** List of links or connections between nodes in the network.
  - **Packet Stack:** List of geocasting packets that are delivered during the simulation session. Includes, the turn when each packet is meant to be introduced to the network.
  - **Link-state stack:** List of links that are going to be dropout during the session.
  - **Reached nodes:** List of reached nodes during the simulation. At the beginning this list is empty. Every time a packet reaches a destination node, both IDs (packet id and source node id) and the reached node are included in the list. When all destinations are reached, the script finishes the simulation session.
  - **Session logs:** List of logs generated during the session. These logs permit to interpret and visualise the results of the simulation in each turn.

Mentioned list objects are related to object classes from “Routing elements lib”.

After those attributes have been filled, all nodes will flood hello packets to directly connected neighbours. Then, nodes exchange update packets or LSPs for spreading link-state information through the network. Finally, with this information, every node is able to create a map of the network topology. Also, other routing tables like coverage database are fulfilled with the relation between coverage address and nodes.

Once the initialization phase is completed, in the next turns the processing and transmission task are performed sequentially in every node. At this point, the simulation environment is ready to run the routing algorithm explained in the previous chapter.

- **Processing phase:**

In the processing phase, nodes process incoming packets and calculate their optimal route.

Each node checks every packet in the inbox table and analyses it. In case a new packet arrives at a node, this is processed to calculate the optimal routing path to reach all the destination nodes.

However, if the packet has an attached path, the node behaves as an intermediate node. It looks in the packet’s path for the next hop and takes the packet to its outbox table.

The node can detect if the previously determined path is not longer valid due to some connection failure or disconnection of a node from the network. If this happens, the current node will be in charge of calculating an alternative route to be able to reach all the destination nodes.

- **Transmission phase:**

In the transmission phase, messages are transmitted from one node to another at each turn.

It must be said that this phase does not contemplate congestion of links, packet priorities, collisions or delays. Transmission is performed for all outgoing messages from a node at the same time and on the same turn. Thus, no message can be lost while it is being transmitted.

The transmission of a packet consists of taking the packet from the outgoing node of the outbox table and placing it in the inbox table of the next hop router following the destination specified in the outbox table.

Finally, messages will be exchanged between routers until all nodes receive their corresponding geocast packets. It is worth to mention, due to link failures happen during the simulation, it could happen that not all the destination nodes receive their corresponding messages.

For this reason, we have implemented a function that permits simulator to control reached and unreachable nodes for each sent packet based on logs. This function verifies for each packet if the destination nodes match reached and unreachable list of nodes in each turn. So, the simulator class is responsible for controlling when the simulation needs to be finished.

5.3. Routing elements library

Some routing elements take part in the simulation process. Every element has a class containing multiple functions that are used to achieve a correct operation of the simulation environment.

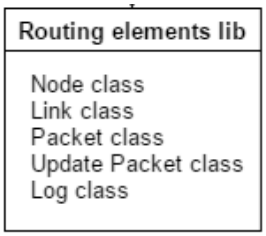


Figure 26. Routing lib main classes

These classes together create a multigraph that represents the topology of the network. The details of those elements are explained in the following subsections.

5.3.1. Node class

A node is a router that can route packets in the geocasting field. It is the most important element because all the processing load and routing decisions are distributed between all nodes of the network.

Nodes are connected to each other by a set of links and had full network knowledge. Nodes exchange information between them using hello packets and link-state packets (LSPs), to achieve full network knowledge.

Once a node has a view of the topology of the network, it can calculate the shortest path to each node. Every time a node receives a new packet sending request, it calculates the best routing path to reach all the nodes belonging to a given destination region.

In Figure 27, it is presented a diagram of node class attributes and tables. We are going to do an overview of node attributes:

- **node-id:** Node-id is a unique number that identifies a node in the network. It is assigned automatically by the simulation class.
  Our simulation implementation uses node id numbers instead of IPv6 addresses to route a packet. It has been done in such way to simplify the model and reduce the processing time of the simulation.
- **Latitude, Longitude:** They are GPS coordinates of the physical location of the node. They are required for drawing the node in its physical location in the graph. Also, these coordinates are used to calculate de coverage address associated with a node automatically.
- **Coverage address:** Each node can automatically calculate its coverage address by using the latitude and longitude as input attributes and add it to the coverage database. This mechanism allows to save time and not manually assigning a coverage address to each node. As a clear example, the coverage address of the node belonging to Groningen (The Netherlands) is '4.1.4.4.2.1.4.1.2.2.1.1.2.1.3.2.2.4'.
- **Label and Country**: Useful data that describes the geographical location of a node. Usually, both attributes refer to the city and the country where the node is located.
- **Inbox:** Incoming box that contains all the packets that need to be processed during the processing phase.
- **Outbox:** Outgoing box where the packets that need to be forwarded are placed during the transmission phase.

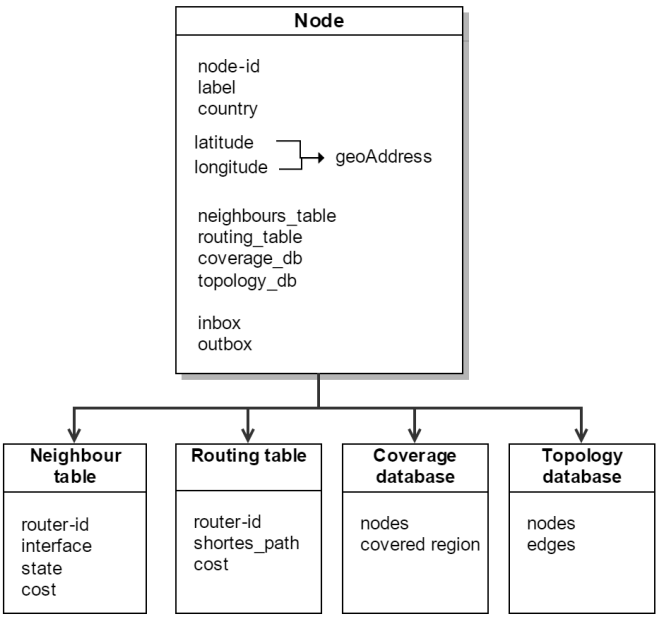


Figure 27. Node attributes

As we can see in Figure 27, each node has multiple tables to maintain updated information on the network. For further information about these tables, go to Chapter 4.

- **Neighbour table:** The neighbour table stores information about adjacent nodes that are directly connected to a node. In this table, we can find one hop neighbour router-ids, interface, the status of the shared links and cost of the link.
  Nodes need to establish a neighbour relationship with their adjacent nodes before exchanging routing updates. Neighbour nodes are dynamically discovered by sending Hello packets from each interface on a router.

Router-id	Interface	State	Cost
R1	-	-	0
R2	L1	Up	1
R3	L2	Up	1
R4	L3	Down	1

Figure 28. Neighbour table

- **Routing table:** This routing table permits to lookup for the best path to each node inside the network. So, this table works as a cache, where the shortest path for each destination has been stored. The node uses it to make forwarding decisions.
  Every time a node notices a topology update, it checks if the previous shortest paths contained in the routing table is still valid to reach the rest of the nodes of the network. Hence, entries that are obsolete are recalculated.
  The main advantage of this tables is that the shortest path to each destination node is already available and there is no need to compute the shortest paths each time a new message appears. This minimises the processing load of the routers.

Router-id	Shortest Path	Cost
R2	-	1
R3	-	1
R4	-	1
R5	[1,4,5]	2
R6	[1,2,6]	2
...	...	...

Figure 29. Routing table

- **Coverage database:** Coverage database contains the list of nodes and the coverage address for each of them, which permits to identify each node in the network.
  The covered region gives information about the coverage area that the routers serve. This coverage area does not necessarily need to be the exact location of the area because even if a router is not physically in an area, it can serve data information to a different location.
  In Figure 30, it is evident the need to simplify or group destination areas in order to reduce the number of coverage database entries.

Nodes	Covered region
0	4.1.4.4.2.1.4.1.2.1.3.1.4.1.4.1.4.
1	4.1.4.4.2.1.4.1.2.4.3.4.1.3.4.4.2.
2	4.1.4.4.2.1.4.1.2.2.1.1.2.1.3.2.2.
3	4.1.4.4.2.1.4.1.2.1.1.1.1.2.2.2.2.
4	4.1.4.4.3.1.4.1.1.4.2.4.4.3.4.4.1.
5	4.1.4.4.3.1.4.4.4.4.1.4.1.1.3.1.1.
6	4.1.4.4.2.1.4.4.3.2.1.4.4.4.1.1.1.
7	4.1.4.4.3.1.4.1.2.4.1.4.4.1.2.4.1.
8	4.1.4.4.3.1.4.1.1.3.4.3.2.2.2.4.3.
9	4.1.4.4.3.1.4.1.1.2.4.2.3.2.3.3.3.
10	4.1.4.4.3.1.4.3.4.3.1.2.4.2.3.2.2.
11	4.1.4.4.3.1.4.3.4.2.4.3.2.2.1.3.2.
12	4.1.4.4.3.1.4.3.1.2.2.2.3.2.3.2.4.
13	4.1.4.4.2.1.4.3.4.3.3.3.3.1.4.4.3.1.
14	4.1.4.4.2.1.4.3.4.2.1.1.2.3.1.1.2.
15	4.1.4.4.2.1.4.3.1.4.2.3.3.1.2.2.3.
16	4.1.4.4.2.1.4.3.2.2.4.4.4.3.3.2.4.
17	4.1.4.4.2.1.4.3.3.3.4.1.1.1.3.3.1.
18	4.1.4.4.2.1.4.3.3.1.1.1.3.3.4.4.2.
19	4.1.4.4.3.1.4.3.4.4.2.4.1.1.1.2.4.
20	4.1.4.4.3.1.1.2.4.4.2.4.3.3.1.1.2.
...	...

