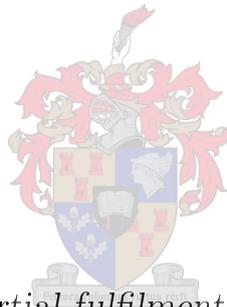


Satellite Network Simulator - Design and Verification using AODV and AntHocNet

by

André-Jan Merts



Thesis presented in partial fulfilment of the requirements for the degree of Master of Engineering(Electronic) in the Faculty of Engineering at Stellenbosch University

Supervisor: Mr. A. Barnard

March 2017

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: March 2017

Copyright ©2017 Stellenbosch University
All rights reserved.

Abstract

Satellite Network Simulator - Design and Verification using AODV and AntHocNet

AJ. Merts

*Department of Electrical and Electronic Engineering,
University of Stellenbosch,
Private Bag X1, Matieland 7602, South Africa.*

Thesis: MEng (Elec)

2016

As access to space increases, the viability of creating satellite networks and dedicated services for satellite clients also increases. However, satellites are expensive to launch and cannot be easily maintained once in orbit, making it beneficial to simulate network performance before launch. Furthermore there is an extensive wealth of research regarding routing protocols available for terrestrial networks that have not been fully researched for satellite networks thus creating a need for a satellite network simulator.

Any network has various ways in which its properties can be interpreted, which means any network metric must be unambiguously defined. Routing protocols can be classified in various ways, showing their inherent advantages, disadvantages, requirements and limitations. Combining these properties with information on the components of routing protocols, an analysis of routing protocols can be made and candidate routing protocols can be chosen. For this research AODV was chosen to allow functional verification and AntHocNet was chosen to show that the simulator is flexible enough to simulate various routing protocol functionality.

Satellite networks were manually analysed with attention given to the key differences between terrestrial and satellite networks. Using this information in conjunction with previous research findings and recommendations, a vast distributed network similar to the Iridium network was chosen as a verification scenario for the simulator. A previously proposed network, called the OuterNet, was chosen to further research the transferability of AODV and AntHocNet to other satellite constellations.

The expansion of SatSim from a basic point-to-point simulator to a network simulator capable of handling complex routing protocols was documented in detail, with clear information given as to which design choices were made and why. The key design drivers were user-friendliness, modularity and re-usability, and the code was structured to allow users to easily adjust and expand the simulator as needed. Due to the limited replicatable results, further testing for the MAC layer channel access for AODV and AntHocNet implementations was performed.

SatSim was successfully expanded to allow for satellite network simulations. Preliminary findings regarding the suitability of AODV compared to AntHocNet for two different satellite constellations are presented.

Uittreksel

Satelliet Netwerk Simulator - Ontwerp en verifikasie met AODV en AntHocNet

(“Satellite Network Simulator - Design and Verification using AODV and AntHocNet”)

AJ. Merts

*Departement Elektriese en Elektroniese Ingenieurswese,
Universiteit van Stellenbosch,
Privaatsak X1, Matieland 7602, Suid Afrika.*

Tesis: MIng (Elek)

2016

Namate die ruimte meer toeganklik word, verhoog die lewensvatbaarheid van die gebruik van satellietnetwerke en toegewyde dienste wat op daardie netwerk gerig is. Satelliete is steeds relatief duur om te lanseer en kan nie maklik onderhou of herstel word na dit in 'n wentelbaan geplaas is nie, gevolglik kan die simulaties van die netwerk voor lansering baie nuttig wees. Daar is verder ook heelwat navorsing oor netwerkprotokolle vir aardnetwerke gedoen wat nog nie volledig bestudeer is vir satellietnetwerke nie en wat dus 'n behoefte skep vir 'n satellietnetwerk simulator.

Enige netwerk se eienskappe kan op 'n verskeidenheid maniere geïnterpreteer word, wat daartoe lei dat alle netwerkmaatstawwe op so 'n wyse gedefinieer moet word dat daar geen dubbelsinnigheid is nie. Roeteringsprotokolle kan op verkeie manier geklassifiseer word wat hul inherente voordele, nadele, vereistes en beperkings teen toon stel. Deur die roeteringsprotokolle se eienskappe saam met inligting oor die komponente van roeteringsprotokolle te gebruik, kan 'n analise van roeteringsprotokolle uitgevoer word en twee kandidaat roeteringsprotokolle is gekies vir toetsing. Vir hierdie navorsing is AODV gekies om die stelsel te help verifieer en AntHocNet is gekies om aan te toon dat die simulator buigsaam genoeg is om verkeie roeteringsprotokolle se funksionaliteit te simuleer.

Satellietnetwerke is handmatig ontleed en die kernverskille tussen aards-en satellietenetwerke wat spesiale aandag geniet het. Deur hierdie informasie saam met vorige navorsing en aanbevelings te gebruik is 'n verspreide netwerk soortgelyk aan die Iridium netwerk gekies om as verifikasie van die simulator te dien. 'n Voorheen voorgestelde netwerk, bekend as die OuterNet, is gekies om verdere navorsing oor die toepasbaarheid van AODV en AntHocNet op ander satellietkonstellasies uit te voer.

Die uitbreiding van SatSim vanaf 'n basiese punt-tot-punt simulator tot 'n netwerk-simulator wat in staat is om komplekse roeteringsprotokolle te simuleer, is volledig gedokumenteer, met alle ontwerpseuses verduidelik en geregverdig. Die hoof ontwerpmaatstawwe van gebruikersgerief, modulariteit en herbruikbaarheid is gebruik, met die kode wat gestruktueer is om uitbreiding van die simulator te vergemaklik. Weens beperkte

herhaalbare resultate is verdere toetsing vir die MAC laag kanaaltoegang en vir AODV en AntHocNet uitgevoer.

SatSim is suksesvol uitgebrei om satellietnetwerke te simuleer. Voorlopige resultate vir die toepaasbaarheid van AODV teenoor die van AntHocNet vir twee satellietkonstellasies word gegee.

Acknowledgements

I would like to thank the following people, without whom this thesis would not have been possible.

- Firstly, my parents Chris and Hilda Merts for their unwavering support and motivation in completing this thesis, especially in the face of many challenges.
- My supervisor Arno Barnard for his valued insights and contributions as well as keeping me on the correct track and consistently motivated.
- My friends, especially my comrades in the ESL who were always cheery and striving hard on various projects. Also my close friends Hendrik, Anita and James, whose company and conversation I truly cherish.

Lastly, I would like to thank all the people who contribute to sites such as StackOverflow in their own free time and made it one of the best resources on the internet.

Contents

Declaration	i
Abstract	ii
Uittreksel	iii
Acknowledgements	v
Contents	vi
List of Figures	viii
List of Tables	ix
Acronyms	xi
Scientific Constants	xiii
1 The Need for a Satellite Network Simulator	2
1.1 Background	2
1.2 Previous Work	3
1.3 Scope of the Research	4
1.4 Objective and Contribution	5
1.5 Thesis Structure	6
2 Literature Review	8
2.1 Definitions of Network Metrics	8
2.2 Routing Methodologies	12
2.3 Routing Protocols Frequently Used for MANETS	18
2.4 Components of Routing Protocols	20
2.5 Overview of Satellite Networks	21
2.6 Review of SatSim V.1	27
3 Routing Protocols and Channel Access	31
3.1 Overview of MAC Layer	31
3.2 Overview of AODV	37
3.3 Components and Functionality of AODV	38
3.4 Overview of AntHocNet	42
3.5 Components and Functionality of AntHocNet	43
4 Expansion and Testing of SatSim	50
4.1 SatSim - Overview of Expansion	50

4.2	Simulator	51
4.3	SimObjects	54
4.4	Testing of Channel Access	60
4.5	Testing of AODV	63
4.6	Testing Of AntHocNet	64
5	Protocol Evaluation	67
5.1	Use of a Distributed Constellation	67
5.2	Baseline Scenario	72
5.3	Comparison of AODV and AntHocNet for the OuterNet	73
6	Conclusion and Recommendations for Future Work	77
6.1	Conclusion	77
6.2	Recommendations for Future Work	78
	Appendices	81
A	SatSim Supplementary	82
A.1	Installation	82
A.2	Configuration and Use	83
B	Routing Supplementary Information	85
B.1	Inherent Issues in MANET routing	85
	Glossary	88
	Bibliography	90

List of Figures

3.1	Successful MAC Handshake	34
3.2	Ant Class Diagram: Showing Base Ant Class and Derived Ants	46
4.1	High Level SatSim Overview	52
4.2	SatSim implementation of the OSI model	55
4.3	OBC UML Class Diagram with Configuration	58
4.4	Generic Frame Class and its Derived Classes	61
4.5	AODV Test 1	63
4.6	AODV Test 2	64
4.7	AntHocNet Test 1	65
4.8	AntHocNet Test 2	65
5.1	Visualisation of the Constellation Showing Four Planes	68
5.2	Representation of the OuterNet	75
B.1	The Hidden Terminal Problem	86
B.2	The Exposed Terminal Problem	87

List of Tables

2.1	Network Metrics	9
2.2	Protocol Layer Challenges for Mobile Ad Hoc Networks (MANETs). Adapted from [11], p3	13
2.3	Comparison of Centralised and Decentralised Routing for MANETs	15
2.4	Comparison of Reactive and Proactive Routing Protocols for MANETs	17
2.5	Summary of Various Routing Protocols	18
2.6	Currently Active Satellite Constellations	22
3.1	Implemented MAC Configurations for SatSim V.2	32
3.2	Custom MAC Implementation	32
3.3	Frame Control Field. Adapted from [34] , p404	35
3.4	Request to Send (RTS) frame. Adapted from [34], p404	35
3.5	Clear to Send (CTS) frame. Adapted from [34], p405	35
3.6	Acknowledge (ACK) frame. Adapted from [34], p405	36
3.7	RREQ Frame. Adapted from [20]	39
3.8	RREQ Flag Field. Adapted from [20]	39
3.9	RREP Frame. Adapted from [20]	39
3.10	RERR Frame. Adapted from [20]	40
3.11	AODV Simulation Parameters	42
3.12	Hello Message Structure. Adapted from [19]	45
3.13	AntHocNet Parameters	49
3.14	Data Sizes of AntHocNet Components	49
4.1	Frame Type Classification	60
4.2	Frame Meta Information	60
5.1	Constellation Satellites Orbital Elements	68
5.2	Constellation End Users	69
5.3	Constellation Simulation Summary	70
5.4	Comparison of AODV and AntHocNet for 1	70
5.5	AODV Reactive Overhead	71
5.6	AntHocNet Reactive Overhead	71
5.7	AntHocNet Proactive Overhead	71
5.8	Baseline Client Satellite	73
5.9	Baseline Client Satellite - 24 hours	73
5.10	OuterNet Satellites Orbital Elements	74
5.11	OuterNet Ground stations	74
5.12	Baseline Client connected to OuterNet for AODV and AntHocNet	75
A.1	Summary of Dependencies	83

LIST OF TABLES

x

A.2 SatSim Network Configuration 84

Acronyms

ACK Acknowledge. 9, 33–36

ANSI Ad Hoc Networking with Swarm Intelligence. 18, 19, 42

AODV Ad Hoc On-Demand Distance Vector Routing. 3, 5, 6, 15, 17, 19–21, 26, 27, 31, 37, 38, 40–43, 50, 55, 59, 60, 63, 67, 69–73, 75–77, 79, 85

AOMDV Ad hoc on-demand multipath distance vector routing. 17

BER Bit Error Rate. 4, 11, 28, 29, 54, 56

CBR Constant Bit Rate. 25

CTS Clear to Send. 9, 33–36

CW Contention Window. 31, 32, 36

DSDV Dynamic Destination Sequenced Distance-Vector Routing. 19

DSR Dynamic Source Routing Protocol. 19

ECC Error Correcting Code. 9, 11, 12, 25, 29, 56, 57

ESL Electronic Systems Laboratory. 2, 27

FIFO First in First out. 59

GUI Graphical User Interface. 51, 79

ISL Inter Satellite Link. 18, 22, 29, 55, 62, 76, 79

LEO Low Earth Orbit. 4, 12, 28, 67, 74

LFN Link Failure Notifications. 71

LOS Line of Sight. 3, 13, 14, 19, 26, 29, 32, 33, 36, 48, 73–75

MAC Media Access. 10, 20, 29, 31–33, 37, 47, 57, 60–62, 67, 70, 76–79, 84–86

MANET Mobile Ad Hoc Network. 3, 7, 8, 13–15, 17–21, 24, 37, 38, 42, 57, 77, 78

NAV Network Allocation Vector. 33, 34, 57, 61

OBC On-Board Computer. 5, 54, 58, 79

OLSR Optimized Link State Routing Protocol. 19, 42

- OOP** Object Orientated Programming. 30, 50, 51
- OSI** Open Systems Interconnection. 5, 13, 50, 54, 55
- PA** Proactive Ant. 45, 49, 71
- QoS** Quality of Service. 11, 38
- RA** Reactive Ant. 25, 45, 71
- RAAN** Right Ascension of the Ascending Node. 54, 68, 73, 76
- RERR** Route Error. 40, 71
- RF** Radio Frequency. 54
- RPA** Repair Ant. 46, 71
- RREP** Route Reply. 38–40, 70, 71
- RREQ** Route Request. 25, 38–40, 70, 71
- RTS** Request to Send. 9, 33, 35
- RWP** Random Waypoint Model. 26, 42
- SIFS** Short Interval Between Frames. 34
- TORA** Temporally-Ordered Routing Algorithm. 20
- TTL** Time to Live. 38, 42
- UDP** User Datagram Protocol. 58
- UML** Unified Modelling Language. 58, 59
- WMN** Wireless Mesh Network. 37, 54
- WSN** Wireless Sensor Network. 13, 15, 21, 24, 25, 54, 57, 58, 80
- ZRP** Zone Routing Protocol. 19, 20, 79

Scientific Constants

Standard gravitational parameter (Earth) : $\mu = 3.986004418(9) * 10^{14} m^3 s^{-2}$

Speed of Light in a Vacuum : $c = 299792458.0 m s^{-1}$

Radius of the Earth : $R_e = 6378136.6 m$

Chapter 1

The Need for a Satellite Network Simulator

1.1 Background

Space is becoming accessible to more users, due to the decreasing relative cost of building satellites, more adaptive markets and the ever increasing knowledge base of the field. Current trends suggest the increase of satellite launches will continue in the future, with more space applications appearing and becoming viable as more users gain access to space. In order to efficiently use these satellites, especially in networks, there must be a platform to study the optimal way for satellite networks to communicate.

When approaching the problems inherent to satellite communication and networks, some novel solutions have become more viable due to more users being involved in space. One such idea that was proposed by the Electronic Systems Laboratory (ESL) at Stellenbosch University is the use of an OuterNet [1], however for such an idea to be financially viable it would need a solid subscription base. In order to justify the cost of subscription users must clearly be presented with the benefits of such a system. While many configurations and routing protocols have been investigated, the vast field of routing protocols has not been explored in the same detail for satellite networks as for terrestrial wireless sensor networks.

It becomes desirable to use this wealth of accrued knowledge, make the necessary adjustments to proven routing protocols and then attempt to use and evaluate them in a satellite environment. However, for many of these routing protocols little to no information is available for their performance in a satellite environment.

As such, there is a need for a new tool that can be used to easily test existing routing protocols in a space-based environment, whilst studying and highlighting the differences between the satellite and terrestrial networks. A key feature of many routing protocols is that values were chosen with specific terrestrial conditions in mind. In order to optimally use these routing protocols with satellites, these values must be evaluated and adjusted if the core assumptions which lead to them are significantly different in a satellite environment.

This research will therefore attempt to implement two routing protocols, each on two different satellite constellations, and evaluate their performance. The research is not primarily aimed at finding an optimal networking solution for a given satellite constellation but rather towards creating a simulation platform on which satellite network protocols can be evaluated. As a secondary objective preliminary performance observations for

both glsaodv and AntHocNet were made.

1.1.1 Satellite Communication Systems

The communications system of a satellite will always be of crucial importance to the satellite's mission - without it the satellite cannot be efficiently controlled via Telecommand or communicate the data it has gathered or is relaying to the end user using Telemetry.

The end-user can differ greatly depending upon whether the satellite mission was undertaken for academical interest (usually a university) or for commercial interests (usually a fairly large or specialist company). In the former case limited budgets mean that it is already difficult to obtain funding for a single satellite and even more so a constellation of them. In the latter case there would have to be proof that there is sufficient benefits to having multiple satellites. Collaboration between multiple satellite entities do occur, with the QB50 project being a good example [2]. Even though the QB50 satellites contribute towards a shared goal, they do not necessarily interact with each other in a networked configuration.

For several university missions, the communication setup may consist of a single satellite and ground station, such as with SunSat and Sumbandilasat [3] [4]. In this point-to-point connection, communication with the satellite can only occur when the two nodes have visibility, or Line of Sight (LOS), of each other. This communication window can be a severely limiting factor, and the previous use of SatSim was to investigate analyse and maximise the data throughput for a given communication window whilst using the AX.25 and FX.25 protocols [5].

The physical setup of a satellite communication system allows multiple configurations for communication, including baud rate, modulation scheme, antenna, and frequency. These and other parameters lead to a very wide variety of different possible settings for any given scenario.

1.1.2 Routing Protocols

Routing protocols for MANETs is a well studied problem for terrestrial networks but has not enjoyed the same attention for space-based applications. Most routing problems stem from two main types, namely distance vector and link state, and new protocols are usually developed to address a specific shortcoming in another protocol or to adapt to a specific subset of scenarios.

There is no clear choice for which routing protocol has the most general applications in a satellite environment, however Ad Hoc On-Demand Distance Vector Routing (AODV) acts as the de facto standard for routing protocols and most other routing protocols are compared to AODV during their conceptual testing.

AntHocNet is a more complex hybrid protocol that covers almost all MANET routing protocol functionality. An implementation of AntHocNet will therefore assist in verifying that the network simulator is sufficiently well-designed to handle a wide variety of other network routing protocols.

1.2 Previous Work

This research acts as a continuation and expansion of research conducted at Stellenbosch University, by Le Roux [5]. Le Roux's work was in part a continuation of Bezuidenhout's

research [6], that was aimed at the selection of modulation schemes and channel access methods for Low Earth Orbit (LEO) satellites. However, an entirely new simulator was developed by Le Roux[5], making use of SimPy in Python rather than DEMSMO-J in Java as Bezuidenhout did. The work would focus heavily on optimising network throughput on a single link basis through the development of a platform called SatSim.

SatSim was verified as an discrete events simulator that could give accurate single link results for a point-to-point connection. This was achieved via the use of dynamic Bit Error Rate (BER) calculations on a packet by packet basis. The AX.25 and FX.25 protocols were tested on the link and a hybrid protocol proposed in order to optimize the communication throughput for that specific communications link.

1.2.1 Previous Simulation Results

As of the time of this research being conducted, there was very limited network simulation data available for either AntHocNet or AODV in a satellite environment. Where acceptable results could be found in literature, such as reported by Liu et al [7], the results were used to verify functional correctness of the protocol simulations as implemented in SatSim.

It should be noted that researchers often make different assumptions regarding a simulation environment for the same scenario. It is therefore important to state which assumptions were used for each simulation to allow the reader to replicate the results.

1.2.2 Other Network Simulators

The choice of network simulator had to satisfy certain criteria, including being freely available, easy to learn, have decent execution times and have great developmental flexibility. An observation from this research is that most networks can see extensive benefits from unique optimizations and configurations, implying that having full control over the network simulator and its objects is highly beneficial. The decision to continue with SatSim was due to the Python and Simpy being well documented tools that could easily be expanded in order to achieve the objectives of this research.

A variety of other network simulators are available, such as ns2, ns3, OPNET, OM-NeT++, and the STK. However, most of these have properties that make them unsuitable as has already been outlined by Le Roux [5]. There are some additional issues that also occur when a full network is considered for simulation, relating to scaling, memory consumption and execution time. Due to the computationally heavy nature of some simulations protocols occasionally have to be modified in order for them to be simulated in worst case conditions.

1.3 Scope of the Research

As access to space and the capabilities of satellites increase so does the feasibility and ease of using satellites in networks. Being able to simulate a satellite network before launch is an attractive option. These simulations will serve multiple purposes, including analysis of the feasibility of networks and testing the performance of existing protocols in a satellite environment.

The goal of this research can be broken into three goals, which will be described below.

1.3.1 Expansion of SatSim

The primary goal of this research is to expand SatSim into a satellite network simulator that can handle complex routing protocols. This will include restructuring the design of the Simulator, optimizing computations wherever possible and to review and adjust all the assumptions that went into the design of SatSim. All the components of SatSim have to be expanded and adjusted to function in a network environment.

An additional contribution to the SatSim environment was adding a channel access method, which resulted in several observations being made regarding the differences between terrestrial and space-based communication systems.

1.3.2 Analysis and Implementation of Routing Protocols

A second research goal was choosing two routing protocols which need to be fully analysed and implemented on the SatSim environment. Two protocols are needed in order to allow additional verification and to allow SatSim to be used to evaluate the performance of two routing protocols in a similar satellite scenario. This includes full testing of the protocols and any optimizations or adjustments needed in order to implement the protocols in the SatSim environment.

1.3.3 Evaluation of Routing Protocols

Lastly, the two routing protocols need to be compared on two different networks in order to evaluate their performance and to attempt to better analyse how SatSim should be used in order to help choose routing protocols for a given satellite network. Observations were made, regarding both the protocols and which properties of the network are significant when deciding on an optimal routing strategy. An additional research contribution resulted from this phase, providing data to indicate when certain protocols may be unsuitable for satellite networks.

1.4 Objective and Contribution

The objective of this research was to create a satellite network communications tool, capable of simulating complex routing protocols and allow research to be conducted on the performance of various networking routing protocols. This tool needed to be highly modular, have reasonable runtime and not be too memory intensive. Furthermore upon verification of this tool the idea was to determine what optimizations and changes to existing routing techniques were possible or needed in a specific satellite environment.