Figure 30. Coverage database

- **Topology database/table:** Topology database contains information about nodes that form the network. It stores the topology structure of a network, the distribution of the nodes and the connection between them.
  Each node will create each own copy of the topology map of the network to have full knowledge of the network. If a link-state change happens or a node is disconnected from the network, topology database information needs to be updated using the update mechanism.

Nodes	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, ...]
Edges	[(1, 2), (1, 3), (1, 4), (2, 5), (2, 6), (3, 5), (4, 5), (5, 8), (6, 8), (6, 7), (7, 9), (7, 10), (8, 9), (8, 10), ...]

Figure 31. Topology database

5.3.2. Link class

A link represents the connection between two or more adjacent nodes. In the networkX library, these links are called edges. A link connection is bi-directional so that nodes can perform the communication both ways (full duplex).

A link class implements an update link method that permits to change the state and some other characteristics.

The link attributes are the following:

- **Link-id:** It is a unique number that identifies a link in the network.
- **Source:** It represents the source node where the link starts.
- **Target**: It represents the target node where the link ends.
- **State:** It gives information about the state of the link, up or down.

- **Capacity:** This attribute represents the bandwidth of a link.
- **Cost:** It represents the metric values on links or edges along the path between minimum two nodes. The lower the cost, the better the path. If this field is empty, we will consider the default cost 1 per link. The link weights represent costs; they are not necessarily distances. We assume that the link weights or costs are positive (or at least one).

5.3.3. Packet class

A packet is a simple data object that represents a geocasting message that needs to be delivered from source to some destination nodes that belong to the same geographic region.

As we have mentioned before, a node knows its location, the location of its neighbours and the destination´s location. The destination where the packet is delivered is specified in the destination geoaddress, that is included in the packet header. The forwarding decision is based on the information contained in this packet.

A geocast packet contains the following fields:

- **Packet-id:** It is a unique number that identifies a geocasting packet in the network.
- **Destination geoaddress:** It is a destination geoaddress based on the Geographical addressing scheme that represents a geographical coverage area in the network.
  A packet is given an arbitrary destination geoaddress that is pick up from the list of available coverage addresses in the network. This address allows identifying destination nodes by looking up to the coverage database (included in every router).
- **Destination nodes:** A list of nodes which coverage address matches with the destination geoaddress of the packet.
- **Shortest path:** It is a dictionary or associative array that contains the optimal path from source node to multiple destination nodes. This shortest path contains a set of destination nodes stored as keys and values. The pythonic solution is to use dictionaries instead of arrays for path store. This makes the task of implementing search functions easier because dictionaries allow one to one matching of a key and value.

```
    === Packet_id 6 ===
src_node: 2
dst_geoAddress: 4.1.4.4.3.1.4.2
dst_nodes: [22, 25, 26, 27, 37, 38]
shortest path:
    {'src': 2, 'dst': 3}
    {'src': 3, 'dst': 1}
    {'src': 1, 'dst': 8}
    {'src': 8, 'dst': [30, 38]}
    {'src': 30, 'dst': [22, 19]}
    {'src': 37, 'dst': 25}
    {'src': 27, 'dst': 26}
    {'src': 19, 'dst': 24}
    {'src': 24, 'dst': 27}
    {'src': 38, 'dst': 37}
```

Figure 32. Packet 6 attributes

- **Size:** The overall size of the packet.
- **Recalculation count:** Counter which tells us the number of recalculation processes that a packet has faced. The moment it reaches a predefined maximum of recalculations, the packet is discarded and the destination nodes that have not been reached are considered unreachable.

In addition, the main methods used by this class are the calculation of the shortest path to all destinations and the draw of the path.

A node needs the previous attributes as input data to apply the shortest path to all destinations algorithm (explained in Section 4.1.1). Mention that the same algorithm is used when it is needed to look for an alternative path for the message.

After the shortest path is computed, it is possible to execute a draw_path function to show the resulting optimal route for the packet.

5.3.4. Update Packet class

The Update packet is the LSP that a node creates when a topology change happens. In this simulation environment, two topology changes are contemplated: link-state change and a node disconnection from the network.

Update packets are used to maintain up to date the rest of the nodes about link state changes through the network. Every time a node receives an update packet, it updates its topology database and forwards the LSP to the next nodes. In order to perform the controlled flooding to forward the update packet, it contains the following fields.

- **Update-id:** It is a unique number that identifies an update packet in the network.
- **Object type and Object id:** It refers to the type of the object (node or link) and its unique identifier. It is essential to know about which object are we updating the information in the databases.
- **State:** Indicates the state of the object, up or down.
- **Update source:** Denote in which node the update packet was generated. Because avoiding duplicates in the flooding process of update packets is fundamental.

```
=== Update Packet ===
update_id: 2004008
update_type: update
obj_type: Link
obj_id: e24
obj_src: 4
obj_trgt: 8
state: down
update_src: 8
```

Figure 33. Update packet 200408 attributes

It is worth to mention that a single Link-state update packet can carry one or more link-state advertisements.

5.3.5. Log class

It has been created a log system to record all the actions taken place when we run a simulation.

This last class does not belong to a routing element. However, it contains fundamental information about the changes in the routing elements during the simulation. Afterwards, the generation of these logs allows to create simulation animations and analyse the behaviour of the network step by step.

The Log class contains the following fields or attributes:

- **turn:** Indicates the turn in which that log occurred during the simulation session.
- **Packet-id:** Contains information about the packet which this log references.
- **Node-id:** Indicates where, on which node, the action of that packet is occurring.
- **State:** Using status codes we can know what action is being taken with that packet. A packet can be found in any of the following states: 'inbox', 'outbox', 'reached *', unreachable *' or 'rejected'.
- **Packet-source:** Indicates the node in which the packet is located at that turn.
- **Packet-destination**: Indicates the next hop or destination in which the packet is addressed at that time.

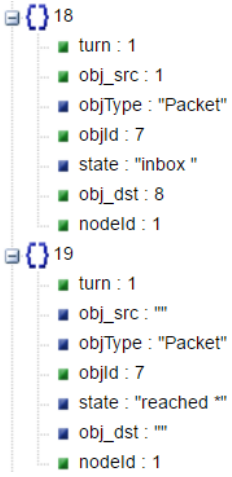


Figure 34. Simulation session logs in JSON format

With those fields, we can identify the processes that are being carried out during the simulation of our model.

The generated logs are saved in a JSON text file and will allow us to analyse and interpret the operations performed in each turn by the routing protocol.

5.4. Geocasting library

This library provides a set of Python functions that let us interpret Geocasting addressing scheme [1] addresses and convert between geocasting address formats (binary, hexadecimal and numerical).

Also, includes converter script which converts from GML into other formats. It creates node and edge objects from imported GML files. For instance, the Basemap_plot script uses Basemap library to perform geographical plotting. It takes a GML file, extracts GPS coordinates of each node and draws nodes in their actual physical location in the graph.

5.5. Generate test data

The simulation environment needs three inputs to run correctly a test simulation session: network topology information, packets and link failures.

As we have mentioned before, topology network information is imported using GML files provided by Topology Zoo website. However, geocasting packets and link drop out characteristics need to be specified before the simulation starts.

For this purpose, a script has been developed in python to generate random test data called “Sim_generate_testdata.py”. This script automatically creates packets and link failure objects that later will be imported to the simulation session.

In order to create some geocasting packets to be sent during the simulation it is needed to select a value for the following parameters of the script:

- Number of packets that we want to create and send.
- Level of the destination geoaddress.
- Turn in which generated packets will be introduced in their respective node’s inbox table.

On the basis of these parameters, packets are created picking up random destination geoaddresses inside de coverage database and sending them to different levels of the coverage area. Also, a random source node is selected to insert the packet into the node inbox while the simulation is running.

We have to choose a value for the following parameters of the script to create link failures that happen during the simulation:

- The number of link failures that need to occur during the simulation.
- Turn in which link dropouts happen.

Once those parameters are selected, links are randomly selected from the edge list of the topology graph.

These parameters will let us customise different types of test data to personalise the simulation inputs. If any of those parameters is left empty, it will be randomly selected by the script.

After packets and link failure objects have been created, they are all saved as JSON object structures in a JSON text file.

The moment we run a simulation session, those objects are imported from the JSON text file. The turn to insert each packet in certain node’s inbox table and apply each link dropout in the network will differ depending on the turn previously assigned.

On the whole, used experimental data consist of randomly generated packets and link failures that are used to measure the performance of the network in different scenarios.

6. Evaluation

In this chapter, we are going to test the implemented functionalities to evaluate system performance in terms of link usage and accuracy.

We first introduce the topology networks where the link-state protocol is used. Then we explain the proposed scenarios being used for the analysis. Finally, we discuss the results for different scenarios and networks.

6.1. Scenarios

In this project, we analyse and compare the performance of the routing protocol in different real network scenarios. We consider various topologies of different sizes and forms, each of which has been simulated in a Python environment.

We have downloaded topology datasets from Topology Zoo webpage [39]. There are lots of GML file format datasets available on this website, where we can find network topologies of any kind around the world. GML files include node and edge object details that are imported to create a network graph.

The evaluation is performed with two typical topology network scenarios to simulate the impact of the proposed link-state protocol on the network performance.

- **Surfnet network**

Surfnet is the network that offers students, lecturers and scientist in the Netherlands access to the internet and ICT facilities. It is composed of a set of 50 nodes and 73 links that connect routers distributed throughout the country.

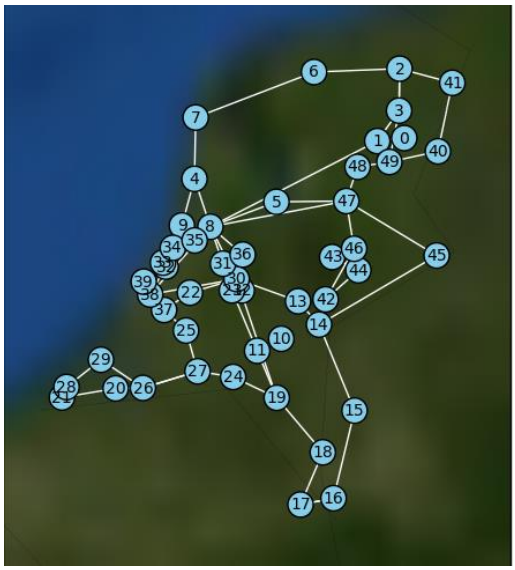


Figure 35. Surfnet network topology

- **Cogent network**

Cogent is a multinational Internet Service Provider that owns one of the top five largest networks in the world. The company offers a service Internet access and data transport over optical fiber along different countries [40].

The Cogent network area shown in Figure 36 is formed by 186 nodes and 214 link connections that interconnect North America and Europe across the Atlantic ocean.

Figure 36. Cogent network topology [39]

The reason why we have selected these two topology networks is that they represent two typical network scenarios. The first one refers to a medium size network that covers the geographical region of a country, whereas the second permits to test the implemented routing protocol in a worldwide network.

We have not chosen a small network topology for evaluation because they are formed by just a few routers and edges. This feature is not suitable for our analysis because the probability of packets that do not reach their destination nodes will be increased as we start removing link connections during a test.

6.2. Test setup

The main goal of this evaluation is to examine the impact of the proposed link-state routing protocol on the network performance. The following performance indicators characterise the behaviour of network elements:

- Link usage: It is the analysis of the number of transmissions used during the simulation sessions to determine successful transmissions, wasted transmissions, packet's path recalculations and discarded packets.
- Accuracy: It is the analysis of the number of reached and unreached destination nodes for each packet during the simulation sessions. It is fundamental to determine if all nodes belonging to the target region have been reached to obtain results about the accuracy of the routing and forwarding mechanisms. For this purpose, we examine if sent packets have reached destination nodes or have been discarded before arriving at all destination nodes.

In order to evaluate the link-state routing protocol, we have defined some cases to test link usage and accuracy in different network conditions.

These cases use the same predefined set of packets and link failures for a given network. This test data stack includes a set of 10 packets and 10 link failures that do not change to assure consistent data for testing the performance of the network.

The experimental data used for each network have been created by running the "Generate_test_data" script, described in Section 5.5. With this script, it is possible to generate random packets and link failures to reproduce any network scenario.

In our analysis, we consider different configurations to create four scenarios. The details of each scenario that make use of the mentioned test data are explained below.

- **Case A:** Multiple packets

The first scenario has the objective of verifying different functionalities of the link-state routing protocol implementation in the simulation environment.

It relates to the ideal case in which multiple packets (10 packets per node) are sent from each router to predefined destination areas. This set of multiple packets permits us to get ideal results for the deployed network configuration.

In case A, there are no link failures or node disconnections during the simulation sessions. Therefore, routing and forwarding mechanisms are not affected by network conditions, such as dropped out connections.

- **Case B:** Multiple packets and one link failure

In the second case, we introduce the same set of packets used in the previous case, which is sent from each router in the network.

However, in this case, we pick up a random link that will be dropped out during the simulation.

This scenario let us collect information about how the protocol reacts when a link failure occurs during the simulation.

- **Case C:** Multiple packets and multiple link failures

The objective of the third scenario is to simulate a disconnection of multiple links at the same turn. Instead of dropping out a unique link like in the previous case, we are going to turn down 10 connections at the same time.

Hence, the routing protocol will need to perform some path recalculations and use more transmissions to reach destination nodes.

- **Case D:** Multiple packets and successive link failures

In case D, multiple packets are sent from source routers to their corresponding target nodes and link failures are introduced consecutively in each turn.

The point of this scenario is to analyse how the successive disconnections affect the network performance during the simulation sessions.

If we analyse and compare between the presented scenarios, we should be able to understand the behaviour of the designed link-state routing protocol in different conditions.

In addition, we have prepared an accuracy test to investigate how link failures affect the reliability of the packets. This will let us have an idea of how the network would respond in such case.

6.3. Results

In this section, we present the results of the analysis. We first introduce a simulation demo to show how the routing and forwarding mechanism work using animation functions. Then we illustrate the results obtained from the analysis performed regarding link usage and accuracy.

6.3.1. Simulation demo

In this section, we show with the help of an example how the routing and forwarding mechanism works in the simulator.

During any simulation session, turn-based simulation routing procedures are applied to build routes and dynamically maintain updated information of the network. The log system keeps a record of those routing procedures. Hence, simulation animations and graphs are created based on logs.

The graph in Figure 38 is formed by a set of nodes and edges. Each node represents the routers of the network, while an edge or link is the connection between nodes. The relation between those elements is explained in Section 4.1.

For instance, imagine that a geocasting packet is sent from node 2 through the Surfnet network. The generated packet attributes are shown in Figure 37 and the shortest path of the packet is illustrated in Figure 38.

```
   === Packet 1 ===
src_node: 2
dst_geoAddress: 4.1.4.4.3.1.4.2
dst_nodes: [22, 25, 26, 27, 37, 38]
shortest_path:
   {'dst': 3, 'src': 2}
   {'dst': 1, 'src': 3}
   {'dst': 8, 'src': 1}
   {'dst': [30, 38], 'src': 8}
   {'dst': [22, 19], 'src': 30}
   {'dst': 25, 'src': 37}
   {'dst': 26, 'src': 27}
   {'dst': 24, 'src': 19}
   {'dst': 27, 'src': 24}
   {'dst': 37, 'src': 38}
```