In order to do this, the SatSim network simulation tool was expanded. This included significant changes to the node in the network, such as an On-Board Computer (OBC) for each node, a more complex Open Systems Interconnection (OSI) implementation, the implementation of a channel access layer and several other optimizations and structural changes. This would allow each node to maintain and react to a large amount of state information.

On the new platform AODV and AntHocNet were implemented and results compared to existing simulations where possible. Extensive use of theoretical calculations and simple test cases was made whenever there were insufficient simulations results available.

The final contribution of the expanded tool was an evaluation of OuterNet and the performance of AODV and AntHocNet in such a network as well as evaluating the performance of AntHocNet in an Iridium like network.

This research shows SatSim has been expanded to be a useful and easily expandable tool. Most satellites networks have such unique operating conditions that a general solution can rarely be found. The main benefits of SatSim is that it can be used to assist in discovering the optimal communication strategy for a given scenario.

1.5 Thesis Structure

The chapters in this thesis are structured as follows:

Chapter 1 - Introduces the research problem approached by this research and provides background data and an overview of recent work in the field. The scope and contribution of the research is also described.

Chapter 2 - Introduces the network definitions which are used to evaluate network performance. The chapter gives an overview on different routing classifications and their inherent properties. Using this information along with a short section describing currently used routing protocols two candidate protocols are chosen for this research. Satellite networks and their salient properties are investigated so that two candidate networks can be selected for this research. The chapter concludes with a review of the previous work performed on SatSim and outlining the design principles which will be used to expand it further.

Chapter 3 - Describes the implementation of a channel access method on the SatSim environment and the challenges presented thereby. It then outlines AODV and how it was implemented, as well as how its overhead is quantified. The chapter concludes with a description of the AntHocNet and its adaptation for the SatSim environment.

Chapter 4 - Details the expansion of SatSim, describing and justifying the design decisions made. The Simulator structure is covered in detail along with descriptions of the various subcomponents that are used thereby. The SimObject class and its expanded behaviour is also explained in details. The testing scenarios for the Channel Access, AODV and AntHocNet are also presented along with their results and some observations.

Chapter 5 - Gives the results for the verification of SatSim on an Iridium like network. The testing procedure and results are described and an analysis of the performance of AntHocNet in such an environment is made, along with some general observations. A Baseline scenario is then tested for both AODV and AntHocNet, which is then compared to their performance when the OuterNet is introduced. The two networks are evaluated against each other for this specific network and observations and suggestions are made.

Chapter 6 - Concludes this research by describing what has been achieved and what could not be achieved. Several suggestions are made towards future work, detailing both the suggest further expansion of SatSim and new research which could be conducted on the SatSim environment.

The appendices contain supplementary information on the use of SatSim and routing protocols.

The appendices in this thesis are structured as follows:

Appendix A - Contains supplementary information regarding SatSim. Detailing its installation and options currently available for use.

Appendix B - Outlines some inherent properties of MANETs and the potential issues that they raise.

Chapter 2

Literature Review

The following chapter gives an overview of the the definitions of metrics needed to evaluate and describe MANET networks, as well as the information needed to fully design and evaluate routing protocols and how they are classified. The main classes of decentralised routing is then expanded upon.

In order to fully cover the details of decentralised routing reactive, proactive and hybrid routing is discussed and evaluated. Attention is given to the inherent properties, advantages , disadvantages, requirements and limitations of the different types, which will allow better understanding of when a specific routing protocol is preferable.

A selection of routing protocols is presented and their suitability to satellite networks is briefly discussed, along with a short section which details the typical routing components which are used by these routing protocols. The discussion of routing protocols concludes with a discussion on why the routing protocols which were chosen for this research were used.

To further describe both networks and protocols in terms of the unique needs and characteristics of satellite networks , satellite networks and their operation must be understood. To do this the research presents an overview of satellite networks, which includes a concise summary of some existing and planned satellite networks and a discussion on the salient characteristics of networks and how these change in a satellite environment. The section concludes with the choice regarding which networks will be investigated by this research.

Finally, the chapter concludes with an analysis of the previous work which led to the development of SatSim and the research which was performed on the simulator in [5]. The main design drivers, limitations and decisions are highlighted and their reasons for being maintained or removed are given, which will be used to guide the expansion of SatSim towards a simulator that will be useful to the satellite community.

2.1 Definitions of Network Metrics

To ensure that meaningful comparisons and conclusions can be drawn using the SatSim environment, the metrics for network analysis must be rigidly defined. These definitions are stated to indicate how the results in this research should be interpreted and can be independently recreated. The definitions given below were derived and adapted from a multitude of sources, and are foremost meant to be an accurate representation of what the values mean and how they are determined in the SatSim environment.

A key point raised by [8] p.8 is that the measurement for certain metrics should be done externally to the perspective of the individual nodes, which is why the DataManager class aref used. It was also noted that it should always be stated whether means, variances or distributions is used.

2.1.1 Summary of Network Metrics

Table 2.1 gives a summary of the metrics discussed for this research as well as the units they are presented with, unless otherwise stated. A possible affect of the metric on a constellation is also given.

Table 2.1: Network Metrics

Metric	Units	Affects
Latency	millisecond	Timing, Full Duplex Communications
Throughput	bits/second	Download Times
Jitter	millisecond	Multi-directional Communications
Quality of Service	ratio	Communication quality, Throughput
Overhead	bits	Throughput, Congestion
Coverage	square kilometre	Accessibility of End-Users
Route Discovery Time	millisecond	Full Duplex Communications
Guarantee of Delivery	ratio	Crucial and Prioritised messages

2.1.2 Throughput

Throughput is used to indicate the rate at which data is effectively being transmitted from the original source node to the final destination node. This value is directly obtained by defining two points (usually referred to as the data source and the data sink), and dividing the amount of data which passed between them by the amount of time it took the data to move from point A to point B.

$$\text{Throughput} = \frac{\text{Data Transmitted (bits)}}{\text{Total Time for Transmission (seconds)}} \quad (2.1.1)$$

However, the value given by the equation above can be interpreted in multiple ways, which may make the same link seem better or worse. For example, you may have a data frame consisting of application data and an extended header file. For different protocols with different header sizes, the throughput may be the same but one of the protocols is transmitting data at a faster rate than the other. In order to get an accurate, comparable value for throughput, only application data should be used for the *Data Transmitted* value. This prevents the problem where the method of interpretation of data can hide the true value of interest, which is how fast useful data is being transmitted. The start time is chosen as the time when the frame was created and the end time is taken as the time at which the sinks application layer has access to the data.

All multi-hop strategies require some form of overhead in order to ensure data is correctly routed over multiple hops from some source node to the eventual destinations. This overhead does not just include the various routing headers and Error Correcting Code (ECC) attached to the packets themselves, but also all handshaking data (often RTS, CTS, ACK frames used for channel access) and wait times incorporated (for a more

thorough description of this see Media Access (MAC)) as well as the additional overhead costs of retransmission (which can be highly significant over lossy links).

When these distinctions aren't made, the effective throughput measured by an application might be considerably lower than the given throughput specified by the protocol. In this study only application data being transferred is used when analysing the throughput - which implies a much lower throughput value will be calculated than other studies, which will be closer to that measured by the applications on the communicating devices.

2.1.3 Latency

Latency is used to quantify the time it takes a transmission to propagate from node A to node B. The first time interval starts at the moment of transmission and the second time interval is taken when the packet has been fully received by the destination. This will usually be given in milliseconds, even in the satellite environment. The latency for a given network can make it unsuitable for certain applications (such as warning systems or interactive peer to peer systems), or otherwise highly undesirable (such as the use of peer to peer interaction).

Latency can be approximated as the time difference between the packet being transmitted from the source and it being received by the sink. However, this time difference can be approximated using the equation below.

$$\text{Latency} = \text{Propagation Delay} + \text{Switching Delay} + \text{Processing Delay} \quad (2.1.2)$$

Propagation delay for a satellite to ground station connection is given as the distance between the two nodes which should be close to the approximate altitude of the satellite divided by the speed of light. Propagation delay for inter satellite links is given as the distance between the two satellites divided by the speed of light.

The switching delay is only relevant for certain satellites and refers to the time needed for the satellite to switch from receiving mode to transmitting mode.

The processing delay refers to the time needed to process the data by the transmitter and is equal to the data in bits divided by the baud rate for the transmitter.

The equation above does not account for retransmissions and collisions, which can make effective latency far lower. In order to measure the effect of lossy links and retransmissions, the jitter metric can be used.

Latency is also often referred to as delay or end-to-end delay.

2.1.4 Jitter

Jitter is the difference in time between the arrival of one packet and the next. While jitter is affected by the network topology, it can also be affected by various other factors, some of which aren't modelled in this simulation, such as the relevant processor overhead associated with the buffering and sending of data and random events.

The equation to calculate jitter is given below, with N being the number of packets transmitted.

$$\text{Jitter} = \frac{\sum_{i=2}^N t_n - t_{n-1}}{N} \quad (2.1.3)$$

High jitter may indicate a link is operating a lossy or highly contested channel, rendering it unsuitable for certain applications.

Jitter is also referred to as the inter-packet delay.

2.1.5 Quality of Service

Quality of Service (QoS) refers to how lossy a link or path is. In order to simulate this in SatSim, a BER is associated with each data packet during propagation with a value determined based upon the network topology at that instant. BER is defined for the equation below. Other studies occasionally make a distinction between Bit Error Rate and Bit Error Ratio where Bit Error Rate is rather defined as the amount of bit errors per a specific time interval.

$$\text{Bit Error Rate} = \frac{\text{Corrupted Bits}}{\text{Transmitted Bits}} \quad (2.1.4)$$

A bad quality of service will result in more packets being unusable which will reduce throughput, increase jitter and effectively reduce average latency. For more information on how the bit errors are assigned to packets, refer to subsection 4.3.3.3.

From a user's viewpoint, QoS can be a deciding factor in deciding whether they wish to participate in the satellite services, especially for services such as satellite internet or global communications.

2.1.6 Overhead

Overhead is all additional data needed to ensure application data can safely be sent from the source and received at the sink. This overhead can be headers which enclose the application data in the frame structure or are needed to create and maintain routing tables. It also includes additional handshaking frames used to control channel access.

$$\text{Overhead} = \text{Total bits transmitted} - \text{Application Data} \quad (2.1.5)$$

Overhead can be broadly defined as the total number of control bits transmitted, or given as a ratio of the control bits needed to transmit data bits [8].

2.1.7 Packet Delivery Fraction

Packet Delivery Fraction refers to the amount of packets which successfully arrived relative to total amount of packets which were sent. It is closely tied to the quality of service of a link and may indicate that specific routing protocols with additional ECC may be beneficial.

$$\text{Packet Delivery Fraction} = \frac{\text{Packets Transmitted} - \text{Packets Successfully Received}}{\text{Packets Transmitted}} \quad (2.1.6)$$

2.1.8 Route Discovery Time

Route discovery time refers to the time a given protocol takes to establish a new route for a data packet. This value can vary significantly based upon the type of routing protocol being used, while also having a significant effect on average latency. Route discovery time also becomes more significant when link failures occur frequently and is far less important for quasi-static networks.

Route discovery time is also referred to as route acquisition time in literature [8].

2.1.9 Coverage

Coverage is the amount of surface area in which end-users can access the network. Typically a single satellite's altitude will be a deciding factor in how much ground coverage it gives, with a LEO satellite covering a few square kilometres while three geosynchronous satellites can provide near global coverage. However, increased coverage via altitude will have a negative effect on latency, as this will increase propagation delay. The antenna type and beam width can also have a very significant effect on the footprint of the network. The amount of coverage a network gives can differ vastly based upon the network configuration and has to be matched to the needs of the network.

Coverage is also referred to as the footprint of the network, especially when referring to satellite networks.

2.1.10 Guarantee of Delivery

Guarantee of Delivery refers to taking additional measures to ensure that a specific packet reaches its destination. This could mean increasing the allowed limit of retransmissions, using additional ECC or even flooding the network via the use of multiple routes.

This idea is not to be confused with prioritized transmission, in which certain transmissions gets preferential treatment, allowing them to be moved ahead in buffer queues. Guarantee of Delivery is needed for certain services and the additional costs thereof should be viewable by the simulation.

2.1.11 Protocol Efficiency

While all the other metrics are externally measured, the protocol efficiency is given by a node's internal measurement of its effectiveness [8]. This value is determined by using only information available to the node, which may occasionally give misleading results but will also give an accurate reflection of how an end-user will view the communications link.

2.1.12 Hops

When a packet has to move across multiple nodes in order to reach a destination, the transmission towards a relay node is counted as a hop. Hops are often viewed as a cost to a route since the additional switching delays and processing delays are often very costly.

For this research, a path refers to a multi-hop route from a source to a destination while a link refers to a direct connection between two nodes.

2.2 Routing Methodologies

Routing protocols can be classified according to a variety of different criteria, in which its functionality is analysed to assign it to a specific classification. Protocols in the same classification will usually share some inherent advantages and disadvantages. In this section the different classifications of routing protocols will be outlined along with a general description of each classification. Wherever possible naming conventions will be

explained to ensure the reader will be able to correctly associate the classification with what is currently being used in literature.

2.2.1 Definition of MANET and Ad-hoc Networking

The term MANET actually refers to a subset of ad-hoc networks, however the two terms are sometimes used interchangeably. The following sections define the two terms and how they are used in this research:

2.2.1.1 Ad-Hoc Networking

Ad-Hoc networks are often defined as a collection of wireless mobile hosts forming a temporary network without the aid of any established infrastructure or centralized administration [9]. However, a clear distinction is often made when explicitly referring to these networks as MANETs, although literature may often assume ad-hoc networks refer to MANETs. To avoid possible confusion, this research will explicitly refer to ad-hoc networks with mobile nodes as MANETs.

2.2.1.2 MANET

For this study, a similar definition of MANETs as the following adapted from [10] is used:

MANETS: A Mobile Ad-Hoc Network is a network of mobile nodes whom communicate with wireless interfaces to exchange data. The network's topology changes with no form of central administration or prior planning.

This leads to the network having a dynamic topology, with the state of the network at a given instant being referred to as a *snapshot*. A difference between MANETs and Wireless Sensor Network (WSN) is that most MANETs rely on intermediate nodes to communicate with nodes they do not have direct LOS with or that are too far away for effective communication [11]. In the MANETs scenario, it often becomes necessary for nodes to organise themselves in order to compensate for the lack of fixed infrastructure or central administration. The OSI protocol layers, initially designed for wired networks, also faces unique challenges as shown in 2.2.

Table 2.2: Protocol Layer Challenges for MANETs. Adapted from [11], p3

Protocol Layer	Challenge
Physical Layer	Rapidly Changing Links
Channel Access	Collisions and Fair Access, see B.1.2, B.1.3
Network Layer	Handle Complex Paths
Transport Layer	Packet Loss and Delay
Application	Random Disconnection and Reconnection

2.2.1.3 Multi-Hop Networks

A multi-hop network refers to a network where nodes frequently use intermediate nodes or Neighbour in order to communicate with other nodes. This could be either because there is no LOS between the two nodes or because the distance between the two nodes makes communication impossible or too lossy. Most satellite networks will be multi-hop networks.

2.2.2 Decentralised versus Centralised Routing

Routing protocols can be classified as being centralised or distributed. Both types have inherent advantages and disadvantages, as well as physical requirements and limitations. Since satellite networks represent an extreme subset of the MANET problems, certain viable network types for terrestrial networks may have to be completely eliminated from consideration. While both types have their inherent advantages and disadvantages, the physical requirements for each type of protocol may have a far more significant role.

2.2.2.1 Centralised Routing

For centralised routing all nodes have the same routing information which they received from a single source.

An example of a centralised routing protocol would be the LEACH (Low-Energy Adaptive Clustering Hierarchy) protocol proposed by [12]. The network described therein assumed a large number of sensor nodes that act only as data gatherers and need to transmit data back towards a single source. A significant physical requirement for such a network is therefore that some single node should have LOS with the entire network, which is not always viable for MANETS, much less the more extreme case of satellite networks.

This scenario does not occur in the satellite environment for very obvious reasons (with the exception of some hierarchical routing protocols which are discussed in subsection 2.2.2.3), but some adjustments can be made in order to allow pseudo-centralised routing.

Since a complete view of the network would be too hard to obtain, extensive use of propagation programs could be made in order to predict the topology. Since the ground station can only communicate with the satellites for a limited window and the data expires, the ground stations will then have to send multiple snapshots of the network's topology as well as a linked list of times at which they are relevant. This would allow each node to have a routing table which should be reasonably accurate at all times.

However, this could lead to some errors as well as generating significant overhead on the ground station to satellite communication link. This problem can be mitigated by relaying the datasets from visible satellites, but this will of course increase overhead even further and lead to additional complexities.

Another issue raised by [12] is that multiple sensors would need to transmit back towards this sink. This limits the effective bandwidth available, which is already limited in the satellite environment.

2.2.2.2 Distributed Routing

Distributed routing refers to a protocol where each node constructs and maintains its own routing table by exchanging information with other nodes in the network. No two nodes are therefore guaranteed to have the same routing table and no single node is crucial to the functionality of the system. However, significant overhead will be generated as each node needs to exchange data in order to learn more about the topology of the network at any given instant.

Many of the protocols make use of complex strategies for and information sharing, diffusion and exchange. A key issue with distributed routing is relative ignorance of individual nodes of the state of the network, which creates the need for broadcasts in order to efficiently learn of the network topology.

Distributed routing protocols can be either reactive or proactive, with the concepts being defined in 2.2.3.

Distributed routing is also referred to as decentralised routing.

2.2.2.3 Hybrid Routing

A hierarchical routing approach has been proposed by [13], [14] specifically for satellite networks. The approach will function as a hybrid of centralised and distributed routing.

In this approach, LEO satellites will be grouped together due to being inside the footprint of a given MEO or HEO. The higher satellite will act as the central administration point for the group of LEO satellites. Each satellite in the group will then have the same routing information as every other satellite in the group, with the routing information only having to be forward to the MEO or HEO satellites. It is also possible that the more powerful higher satellites will be able to use propagation models in order to estimate the state of the network topology. Some of the challenges emanating from this approach would be the relative cost and the handover process when LEO are near the borders of multiple footprints.

A potential issue key points here is the accuracy in simulation - while in reality there will be the propagator's version of where the satellite is and the actual position of the satellite, in the simulation these will be one and the same. Certain errors which can occur in reality are therefore not likely to surface in simulation.

2.2.2.4 Summary

Table 2.3: Comparison of Centralised and Decentralised Routing for MANETs

Property	Centralised	Decentralised
Needs central administration	Required	Not needed
Individual nodes important	Incredible	Negligible [10]
Bandwidth Limited	Limited to sinks	Spread over network

When analysing table 2.3 , some of the disadvantages have a far heavier weight than others. The difficulty of putting some form of central administration in place for a large scale satellite network is the main deciding factor. While this can be offset using the hierarchical routing strategy, the difficulty of creating the necessary system and the limitations placed upon throughput make distributed routing a more attractive alternative. This research will therefore only consider decentralised routing protocols as viable for satellite networks.

2.2.3 Reactive versus Proactive Routing

Distributed routing protocols will use either proactive, reactive or a hybrid functionality. Of the two main classes, proactive routing protocols has had more research done in the WSN field [15] but several reactive routing protocols such as AODV have also been popular.

This section includes a comparison of the two different types, however unlike the distinctions between centralised and distributed routing protocols there is no clear decision which can be made to the inherent requirements. Both reactive and proactive routing protocols have unique advantages and disadvantages which make them more suited to

certain scenarios than others. In order to be usable for a wide variety of scenarios, SatSim needs to be able to simulate reactive, proactive and hybrid protocols.

2.2.3.1 Proactive Routing

Proactive routing protocols refer to protocols where routing information is constantly gathered and exchanged regardless of network activity. Additional information exchanges are also triggered by sudden topology changes or link breakages [11].

Proactive routing therefore implies a strategy where network information is gathered at fixed rates and processed in order to generate routing tables. At the very least this will usually involve the use of Hello Message in order to establish the visible neighbours. Some protocols such as AntHocNet also pro-actively exchange information over the network through diffusion processes - for instance node A may be visible to node B. Node A can see node C, but node B cannot. A will inform B of its route to C, giving B a route to C via A.

Due to the constant information exchanging inherent to these strategies, proactive routing strategies will usually have higher total and average overhead than reactive strategies. This will also not fluctuate as severely with network activity. However, when a route to a specific node is needed, it is more likely to be readily available, which can have positive effects on latency and throughput, especially for sporadic data transmissions to multiple destinations.

Proactive routing is also referred to as table-driven routing.

2.2.3.2 Reactive Routing

Reactive routing protocols only create and maintain routes when an active communication session requires them [11]. Such protocols will then usually flood the network with broadcast messages, which will be broadcast again by additional nodes until a route is found. Additional information must be used to avoid resending and reprocessing superfluous messages.

A reactive routing strategy will therefore do nothing when there is no pending data and then send out control packets during the route discovery process. Once one or more routes have been established, the protocol will usually then execute its route maintenance process[15].

Reactive routing will usually have less total overhead than proactive routing but will have larger spikes of activity whenever new routes are required. This makes it highly effective when a great deal of data needs to be sent along single routes and less so when routes expire frequently and are needed often.

Reactive routing is also referred to as on-demand routing.

2.2.3.3 Hybrid Routing

Hybrid routing strategies make use of a combination of reactive and proactive functions in order to create and maintain paths. This is usually done in such a way as to mitigate the downsides of each while maintaining some of the core strengths.

AntHocNet is an example of a Hybrid routing strategy. It uses proactive functionality such as Hello Message to constantly maintain lists of nearest neighbours and proactive ants to main and explore new routes. However, reactive functionality such as reactive ants and repair ants are used to find new routes.

2.2.3.4 Summary

As can be seen in table 2.4 , reactive and proactive routing will favour different applications and scenarios at different times. As such, in order to allow SatSim to be an efficient tool for simulating a wide variety of satellite networks, it needs to be able to simulate reactive, proactive and hybrid protocols.

Table 2.4: Comparison of Reactive and Proactive Routing Protocols for MANETs

Property	Reactive	Proactive
Routes	On Demand	Always
Power Consumption	Lower	Higher
Overhead	Lower	Higher
Route Discovery Time	Critical [8]	Minimal Effect
Latency	Slightly Higher	Slightly Lower
Jitter	Slight Higher	Slightly Lower
Route Maintenance	Negligible	Constant
Quality of Service	Slightly Lower	Slightly Higher
Route Repair	Lower	Higher

2.2.4 Additional Routing Classifications

The classifications discussed above are the main criteria of interest for this research. However, there are also additional sub classifications which may be applicable based on the user's needs for a specific application.

2.2.4.1 Multipath Protocols

Multipath protocols create multiple paths between two nodes, as opposed to only creating a single path. AODV is an example of a single path routing protocol. To gain the advantages of multiple paths, Ad hoc on-demand multipath distance vector routing (AOMDV) was proposed [16]. Most proactive protocols will be multipath while reactive protocols can favour single path protocols.

2.2.4.2 Power Aware Routing Protocols

Power aware routing protocols use information based upon the power available to each node to make or help make routing decisions [15]. Nodes that are battery driven or have limited supplies of power are referred to as energy-constrained. In order to simulate this, far more state information and calculations for the use of power would be needed. While this may be of interest for future research or specific scenarios it will not be evaluated in this research.

2.2.4.3 Link State Routing

For link state routing, each node maintains a view of the network topology with a cost for each link [17]. Nodes will exchange information by periodically sending link cost information during a flooding process.

2.2.4.4 Distance Vector Routing

For distance vector algorithms, each node maintains a set of distances for each destination over all of its available neighbours [17]. In order to keep the distance estimates up to date, each node will periodically broadcast estimates of the shortest distance information to each of its neighbours. For Inter Satellite Link (ISL) distance is not as severe as in terrestrial connections, since there are far fewer sources of noise in space and moving more or less in the same atmosphere has a far less significant effect on the attenuation of the signal.

2.3 Routing Protocols Frequently Used for MANETS

The following section covers some of the routing protocols often used by MANETs and analyses them while using the information given earlier in this research to make an informed choice regarding which protocols will be most suited for this research.

Most routing protocols are expansions or changes upon earlier protocols with specific scenarios or deficiencies in existing protocols in mind. It therefore becomes possible to match routing protocols to compatibility with satellite networks.

2.3.1 Summary

Table 2.5 provides a summary of different routing protocols and what they can be classified as. An ideal choice for the evaluation of SatSim would cover most routing functionality.

Table 2.5: Summary of Various Routing Protocols

Routing Protocol	Type	Multipath
AODV	Reactive	No
AntHocNet	Hybrid	Yes
ANSI	Hybrid	Yes
DSDV	Proactive	No
DSR	Hybrid	No
OLSR	Proactive	No
TORA	Reactive	No
ZRP	Hybrid	No

2.3.2 ANSI

Ad Hoc Networking with Swarm Intelligence (ANSI) is a hybrid routing protocol proposed for hybrid ad hoc networks. ANSI makes use of elements from both AODV and AntHocNet [18].

Pure MANET nodes in ANSI use only reactive routing and choose routes deterministically while nodes belong to the more infrastructures, fixed network makes use of both proactive and reactive routing to perform stochastic routing.

ANSI has many of the same routing elements as AntHocNet.

2.3.3 AntHocNet

AntHocNet is a hybrid routing protocol which was proposed by [19] as an extension of the ant colony optimization strategy specifically for MANETs. Refer to 3.4 for a detailed description of this protocol and its implementation in the SatSim environment.

One variation which may solve some of the issue caused by a quasi-static network is HopNet , a modified protocol which combines AntHocNet with aspects of Zone Routing Protocol (ZRP).

2.3.4 AODV

AODV is a reactive routing protocol which was proposed by [20]. Refer to section 3.2 for a detailed description of this protocol and its implementation in the SatSim environment.

There has been extensive research regarding the performance of AODV and it is often used as the benchmark to compare new routing protocols , with [19] referring to it as the de facto standard.

2.3.5 DSDV

Dynamic Destination Sequenced Distance-Vector Routing (DSDV) [17] is a distance based routing protocol where each node broadcasts information to the other nodes to allow all nodes to maintain routing tables. Distance is the main metric to determine routes, which is not often the case for satellite networks, where LOS is often more important. DSDV does not perform well in a highly dynamic network.

2.3.6 DSR

Dynamic Source Routing Protocol (DSR) [21] is a hybrid routing protocol which makes use of on-demand elements in order to find and maintain routes. It makes use of a discrete Route Discovery and Route Maintenance processes. Both of the processes operate on demand. Multiple routes are stored towards a destination, which means that upon a link failure another route can be tried. However this also presents a new potential problem, known as Route Reply Storms (as opposed to Route Request Flooding) in which multiple nodes keep sending Route Replies. DSR is similar to AODV but has less additional information available.

2.3.7 OSLR

Optimized Link State Routing Protocol (OLSR) [22] is a proactive algorithm that periodically exchanges messages to allow all nodes to correctly estimate topology information.

Some variations of this protocol include OSLR-H and OSLR-SAT, which makes use of satellites to broadcast their terrestrial networks messages. The terrestrial network is vastly helped by the satellites coverage, covering far more nodes than a single terrestrial node would be able to on its own.

2.3.8 TORA

Temporally-Ordered Routing Algorithm (TORA) is a routing protocol that favours multiple, loop free routes that can be easily found above locating shortest path routes. Steps are

also taken to minimize the communication overhead. Each node only explicitly initiates a query when it needs to send data towards a specific destination.

Its advantages include multiple paths being created, good performance in dense networks and disadvantages include not being scalable and being outperformed by both AODV and other protocols.

2.3.9 ZRP

ZRP [23] is a reactive routing protocol that works in part by grouping the nodes in the MANET into zones that can then communicate with each other on a zone by zone level. Several strategies can then be implemented in order to increase the effectiveness of the communication, such as using proactive methods between the various zones and reactive methods within individual zones.

2.4 Components of Routing Protocols

Although the above routing protocols differ significantly in how they function and how they achieve their functionality, most of the building blocks of a protocol can be grouped and classified according to some basic properties. These are given below and it would be desirable that the simulator can provide support or be verified for all the listed components.

2.4.1 Hello Messages

Hello Message are short messages intended to indicate to a node which Neighbour are currently visible. Hello Messages are broadcast to all visible nodes, and since they are usually quite small they may sometimes ignore the MAC layer functionality, resulting in collisions.

2.4.2 Failure Notification Messages

Failure messages are used when a node detects a link break, either through a failure to transmit a specific message or some other means such as the MAC layer. This message must contain enough information for any node to determine which routes are no longer usable, and also which other nodes need to be informed. This can lead to considerable overhead in the case of a single node reset at a critical point.

2.4.3 Routing Tables

Each protocol will interpret the information it has received or exchanged with other nodes to create some form of routing table. To put it simply the routing table will contain information as to what the next destination should be to eventually at some final destination. They can differ vastly in what information is contained in them and how the validity of the data is shown.

2.4.4 Proactive Elements

Proactive elements are used to either share information or to gather new information regarding the network, regardless of the current needs of the network. This can be used

to explore new routes or to ensure certain connections are still valid. Proactive elements are usually periodically sent regardless of network activity.

2.4.5 Reactive Elements

Reactive elements are used to gather very specific data, usually to allow a specific end-to-end-route. Reactive elements are only sent in response to network activity.

2.4.6 Sequence Numbers

Sequence numbers are used to guarantee Routing Freedom in a given routing protocol. This is achieved by using the sequence numbers to order data, so that the new or fresh data can take precedence over stale or old data. It also helps prevent old or stale data from accidentally being used to update nodes incorrectly. When combined with unique id's, this can be used to determine the freshness of data and to ignore or discard stale data.

2.4.7 Load Balancing Mechanisms

Load balancing mechanisms are used to spread the data load over multiple paths in order to avoid congestion and thereby increase throughput. Certain single path protocols have to have additional functionality to accommodate this, while multipath protocols may automatically contain such functionality by forwarding data stochastically rather than deterministically.

2.4.8 Choice of Routing Protocols

The choice of routing protocols has to achieve two different goals. Firstly, the routing protocols must be chosen so that the majority of proactive and reactive functionality can be tested on the SatSim platform. If this is achieved it should ensure that any future additions of routing protocols will not be too hard to achieve. Secondly, at least one of the routing protocols should have some replicable results that will serve as additional verification of the SatSim platform.

The hybrid protocol AntHocNet was chosen due to it being a very versatile protocol that covers almost all routing functionality. It is also relatively complex and memory intensive, that suggests that if AntHocNet can function in the SatSim environment it should be able to simulate most routing protocols.

The reactive protocol AODV was chosen since it has a wealth of information available and is used by most other routing protocols as the baseline to compare their protocol to.

2.5 Overview of Satellite Networks

Satellite networks are MANETs by nature and can be viewed as highly changeable WSN with many satellites acting as sensors to gather data from the environment. They have several unique or specifically significant issues however, due to the high speed at which the nodes move and the possibility of additional obstructions. There are also less sources of noise present in space than there are in terrestrial networks.

In order to evaluate the performance of certain protocols for a specific scenario, the unique properties of a specific network must be analysed and defined first. Typical network performance analysis metrics were discussed in section 2.1, while the design parameters for networks will be described after giving a summary of satellite constellations.

Satellite constellations are very expensive to implement and require a large amount of resources both in terms of manpower to create operate and maintain. Orbital space is also a valuable resource, with certain orbits being more desirable than others. The increasing prevalence of space debris further increases congestion for optimal orbits and severely limits amount of favourable orbits available whilst also also raising de-orbiting as a critical issue.

Each satellite constellation needs a significant amount of start up capital, a dedicated communication band and operational costs, which needs to be favourable compared to its benefits.

It can be noted that satellite constellations are often configured and launched as planes with set ISL .

In order to illustrate the currents needs and expectations from satellite constellations the section below includes typical uses of satellite constellations, including scientific constellations (A-train)[24], successful commercial constellations(Globalstar) and failed commercial constellations that managed to reinvent their business model and become successful using modern technology(Iridium).

2.5.1 Satellite Constellations

The following section summarises information on existing satellite networks. This shows realistic ballpark figures to see the typical conditions satellites networks are expected to function under.

2.5.1.1 Summary

Table 2.6: Currently Active Satellite Constellations

Name	Satellites	Approximate Latency
A-Train	6	23ms
Iridium	66	270-390ms
Globalstar	84	60ms
Inmarsat	20+	15-60ms
Orbcomm	44	450ms

2.5.1.2 A-train

The A-Train(Afternoon constellation) consists of six active Earth Observational Satellites placed inside a sun synchronous orbit on the same orbital track [24]. The satellites are placed so that their observations can be combined to construct high-definition three dimensional images of the Earth's atmosphere and surface.

Some of the main goals of the constellation include studying atmospheric composition, the carbon cycle, climate change, water and energy cycles, weather, changes in the global Earth system and the earth surface and interior.

2.5.1.3 Iridium

Iridium is meant to provide data connection to various mobile users, such as voice and text data for cellular phones. It was developed by Motorola. The networks consists of 66 satellites, which is enough to provide global earth coverage. The 66 satellites are placed within 6 orbital planes with each plane containing 11 satellites and being 30 degrees apart.

An issue with Iridium was that the entire constellation had to be in place before its functionality could be properly utilised, several ramping up capital costs before any return on investments can be achieved.

2.5.1.4 Iridium NEXT

Iridium NEXT [25] is a planned second generation satellite constellation that will supplement and expand the existing Iridium network and services.

Iridium NEXT will consist of 66 cross-linked LEO satellite which will include full global coverage (including oceans, airways and polar regions - most geostationary constellation do not provide coverage above 70 degrees). The configuration will be supplemented by extensive ground level architecture.

2.5.1.5 Globalstar

Globalstar [26] is a satellite constellation that offers users telephonic communications. It combines satellites with ground stations, known as GlobalStar Gateways. Users have to be in range of a GateWay in order to make use of the network.

What is noteworthy of the Globalstar System is that parts of the constellation has become inactive, causing users to only have functionality when certain satellite are in range. To help users with this Globalstar store created an Optimum Satellite Availability tool which will tell a user when a satellite will be well positioned for a specific geographical position.

2.5.1.6 Inmarsat

Inmarsat [27] provides global mobile and telephone coverage using 12 geostationary satellites, of which 11 are still operational. Inmarsat offers a a wide variety of services using different satellite constellations , including the Global Xpress service, which is a global high-speed broadband network. It also offers M2M, a global machine-to-machine service to monitor and manage remote assets.

2.5.1.7 Orbcomm

Orbcomm [28] is a company which offers various data communication solutions and has a network of 31 LEO satellites and 16 ground stations that act as additional gateways for communication. Many of the satellites launched by the company are no longer functional. The ORBCOMM systems have an inherent latency that makes them unsuited for timing critical applications.

2.5.2 Salient Characteristics. Adapted from [8])

As explained earlier in this chapter, MANETs have several characteristics that distinguish them from the more traditional, hard-wired WSN. Some characteristics needed to be

adapted for the Satellite environment and details of the adaptations made are discussed in the following paragraphs.

1. **Dynamic Topologies:** the nodes will move in fixed, arbitrary orbits, leading to a changing network topology. This movement is less random and more predictable than in most terrestrial MANETs meaning snapshots of the network can be calculated once and considered constant for multiple runs while other parameters are being adjusted.
2. **Bandwidth-Constrained:** the wireless links will have significantly lower capacity than the hard-wired counterparts. Throughput will also be influenced by additional effects, such as multiple access, fading (worse due to long distances), noise (better since there are significantly less sources of ambient noise in space) and interference. True bandwidth is still often considerably less than the transmitter radio's maximum transmission rate. It is suggested that this low link capacity will lead to congestion being the norm rather than the exception, highlighting the importance of maximizing throughput and effective load balancing techniques.
3. **Energy constrained operation:** the satellites in the networks will rely partly on exhaustible means for energy, as well as having limited recharge capability via the use of solar panels. Energy conservation can therefore be considered a significant factor for optimization.
4. **Limited Physical Security:** MANETs are in general more prone to physical security threats such as spoofing, eavesdropping and denial-of-service. Satellites also have the very real possibility of losing one or more satellites due to collisions, either with defunct satellites or debris. There can also be random resets due to radiation upsets or solar flares, leading to the need for robustness against single points of failure.

2.5.3 Network Context

In order to give accurate and useful data regarding the performance of a routing protocol, the full context of the characteristics for the network must be given, and then run under various different values [8]. Some of these are not applicable when only using Satellite propagation models, since the movement of the satellites will be the same for multiple runs due to the deterministic nature of satellite orbits.

2.5.3.1 Network Size - Number of Nodes

Networks with few nodes are usually referred to as sparse, whilst networks with many nodes are referred to as dense. When considering the number of nodes in a satellite network, the most important value is the amount of nodes that have vision of each other at a given instant, since this becomes your effective communication network and also limits the effect of broadcasts.

Most MANETs make use of hello messages and other forms of proactive network maintenance, with each message usually triggering a reply from all visible nodes. More nodes therefore means more overhead generated by these mechanisms, and there is usually a potential for the network getting flooded.

Any frame that is broadcast (such as a Reactive Ant (RA) or a Route Request (RREQ)) will also result in many more broadcasts, leading to a snowball like effect often referred to as flooding the network.

Densely populated networks may still be desirable for satellite networks, as they can provide coverage over larger areas and better serve large numbers of user, provided that efficient load balancing techniques have been implemented. Although satellites with higher altitudes may be larger and have superior computational power and memory, allowing them to better service multiple users, the increased altitude will have a negative effect on latency.

2.5.3.2 Network Connectivity

The number of links a node has at a given time can have a significant effect on the effect of broadcasts. The network connectivity refers to the amount of neighbours a node has at a given time.

2.5.3.3 Packet Generation Frequency

Data levels refer to the amount of data being sent through the network, with the worst case scenario being all nodes and end-users continuously sending data. In such a scenario, the amount of time lost due to contention for the medium and inadvertent collisions that occur, will most likely cripple the efficiency of the network.

When a network becomes saturated, there are multiple ways of attempting to mitigate the time lost to collisions and additional wait times and retransmissions. If collisions are highly likely, smaller packet sizes may be used to reduce the cost of any specific collision.

For this research a Constant Bit Rate (CBR) is assumed for packet generation, and typically flood the network from a source node. This does not always match up to typical satellite packet generation patterns and represents the worst case scenario.

In most WSN literature, some nodes are specified as sources of data that generate and transmit data while others are specified as sinks that receive data.

The packet generation frequency is also referred to as the data levels or traffic patterns of the network.

2.5.3.4 Overhead Generated

The amount of overhead generated by any strategy can create a false impression of the efficiency of said strategy. For overhead as an adjustable parameter, the strategy must contain some extra form of insurance that a packet will arrive, usually at the expense of additional ECC bits attached to the packet, or with additional strategy mechanisms.

Increasing the amount of overhead can actually increase the amount of throughput, as shown by [5] testing on the use of FX.25. The guiding principle is to find out when is the extra overhead justified: in a highly lossy channel extra ECC bits allow additional frames to pass through the channel uncorrupted. In a more ideal channel the extra bits essentially serve no purpose and therefore reduce throughput lower than the optimal case, as shown by [5].

2.5.3.5 Strategy Specific Parameters

Most strategies have various parameters that can be set. This usually includes the length of the period at which proactive data is sent, which can have costs and benefits. Increasing

the interval will reduce overhead, but increase the time it takes for new routes to be discovered or routes to be repaired.

Many of the parameters have to be carefully set in order to avoid protocol specific pitfalls. As an example, AODV can use the NETWORK DIAMETER parameter to avoid flooding a network with broadcasts. If this parameter is set too high, flooding can readily occur. If it is set too low, the destination may be viewed as unreachable. All parameters have trade-offs which must be considered on a network by network basis.

2.5.3.6 Configuration of the Network

The Unique configuration adopted by a specific network may also have some significant effects on the network itself. For example, when two satellites approach each other, they will have LOS with each other, however the Doppler shift may be too great for communication to take place. Furthermore this research assumes omnidirectional antennas are used, but pointing can be simulated through the use of the wizards. Modifications to the environment class can allow users to create specific link requirements and configurations.

2.5.4 Orbit Propagation Models

The propagation model used by SatSim is defined by the PyEphem module [29]. Most network simulators offer a variety of propagation models, with the models being used by terrestrial networks being significantly different to satellite models. Some key differences between a satellite propagation model and a terrestrial model such as Random Waypoint Model (RWP) are that satellite movement is deterministic and there is no sudden stops or random changes in direction. It should be noted that SatSim can be expanded to be a more general case network simulator by including additional propagation models and expanding the SimObject class slightly.

2.5.5 Higher Satellites versus Lower Satellites

The altitude of a satellite has a significant effect on various network parameters, but also has a significant effect on the property of the satellites themselves. Since higher satellites can be significantly heavier, they can also have more memory, computational power and transmission capabilities. A higher satellite will have higher latency and a larger footprint.

2.5.6 Satellites in Networks

Three different network types were considered for this research, based upon existing satellite constellations and what has been suggested by other research.

2.5.6.1 Single Ground Station with one or more Satellites

This is used as the baseline scenario for most of the research since it is the approach typically taken by Universities or small commercial entities. With only a single ground station the orbit of the satellites must be carefully chosen - there has to be line of sight of the ground station at some point, preferably for the maximum amount of time. The communication window is typically limited to only certain passes per day, with four 12 minutes passes per day being normal for a LEO satellite. A significant issue is therefore the large amount of downtime in which the satellite has no signal with the ground station and

communication is impossible, implying that throughput is severely limited. Improvements on this scenario must bear in mind the limited budgets of the entities involved, meaning improvements must rely on maximising the use of the communication window or making use of a dedicated relay network, such as the OuterNet.

2.5.6.2 Dedicated Relay Network

A dedicated relay network refers to a constellation in which there are dedicated satellites in the networks whose only purpose is to relay information from client satellites. An example of this would be the OuterNet proposed by the ESL. The research regarding this will have to take several economical factors into consideration as well. Since the constellation only relays data, it must generate revenue by relaying data for which a subscription fee can be charged. In order for such constellations to be viable, they need to give quantifiable benefits that they can charge a subscription fee for.

This OuterNet is not to be confused with another OuterNet concept [30], a project to give free internet access to the entire world.

2.5.6.3 Vast Network

A vast network refers to a constellation with a large amount of constantly connected satellites providing a wide area of coverage. These satellites typically have quasi-static links except in specific conditions, usually determined by the effect of Doppler shift. It is important to determine the amount of coverage given per the number of satellites as well as typical operating conditions. Although this network can provide a large area of coverage, it also has much more possible overhead and a greater potential for flooding. An example of such a network would be the Iridium network, which needed the large scale coverage and interconnectivity to provide telecommunication services.

2.5.7 Networks to Evaluate

The choice of networks has to serve two purposes. Firstly, it will ideally be a general case which has reliable published results for AODV or AntHocNet. This will not only ensure SatSim can cope with the large memory and computational needs of such a simulation, but also serve as additional verification for the platform. The Iridium network has enjoyed a lot of attention, and with a results such as [7] can be used as additional verification for SatSim. Secondly, one of the networks should offer some real world research that can be used to evaluate two protocols against each other and add new information to the field. The OuterNet serves as a novel network which has no research done for it and case serve as a test case as well as a platform for comparing AODV and AntHocNet in a unique scenario.

2.6 Review of SatSim V.1

For this research, we will evaluate an existing, freely available satellite simulator that was developed at the University of Stellenbosch by [5]. In this section the simulator will be evaluated with potential optimization and areas for expansion highlighted. Some plans for the future development of SatSim will also be highlighted.

2.6.1 The Need for SatSim

The original need for SatSim was already described by different studies such as [6]. This research showed some of the choices that need to be made when considering protocol and other satellite configuration choices specifically for LEO satellites.