Figure 37. Packet attributes

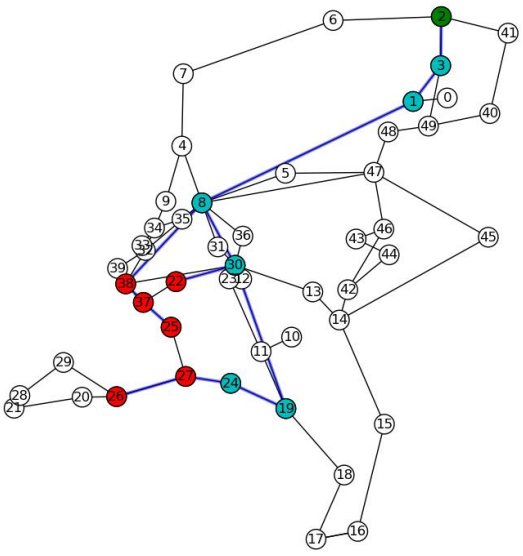
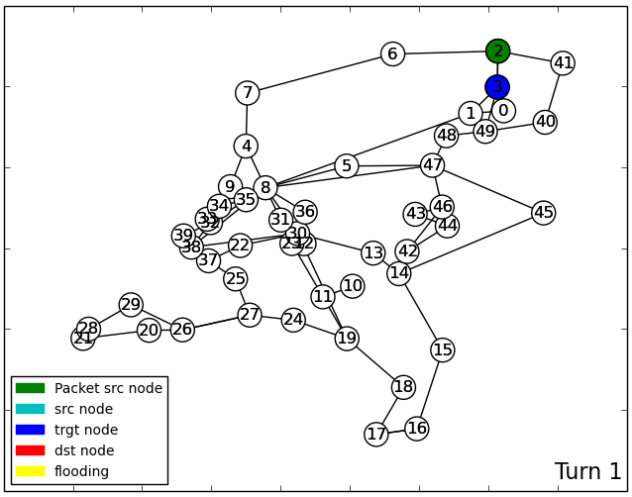
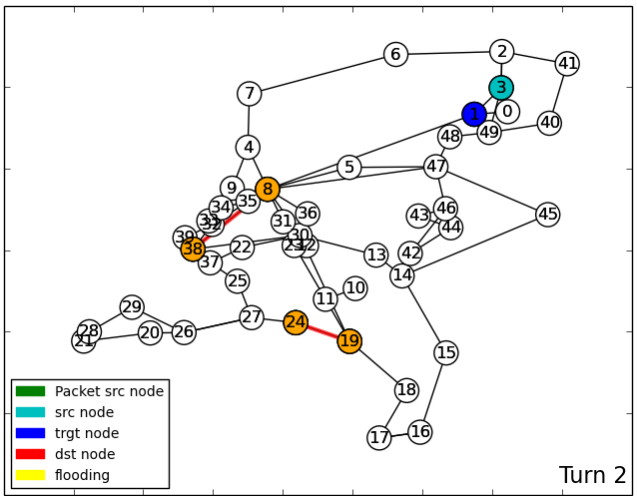
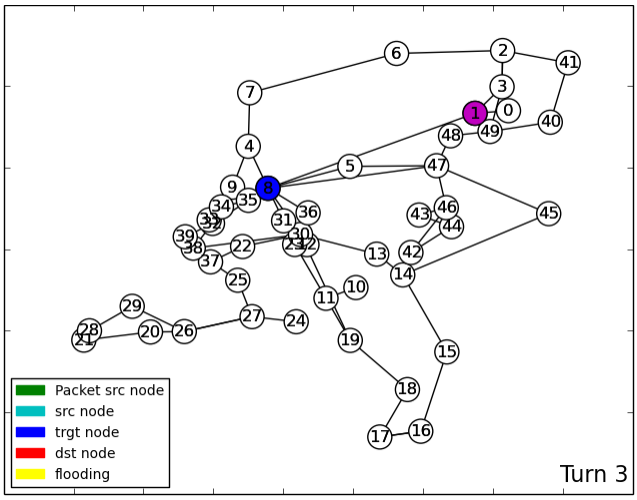


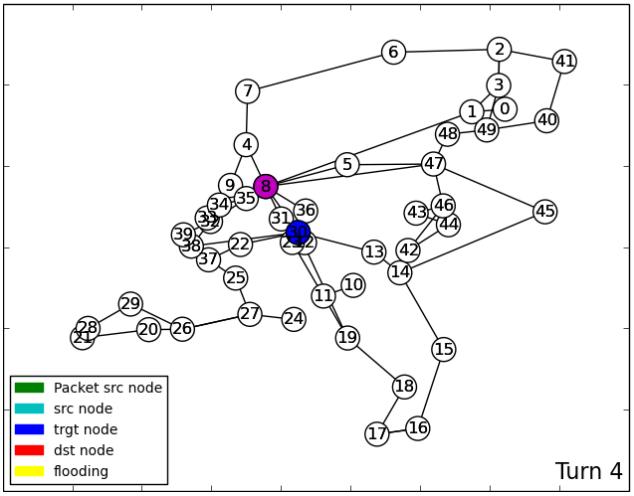
Figure 38. Packet shortest path

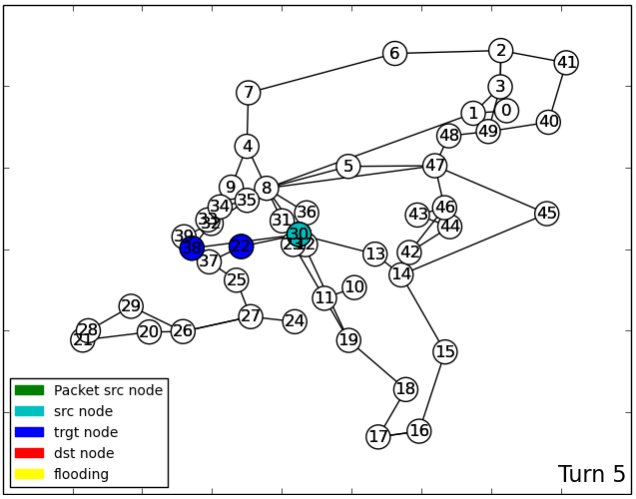
We run the simulation session where the mentioned packet is transmitted and two link failures happen. The procedure of the routing mechanism is illustrated through simulation animation snapshots, shown in Figure 39.

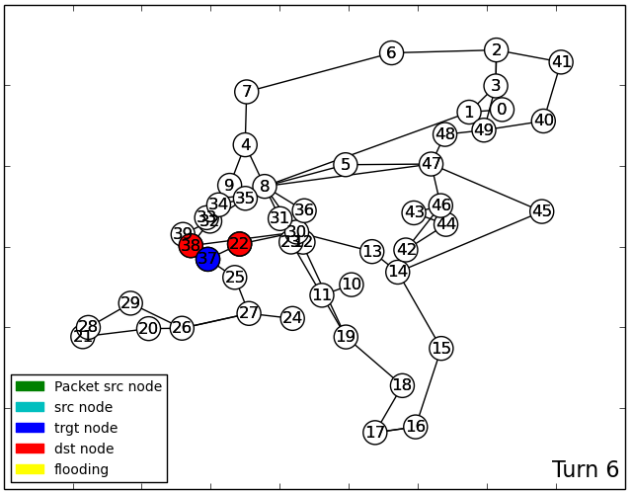

Packet src node
src node
trgt node
dst node
flooding
Turn 1


Packet src node
src node
trgt node
dst node
flooding
Turn 2


Packet src node
src node
trgt node
dst node
flooding
Turn 3


Packet src node
src node
trgt node
dst node
flooding
Turn 4


Packet src node
src node
trgt node
dst node
flooding
Turn 5


Packet src node
src node
trgt node
dst node
flooding
Turn 6

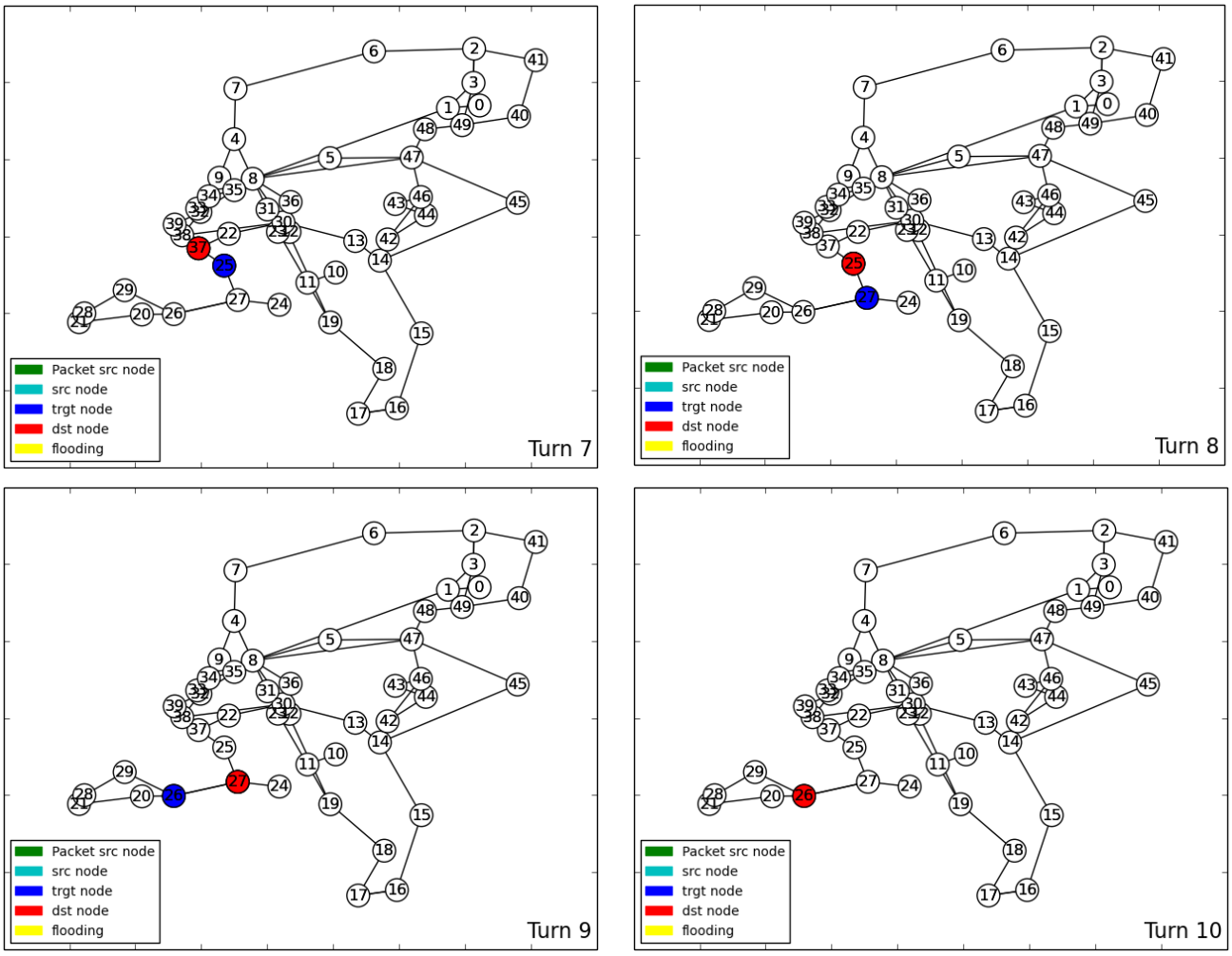


Figure 39. Simulation demo animation snapshots

In the first turn, source node 2 calculates the shortest path for packet 1. In the following turns, intermediate nodes that receive the packet will forward the message 1 looking up to the shortest path included in the packet.