The following choices for optimization were made:

1. Modulation scheme
2. Packet Size
3. Transmission Power
4. Protocol Used CSMA-CA, Round-Robin Polling

The simulation made use of the OpNet platform.

A crucial finding of this research has been confirmed by this research: the variables created by the different network constellations cause a large enough difference that a general best case does not exist. It will therefore be advantageous to be able to quickly and efficiently find the best possible configuration for a given constellation.

Other reasons for the creation of SatSim included the lack of an easy to learn, freely available and highly adaptable Satellite simulator. A crucial shortcoming on some of the freely available satellite simulators was their inability to handle dynamic links: links would be simulated with a single BER value rather than a dynamic value. Other downsides, especially for the academic community, include the high cost. For the commercial community the difficulty in mastering and using the simulator can have a negative effect on schedules.

2.6.2 Main Design Drivers for SatSim

The following design drivers were considered crucial to the development of SatSim [5]:

- Easy to Learn / User Friendliness
- Modularity
- Re-usability

The majority of these design drivers were maintained, since they all lead to a satellite simulator that is easy to understand, modify and expand. A new driver to add was that additional code should be highly readable and strive towards natural language - which is well suited to Python.

2.6.3 Main Contribution of SatSim

SatSim provided an eloquent framework for the simulation of a satellite network, although several crucial network requirements could not be simulated yet. The research conducted on SatSim therefore only made use of the connectionless versions of protocols such as AX.25 and FX.25. SatSim was by definition only a point-to-point simulator that could provide accurate and realistic results for the communication link between a ground station and a satellite.

A SatSim simulation would occur as satellites are created as simObjects, with fixed orbital parameters. Groundstations would be created with fixed positions.

SatSim was verified using comparisons with existing simulators and theoretical results, as well as early stage comparisons with [6].

2.6.4 Results Gathered from SatSim V.1

SatSim was verified using a connectionless AX.25 protocol to simulate a single satellite pass over a ground station. The communication link had a dynamic BER calculated on a packet-per-packet basis, which resulted in the BER starting high when signal was acquired, reaching a minimum value at the best possible elevation before going higher again. This leads to examining the cost of having ECC versus the benefits of not having any. The graphs generated for AX.25 and FX.25 were significantly different which suggested benefits to a hybrid protocol and subsequently led to its creation.

The hybrid AFX.25 protocol was then designed and simulated, which resulted in a considerably increased throughput for the same communication window.

2.6.5 Recommendations for Future work on SatSim

Some recommendations for SatSim were made, in order to improve the capabilities of the simulator as it was in version 1. These mostly included changes aimed towards making the simulation more realistic by removing some simplifications and changing key assumptions.

The following lists details the suggested change from SatSim and what was done:

- **Spherical Earth Model** - Kept since it has minimal influence on Satellite Networks and is not justified in terms of increased runtime.
- **Communication** - only relies on LOS. Signal diffraction and other phenomena are not simulated. This was maintained since ISL have even less attenuation.
- **Burst Errors** are not simulated since they are caused by random events not determined by protocol choice. Implementing random burst errors may therefore have created an unfair data bias against certain protocols. There may still be interest to see how protocols react to them, but it should not be used when comparing two protocols.
- **Channel access model** - An environment class and MAC layer simulations have been added in order to control access to the transmission medium.
- **Weather Model** - this has not been considered since its cost in terms of execution would most likely render a network simulation near unusable, while not adding significant value to the simulation.
- **Circular Orbits** - Satellites can only orbit in circular orbit paths - changed due to what is being simulated. With sufficient passes the issues will average out.

2.6.6 OOP

As stated by [31], all computer programs consist of code and data. A program can be conceptually structured around currently occurring, or what data is currently being affected. The first paradigm is a process-orientated model, where code acts on data.

The second paradigm was proposed to manage increasing complexity of computer programs and was named Object Orientated Programming (OOP). The main characteristic of an object orientated program is that data controls the access to code.

A key advantage of OOP is abstraction, which is very useful for a simulator environment since it helps users better interpret what is actually happening in a given simulation. By using OOP correctly the simulation becomes understandable in human terms.

In order to best serve the design drivers given earlier, it is desirable that code is easy to maintain and understand. The complexity of the simulation code is therefore best approached by using OOP principles.

Chapter 3

Routing Protocols and Channel Access

This chapter describes the addition of crucial functionality needed for a network simulator, namely some method of regulating and controlling channel contention and access. To achieve this an implementation of the MAC 802.11 layer was implemented in the SatSim environment.

This chapter also details the fundamental descriptions of the two routing protocols currently implemented to assist with the verification and testing of the SatSim environment, namely AODV and AntHocNet. This section will give an overview of both protocols as well as including and justifying any changes that were made to allow the protocols to function in the SatSim environment.

3.1 Overview of MAC Layer

SatSim used to function as a point-to-point simulator. Due to the one way communication link, channel contention and access as well as collisions did not need to be simulated. When simulating a network of nodes, an arbitration scheme is needed to allow nodes to communicate fairly while minimising and detecting collisions. To this end a MAC layer needs to be added to the SatSim environment.

A full implementation of the MAC 802.11 standard and its derivatives falls outside of the scope of this research. For this research, only MAC 802.11a and MAC 802.11b were included that would both retain the following core functionality:

1. Virtual and Physical channel sense
2. Communication Handshaking
3. Effective Back-off Mechanisms and wait times
4. Effective collision detection and processing
5. Fair channel access

It was noted during the implementation of the MAC protocol that the various inter frame spaces and Contention Window (CW) sizes were chosen to best suit terrestrial networks, that can at times be suboptimal for satellite networks. The two implemented MAC layers, with their relevant inter frame spaces and slot time intervals, are given below in 3.1.

The preliminary testing results suggested that the slotTime interval caused an unacceptably high amount of collisions and retransmissions. The difficulties of using MAC 802.11 for satellite networks have been documented by various authors, including [32] and [33].

Table 3.1: Implemented MAC Configurations for SatSim V.2

Value	MAC 802.11a	MAC 802.11b
SIFS	16 μ s	20 μ s
slotTime	9 μ s	10 μ s
DIFS	34 μ s	40 μ s
CW Min	7	7
CW Max	255	255

These difficulties arise since many of these values were chosen for terrestrial networks with distances no greater than 300 meters separating the nodes. The biggest difference between the terrestrial and space-based networks is the size of the propagation delays. The massive difference in distance between nodes suggests that the interval times will be far too short for space-based application.

Additional testing and the research from [32] and [33] suggested various techniques to deal with the far greater propagation delay, including variable contention windows. However, for this research the more basic approach was taken of increasing the slotTime to better match the average propagation delays experienced in a satellite environment. The idea of a variable contention window is very appealing however, due to the massive difference in distances which can occur in a satellite environment.

A customised MAC 802.11SAT implementation was designed and used for this research and its configuration is detailed in table 3.2.

Table 3.2: Custom MAC Implementation

Value	MAC 802.11SAT
SIFS	30 ms
slotTime	15 ms
DIFS	60 ms
CW Min	7
CW Max	255

3.1.1 Channel Sense

The channel sense methods implemented by the MAC layer consists of physical and virtual channel sensing. The physical channel sense involves the source node scanning the communications medium for activity. The virtual sense implies the node is making use of header information it has gathered to determine when the channel will be available again. In this research, when a node senses the channel, it is implied it is performing both virtual and physical sensing.

3.1.1.1 Physical Channel Sense

This research only considers wireless channels, with nodes being aware of the channel being in use if another node they have LOS with is transmitting. If a node detects the channel is busy it will wait for a small interval before sensing the channel again. Physically sensing the channel helps to avoid collisions, but some collisions can still occur, especially in scenario's such as the Hidden Terminal Problem, described in B.1.2. Furthermore the

sensing mechanism can prevent some viable data transmissions from occurring as efficiently as they would occur if channel sense and collision avoidance wasn't implemented, especially in scenarios such as the Exposed Terminal problem, described in B.1.3.

3.1.1.2 Virtual Channel Sense

The virtual channel sense mechanism makes use of additional Network Allocation Vector (NAV) data attached to frames to allow nodes to share data regarding how long they will use the channel for that specific transmission. The virtual channel sense mechanism assumes that each node will read this header information when MAC handshake frames are being exchanged. The header field of interest is the duration field that will give an indication of till when the channel will be busy. As with physical sensing, a node needs to have the appropriate LOS in order to make full use of virtual sense. Unlike physical sense, where a node waits a set time before attempting to sense the channel again, a node that used virtual sense to detect that the channel is busy will wait until the time indicated in the duration field has elapsed before competing for the channel again.

3.1.2 Communications Handshaking

The MAC layer uses a simple handshake procedure to implement collision avoidance and to allow nodes to be aware of when retransmissions are necessary.

The typical handshake operation is shown by figure 3.1, for the scenario of Node A attempting to transmit towards Node B. This handshake occurs after Node A has successfully been granted access to the channel.

1. Node A creates a RTS frame and transmits it towards Node B. The RTS frame contains enough information to allow other nodes that have vision of Node A to defer for the duration of the transmission.
2. Node B will receive the RTS. If it can receive data, it will create and send a CTS frame towards node A. Otherwise it will do nothing, causing Node A to wait until it determines no CTS is coming. Node A will then attempt to restart the process.
3. Node A transmits the data.
4. Node B will then analyse the received DATA frame. If it was correctly received, it will reply with an ACK frame. Otherwise it will do nothing.
5. Upon receiving the ACK frame, Node A will consider the transmission a success and wait a short interval before contending for the channel again. If no ACK frame is received it will consider the transmission a failure and attempt to restart the process.

3.1.3 MAC Frame Formats

Each Mac frame has specific fields that are used by the MAC layer. Some of these fields are defined below for clarity as to how they are used in the SatSim environment. The most significant field is the duration field that is defined below:

Duration: Duration is the total time, in seconds, that this frame will make use of the channel. This consists of the time needed to transmit the pending data frame or

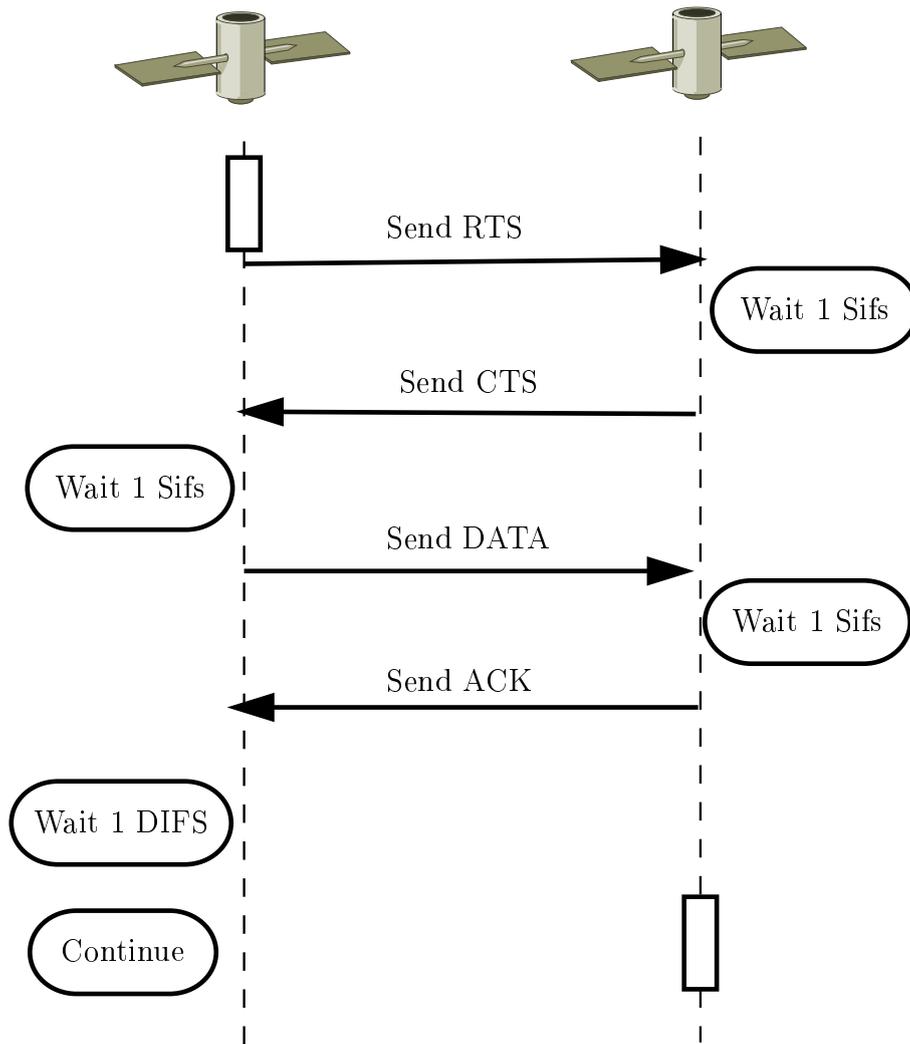


Figure 3.1: Successful MAC Handshake

control packet, as well as one CTS frame, plus one ACK frame, plus three Short Interval Between Frames (SIFS) intervals. In SatSim this value is set to the instance of `simTime` the channel should be available again, in order to reduce computational complexity.

$$Duration = 3 * SIFS + Tx_{DATA} + Tx_{CTS} + Tx_{ACK} \quad (3.1.1)$$

Using the above equation we calculate an approximate time when the current transmission should no longer be using the channel, allowing for minimal calculations to be needed in order to do NAV calculations. The above equation, upon testing for efficiency, was adjusted to be the following:

$$Duration = 3 * SIFS + 4 * PropagationDelay + Tx_{DATA} + Tx_{CTS} + Tx_{ACK} \quad (3.1.2)$$

Where the Transmitter delay is given as:

$$Transmitter\ Delay = \frac{Bits\ Transmitted}{BitRate\ of\ Node} \quad (3.1.3)$$

It is important to note that the SIFS is set to the maximum expected time for the transceiver switching time, physical propagation delay and the mac delay. Of these,

the physical propagation delay can be significantly different to that expected within a terrestrial network. The virtual sense mechanism was adjusted to use this new value for duration, reducing the amount of calculation needed by the mechanism.

3.1.3.1 Frame Control Field

The frame control field is part of all handshake frames and takes up a significant amount of bits. It contains the data shown in below in 3.3.

Table 3.3: Frame Control Field. Adapted from [34], p404

Protocol Version	Type	Subtype	To DS
2	2	4	1
From DS	More frag	Retry	Power Management
1	1	1	1
More Data	Protected Frame	Order	-
1	1	1	-

3.1.3.2 RTS frame

The RTS frame is sent at the beginning of the handshaking process and contributes the most to overhead since it is the largest handshake frame and is usually transmitted with the highest frequency. A handshake will always be a point-to-point link with the *source* in table 3.4 being the transmitting node and the *dest* being the receiving node.

Table 3.4: RTS frame. Adapted from [34], p404

Frame Control	Duration	Dest Address	Source Address	FCS
16	16	48	48	32

3.1.3.3 CTS frame

The CTS frame is transmitted upon reception of an RTS frame, given that the node can currently receive data. The *duration* and *source address* fields are copied from the RTS frame.

Table 3.5: CTS frame. Adapted from [34], p405

Frame Control	Duration	Source Address	FCS
16	16	48	32

3.1.3.4 ACK frame

An ACK frame is sent if data has been successfully received, in other words it can be interpreted correctly and does not have more bit errors than some arbitrary threshold. The *Source Address* is maintained from the original RTS frame.

Table 3.6: ACK frame. Adapted from [34], p405

Frame Control	Duration	Source Address	FCS
16	16	48	32

3.1.4 Back-off Procedures

Back-off procedures refer to how nodes will attempt to access and defer access to the channel in such a way as to ensure fair channel use. No node should wait indefinitely while another node enjoys constant access to the channel.

To achieve this functionality, inter frame wait times and slotTime intervals are used, however as already mentioned these were chosen on the assumption that the network will operate under terrestrial conditions. There is therefore potentially room for improvement by adjusting the lengths of the contention windows and the inter frame intervals.

When a node wishes to access the channel, it will first implement channel sense procedure. If the channel is idle, it will implement an exponential back off timer. This is done by choosing a random value between 0 and CW_{min} . It will then wait that many slotTime intervals of the channel being idle before it will attempt to use the channel. If it detects channel activity during that time, either through virtual or physical sense, it will freeze the countdown until the channel is available again. It will then resume the countdown at the same point, ensuring fair channel contention.

In case of a failed transmission, usually caused by a collision, CW_{min} will be doubled and a new random value between 0 and CW_{min} chosen. This process will repeat until it reaches CW_{max} . After a set amount of failed retransmissions, the destination will be declared to be unreachable and the transmission abandoned for a set amount of time.

3.1.5 Collision Detection

Collisions are detected by the environment class. At any given time, nodes are only aware of transmitting node which they have LOS with. However, if no LOS is established, multiple nodes can transmit to a single sink. This will result in all those packets being dropped by the relevant nodes. These collisions are then detected by the transmitting nodes due to specific wait functions timing out. The mac layer will then inform the Data Link layer of the collision, which will trigger the protocol specific response.

3.1.6 The Wait Functions

When a node is transmitting, there is always a chance that the transmitted frame will not be received correctly, due to collisions or losing signal with the destination. During the handshake process, the transmitting node waits first for a CTS frame and then for an ACK frame. If these wait times out, the node will assume the data did not safely arrive and attempt a retransmission.

The wait-time for the CTS is given by:

$$CTS\ WAIT = (2*SIFS + 2*Propagation_Time + RTS_TX + CTS_TX) * 1.2 \quad (3.1.4)$$

The wait-time for the ACK is given by:

$$ACK\ WAIT = (2*SIFS + 2*Propagation_Time + RTS_TX + CTS_TX) * 1.2 \quad (3.1.5)$$

The padding factor of 1.2 is used to ensure that the wait doesn't time out too early. This means the wait can only perform its checks when a new transmission has started. Therefore sequence numbers have been added in order to ensure that the waits are synchronous.

3.1.7 Penalty for Channel Contention

Given the same direct point-to-point link, a communication scheme which implements channel access will have a significantly lower throughput than a communication scheme which has no channel sense. The best case scenario for a scheme which uses an implementation of the MAC 802.11 standard includes the short interval wait time before transmitting and the communication handshake. While the handshake frames themselves are quite small, the addition of three additional propagation delays can be very significant for a satellite network. The benefits of collision detection and avoidance may ideally have to be replaced with dedicated links which do not need them, which may lead to more efficient networks.

3.2 Overview of AODV

AODV was proposed by Perkins et al as a reactive routing protocol for MANETs [20]. It has become one of the most widely used routing protocols and is often used as a benchmark to compare the performance of a new or novel routing protocol. Ducatelle [19] refers to AODV as the de facto standard routing algorithm for MANETs and Wireless Mesh Networks (WMNs) as well as for benchmarking.

Implementing this protocol in the SatSim environment will serve two purposes. Firstly, it will serve as an analysis of the ease with which a complex routing protocol can be implemented within the SatSim environment. Secondly, it will allow more thorough validation of the SatSim environment since there exists more data on network performance under AODV for verification.

AODV is a Reactive Protocol that works in a completely decentralised manner where minimal routing information regarding routes is stored. Furthermore, after a certain Lifetime expires without the data being used, the data is discarded. Any activity on the route will increase the Lifetime of the entry, ensuring frequently used routes aren't discarded. Furthermore, each route only stores the next hop for the route, and no further direct route information.

AODV plays no role in the routing of a network unless some node wishes to transmit data to a node it does not have valid routing information for. The node will then initiate a route discovery procedure towards the destination. The data is then buffered until a route is found (this can be due to receiving any control packet with the relevant information, not just the route reply triggered by the route discovery.) It is important to note that, when a route has been discovered, all the data will move along that single path, which leads to the possibility of congestion.

3.2.1 Research Regarding Performance of AODV

AODV has been far more extensively studied and simulated than AntHocNet. It is often used as the baseline comparison for new routing protocols. Some of these studies have been for Ad-Hoc networks with static nodes, while there have also been studies focusing on MANETs. The results published in [7] are of particular interest to this research and were used for additional verification.

3.3 Components and Functionality of AODV

The following section details the control packets used by AODV to create and repair routes as well as how this functionality is achieved.

3.3.1 Route Request Frames

A node will broadcast RREQ frames whenever it needs a new, active route to transmit data. The RREQ will contain a unique identifier and be broadcast to all visible nodes, which will react based on whether they have a route to the intended destination or if they are the destination. If they do not they will also broadcast the RREQ after updating the hop count. This behaviour can rapidly flood a network and care has to be taken to ensure the unnecessary broadcasting of RREQ frames occur.

To help prevent this each RREQ has a Time to Live (TTL), that can be measured in hops or seconds. For satellites it is usually better to use time than hops, since the networks can be quite sparse and no information exists on how far away the other node will be. Almost all operations in AODV are associated with a specific TTL, that will usually be increased to some upper limit upon each retransmission or reattempt. After the maximum number of retries has been attempted the relevant data will simply be discarded.

If a node does not have a route after sending out a RREQ it can send additional RREQ's, as long as it doesn't generate more than $RREQ_{RateLimit}$ per second. This process is repeated until either the transmission is cancelled, after which the destination will be considered *unreachable* or the relevant routing information has been received. AODV does not explicitly wait for an Route Reply (RREP) to return with the relevant information and will only check that the information exists and uses sequence numbers to ensure that only the most recent information is maintained. The RREQ frame structure is shown in table 3.7, with the the field sizes in bits.

3.3.1.1 Flag Field

The flag field denotes different flags that can be used as needed. Most of these aren't used in general simulations, but they can be used to improve certain functionality. As an example, the G flag can be set improve QoS. The different flag options are listed in table 3.8.

3.3.2 Route Reply Frame

A RREP is sent back towards a data source upon reception at the intended data sink or a node that has a route towards the intended sink.

Table 3.7: RREQ Frame. Adapted from [20]

Type	Flags	Reserved	Hop Count
8	5	11	8
RREQ ID			
32			
Destination IP Address			
32			
Destination Sequence Number			
32			
Originator IP Address			
32			
Originator Sequence Number			
32			

Table 3.8: RREQ Flag Field. Adapted from [20]

Flag	Flag Description
J	Join Flag, reserved for multicast
R	Repair Flag, reserved for multicast
G	Gratuitous RREP flag, ensures unidirectional communications
D	Destination Only Flag, only dest may reply to RREQ
U	Unknown Sequence Number, indicates sequence number is unknown

A node will unicast a RREP back towards the original source of an RREQ, given that it has sufficient or fresh information regarding the destination. Multiple nodes can therefore reply to a given route discovery procedure, however only the first received RREP will be utilised by the original node.

RREP have additional flags that can be set, based upon the needs of the network. For example, if unidirectional communication is required, the acknowledge flag on the RREP can be set to indicate it. The RREP frame is shown in table 3.9, with field sizes indicated in bits.

Table 3.9: RREP Frame. Adapted from [20]

Type	Flags	Reserved	Prefix Sz	Hop Count
8	2	9	5	8
Destination IP Address				
32				
Destination Sequence Number				
32				
Originator IP Address				
32				
Lifetime				
32				

3.3.2.1 Flag Field

The RREP frame only has two flags, an R Repair Flag and an A Acknowledgement flag, that is used over unreliable links to ensure the that source sends an RREP-ACK to the sender of the RREP upon reception.

3.3.2.2 Hello Messages

AODV also allows the user to make use of hello messages. These are actually special RREP frames which aren't sent in response to a RREQ. These messages are not critical to the functionality of AODV and are optional for any given simulation.

3.3.3 Route Error Frames

A node will transmit one or more Route Error (RERR) messages upon discovering the failure of some data transmission or upon detection of a link break. A node will notify all nodes in the relevant precursor list upon discover of a link failure using RERR. The RERR frame structure is shown in table 3.10 with the field sizes given in bits.

Table 3.10: RERR Frame. Adapted from [20]

Type	Flags	Reserved	Dest Count
8	1	15	8
Unreachable Destination IP Address			
32			
Unreachable Destination Sequence Number			
32			
Additional Unreachable Destination IP Address			
32			
Additional Unreachable Destination Sequence Number			
32			

3.3.3.1 Flag Field

N - Represent the *no delete* flag that indicates that a local repair is taking place.

3.3.4 Routing Entry

Each node maintains a number of routing entries in order to allow data packets to be forwarded correctly. Routing entries are only created by nodes when a route discovery process is triggered. A routing entry supplies a path to a specific destination by indicating the next node to transmit a given packet to.

Each routing entry contains the following information:

- **Destination IP** - A unique address for the destination of the route that the entry supplies
- **Destination Sequence Number** - A sequence number for the entry, giving an indication of the order in which this entry was updated and its validity

- **Next Hop** - The next node on the route towards the destination
- **Hop Count** - The amount of hops from this node towards the destination
- **Route Active** - Flag that indicates whether or not the route is currently active
- **Sequence Number Valid** - Flag that indicates whether the sequence number is currently valid
- **Precursors** - List of all nodes that may be dependant on this route and need to be informed if a link breakage occurs.

Precursors are all the other nodes that may make use of the route. These nodes need to be informed when a link break occurs, since it changes the validity of their route entries.

It's important to note that the only information a node has to forward data towards a specific destination, is the next node it has to forward the data to.

3.3.5 Routing Data Expiry

One challenge in implementing AODV was that routing entries expire after a certain period without use. Furthermore, whenever a node sends out some message it will wait a certain amount of time before deciding if a resend is appropriate. While implementing all these *wait* functions did not require any significant changes they can often cause bugs, memory leaks and in the worst case, loops. The Sequence Number system of AODV helps prevent the loops but the *loop freedom* of AODV is doubted by some as detailed in [35].

3.3.6 Data Forwarding

AODV has a far simpler method of forwarding data packets than AntHocNet, since any data packet will only have one option for a next hop at any node. This can lead to node path congestion. This situation can be less than ideal, if a suboptimal link is chosen first and kept for an extended period of time.

3.3.7 Significant Parameters

Certain parameters have a far more significant effect on the functionality of the protocol than others.

Many of the *RateLimits* parameters have negligible effects when retransmissions aren't required, but can have significant effects in a lossy network with many collisions.

The *Time to Live* parameter has a very significant effect on how far packets are broadcast. This will always have a big effect since dead-end routes will be followed far more with much more overhead being generated than what is needed.

Many of the parameters are dependent on the other parameters, meaning whilst they have a significant effect on the functionality of AODV they aren't directly manipulated in order to view the effect their changing value has on the simulation, but will be changed along with the variables they are dependent on. Table 3.11 shows just some of the parameters which can be configured for an AODV simulation. As can easily be seen this means there are many possible configurations possible for a given simulation.

Table 3.11: AODV Simulation Parameters

Parameter	Default Value	Description
Active Route Timeout	3000	Time before data expires/invalidates
TTL_{start}	20	Sets the minimum lifetime for a message
$TTL_{increment}$	2	TTL increase for retransmission
$TTL_{threshold}$	7	Maximum value for TTL
$RERR_{ratelimit}$	10	Maximum RERR's per second
$RREQ_{ratelimit}$	10	Maximum RREQ's per second
$RREQ_{retries}$	2	allowed RREQ retransmissions
NetDiameter	NA	Maximum diameter of the network

3.3.8 Quantification of Overhead generated by AODV

AODV has multiple modes of operation as well as adjustable parameters that can drastically impact the amount of overhead generated. The primary sources of overhead are discussed below along with the effects different parameters have on them.

It is important to note that upon the reception of any control packet a node will attempt to either update or create a routing table entry for the relevant node, regardless of it needing the route for itself, therefore flooding the network not only increases the overhead, it also triggers several timing and memory events, making the simulation itself more computationally expensive.

Unlike AntHocNet the size of the simulated overhead packets will be constant, allowing for far easier calculation of the total overhead generated.

3.4 Overview of AntHocNet

AntHocNet was proposed by Frederick Duecelle [19] as a modification of Ant Colony Optimization techniques in order to make them more applicable to MANETs. In the conceptual study it was compared to AODV, OLSR and ANSI and produced favourable results. However, there have been little to no research as to the implementation of AntHocNet in a satellite environment as opposed to a terrestrial environment.

AntHocNet is a Hybrid Protocol, that contains both reactive and proactive routing functionality and is overall a highly complex algorithm with many routing components. An implementation of AntHocNet would contain most of the elements needed by other routing protocols as well being relatively difficult, which in turn will highlight potential issues with the implementation of additional routing protocols. The implementation of the AntHocNet protocol should therefore help with expansion and design of SatSim. Furthermore it will contribute to the field by investigating the performance of AntHocNet in a satellite environment.

3.4.1 Research Regarding AntHocNet

AntHocNet has not been extensively researched in the satellite environment and was mostly tested with regards to a terrestrial network such as [36]. However, this simulation makes use of nodes being extremely densely populated (relative to what is normal in satellite networks) in a small area and making use of a RWP as a propagation model, which differs highly from the movement of satellites. In this simulation AntHocNet outperformed

AODV, however the network topology and behaviour was very different from what is expected in a typical satellite environment.

3.4.2 Performance Based Pheromone Value

AntHocNet's properties causes it to focus on specific performance metrics in a given network constellation since AntHocNet uses specific metrics in order to assign a goodness or a cost to a specific route. This means that the network can be optimized in terms of that cost or goodness, while not focusing as significantly on other metrics. For most of the testing latency was used to indicate the cost of a route, leading to paths being chosen to optimize latency and route discovery times. This is also the value that most closely corresponds with how AODV functions.

3.5 Components and Functionality of AntHocNet

The AntHocNet routing protocols consists of both reactive elements and proactive elements that are used in unison. This section describes these elements along with their limitations in a satellite environment and their implementation details for SatSim.

3.5.1 Pheromone Tables

Each node maintains two distinct pheromone tables, one for real pheromone(considered more reliable) and one for virtual pheromone(considered less reliable). Real pheromone is deposited by all ants on their way back towards a source after successfully reaching a destination whilst virtual pheromone is diffused over the network via hello messages.

The pheromone values will either represent a goodness or a cost for that specific route or hop. For instance, a cost might indicate the delay associated with a specific route whilst the goodness may simply be a success rate for data transmissions.

Pheromone tables are structured so that rows will represent specific final destinations while columns will represent hops, or next destinations. If a node wants to send towards node 4, it checks the correct row, with the column indicating which node to send to next in order to reach that destination.

3.5.1.1 Real Pheromone

The real pheromone table contains indicative values that are considered reliable. Data will only use real pheromone to determine routes, and reactive and repair ants will only follow real pheromone when they are unicast. Real pheromone is updated more stringently than virtual pheromone, in an attempt to ensure reliability.

3.5.1.2 Virtual Pheromone

The virtual pheromone table contains indicative values that may be considered unreliable. For instance, it can contain values obtained through pheromone diffusion, in which the node becomes aware it has a route to some destination through an intermediate node. However, this route only exists as long as the route via the intermediate node exists, which will have to be corroborated by proactive ants or additional conditions.

3.5.1.3 Scaling of Pheromone Tables

The implementation of pheromone tables on the AntHocNet does scale with a rather significant effect on resource consumption. Each node will have two NumPy matrices with [Amount of Nodes][Amount of Nodes] elements. Furthermore, pheromone values have to be represented by floats, in order to allow accurate differentiation of values. This leads to AntHocNet having significant memory consumption. This can also cause significant delays in execution times, since many search functions have to traverse through larger data structures.

3.5.1.4 Assigning Values to Paths

In AntHocNet, direct hops (point-to-point) and paths (multiple hops) will each have an associated cost or goodness assigned to them by ants.

A goodness of cost for a path can be indicated as follows: R_{ij}^d

The R_{ij}^d represents a real pheromone value, indicating a cost or goodness of moving towards node d via a hop over neighbour j from the source node i .

Another cost value could be the average number of hops, given as h_{ij}^d . It's important to note that the frame has no further control or information about what exactly will happen at node j , only that it has recently been indicated that the value for the route is true.

The average number of hops is calculated using a moving average, where α determines how quickly the value adapts to new information. The equation, adapted from [19], is given below:

$$h_{in}^d = \alpha h_{in}^d + (1 - \alpha) h_{new} \quad (3.5.1)$$

When assigning values to a path, the values can be summed as either a cost value or a goodness value as long as it is done consistently throughout the equation. This differs slightly from the equation used by [36], since it removes one inversion. In SatSim either a cost or a goodness is used and that value type is then used consistently throughout the simulation.

The equation used to update pheromone information is given below, where γ determines how quickly the value adapts to new information. The equation is adapted from [19]:

$$R_{ij}^d = \gamma R_{ij}^d + (1 - \gamma) R_{new} \quad (3.5.2)$$

3.5.1.5 Neighbour Tables

Each node maintains a neighbour table, that uses binary values to indicate whether or not a given node is visible or not relative to that node. If it is visible, the node considers that node to be a *Neighbour*. The set N_i indicates all the neighbours of a node. This set is extensively used, both for the detection of link failures and for many of the stochastic unicast equations and is updated by the use of hello messages.

3.5.2 Hello Messages

Hello messages are broadcast periodically from every node every t_{hello} seconds. These messages carry pheromone diffusion information as well as supplying information to a

node regarding its neighbours. If a node receives a hello message it replies and notes that it has a connection with the source node. A node that receives the reply will then also list the responding node as one of its visible neighbours, before updating its pheromone table with the information included for pheromone diffusion. Typical data fields for a hello message is given in table 3.12, with the value of k being the number of pheromone diffusion entries which are included per hello message. The data sizes of the fields are in bits.

Table 3.12: Hello Message Structure. Adapted from [19]

Source	Signature
8	8
Diffusion Addresses	
$k * 8$	
Diffusion Pheromone Values	
$k * 16$	
Diffusion Bit Flags	
$k * 1$	

3.5.2.1 Pheromone Diffusion

The pheromone diffusion process allows nodes to share routing information with its neighbours. The pheromone diffusion process works since each node will be aware of the one-hop routes to its neighbours via the exchange of hello messages. This process will then spread additional information, on the simple principle that if a node has a route to node x , each of its neighbours also have a route to node x via that node. Pheromone information is exchanged this way via hello messages, reducing the necessary overhead needed to exchange information.

Each hello message bootstraps pheromone values towards specific destinations by choosing the best value towards that destination - virtual or real. When a node receives the hello message, it derives a bootstrapped pheromone value by combining the value of that route indicated by the hello message with the value of the hop to the neighbour. The equation has once again been changed from the source [19], to remove inversions.

$$\kappa_{ji}^d = R_j^d + R_j^i \quad (3.5.3)$$

This pheromone value is not automatically considered reliable and will usually be used to update the virtual pheromone table, unless certain strict conditions are met.

3.5.3 Ants

The ants of AntHocNet use pheromone tables, pheromone diffusion and broadcasting in order to create, maintain and explore new routes. As the name implies, RA are created on demand, whenever a transmission towards a then unknown destination is scheduled. Proactive Ant (PA) are created periodically while data is being transmitted, but may be set to only be sent out in favourable conditions (usually when the virtual pheromone table contains far more favourable values than the real pheromone table), in order to reduce

overhead. Repair Ant (RPA) are sent from nodes that attempt to repair link breaks upon detection of a link failure.

All ants are called Forward Ants when they are moving from a source towards a sink and Backward Ants when they are moving from a sink back towards a source. In order to reduce overhead and allow the shortest possible path to be found, a destination will only convert the first Ant for a given session back towards the source while discarding the rest.

A crucial fact about ants is that they collect a list of all the nodes they have visited on their path from their source to their destination. This means the size of the Ants, and therefore their contribution to overhead, changes over time.

The implemented base ant class, that all ants inherit from is presented in figure 3.2 along with the details of the other ants.

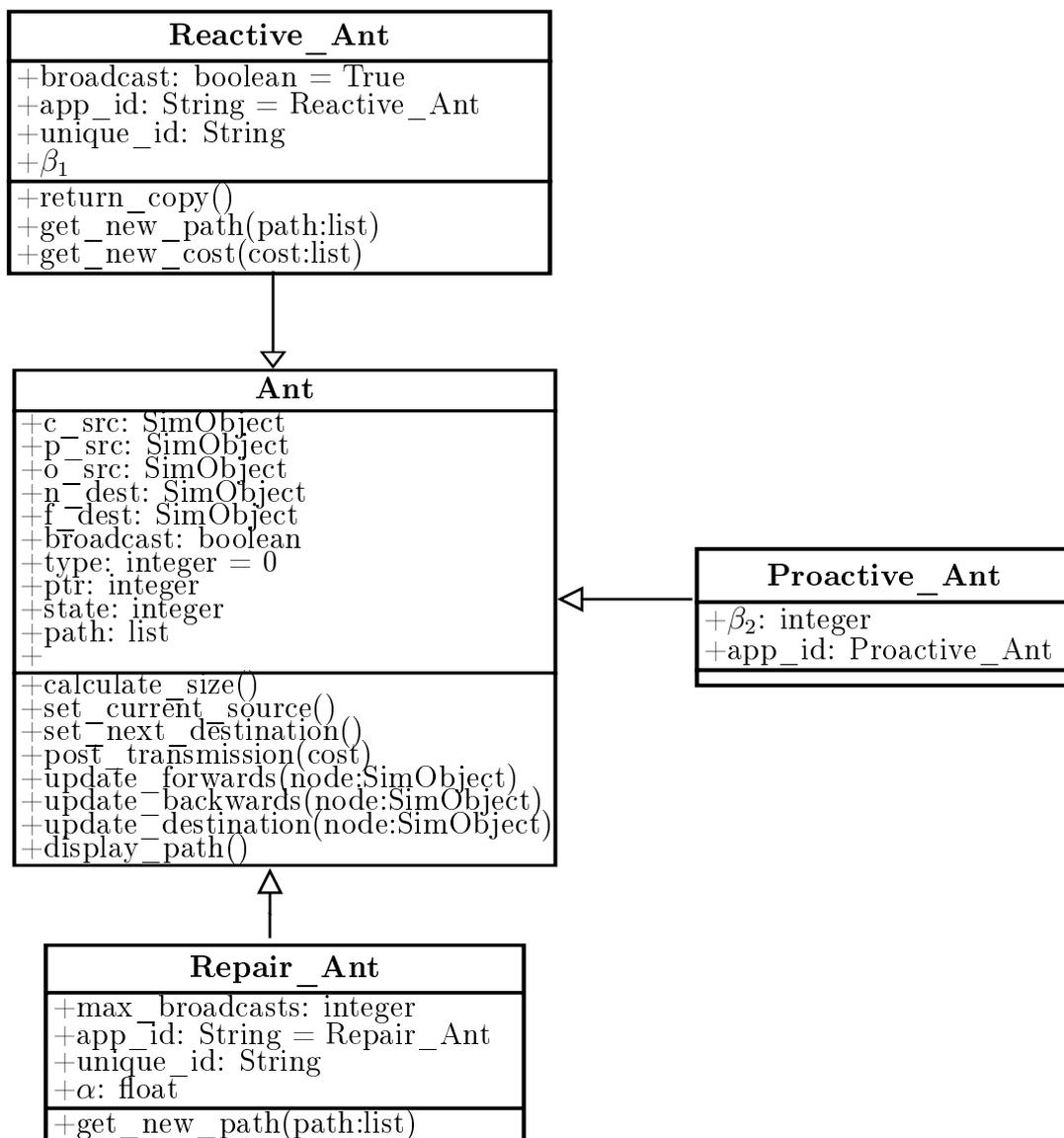


Figure 3.2: Ant Class Diagram: Showing Base Ant Class and Derived Ants

3.5.3.1 Reactive Ants - Path Creation

The reactive ants are sent out whenever a data session between two nodes occur for the first time. Reactive ants attempt to establish a path between the final destination and original source of the data, via broadcasting and the use of established pheromone information.

3.5.3.2 Proactive Ants - Path Exploration and Maintenance

Proactive ants are used to prevent the overuse of a path once a better path has become available in a network. Proactive ants can be sent out periodically or whenever the virtual pheromone value is of near or greater value than the real pheromone. In the original study of AntHocNet proactive ants were only optimally sent out when the virtual pheromone is within 10 percent of the real pheromone or better. They will stochastically follow values for the given path, using either the virtual or the real pheromone values.

3.5.3.3 Repair Ants - Path Repair

Repair ants are used when a link failure is detected or the MAC layer indicates that a data frame cannot be sent. These ants act as reactive ants, although their total number of broadcasts is limited. This prevents them from deviating too far from the original route.

3.5.4 Link Failure Messages

Link failure messages are used to indicate a failure in the constellation when a node attempts to send to an unavailable node. AntHocNet makes use of *Link Failure Notifications* to inform multiple nodes and *Unicast Warning Messages* to inform single nodes of link failures. However the method for handling link breaks is highly unsuited for large quasi-static networks, as was discovered during the verification process of SatSim.

3.5.5 Stochastically Choosing Next Hops

AntHocNet has multiple scenarios in which a single destination for unicast must be chosen. This includes route discovery by reactive ants, data forwarding by nodes and path exploration by proactive ants.

Each of these equations were verified by setting two different pheromone values to 80 and 20. Upon a large amount of calls, roughly more than 100, to the equation the function properly chose option one 80% of the time and option two 20% of the time.

3.5.5.1 Route Discovery

Routes are discovered via Reactive ants that are broadcast and unicast until they reach their destination node. Reactive ants will be broadcast whenever no routing information for the sink is available. Otherwise, they will be stochastically unicast towards the relevant next hop. The equation below, adapted from [19], describes how reactive ants are unicast when multiple paths are available.

$$P_{in}^d = \frac{(R_{in}^d)^{\beta_1}}{\sum_j = 1^N (N_i^d)(R_{ij}^d)^{\beta_1}} \quad (3.5.4)$$

where the symbols represent the following:

- P_{in}^d Probability of using n as the next hop towards d from current node i
- R_{in}^d Pheromone value associated with the path to d via n from i
- β_1 Route discovery variable - high implies the best routes are more likely to be used
- N_i^d Set of all visible neighbours, with 1 for visible and 0 for no LOS

3.5.5.2 Data Forwarding

Data is forwarded using pheromone values from the real pheromone table. Virtual pheromone is not considered reliable enough. It uses a weighted version of each value to stochastically chose a next destination. This means that the load will be more or less spread according to the goodness or cost of the route. This will over time lead to effective load balancing of the data over a given link.

Data is stochastically forwarded using the following equation, adapted from [19]:

$$P_{nd} = \frac{(R_{in}^d)^{\beta_3}}{\sum_j = 1^N (N_i^d)(R_{ij}^d)^{\beta_3}} \quad (3.5.5)$$

where the symbols represent the following:

- P_{nd} - Probability of using n as the next hop towards d from current node i
- R_{in}^d - Pheromone value associated with the path to d via n from i
- β_3 - Data forwarding variable
- N_i^d - Set of all visible neighbours, with 1 for visible and 0 for no LOS

3.5.5.3 Path Exploration and Maintenance

The path exploration functionality of AntHocNet can be set in order to make path exploration more aggressive, leading to increased overhead and knowledge of more paths, or more passive, reducing the amount of overhead but also reducing the amount of information regarding the network. The path exploration equations, adapted from [19], is given below:

$$P_{nd} = \frac{\max(R_{in}^d, T_{in}^d)^{\beta_2}}{\sum_j = 1^N (N_i^d) \max((R_{ij}^d), T_{ij}^d)^{\beta_2}} \quad (3.5.6)$$

where the symbols represent the following:

- P_{in}^d - Probability of using n as the next hop towards d from current node i
- R_{in}^d - Pheromone value associated with the path to d via n from i
- β_1 - Route discovery variable
- N_i^d - Set of all visible neighbours, with 1 for visible and 0 for no LOS

3.5.6 Adjustable Parameters for AntHocNet

The list of parameters, their default values and their effect on the network is shown in table 3.13.