During the simulation, two link failures occur in the second turn; where links (8,38) and (19, 24) are dropped-out from the graph. As a consequence, affected nodes (8, 38, 19 and 24) propagate update packets so that the rest of the nodes are aware of the topology changes.

Remember that source routers are the ones that calculate the path of the packet. However, if any connection failure happens, intermediate nodes will need to recalculate the packet path which has been affected.

This condition causes the packet's path no longer to be valid. Therefore, nodes 1 and 8 recalculate the path to remaining destination nodes in turn 3 and 4 respectively. The new available path data is presented in Figure 41.

Once the new path is calculated, intermediate nodes forward the copies of the packet. Eventually, in the following turns, packet 1 copies reach all destination nodes (22, 25, 26, 27, 37, 38).

```
    === Packet 1 ===
src_node: 2
dst_geoAddress: 4.1.4.4.3.1.4.2
dst_nodes: [22, 25, 26, 27, 37, 38]
shortest path:
   {'dst': 3, 'src': 2}
   {'dst': 1, 'src': 3}
   {'dst': 8, 'src': 1}
   {'dst': 30, 'src': 8}
   {'dst': [22, 38], 'src': 30}
   {'dst': 25, 'src': 37}
   {'dst': 26, 'src': 27}
   {'dst': 27, 'src': 25}
   {'dst': 37, 'src': 22}
```

Figure 40. Packet's attributes after recalculations

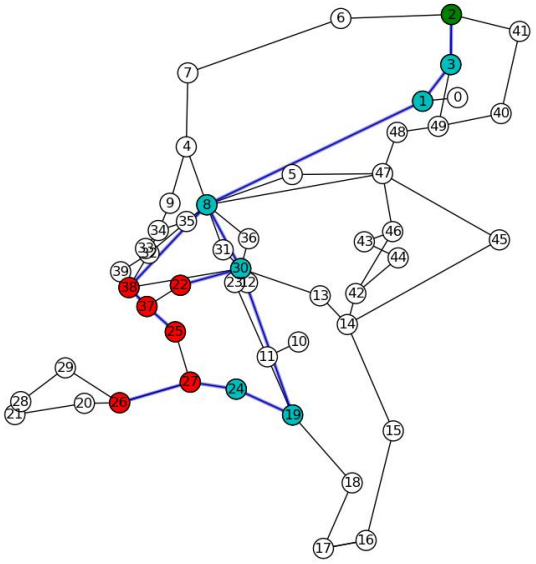


Figure 41. Packet's shortest path after recalculations

In addition, we are going to show how the network update mechanism works. Each time a link failure happens, affected nodes send update packet to the rest of the nodes using the flooding mechanism. An orange colour illustrates affected nodes and the nodes reached by flooding mechanism are coloured in yellow. In this case, there have been dropped out links (8, 38) and (19, 24). Flooding mechanism needs 7 turns to send a copy of the updated packets to update topology databases of all the nodes in the network.

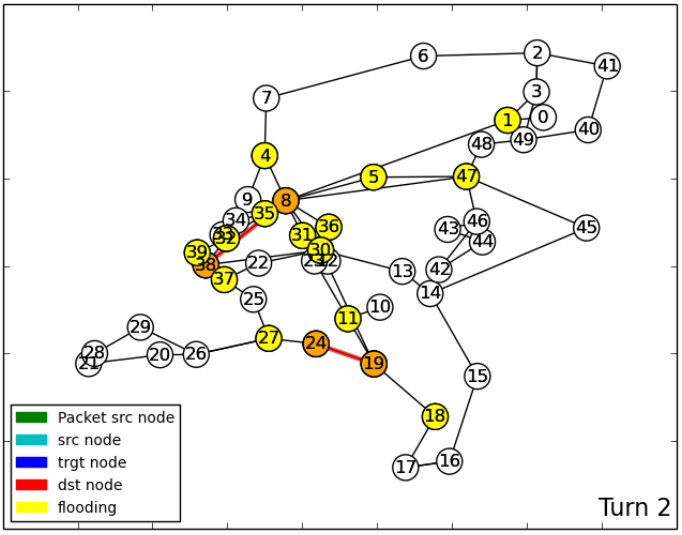


Figure 42. Update mechanism in turn 2

6.3.2. Analysis of different scenarios

In this section, we do an analysis of the four configuration scenarios regarding link usage, path calculations and accuracy in both networks. The results corresponding to the Surfnet network can be seen in Figure 43, Figure 44 and Figure 45; while results from the Cogent network are shown in Figure 46, Figure 47 and Figure 48.

It is important to point out that the results depend on generated packets parameters and link-state conditions of the networks. We are considering a successful communication only when a packet sent from a source node reaches all the target nodes within the destination geoaddress.

If we compare figures of both networks, we can see how the network responds to different scenarios.

The case A values match with the ideal scenario, in which a set of multiple packets have been sent from every source node with no link failures during the simulation. Therefore, we have only path calculations in turn 1. We have taken this ideal case as a reference for following comparisons.

Common sense tells us that if in the other scenarios (B, C and D) link failures happen, link usage and path calculations curves offset will be increased. However, they are decreased because the introduction of multiple link failures produces the growth of unreachable nodes for the packets. If a node detects that there is no way to reach a destination node, the node marks the destination node as unreachable and discards the message. Therefore, we can notice that the decrease in the group of transmissions and path calculations in cases C and D is related to the increase of unreachable nodes in Figure 45 and Figure 48.

The link usage results are presented in Figure 43 and Figure 46, which corresponds to the number of transmissions used in each turn of the simulation session. The reason why the values on the Y axis are high is because we have sent a set of 10 packets from each node of the network. This means that the count of the number of transmissions, number of path calculations and number of reached and unreachable nodes are calculated in relation to 10 sent packets per node. With this procedure, we have consistent results throughout the simulations.

Also, it must be said that the implemented protocol does not contemplate congestion of links, packet priorities, collisions or delays. A node will process transmissions of the outgoing messages on the same turn. Thus, no message can be lost while it is being transmitted.

According to this results, the behaviour of the network works as follows. Messages are introduced in their respective source node inbox in turn 0. Then, each node calculates the shortest path of new packets in turn 1. The links, which are dropout after turn 2, produce the rest of recalculations of the path of packets due to topology changes.

Comparing figures Figure 43 and Figure 46, we can notice that the case C has more overall transmissions than case D in both networks. This means that the drop out of multiple packets in the same turn increases the number of transmissions more than a successive drop out of links in each turn.

Also, looking at the link usage curves, we can conclude that the number of transmissions is higher in the first moments of the simulation.

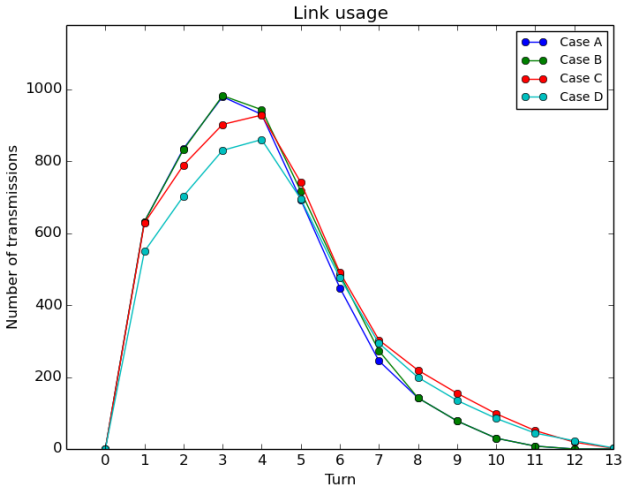


Figure 43. Surfnet: Link usage of different scenarios (10 packets/node)