Table 3.13: AntHocNet Parameters

Parameter	Default Value	Affects
Hello Interval	1 second	Overhead/Network Fineness
Mode	True	True for Cost, False for Goodness
PA	1	link breakages
β_1	20	Greed for Reactive Ants
β_2	20	Greed for Exploratory Nature
β_3	20	Greed for best routes(Repair)
k	10	Pheromone Diffusion Entries
γ	1	Weight of new hop information
α	1	Weight of new pheromone information

3.5.7 Quantification of AntHocNet Overhead

An important property of AntHocNet is that some of the components sizes change as they are propagated over the network. For example, Hello Messages will include additional pheromone information for k other destinations (k is 10 by default but may be configured in the AntHocNet parameters class). Which will increase the size of the packets.

Ants also store routing information based upon the path they follow towards their destination, meaning their size will increase based upon the amount of hops they undertake between source and destination. Table 3.14 shows how the overhead for each element was approximated in this research.

Table 3.14: Data Sizes of AntHocNet Components

Component	Formula for Size	Frequency Determined By
Reactive Ant	$\text{len}(\text{path}) * 32 + \text{base}$	Data Activity
Proactive Ant	$\text{len}(\text{path}) * 32 + \text{base}$	Quality of Virtual Pheromone
Repair Ants	$\text{len}(\text{path}) * 32 + \text{base}$	Frequency of Link Breakages
Hello Message	266	Set Interval : t_{hello}

Chapter 4

Expansion and Testing of SatSim

4.1 SatSim - Overview of Expansion

In order to implement the chosen routing protocols on a network of satellites, SatSim needed to be considerably expanded. Several key features of SatSim were maintained, while the core design of SatSim was adapted to conform to OOP principles.

As a satellite simulator SatSim should strive to maintain its original design drivers as far as possible as well as determine accurate results with decent execution times and reasonable memory usage. Another important facet of SatSim is that it should be highly readable, with the code being structured to closely resemble natural language. This is due to unique network configurations often needing custom code to be implemented, resulting in it being highly beneficial for a user to quickly be able to understand and modify the code as needed.

It should be noted that AntHocNet was originally tested on the Qualnet simulator [19] due to computationally expensive nature of the algorithm.

In expanding SatSim each simObject has been expanded to be able to write and read a large amount of state information in the Memory class. The Strategy class allows complex protocols to be correctly implemented and adjusted for optimization. The RFChannel has been optimized in terms of runtime and has become a subclass of the new Environment class, which controls channel access and allows a network to be effectively simulated. The OSI implementation has been considerably expanded and refined to allow complex protocols to be easily slotted into the simulator. A channel access layer has been added along with various other additions.

4.1.1 Developmental Goals for SatSim

The main goal of this study was to expand SatSim to a simulator that can handle complex routing protocols and network characteristics. This was done by implementing the hybrid protocol AntHocNet and the reactive protocol AODV for verification. Additional work and research in the field also suggested additional components that could be necessary in order to realistically simulate a satellite network. After these components were implemented and tested to satisfaction a novel network suggested by Kearney et al [1] was simulated and evaluated in order to give more realistic results from what can be expected from such a network.

4.1.2 Changes made to SatSim

A number of changes were made to the existing structure, for various reasons. In order to better be in line with OOP principles, a central Simulator object was created. Many of the files were then converted into objects that can be easily adjusted and could all be accessed via the simulator. This also ensured that they could be changed and all nodes would have access to the same information. This made SatSim more modular and made it easier to understand the overall structure for a given simulation.

4.1.3 Features which weren't maintained

The Graphical User Interface (GUI) and the Animator classes weren't maintained for the expanded version of SatSim. This was in part due to the sheer scope of variables now required for single simulation being difficult to set with a GUI, resulting in scripting being a more attractive choice. These issues can be mitigated and worked around with proper use of configuration files and proper settings and dropping the GUI will unfortunately have a negative impact on user-friendliness. It is suggested that the GUI be added back to SatSim when it is nearer to a finished product. Lack of time and focus on other issues also contributed to the choice not to continue with the GUI for this expansion of SatSim.

The Animator wasn't maintained due to issues with cross platform compatibility and scaling. As one of the test cases for verification has 122 nodes, it would not be realistic for a human user to realistically interpret meaningful results from such an animation. Furthermore the memory consumption and execution times for such an animation can not be reduced to acceptable values with the current configuration.

4.2 Simulator

The simulator is the top object that controls the entire simulation. It contains a SimPy environment and a list of simObjects which have various interactions with each other. The simulator is created at the start of any simulation and schedules events to occur within the simulation.

The simulator also contains a large number of utility functions that can be used by the protocol implementations. The following section details the components used by the simulator in order to efficiently run a simulation.

4.2.1 Updated Simulator Structure

A high level overview of a SatSim simulation is given by figure 4.1:

A great deal of work went into setting up the simulator as good central object to provide full control over the entire simulation. The expansion of the SimObject class also generated a great deal of work since the role and capabilities of SimObject's had to be expanded considerably. Some classes already mostly followed OOP principles and therefore only required some minor adjustments.

4.2.2 Event Wizards

The event wizards allow specific routing information and conditions to be calculated pre-simulation, which can lead to a significant reduction in simulation time. These event

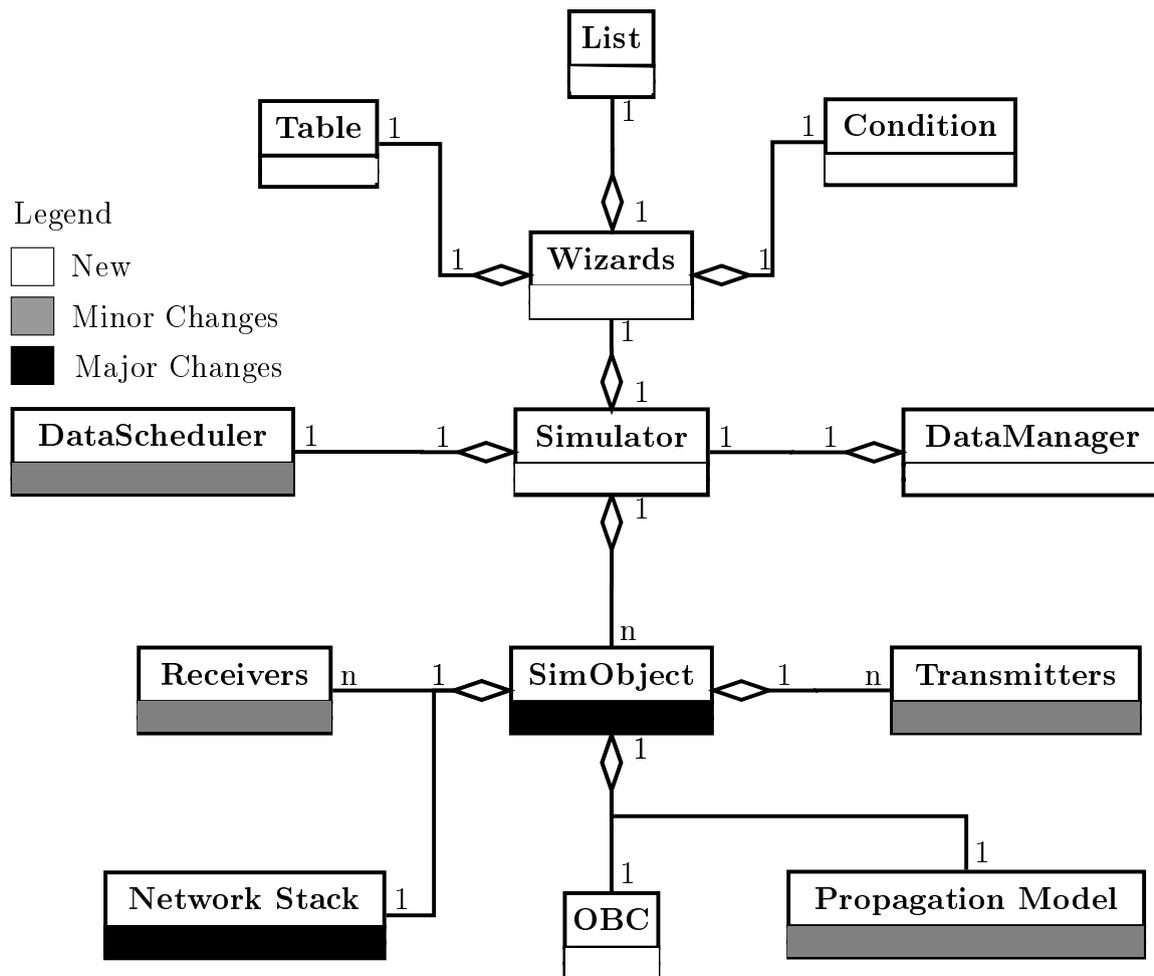


Figure 4.1: High Level SatSim Overview

wizards make use of the deterministic nature of the satellite orbits and the lack of random propagation in their orbits.

4.2.2.1 The Table Wizard

The table wizard allows the automatic generation of tables for centralized routing. The table wizard performs a complete run of the simulation, but only calculates the position of the nodes at fixed minimum intervals. This interval is chosen to be as close as possible to the smallest amount of time it takes for two satellites to gain or lose sight of each other, which is usually in the order of milliseconds but can be set much higher for a sparse network. Whenever a unique new network topology occurs a visibility matrix is generated for that specific topology. This visibility matrix is a simple binary matrix with an entry of 1 indicating the two nodes have vision of each other and 0 indicating otherwise. In order to minimize memory requirements the matrices are only saved if they are unique. The sequence of network topologies is saved as a sequence of unique id values along with the transition times.

4.2.2.2 The List Wizard

The list wizard is used to create lists indicating which nodes are visible to one specific node, which is usually used to reduce execution time. This can be quite efficient if only

one node needs to transmit data.

4.2.2.3 The Condition Wizard

The condition wizard allows the user to simulate mission specific conditions with minimal addition of code. For instance, the user might want to simulate a ground station that always points to the satellite with the highest elevation. This code can be implemented to determining the best satellite available throughout the simulation, allowing the ground station to select from the appropriate list during the simulation which remove a significant amount of runtime calculation.

The condition wizard allows the user to make a series of selection choices that will then be given to the appropriate node at the appropriate time, with the user only having to define what the scenario specific conditions are.

4.2.3 The DataManager

SatSim originally used ordered dictionaries in order to store its useful simulation data. However, this was set up for a a single point to point communication session. The DataManager class creates a DataSet for each node in the simulation, which is used to keep track of all the relevant information when packets are forwarded. Typically the amount and type of overhead generated will be tracked along with certain key factors which are being compared for the simulation, such as route discovery time or latency.

4.2.4 ResultManager

The Result Manager is used to gather overall simulation results, rather than summarised results on a link by link basis. The Result Manager will therefore keep track of metrics such as total generated overhead and average route discovery times.

4.2.5 Data Scheduler

The DataScheduler is used to schedule data transmissions for a specific SimObject at a specific instance of simTime.

Each data event needs an Application ID (which ties the data to a unique Application on the specific SimObject). Although the applications themselves are not simulated, this does allow the data to be tracked better and will allow ease of further expansion.

Each data event also requires an intended destination for the data, as well as the size of the data in bytes.

4.2.6 Routing Scheduler

The routing scheduler is used for centralised routing and takes advantage of the fact that the movement of the nodes will be deterministic. This is done to pre-calculate all the routing tables for a given network before a simulation is a run. This is done by taking snapshots of the network topology and creating routing tables which the central node can send to all other nodes. This class was mostly used by the testing and early expansion phase, but could be utilized in order to implement hierarchical routing protocols.

4.3 SimObjects

SimObjects act as the principle actors in the SatSim environment. In order to facilitate network capabilities, the SimObject had to be expanded to include an implementation of the OSI model and be capable of storing state information. This information is stored in the OBC, which has several subclasses including memory, strategy and a data buffer.

4.3.1 Propagation Models

A satellite will be have a PyEphem object with the following orbital elements as its propagation model. It should be noted that by adding additional propagation models for terrestrial vehicles SatSim can be used to model WSNs and WMNs.

4.3.1.1 Satellite orbit configuration

The orbital elements of individual satellites can be set in order to configure a constellation as needed. The following values are used to configure a satellites orbit, with the mean motions in revolutions per day being derived:

- Inclination
- Mean Anomaly
- Right Ascension of the Ascending Node (RAAN)
- Eccentricity
- Altitude
- Argument of Perigee
- Mean Motion

4.3.1.2 Ground Station Location Configuration

Ground stations only have to be configured for their location using the following:

- Latitude
- Longitude
- Altitude above sea level

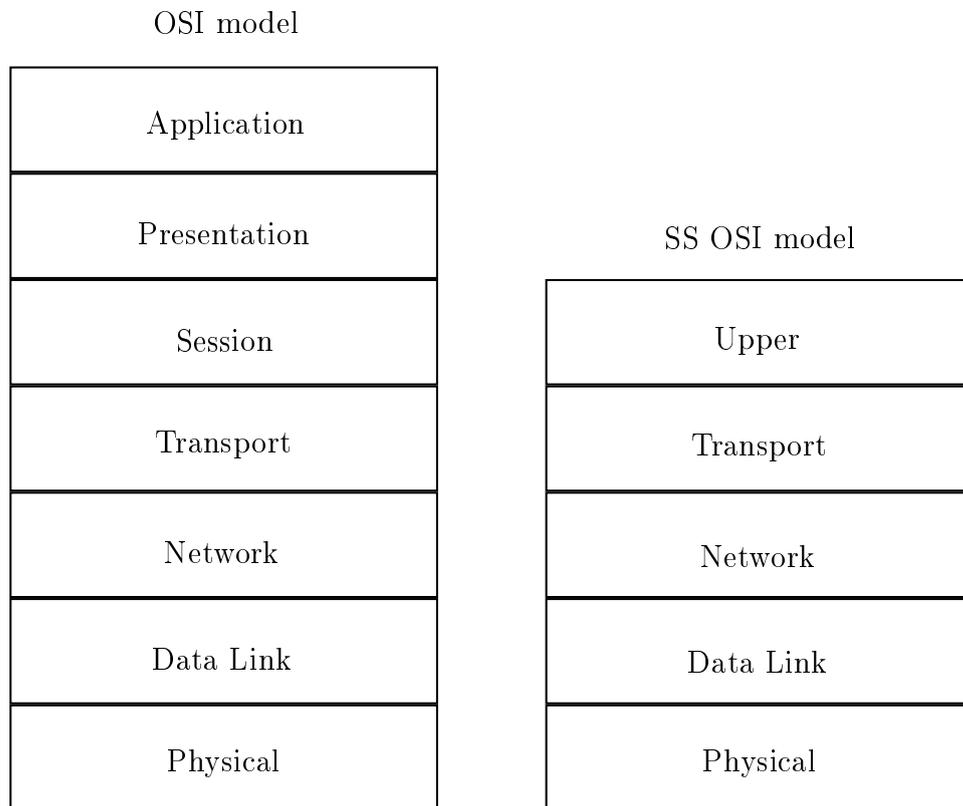
4.3.2 Implementation of the OSI Model

SatSim makes uses of an modified version of the standard OSI model [37] to simulate the exchange of data packets between nodes. Each node contains an instance of each the layers of the modified OSI model, as shown by figure 4.2:

4.3.3 The Physical Layer

The physical layer consists of the an Environment and Radio Frequency (RF) Channel class. Every packet transfer will be associated with a receiver and a transmitter pair. A broadcast will consist of multiple such pairs with the frames being identical copies except for crucial meta information.

The environment updates the positions of the satellites during the simulation and ensures that the position of each satellite is updated for each frame. Whenever a data transmission occurs a link is set up between the transmitter and a receiver. A BER is

Figure 4.2: SatSim implementation of the OSI model

then determined for the link at that given instant using a link budget equation and used to randomly assign bit errors to the packet.

When a packet is broadcast, such a link pair is created between the source nodes transmitter and every receiver that can receive the packet. To achieve this, the simulation can either make use of defined fixed links, as was described by [7] with some variable links or make use of a completely dynamic topology, that is far more computationally intensive. The issue of constant recalculations could be mitigated using a central visibility table, that provision has been made for in the Simulator. This table should contain the visibility information for the network at a given time, acting as an effective snapshot of the network which would be especially useful for the calculations regarding collisions and channel sensing.

4.3.3.1 Environment

The Environment class is used to control data transmissions and to keep track of which nodes are transmitting data and which nodes are receiving data. When centralised routing or specific quasi-network configurations are used the environment also defines the ISL and other links. Two examples included in the research are the constellation implementation given in [7] for both AODV and AntHocNet.

When a frame is unicast, the environment performs some preliminary calculations and updates its state variables before handing the frame to the RFChannel class, establishing a link.

When a frame is broadcast, it is used as a base case. All the visible objects are detected, an a copy of the frame is handed to the RFChannel for each one. This has the

same effect as an actual broadcast, with each transmission being handled uniquely and the frames differing (at that instant) only in some meta-information.

4.3.3.2 RFChannel

The RFChannel represents an instantaneous link between two nodes and all the functionality necessary to simulate the link. The RFChannel class can also be configured in various ways in order to determine how links are simulated, such as using dynamic or static BER calculation.

By default the BER calculation is set to dynamic, however it can be set to some static value or supplied with a constant offset to simulate worst case scenario conditions. The RFChannel can have multiple features active which can make a simulation more realistic at the expense of additional computations.

4.3.3.3 Assigning Errors to Frames

In order to simulate bit errors on a packet-by-packet basis a specific procedure is followed for each packet. Firstly, a BER is determined for the exact instant the packet starts its transmission. This is done using the valid network topology and a link budget equation which was defined by Le Roux [5]. This can be adjusted by setting a constant BER or adding additional losses to simulate worst case or lossy scenario conditions.

After a BER has been obtained negative binomial trials are performed using the size of the packet in bits and the BER. If the frame has more errors than its threshold, determined by the ECC and other factors, it will be set as corrupted and it will be viewed as dropped.

Negative binomial trials are a discrete probability distribution of the number of successes in a sequence of independent and identically distributed Bernoulli trials before a specified number of failures occurs. For SatSim to assign errors, a success is defined as a successful bit transfer, a trial is simply a bit transfer and a failure is a bit error.

The equation for negative binomial trial distribution is given below:

$$P(N; n : p) = \binom{N + n - 1}{n - 1} p^n (1 - p)^N \quad (4.3.1)$$

where:

- n is the number of trials (bits)
- p is the probability of success ($1 - \text{BER}$) in the interval $[0, 1]$
- N is the output shape (1)

The function returns the drawn samples, in other words, how many of the bits were corrupted. The drawn samples are then summed to determine the exact amount of bit errors which occurred for the specific transmission.

This process was verified by running the process several times with a BER of 10^{-6} and a million bits and ensuring that on average only 1 bit error is generated per million bits.

Burst errors have not been implemented in this research.

4.3.3.4 Adjustments for Quasi-Static Networks

Some quasi-static networks may want to simulate effectively fixed links with no error assignment. Additional functionality for these fixed links are included in the RFChannel class and can be easily adjust for a specific network configuration.

4.3.4 The Data Link Layer

The data link layer controls a direct point to point communication session by providing direct access to physical layer via either a receiver or a transmitter. It can also contain the MAC layer and other forms of low level flow control, depending on what is being simulated.

When the MAC layer is active, the primary function of the data link layer is to determine the type of the frames coming through, and process them accordingly. It also controls the transmit queue which is directly linked to the transmitter and processes collisions which occur at the receiver.

4.3.4.1 The Mac Layer

The MAC layer of a communications node directly controls the communication between the various nodes in the network. For MANET there are various additional challenges which occur, such as the hidden and exposed terminal problem, and the mobile nature of the network. Most network simulators make the assumption that the nodes do not move during a communications session - SatSim works on the assumption that they do not move during the transmission of a single packet.

In order to accurately simulate the workings of the MAC 802.11 standard, an effective medium for physical channel sensing and virtual channel sensing must be implemented. The above mentioned central table is used for virtual channel sensing - all visible nodes will collect NAV data from the relevant packets. Whenever a node uses the channel it is added to a list of currently transmitting nodes. Whenever a node attempt to physical sense the channel it will compare the list of its visible nodes to the list of currently transmitting nodes. If any nodes appear in both lists it will determine the channel is busy and proceed to the appropriate backoff procedure.

Some assumptions regarding the design of the implementation of the mac layer can be changed in the satellite environment. For instance, the inter frame wait times were chosen with small WSN. Likewise the design of the contention window was designed while assuming different operational conditions.

4.3.5 The Network Layer

The network layer controls the relaying of packets on a multi-hop route. Where the data link layer considers only the current source and next destination (as a point-to-point connection), the network layer determines how a packet can move from an initial source to a final destination.

Most control packets are handled by the strategy class, whereas data packets are routed in the network layer.

A data packet moving through an intermediate node will only interact with the data link and network layers of that node. The network layer will analyse a packet and call the strategy specific function to determine the next destination for the packet before being relayed.

4.3.6 The Transport Layer

The transport layer handles the process of turning the data set into a series of discrete packets, usually with ECC headers and other additional information. This allows data to be send via multiple routes to a destination and recreated as the original data in the

correct order at the destination. Most WSN use User Datagram Protocol (UDP) for their transport layer, although several custom transport layers are provided with the current implementation of SatSim.

4.3.7 The Upper Layers

The upper layers are treated as a single entity in SatSim since their functionality is usually fairly limited on a nano-Satellite and not of primary interest to this research. All data does have an AppID however, allowing the different data from different applications to be differentiated and data streams to be recreated.

Another function of the upper layers is to provide the DataManager with the relevant data regarding a specific transmission - this is done here since data transference is only fully complete once the application data is fully available to the relevant application.

4.3.8 OBC

Each SimObject includes an OBC model which consists of two discrete classes: strategy and memory. The OBC can also be shut down or rebooted to simulate network failure and to analyse the effects of a node suddenly being dead to the network, as often can happen within satellite networks. The Unified Modelling Language (UML) class diagram for the OBC is given in figure 4.3. Each protocol will require its own specific strategy and memory class.

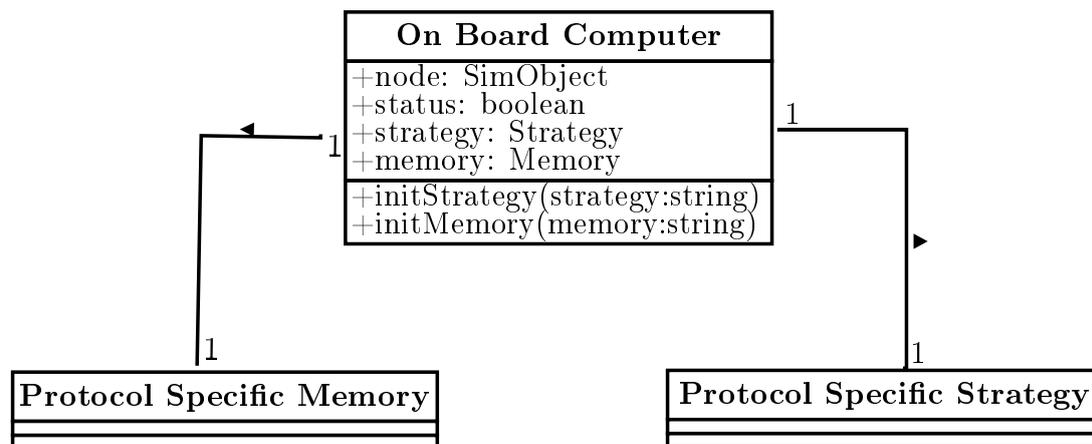


Figure 4.3: OBC UML Class Diagram with Configuration

4.3.8.1 Strategy - Protocol Specific

The Strategy class defines how a specific network protocol responds to different control packets (most control packets are counted as overhead in this study). This allows the processing of control packets and both proactive and reactive messages. Each routing protocol will have its own unique strategy class, determining how it receives, processes and reacts to control packets as defined by the protocol.

Due to the differences in most protocols no effective plug-and-play framework can be provided in which the user can simply change the relevant variables. However, two example implementations are provided along with skeleton code for some often used procedures.

4.3.8.2 Memory - Protocol Specific

The memory class consists of information a specific SimObject would have at a specific instant in simTime. For decentralised routing this usually means a set of tables or lists it has gathered itself. For centralised routing it usually refers to the routing tables it has been sent.

The memory of different routing protocols can be handled very differently. For instance, in AODV data can expire and must be removed from the memory after a certain amount of time, while AntHocNet maintains data without doing additional timers of the accuracy of said data.

4.3.8.3 Parameters

The parameters file stores the protocol specific variables which are used to configure a routing protocol. These parameters are stored in a single class for ease of use and to allow them to be adjusted as needed and the corresponding changes in protocol performance measured. These parameters should always be configured with default values and allow the user to set them as needed. If the effect of these parameters are being investigated they should be set at the beginning of a given simulation, otherwise default values will be used.

4.3.8.4 Data Buffer

During the verification process it was discovered that most link breaks cause large amounts of data to accumulate at a given node without being sent. The upstream nodes will most likely be aware of the link break but not of any data packets they are not receiving. The downstream nodes will mostly keep forwarding data packets until they have been informed of the routes, resulting in a large number of packets accumulating at the node adjacent to the link breakage. The data buffer allows these packets to be stored and send out in a First in First out (FIFO) manner, unless route availability makes it so that certain packets further along in the queue can be sent first.

Whenever a new route becomes available or an existing route is repaired, the data buffer is checked to see whether or not there is data for that specific route.

4.3.9 Frames

The frames section contain descriptions of the frame or packets used by the various communications protocols. Each frame consists of its actual data, in other words the data the real frame will consist of as well as its meta data. The meta data contains information needed by the simulation itself and will not count towards the size of the frame in calculations.

4.3.9.1 The BaseFrame Class

Each frame inherits from the BaseFrame class, which contain the necessary information used to transmit frames over communication links as well as additional information used by the simulation, even if it is not needed by the real protocols. The UML for the BaseFrame class is shown below:

In order to improve simulation execution time, only the critical frame information is maintained such as size. In an earlier version of SatSim packets were simulated in a

bit-by-bit basis, which would have allowed some specific errors to be simulated but have a drastic effect on execution time.

Most frames will be protocol specific and the constructors will have to be custom written for each one, however a wide variety of general purpose frames have been provided, including all the frames needed for AntHocNet, MAC 802.11 and AODV.

Each frame also contains a type parameter, defined in table 4.1 that is used by the data link layer for differentiation.

Table 4.1: Frame Type Classification

Type	Frame Class
0	Control Packet
1	Data Frame
2	Mac Frame

Each frame will also contain meta-information shown in table 4.2 to aid debugging, allow functionality for certain protocols, and for determining the results of a given simulation. These fields do not contribute to the effective size of the frame, unless the protocol explicitly uses it.

Table 4.2: Frame Meta Information

Field	Description
<code>o_src</code>	Original Source of Frame (quasi-static)
<code>p_src</code>	Previous SimObject which handled frame
<code>c_src</code>	SimObject Currently Handling frame
<code>f_dest</code>	Final Destination of Frame (quasi-static)
<code>n_dest</code>	Next SimObject to handle Frame
<code>broadcast</code>	Indicates whether frame is broadcast or unicast
<code>mac_attempts</code>	Amount of Mac Transmissions

Each frame has to contain a signature(for differentiation) and a size(for error checking and the calculation of processing and delay).

Any frame which is broadcast needs its own copy function, since the direct copy class cannot be used inside a generator.

4.3.9.2 The Generic Frame Class

All the frames used in the SatSim environment inherit from a single base class. This class is needed in order to ensure that the Simulator has enough information to properly process the frame. The generic frame class is given in figure 4.4, highlighting the meta data used by the simulator.

4.4 Testing of Channel Access

In order to verify that the MAC layer functions properly, a set of performance goals have been created. To verify the results obtained in the following tests the expected results for each scenario were computed analytically and then compared with the simulated

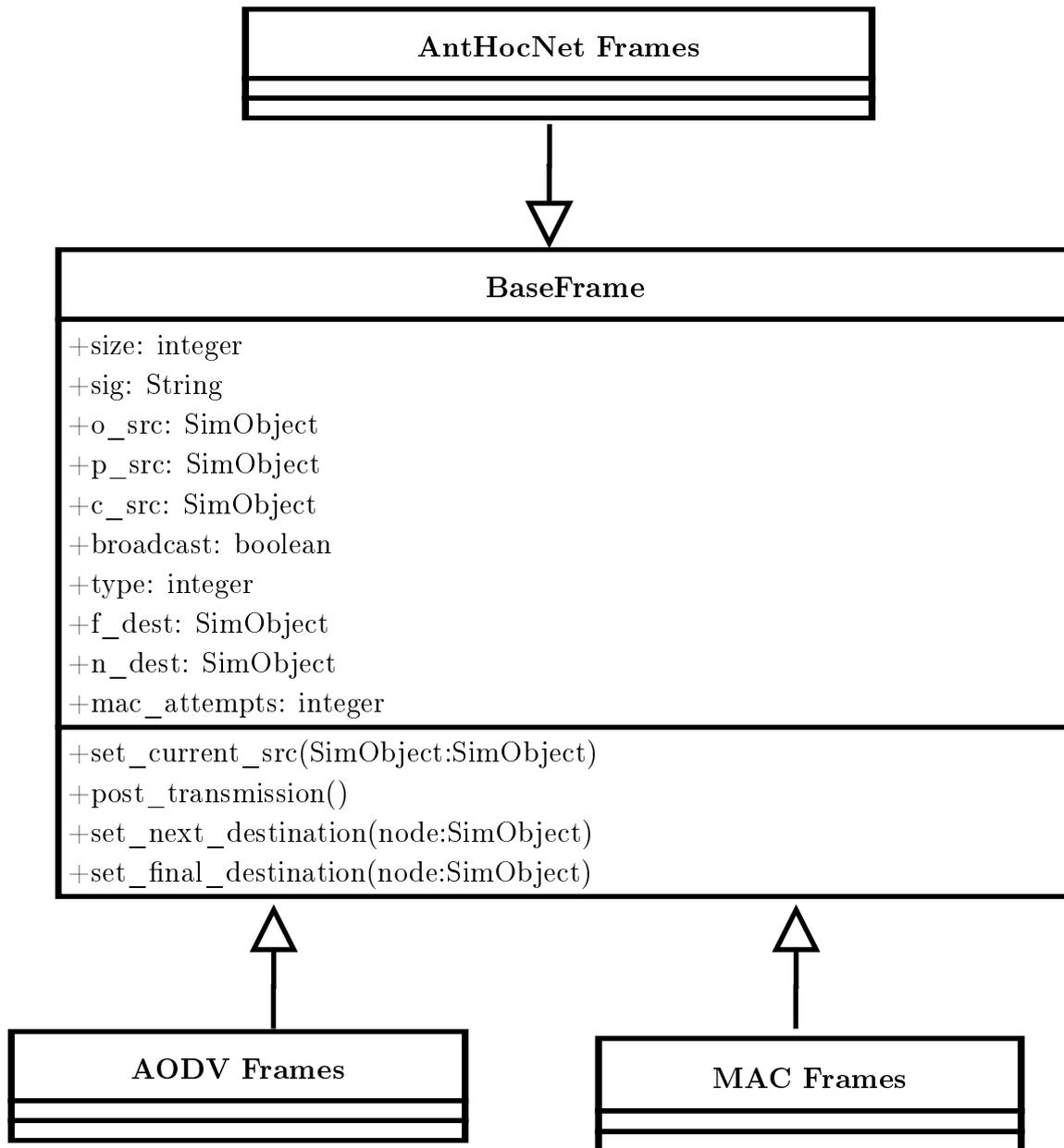


Figure 4.4: Generic Frame Class and its Derived Classes

results. Tests were only concluded to be successful if their results were close enough to the analytical values. The following functionality is therefore critical towards a well implemented MAC layer:

1. Nodes will physically sense the channel before transmitting, and defer or transmit accordingly (physical sense). If a node has visibility of a transmitting node, it senses the channel as busy.
2. Nodes will gather NAV data and defer or transmit accordingly (virtual sense). Nodes will gather NAV data from the appropriate headers and then defer the channel usage until such a time has elapsed.
3. Nodes will, given a large enough frame to transmit, first attempt and finish a handshake procedure before transmission. Retries as needed (handshake)

4. Nodes will wait the appropriate amount of time, including inter frame spaces and back off interval, before attempting to utilize the channel.
5. Nodes will defer the channel with the correct periods of times, and should happen fairly - no channel domination by a single node (back off, fairness)

Even a small scale simulation triggers a very large amount of (fortunately short) events within this context. Different issues will also arise based on different topologies (see the Hidden Node Problem and Exposed Node Problem). In order to ensure that the correct behaviour occurs as expected, multiple testing scenarios were created and tested with special attention given to the worst case scenarios.

These scenario's made use of Satellites in perpendicular orbits whom communicate with ISLs links.

4.4.1 MAC Layer Test 1

The first test which needs to ensure that the building blocks for the MAC layer are in place and working as we expect. A critical property which must be ensured is that channel access is fair.

This test was performed as a simple direct point to point connection which allowed the full function of the mac layer to be verified by occurring in the correct order via debugging. This tests was performed using MAC 802.11a and MAC 802.11SAT. As can be expected, due to their being no actual contention in this scenario, the MAC 802.11a performed much faster.

4.4.2 MAC Layer Test 2

For the second test, the exposed terminal problem will be evaluated. In order to successfully test this scenario, multiple nodes which can detect each other must be compared. A baseline case is taken of a single source transmitting to a single sink. This is compared with the exposed terminal problem, where an additional source is added which also transmits to the sink.

The expected results is a very high occurrence of blockages and broadcast errors. In this simulation it is possible to get multiple nodes to attempt broadcasts at the same time which will guarantee collisions. With two nodes competing for the channel both nodes throughput suffer.

For this case the MAC 802.11a configuration results in virtually no data being successfully transmitted. This can be attributed to the slotTime intervals being too short to allow the MAC layer to function properly.

4.4.3 MAC Layer Test 3

For the third test, the hidden terminal problem will be evaluated. The baseline case will contain a single node transmitting towards a sink. The same scenario will be repeated, however another node not visible to the original sink will attempt to sent towards the same sink.

The result is as expected: a high occurrence of collisions and retransmissions which will severely limit throughput.

4.5 Testing of AODV

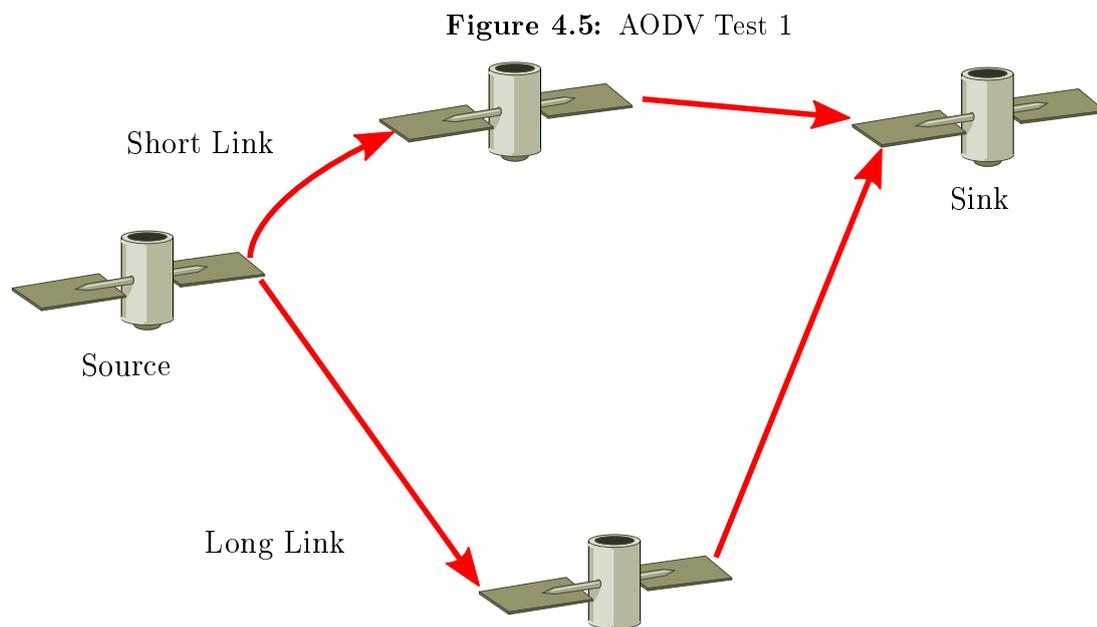
The testing procedure for AODV can be less rigorous than that of AntHocNet. This is due to there being comparable, reputable published results available for AODV [7]. However, some basic tests were included, specifically to cover scenarios not occurring in [7]. The key functionality which needs to be tested is described below.

1. **Route Discovery** - Ensure that only one path is active at a given time for a given logical connection.
2. **Route Recovery** - Ensure that data will be buffered correctly and routes will be searched for whenever possible.
3. **Route Repair** - Ensure that routing entries are correctly updated upon a link break and the appropriate action taken.

In order to test these conditions, scenarios were created where the results can be easily deduced via analysis and confirmed via test scripts. This will also serve as a test of the ability of SatSim to evaluate the routing metrics, since they can also be easily calculated and compared to what the simulators generates.

4.5.1 AODV - Test 1

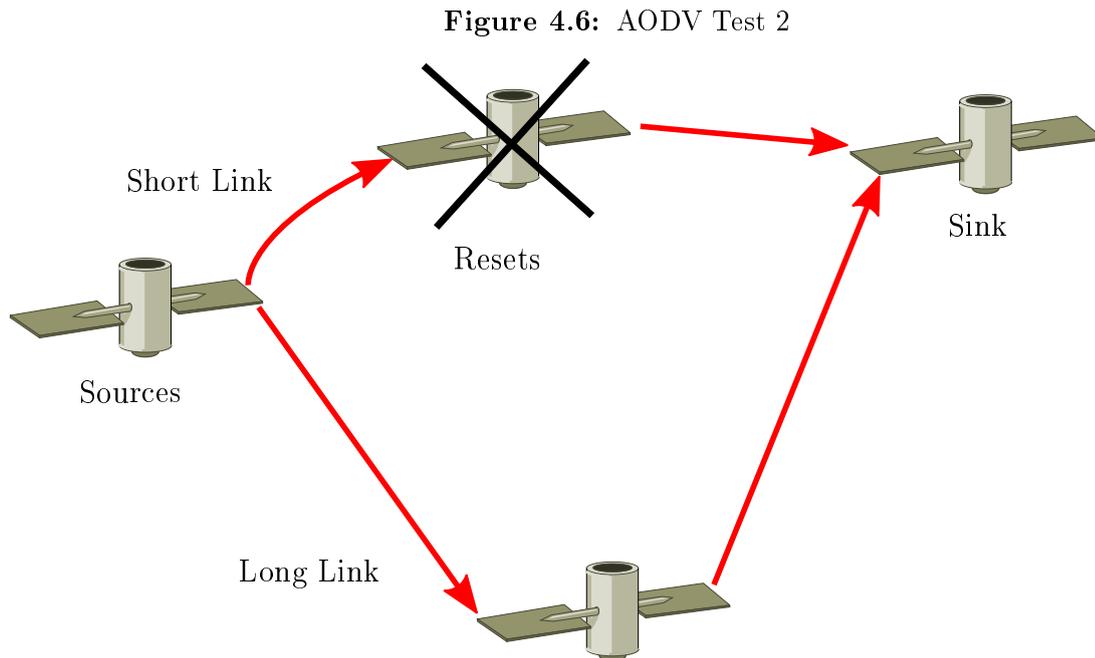
The first test is extremely basic and focuses on testing route discovery as well as verifying the expected values given for the network metrics. Due to the limited nature of this simulation, many of the results can be analytically determined which will act as an additional layer of verification.



The results given by the simulation were as expected and key values were analytically verified with test scripts. The path was chosen over the short link, as expected and all data was transmitted via that link.

4.5.2 AODV - Test 2

This tests uses the same constellation as test 1, however the preferred path resets during the data transmission. This means only one route is left for data to be transferred, which will allow the Route Repair mechanism to be tested.



Once the node resets has occurred, the protocol was verified to follow the appropriate route repair process. The old routing data expires and a new route request is generated, this time with the only remaining, less efficient route. Data transmissions were then verified to continue normally.

4.6 Testing Of AntHocNet

Due to the lack of available research on AntHocNet in a satellite environment, specific tests were used with specified input and the expected output determined with analysis.

In order to do verification of AntHocNet, the following functionality needs to be implemented and verified.

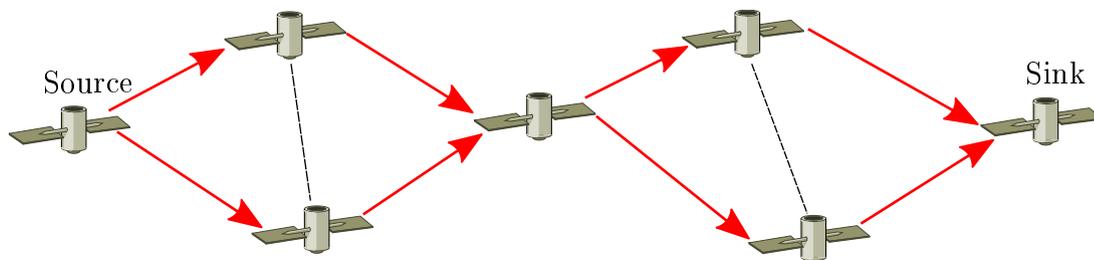
- **Route Discovery** - It needs to be verified that real pheromone routes are created upon route requests, and reactive ants are correctly broadcast and unicast back towards their source.
- **Route Maintenance** - Proactive ants should be created if high enough quality virtual pheromone is available and explore additional routes.
- **Route Repair** - Upon detection of a link break repair ants should be created and the route should be repaired if possible.
- **Pheromone Diffusion** - Hello Messages should share routing information with other nodes, eventually spreading out and describing the network as far as possible.

- **Balanced Data Forwarding** - Given multiple paths for data to travel, the path with better pheromone values should have the majority of the data being transmitted via it.

4.6.1 AntHocNet - Test 1

For this test, we consider a very basic satellite constellation in which most of the expected behaviour of the network can be easily analysed. This will be a quasi-static network, resulting in easily predictable behaviour.

Figure 4.7: AntHocNet Test 1



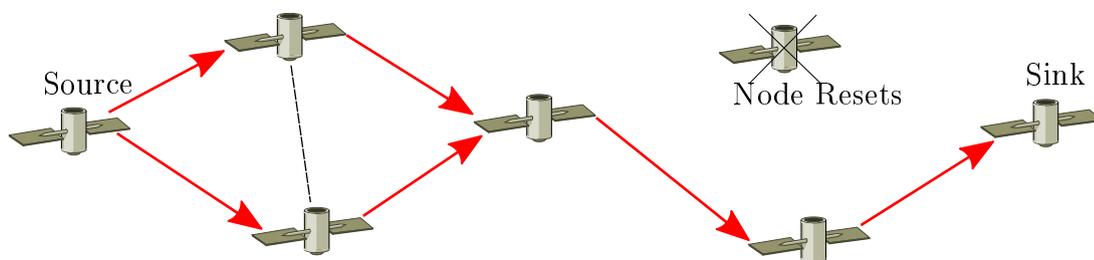
In figure 4.7, the red arrows indicate the path we expect data to take, with the shorter paths being favoured. Since the top path is shorter, we expect the majority of the data to travel via that path. Furthermore it is expected that the Source will have virtual pheromone values for every node in the system, but only real pheromone values for its neighbours and the sink.