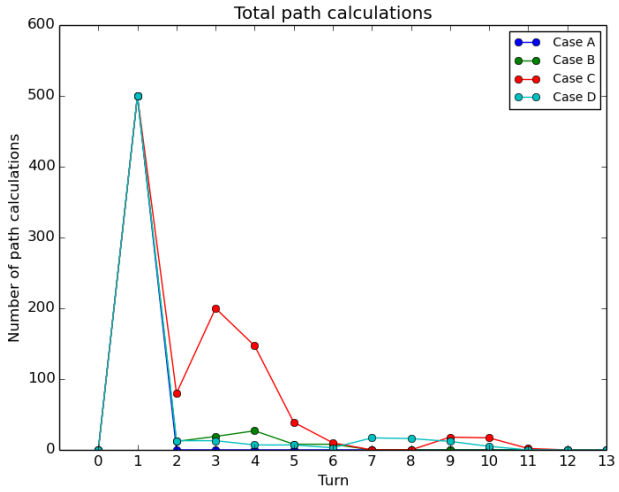


Figure 44. Surfnet: Path calculations of different scenarios (10 packets/node)

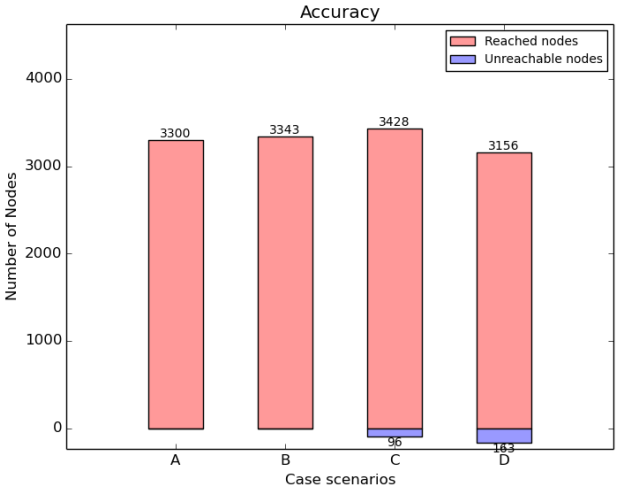


Figure 45. Surfnet: Accuracy of different scenarios (10 packet/node)

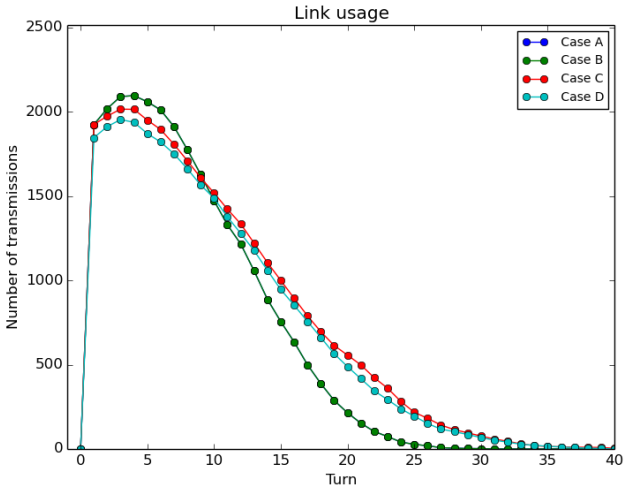


Figure 46. Cogent: Link usage of different scenarios (10 packet / node)

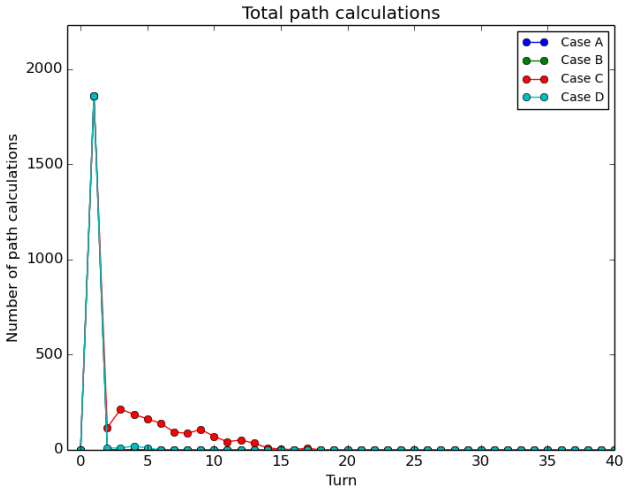


Figure 47 Cogent: Path calculations of different scenarios (10 packets/ node).

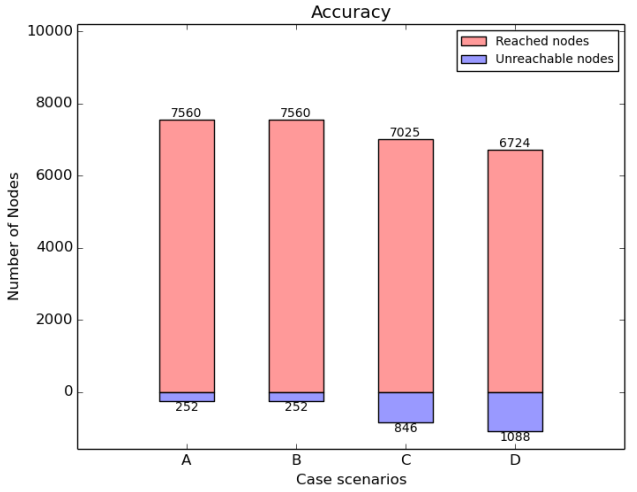


Figure 48. Cogent: Accuracy of different scenarios (10 packet/node)

Figure 44 and Figure 47 show path calculations of the simulations. If we focus on the curves, we notice that most of the path calculations are performed in the first turns. Hence, main processing load in routers is concentrated on the first turns, like happened in link usage.

The maximum peaks encountered in turn 1 of Figure 44 and Figure 47, is related to the fact that in turn 0 packets are introduced in their respective node inbox. In turn 1, each node calculates the shortest path of those packets. The links dropped out after turn 2 produce the rest of recalculations of the path of packets due to topology changes.

It is worth to mention in case C of Surfnet network (Figure 44) a significant increase in the number of recalculations in turn 3 and 4. This increase has been produced by the dropout of one or more critical links in turn 2. As a consequence, several packet paths are affected and need to be recalculated by the intermediate nodes.

A critical node is a router that is located in the centre of the network and is usually elected as an intermediate node in the path calculations of the packets. The links connected to a critical node are known as critical links. This way, the dropout of a critical link causes adjacent nodes to have to change their optimal paths in order to reach the predefined destinations in each packet.

Figure 45 and Figure 48 show graphs of the reached and unreachable nodes regarding accuracy for the four scenarios. If we compare the figures of both networks, we can conclude that the succession of link failures (Case D) causes a major decrease of reached nodes than with multiple link failures in the same turn does (Case C). The reason why this happens is that it is harder to keep convergence between routers when Case D conditions occur.

In this thesis, we understand convergence time as a number of turns (instead of time) it takes to propagate changes in network topology throughout the whole network.

To conclude, we can say that the network is more susceptible to increase the number of transmissions, path calculations and unreachable nodes when multiple link failures happen in the same turn (Case C).

6.3.3. Analysis depending on link failures

Here we are going to analyse how the two networks respond when we increase the number of dropped links. The following figures present the results obtained after running 15 simulation sessions. We introduce an additional link failure in every iteration. So that in test 0 we have no connection failures and in test 15 we have 15 link failures.

It is true that a destination is reached, as long as there exist a path that can reach from the source node to the destination node. If there is no path available, the packet will be discarded.

This mechanism minimises undesired loops within the network, which can subsequently affect the performance of the network. Hence, the implemented link-state routing protocol does not assure reliability and work on a best-effort basis to prevent routing loops.

The Figure 49 and Figure 50 show the accuracy of each test; presenting the number of reached and unreachable nodes per link failure. If we compare both graphs, we can determine that overall reached nodes will decrease as the link failures increase regarding the size and the number of nodes. Although, larger networks, like the Cogent network, are more susceptible to link dropouts and, hence, more prone to have less reached nodes.

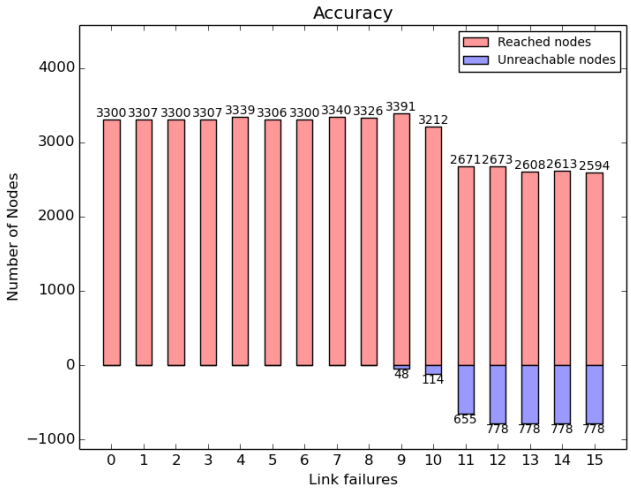


Figure 49. Surfnet: Accuracy dependent on link failures

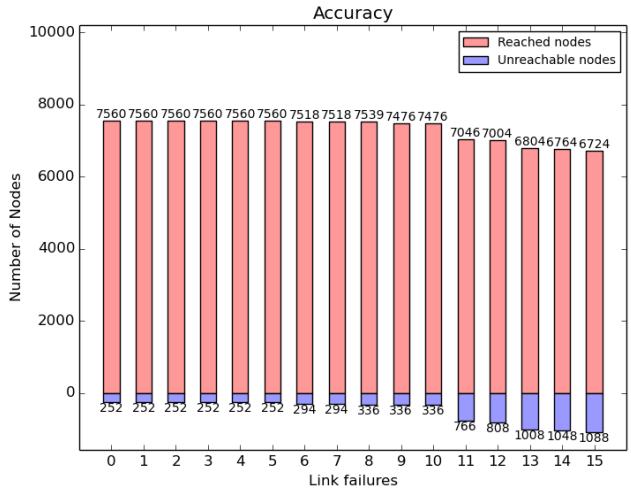


Figure 50. Cogent: Accuracy dependent on link failures

As shown in Figure 49 and Figure 50, the tests with 11 link failures or more will lead to a considerable increase of not reached nodes. In Figure 51 and Figure 52 we see that multiple link failures also affects to the link usage. However, path calculations ratio is mainly constant through the tests.

After analysing numerous results, we can say that it is necessary to keep the number of routers at 50 or below per area to guarantee correct convergence times of the updating mechanism. If we have more than 120 nodes the convergence time gets higher and the network will not work optimally.

In can be concluded that the proposed protocol has better performance in a medium size network regarding accuracy, link usage and path calculation terms.

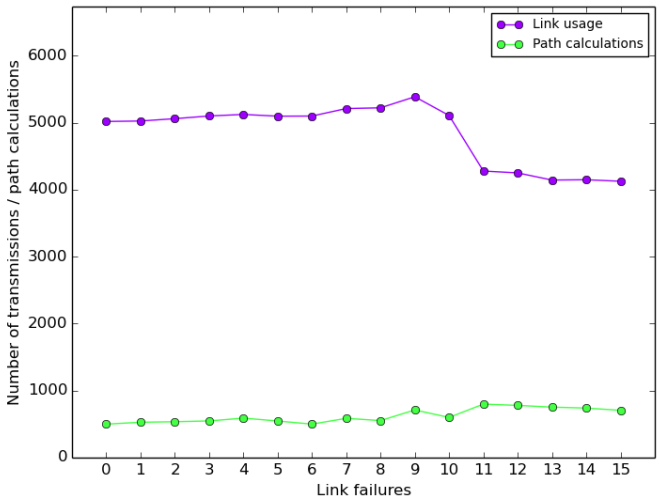


Figure 51. Surfnet: Link usage and path calculations dependent on link failures

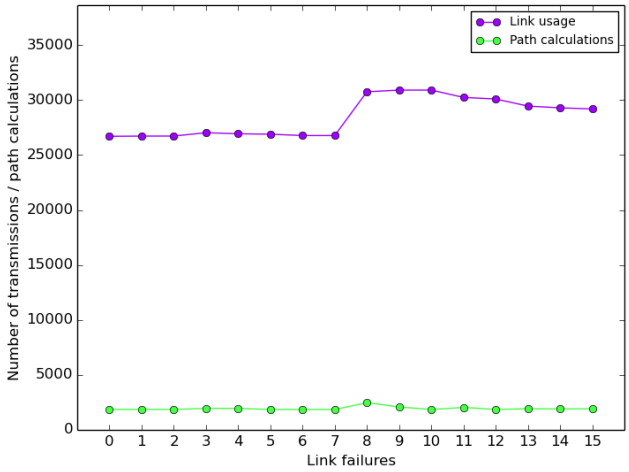


Figure 52. Cogent: Link usage and path calculations dependent on link failures

6.4. Discussion

The simulation tool has facilitated the design, visualisation and debugging of the proposed protocol through the project. However, we have noticed that the obtained results depend strongly on the codification of the simulation functions and used test data.

The link-state protocol for Internet-wide Geocasting is designed to work in a completely distributed manner and not rely on any central entity. The main advantage here is that any node can perform link-state routing and forwarding techniques, like shortest path calculations, when needed.

On the contrary, the evaluation shows a high processing cost. The processing load is concentrated mostly at the first turns of the simulation sessions, mainly in source nodes.

However, once the source node has predefined the path, there are no long delays in the forwarding and transmission process of the packet. In a real scenario, we should need high transmission rates to send all incoming packets of a node in a turn and that way avoid congestion and collisions between outgoing packets.

The simulation results illustrate that flooding is quite effective for maintaining up to date link-state data in each node. However, if convergence time is high, a node could not receive the update packet's data fast enough to recalculate on time the new path for the geocast packet.

The behaviour of the link-state protocol during the results specifies its working domain is a wired network formed by around 50 nodes. As a result, we can say that medium size network areas (like Surfnet) generate better performance and higher levels of availability than larger network areas. Also, convergence time is lower and the probability to reach destination nodes is greater in a medium size network.

According to the obtained results, problems caused by multiple link failures have been solved using the best effort approach. In fact, it is a costly task to track all the packets through the network. That is why when a link failure happens; the intermediate node is responsible for recalculating a valid path for the packet and this way prevent from routing loops. If the recalculation of the path is not possible, the packet is discarded.

Therefore, this protocol does not guarantee reliability. Because if multiple link failures occur, the packet would not be able to reach all destination nodes. However, there are high chances of reaching destination nodes if a path to the destination node(s) is still available.

7. Conclusion and Future Work

The current chapter will go through the conclusions and possible future lines to be carried out in order to give a continuation to this project.

7.1. Conclusions

In this project, we propose and discuss the design of a link-state routing protocol for Fixed infrastructure networks. The described routing protocol enables the distribution of geocasting messages between routers in an already deployed topology network.

In this thesis, we have performed a background research about the concepts and technologies used in the geocasting environment and associated routing protocols. Then, it has been established that the best solution for the main problems stated in Section 3.1 is to adapt a link-state routing protocol.

In order to have a general view of how the proposed protocol responds to different scenarios, we have implemented a simulation environment with the key features of the design. The simulation software has facilitated the design, visualisation and debugging of the proposed protocol, letting us analyse how the network performs under test. However, we need to keep in mind that network simulators are not perfect; because the model used for simulation purposes cannot model all details of the real network behaviour.

The proposed routing protocol performs routing and forwarding tasks in an Inter-area network based on the Geographical addressing scheme. The mentioned addressing scheme is accurate enough to get packets relatively close to their destination nodes.

The protocol has been designed to operate in a completely distributed manner and not depend on any central entity. It uses periodic exchange of messages to maintain updated topology information at each node. Each node uses this information to calculate loop-free routes for every geocast packet.

The main feature of the designed solution is that each node can calculate the optimal routes to reach destination nodes belonging to a geographical region. This is possible because all nodes of the network maintain an updated full network knowledge.

Also, it is important to know that processing load is concentrated in source nodes (path calculation and routing decisions). Therefore, source node establishes the predefined path for a message.

Hence, intermediate nodes just need to forward a received packet. Only if a link is dropped out and the topology changes are detected, intermediate nodes will have to recalculate a packet's path.

We can conclude that if there is a path available, it will be possible to reach all the destination nodes of a message. In fact, if a packet cannot follow the predefined path or recalculate an alternative one it will be discarded. Otherwise, loops could be created.

One of the main advantages of this routing protocol is that the chances to reach most of the nodes belonging to a geographical region are high. However, this protocol does not guarantee reliability. If multiple link failures occur at the same time, the packet would not be able to reach all destination nodes.

The behaviour of the protocol during the results specifies that its working domain is a wired network formed by around 50 nodes. We have seen that convergence time is critical to assure nodes to have full network knowledge. This could be a considerable limitation when multiple link failures occur in the same turn.

It is worth mentioning that to achieve a correct operation of the protocol; routers need to handle a considerable processing load. According to the obtained results, this processing load is present especially during packet's paths calculations, intensive lookup tasks and maintenance of routing tables.

Even though some of the actual routers deployed in the fixed network infrastructure could not be prepared to manage those processing loads, it is feasible to think about routers that will support the mentioned workload in a not too distant future.

7.2. Future work

This project is a first approximation to a routing and forwarding mechanism capable of operating in a real geocasting environment. It is a proof of concept that has allowed us to verify the feasibility of the elaborate design based on the Geographical addressing scheme.

The results of this thesis point to several interesting directions for future work:

- Reliable communications

The designed routing protocol uses a best-effort mechanism to reach target areas. As we have no feedback or tracking mechanisms, we cannot guarantee that a sent packet had reached all the destination nodes.

Future research work should be done on improving reliability in the communications between nodes. This is a significant concern because geocasting is meant to be used in traffic safety applications.

- Reduce entries of routing tables

Another problem is that nodes need to manage numerous routing table entries. This problem increases in large networks. For example, it is evident the need to simplify or group destination areas to reduce the number of coverage database entries.

An improvement in the organisation of Geographical addressing scheme is required to reduce processing load, decrease lookup time and reduce required storage in each router. As a result, these improvements will significantly decrease delays due to routing a geocasting packet from a source node to a set of destination nodes.

- Reduce impact of geocasting routing in packet size

During this project, we have considered that a packet contains the destination specification and the predefined path. If a packet exceeds its size limit, it could be discarded.

Therefore, a study should be done to find out how to carry routing information inside the packet that allows keeping the packet's size below that limit.

- Support for packet priority

A priority system is needed to classify the transmission of outgoing geocast packets in order of importance. Packet priority support should permit to prioritise critical traffic such as accident alerts or important driver information. It is easy to think that we should work on algorithms that manage or control the priority of said packets.

- Use Steiner tree based solution to calculate the shortest path

Although Dijkstra's algorithm works effectively to determine the shortest path from a source node to multiple destination nodes, it could be interesting to study the benefits of using an alternative algorithm, such as Steiner tree.

- Traffic impact

In the geocasting field, the communication frequently occurs between a large number of nodes. In this project, we are not taking into account the impact of this geocasting mechanism on other types of network traffic.

A study should be performed on how this geocasting system affects other traffic in order to minimise the impact on other types of traffic.

- Specify criteria on how to select area border nodes between areas

This design model works based on the premise that all the nodes know the location of each neighbour. If the objective is to implement this kind of routing protocol in the Internet-Wide Geocasting scenario, we should divide this network into different areas. Only this way this type of protocol can be useful in the vast Internet network.

So, the next step in the design would be to determine how area border nodes communicate between each other. In consequence, it should be established how to select area border nodes that make the communication between nodes in different network areas possible.

This implementation of this approach will lead to smaller routing tables and less traffic control messages.

- Packets and nodes security and privacy aspects

Security and privacy are a priority for connected cars and future autonomous vehicles. An Internet-wide geocasting solution should also provide confidentiality to both personal and public data such as the relation between nodes and their location or data contained in the packets. A security and privacy mechanisms to guarantee both integrity of the data and avoid data interception should be developed.

Additionally, future work may focus on the performance of the protocol and implementation to be able to use this system in heterogeneous topology networks. For instance, delay produced by processing task and response times in the routers can be evaluated more extensively.

In conclusion, more research needs to be done on reliable routing protocols that can route geocast packets to provide improved mobility services to vehicles in the future vehicular networks.

8. Bibliography

[1] B. Meijerink, M. Baratchi, and G. Heijenk, “An Efficient Geographical Addressing Scheme for the Internet,” 2016, pp. 78–90.

[2] S. Zeadally, R. Hunt, Y.-S. Chen, A. Irwin, and A. Hassan, “Vehicular ad hoc networks (VANETS): status, results, and challenges,” *Telecommun. Syst.*, vol. 50, no. 4, pp. 217–241, Aug. 2012.

[3] S. Yousefi, M. Mousavi, and M. Fathy, “Vehicular Ad Hoc Networks (VANETs): Challenges and Perspectives,” in *2006 6th International Conference on ITS Telecommunications*, 2006, pp. 761–766.

[4] F. Li and Y. Wang, “Routing in vehicular ad hoc networks: A survey,” *IEEE Veh. Technol. Mag.*, vol. 2, no. 2, pp. 12–22, 2007.

[5] D. Y. Parrado N, “Representation of V2V and V2I technologies.” [Online]. Available: https://openi.nlm.nih.gov/detailedresult.php?img=PMC4431278_sensors-15-07768-g002&req=4. [Accessed: 09-Mar-2017].

[6] *Vehicular Networking*. Google books.

[7] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti, and H. Zedan, “A comprehensive survey on vehicular Ad Hoc network,” *J. Netw. Comput. Appl.*, vol. 37, pp. 380–392, 2014.

[8] Institute of Electrical and Electronics Engineers, IEEE Vehicular Networking Conference (3 2011.11.14-16 Amsterdam), and VNC (3 2011.11.14-16 Amsterdam), *IEEE Vehicular Networking Conference (VNC), 2011 : 14 - 16 Nov. 2011, Amsterdam, Netherlands*. IEEE, 2011.

[9] B. Bilal and M. M. A. Peer, “Classification of Current Routing Protocols for Ad Hoc Networks -A Review,” *Int. J. Comput. Appl.*, vol. 7, no. 8, pp. 975–8887, 2010.

[10] J. C. Navas and T. Imielinski, “GeoCast---geographic addressing and routing,” in *Proceedings of the 3rd annual ACM/IEEE international conference on Mobile computing and networking - MobiCom ’97*, 1997, pp. 66–76.

[11] J. C. Navas, T. Imielinski, J. C. Navas, and T. Imielinski, “On reducing the computational cost of Geographic Routing 1,” 2000.

[12] V. D. Park and M. S. Corson, “A performance comparison of the temporally-ordered routing algorithm and ideal link-state routing,” in *Proceedings Third IEEE Symposium on Computers and Communications. ISCC’98. (Cat. No.98EX166)*, pp. 592–598.

[13] C. Maihofer, “A survey of geocast routing protocols,” *IEEE Commun. Surv. Tutorials*, vol. 6, no. 2, pp. 32–42, 2004.

[14] A. Fox, C. Eichelberger, J. Hughes, and S. Lyon, “Spatio-temporal Indexing in Non-relational Distributed Databases.”

[15] T. Imieliński and J. C. Navas, “GPS-based geographic addressing, routing, and resource discovery,” *Commun. ACM*, vol. 42, no. 4, pp. 86–92, Apr. 1999.

[16] T. Fioreze and G. Heijenk, “Extending the Domain Name System (DNS) to provide geographical addressing towards vehicular ad-hoc networks (VANETs),” in *2011 IEEE Vehicular Networking Conference (VNC)*, 2011, pp. 70–77.

[17] M. Nosrati, R. Karimi, and M. Hariri, “An Overview of Routing Protocols,” *World Appl. Program.*, no. 15, pp. 355–358, 2011.

[18] “Routing Overview.” [Online]. Available: http://networking.ringofsaturn.com/IP/Routing.php. [Accessed: 02-Mar-2017].

[19] “Exterior Gateway Protocol.”

[20] Y. Rekhter, T. Li, and S. Hares, Eds., “A Border Gateway Protocol 4 (BGP-4),” Jan. 2006.

[21] G. Malkin, “RIP Version 2 - Carrying Additional Information.”

[22] C. L. Hedrick, “SIP-RIP.”

[23] Cisco, “Enhanced Interior Gateway Routing Protocol - Cisco.” [Online]. Available: http://www.cisco.com/c/en/us/support/docs/ip/enhanced-interior-gateway-routing-protocol-eigrp/16406-eigrp-toc.html. [Accessed: 02-Mar-2017].

[24] M. Bornhager, “Link-State Routing Protocols.”

[25] A. Kaur and E. Dinesh Kumar, “Comparative Analysis of Link State Routing Protocols OPSF and IS-IS,” *Int. J. Comput. Sci. Trends Technol.*, vol. 3, no. 4, 2013.

[26] H. Smit, “RFC 3784 - Intermediate System to Intermediate System • IS-IS, Extensions for Traffic Engineering □TE□,” 2004.

[27] “OSI IS-IS Intra-domain Routing Protocol.”

[28] R. W. Callon, “Use of OSI IS-IS for Routing in TCP/IP and Dual Environments.”

[29] R. Perlman, “A Comparison Between Two Routing Protocols: OSPF and IS-IS.”

[30] S. E. Deering and D. R. Cheriton, “Multicast routing in datagram internetworks and extended LANs,” *ACM Trans. Comput. Syst.*, vol. 8, no. 2, pp. 85–110, May 1990.

[31] M. L. Shore, L. R. Foulds, and P. B. Gibbons, “An algorithm for the steiner problem in graphs,” *Networks*, vol. 12, no. 3, pp. 323–333, 1982.

[32] A. Santuari, “Steiner Tree NP-completeness Proof,” 2003.

[33] J. Pan and P. R. Jain, “A Survey of Network Simulation Tools: Current Status and Future Developments,” 2008.

[34] OpenSim Ltd., “OMNeT++ Discrete Event Simulator - Documentation.” [Online]. Available: https://omnetpp.org/documentation. [Accessed: 03-Mar-2017].

[35] GNS3 Technologies Inc., “Software | GNS3.” [Online]. Available: https://www.gns3.com/software. [Accessed: 03-Mar-2017].

[36] Mininet Team, “Mininet Overview - Mininet.” [Online]. Available: http://mininet.org/overview/. [Accessed: 03-Mar-2017].

[37] Cisco Systems, “Open Shortest Path First (OSPF),” 1999. [Online]. Available: http://www.cisco.com/cpress/cc/td/cpress/fund/ith2nd/it2442.htm. [Accessed: 10-Jan-2017].

[38] M. Sivabalan and H. T. Mouftah, “On the Design of Link-State Routing Protocol for Connection-Oriented Networks,” *J. Netw. Syst. Manag.*, vol. 9, no. 2, pp. 223–242, 2001.

[39] University of Adelaide, “The Internet Topology Zoo.” [Online]. Available: http://topology-zoo.org/index.html.

[40] Cogentco, “Cogent - Network Map.” [Online]. Available: http://www.cogentco.com/en/network/network-map. [Accessed: 09-Feb-2017].