The output of the test was analysed, and the results were as expected. The pheromone diffusion process led to the nodes being aware of all the other nodes in the network. The source node had real pheromone for the routes with its neighbours, that it is aware of due to the hello messages. A path is created towards the sink over the shortest path and data is transmitted as expected.

4.6.2 Link Break Test

For this test, we consider a basic satellite constellation where multiple routes from the source to the destination are possible and some key nodes are then removed from the most effective path. As is shown by figure 4.8, the simulation contains multiple routes to the sink. However, one of the nodes reset, which leads to all of the data being rerouted over the remaining path.

Figure 4.8: AntHocNet Test 2



The same behaviour as test 1 is seen, up until the node resets. The resulting link breaks cause the expected reactions of link failure notifications and a route repair process. The middle node learns that the route is broken and send out repair ants to establish a new route. Once this occur data keeps being forwarded along that path. The source node, is informed of the broken link but due to the link repair has to take no further action itself. The resulting pheromone tables correctly represent the new network topology at simulation end.

Chapter 5

Protocol Evaluation

This chapter describes the configurations and results for a variety of different network scenarios that were simulated on SatSim.

The first simulation describes an Iridium like network, using a large number of satellites to enable communication between two ground stations. This simulation was used for verification of the simulator as well as for testing its operation under very high data loads. However, it made no use of a MAC layer and raised some issues regarding the functionality of AntHocNet in a quasi-static network.

The second scenario discussed is a baseline case, involving a single satellite making passes over a single ground station. The data gathered will allow users to compare the benefits of a satellite network to the cost of launching and maintaining them.

The third simulation simulates the OuterNet in order to evaluate the performance of AntHocNet compared to AODV for a unique network. It is important to note that neither of these routing protocols were intended for a scenario such as the OuterNet, implying far more optimizations are possible.

5.1 Use of a Distributed Constellation

The performance of a vast distributed constellation, very similar to Iridium, can greatly benefit from a routing protocol such as AODV or AntHocNet. This is due to the lack of knowledge regarding the network topology each node has at any given instant - a node has to request information in order to know its visible neighbours, rather than there being some assumptions which can aid with the decision. However orbits can sometimes still be chosen to allow some relatively safe assumptions to be made, such as in the study this research will be using to verify against[7], which allow quasi-static network connections to appear.

The benefits of distributed constellations include superior latency, lower cost and less dependence on single satellites. The downsides include limited computational power and memory per satellite and the possibility of the large number of clients causing limited bandwidth to be available to each client.

5.1.1 Structure of the Distributed Constellation

The constellation which will be evaluated consists of a quasi-static network of LEO satellites which are dynamically connected to two ground stations. The network topology is

also affected by Doppler Shift, which leads to serious link breaks which occur whenever the satellites go above 75 degrees or below -75 degrees.

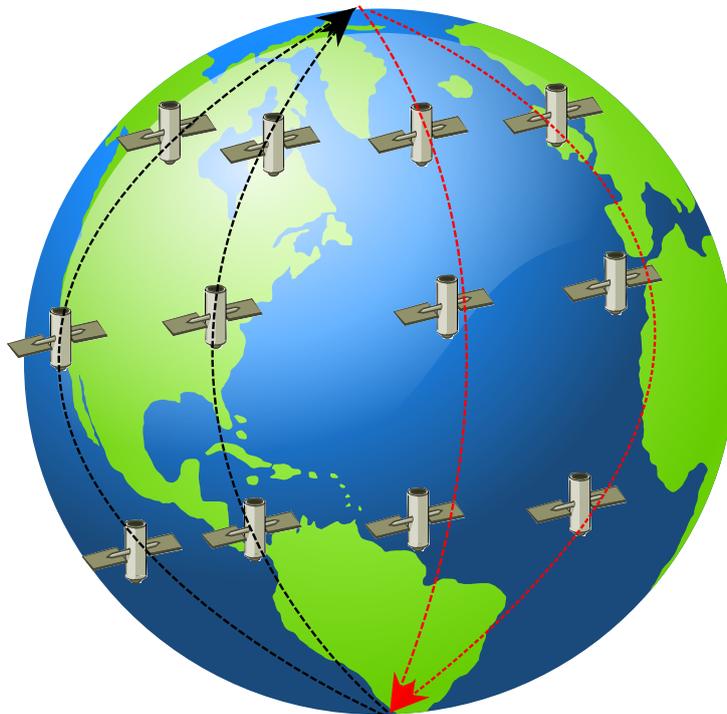
The orbital elements for the distributed network satellites are given in table 5.1.

Table 5.1: Constellation Satellites Orbital Elements

Orbital Element	Value
Altitude	1400 km
Inclination	86.4
Eccentricity	0
RAAN	18 x plane
Argument of Perigee	0
Mean Anomaly	30 x position

The physical arrangement of the satellites consists of 10 planes of 12 satellites each. The planes are positioned next to each other and therefore most of the satellites have four links in total, two in its own plane and two inter-plane links. However there exists a seam in the middle of the constellation that occurs due to the satellites in those planes moving in opposite directions. The constellation along with the seam is shown in figure 5.1. The position of the seam is arbitrary but has to occur between two of the planes.

Figure 5.1: Visualisation of the Constellation Showing Four Planes



The end-users of the network are two ground stations, whose locations are given in table 5.2. The first ground station will send a flood of packets towards the second for the duration of the simulation.

Table 5.2: Constellation End Users

Ground Station	Latitude	Longitude	Altitude
John Hopkins University, Washington DC	39.1886	-76.8833	140
Los Angeles	34.0522	-118.24	87

5.1.2 Analysis of AODV and AntHocNet on a Distributed Constellation

In order to validate the results of SatSim, it was attempted to replicate the results of an experiment outlined in [7]. This paper also proposed a novel routing protocol that was only tested on that specific network. Whilst this was not a fully tested protocol, it gave superior results to AODV which suggested that most network configurations can have specific protocols be optimally suited to them, by using the unique properties of the network in the protocol design.

5.1.3 Adjustments to AODV and AntHocNet for the Quasi-Static Network

In order to match the conditions described by [7], some adjustments had to be made to SatSim due to differences between how Lui et al described the simulation scenario and how SatSim was designed. Firstly, [7] does not make use of dropped packets due to lossy links and creates quasi-static links between the satellites. Secondly no channel access method was implemented, allowing only a functional verification of AODV and AntHocNet to be performed rather than a full practical simulation. Adaptions were made to the SatSim environment class in order for it to more closely match the scenario described by Lui et al.

5.1.4 Inter-plane connection of the Constellation

For this analysis a satellite constellation was created by organising the satellites into specific planes. This resulted in a quasi-static set of connections between the satellites, with a dynamic connection handover for the ground stations. Each ground station will only be connected to one satellite at a time, specifically the satellite with the highest elevation angle relative to itself at that instant. This specific constellation's attributes will favour certain protocols while placing others at a distinct disadvantage. Upon analysis it would seem that reactive protocols would perform far better in general in such a scenario.

This constellation study also made use of some physical phenomena that is not fully included in the standard SatSim configuration. These include antennas pointing only at specific nodes based upon certain conditions and Doppler Shift rendering satellites which approach each other incapable of communicating with each other. Most SatSim simulations assumed omnidirectional antennas, resulting in the addition of a condition wizard in order to allow pointing to be simulated. For this case the condition wizard would determine, for the full run of the simulation, which satellite had the highest elevation relative to the ground station and give it a connection only to that satellite, effectively simulating the antenna pointing towards the satellite.

The simulations were run for four hours of SimTime, since it is roughly the time it takes the satellite to complete two full orbits. The full simulation setting for this scenario is given in 5.3.

Table 5.3: Constellation Simulation Summary

Network Property	Value
Simulation Time	14400 seconds
Packet Size	122 bits
Nodes	122
Carrier Frequency	2.4 GHz
Data Rate	76000 bits/s
MAC Protocol	Not Used
Packet Generation Rate	Flood

The MAC layer was not tested for this constellation since it was not used in the original study and there is a very low likelihood of collisions due to the nature of the inter satellite-links and the ground stations only pointing towards one satellite at a time.

Another important aspect of this scenario is the large network size. The simulator did struggle to an extent with the memory consumption of the simulation, indicating that some programming optimizations may be beneficial to the implementations of AODV and AntHocNet on SatSim.

5.1.5 Results

The following subsection describes the results of the above simulation as well as giving an analysis of why the results occurred as they did. These results also appear in [38].

5.1.5.1 Latency and Route Discovery Time

The latency and route discovery times for both the SatSim evaluation and those generated by Lui et al in [7] are given in 5.4. As can be seen from the table the latency results generated by SatSim are similar to those generated by [7] on the Qualnet 3.9.5 implementation of AODV.

AntHocNet's results are similar to AODV, since the quasi-static network favours certain paths. The multipath nature of AntHocNet does not play such a major role in the simulation due to certain routes having far superior pheromone values.

Table 5.4: Comparison of AODV and AntHocNet for 1

Protocol	Average	Minimum	Maximum
AODV Latency	130	28.5	157
Lui et al [7] Latency	130	-	-
AntHocNet Latency	129	26	182
AODV Route Discovery Time	344	90	482
Lui et al [7] Route Discovery Time	365	-	474
AntHocNet Route Discovery Time	275	75	322

5.1.5.2 Overhead Generated

The overhead generated by the two protocols differs significantly due to their different functionality, as a Reactive Protocol Protocol AODV will generate RREQ and RREP

upon route discovery and RREQ, RREP and RERR upon a link break. As a Hybrid Protocol AntHocNet will send Reactive Ants upon route discovery and Proactive Ants while data is being transmitted. Repair Ants and Link Failure Notifications will be sent after a link break and Hello Messages will be periodically sent.

Table 5.5 compare the SatSim results to the Liu et al results. As can be seen the results for the amount of RREQ compare very favourably, although the amount of RERR and RREP generated differ significantly. It is assumed that this is due to a different method of counting the overhead frames, although it is also possible that there were some differences in the configurations of the networks, since some configuration settings had to be extrapolated or approximated from [7].

Table 5.5: AODV Reactive Overhead

	RREQ	DATA	RERR	DATA	RREP	DATA
SatSim	15042	192	389	160	396	160
Liu et al [7]	14922	192	2151	160	1671	160

The overhead generated by the AntHocNet implementation is given by 5.6 for the reactive elements of AntHocNet and by 5.7 for the proactive elements of AntHocNet. The overhead generated by AntHocNet is far more significant than that generated by AODV, effectively congesting the network unless specific changes were made to the protocol. These are detailed in the analysis of the results.

Table 5.6: AntHocNet Reactive Overhead

RA	DATA	RPA	DATA
452	102	1432	64

Table 5.7: AntHocNet Proactive Overhead

PA	DATA	HELLO	DATA
121	32	$2,17 \times 10^6$	320

5.1.6 Analysis of Results

For this constellation it was noted that AntHocNet has issues with link breaks when the connections in the network lead to a quasi-static network. The pheromone diffusion process used by AntHocNet pro-actively allows nodes to share routing information with each other and build new routes. Essentially, if node A informs node B it has a route to route C, route C will then note that it may have a new route to B via route A. This means that when a link break occurs a large number of routes become unusable. Due to the way AntHocNet reports these breaks, a large amount of overhead is generated which in turn generates more overhead.

This reaches a point where the network becomes nearly unusable due to the sheer amount of Link Failure Notifications (LFN) that are sent over the network. This also becomes a functional issue for SatSim, as it can result in memory errors due to the sheer amount of extra traffic generated. In order to implement AntHocNet in this scenario

an alternative method of handling link breaks was implemented, which cleared fully and reacted better to link breaks in a quasi-static network. Essentially for this alternative implementation of AntHocNet, the pheromone entries were manually cleared upon detection of a link break, allowing the network to rapidly rebuild itself without the use of excessive link failure notifications or unicast warning messages. This implementation of AntHocNet performed slightly better than SatSim in terms of latency and route discovery time, however it is only applicable to this specific network.

5.1.7 Observations on Results

Other than providing additional verification for SatSim, this scenario showed quite clearly that certain routing protocols can have far superior performance to another for a given scenario. This can usually be derived from the properties of the routing protocol itself - AODV was meant to find a single path quickly and then use that path until it breaks, regardless of the availability of other paths. This scenario works very well with AODV since the path has to exist between the ground station and a satellite. Due to the ground station switching periodically a new path is often created and maintained. AntHocNet, however is meant to discover a path to a destination, but then constantly gather and update new routing information allowing the data to be sent over multiple routes. However, these routing data gathering techniques are much more cumbersome and vulnerable to link breakages. The modified version of AntHocNet with dynamic pheromone table clearing has comparable results to AODV in terms of route discovery and latency, however this is not a true representation of AntHocNet in this scenario, where almost no data passes through the network due to the sheer congestion caused by overhead.

A very important observation that emerges from the study [7] is that a custom, scenario specific routing protocol can have far superior results for that scenario than a general purpose routing protocol. SatSim can therefore add value to the field by acting as an efficient tool to research new, scenario specific routing protocols with.

5.2 Baseline Scenario

This section details the comparison of AODV and AntHocNet on a simple baseline scenario that consists of a single client satellite and ground station, as can typically be afforded by a single organisation or university. This baseline scenario will not cover certain issues such as collisions of competition for a communication channel, but will still give an accurate representation of data throughput for the given communication windows.

The critical parameter to consider here will be access time, in other words, the amount of time a communication link is possible between the client satellite and the ground station. This will be an important factor to compare with the OuterNet which will seek to maximise the access time for a given client satellite.

For the orbital elements of the client satellite refer to 5.8, which describes a basic polar orbit.

In order to present an accurate summary of the performance of the client satellite, the simulation should preferably simulate at least one full cycle of the satellites behaviour. For this research the arbitrary time of one day is chosen. For comparison with the OuterNet the Satellite Access Ratio is established with the equation given below:

$$\text{Satellite Access Ratio} = \frac{\text{Access Time}}{\text{Orbit Time}} \quad (5.2.1)$$

Table 5.8: Baseline Client Satellite

Orbital Element	Value
Altitude	650 km
Inclination	85
Eccentricity	0
RAAN	0
Argument of Perigee	0
Mean Anomaly	0

Using the above equation we get an access ratio of roughly 14 percent, as we only have communication for 13 and a half minutes of the 1.62 hours orbit.

The results obtained for the baseline case are summarised in 5.9, which shows slightly better results for AntHocNet.

Table 5.9: Baseline Client Satellite - 24 hours

Protocol	Data Frames Transmitted
AODV	6538
AntHocNet	7522

AntHocNet was never intended to be used as a point-to-point protocol, however an interesting result occurs in this scenario in that almost no overhead other than hello messages are generated. When a hello message is successfully received, the only possible path is present rendering the ants superfluous and data can instantly be transmitted. AODV however, has to go through its basic route discovery process first.

For this simulation, it should be noted that for every pass there will be a communication window during which data can be sent from client satellite to the ground station. However, additional overhead will be needed for the more complex networking routing protocols. The optimal way to handle just that single communication would be the use of a connectionless routing protocol, that would not allow the OuterNet to function as a network. This suggests that an alternative to a network can be used to forward data from the OuterNet to the ground stations.

This alternative will involve the OuterNet relay satellites functioning as data mules, as suggested by [1]. Each relay satellite will accept as many data frames from a client satellite as possible and keep them buffered until it has LOS with the ground station. It will then forward the data effectively transmitting the clients satellite's data frames.

5.3 Comparison of AODV and AntHocNet for the OuterNet

The OuterNet was a novel relay network proposed by [1] that will serve as a platform to evaluate the performance of AODV compared to AntHocNet. It is important to emphasise that for this case study only the differences in performance between AntHocNet and AODV for a network with known properties are being tested. Neither of the routing protocols are specifically optimized either in terms of parameters or design for the OuterNet. The results generated therefore do not give a fair reflection of what is possible on the

OuterNet. As [7] suggested, a custom routing protocol can have vastly superior results compared to a general case routing protocol and considering the resources used to get a satellite network operational, the additional time to choose the best routing protocol during mission planning phase is justified.

5.3.1 Structure of the OuterNet

The OuterNet consists of a ring of equatorial satellites that act as relay satellites or data mules for client satellites. The orbital elements for the relay satellites are given in 5.10 and shows the OuterNet will be able to service LEO satellites that have an altitude lower than 900 km. Since these satellites have identical orbits and only differ in their positions in the orbit, this effectively means there is a quasi-static communications ring available to all the client satellites.

Table 5.10: OuterNet Satellites Orbital Elements

Orbital Element	Value
Altitude	900 km
Inclination	0
Eccentricity	0
RAAN	0
Argument of Perigee	0
Mean Anomaly	25.7 x position

The OuterNet satellite will not deviate from a path more or directly above the equator, which limits the viable ground stations for such a network. [1] suggests the following potential ground stations, described by 5.11.

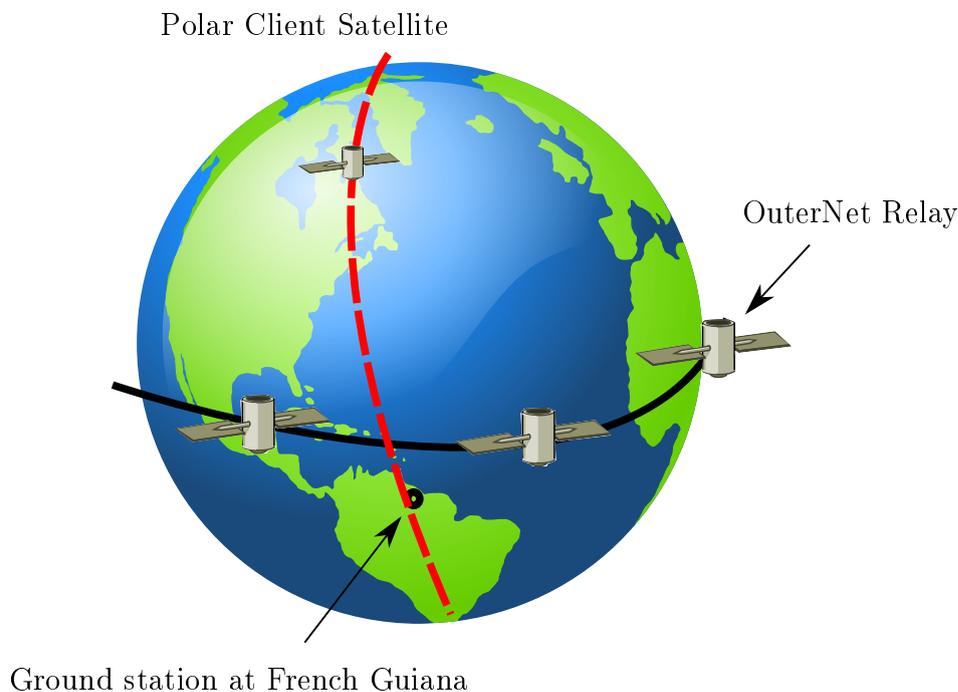
Table 5.11: OuterNet Ground stations

Ground Station	Latitude	Longitude	Altitude
European Space Centre, French Guiana	5.2909	-52.77	0
Remote Sensing Agensi, Malaysia	3.162	101.69	140
Luigi Broglio Space Center, Malindi Kenya	-2.99	40.19	0

For this research the European Space Centre was used as the ground station and effective sink for the data.

5.3.2 Function of the OuterNet

The OuterNet can increase the throughput of the baseline scenario in a fairly straightforward way. Whilst the baseline satellite only acquires LOS with the ground stations communication range x times per day. However, it will acquire LOS with one of the OuterNet relay satellites y times per day. A close approximation of what the new throughput will be, is to simply multiply the communication window throughput with the amount of additional communication windows. By simulating the same client satellite with the addition of the OuterNet ring, the exact increase in throughput can be quantified.

Figure 5.2: Representation of the OuterNet

5.3.3 Results

For this section of the research, it is of interest to see how easily the OuterNet can be implemented on the SatSim environment and if it provides a good platform to analyse the performance differences for AODV and AntHocNet for the OuterNet.

Table 5.12 shows the results for a simulation of the OuterNet for a single client satellite. When compared with table 5.9 it can be seen that the total throughput for the simulation decreased. The reasons therefore were analysed using SatSim and are given in the next subsection.

Table 5.12: Baseline Client connected to OuterNet for AODV and AntHocNet

Protocol	Data Frames Transmitted
AODV	2480
AntHocNet	4408

5.3.3.1 Inefficient Routing

Several issues occur with AODV due to inefficient routing decisions. As AODV is a single path algorithm, it will not deviate from a suboptimal path unless a link break occurs. This means that for a given path it can often be using a suboptimal path for an extended period of time. As LOS is often achieved with a relay satellite first, suboptimal paths can often form making several passes less efficient. The client will then continue sending over a longer path utilising the relay satellites, even if a direct connection is possible. AODV also sometimes has routing information expire if enough collisions occur, due to certain time-outs occurring.

This issue does not occur with AntHocNet. Data is stochastically forwarded when multiple paths are available, allowing the better paths to be utilised. Multipath algorithms therefore seem to perform better in an OuterNet type scenario.

However, both protocols suffer due to path discovery time being high relative to the size of the communication window, and extremely long retransmission times due to the adjusted MAC layer. Collisions therefore have an extremely significant effect on this simulation, and avoiding channel contention and collision via specific schemes such as special band for ISLs should be highly beneficial.

5.3.3.2 Suboptimal Configuration of the Protocols

Neither of the protocols were fully optimised in terms of their parameters for the given simulation, although adjusting the interval at which AntHocNet sent Hello Messages already started to produce favourable results. However, the two protocols were kept in their default configurations in prevent one being favoured over the other one as the its was more of interest to see what the typical behaviour of the protocols in such a network would be, rather than their optimal behaviour.

5.3.4 Observation on Results

As can clearly be seen from the results, neither AntHocNet nor AODV were designed for a scenario such as the OuterNet. While the OuterNet does increase the amount of passes and total access times, it lowers the efficiency of every single pass. This research does not claim that these are the optimal results available for this scenario - there are many different configurations and some simple optimizations possible. For example, a modified version of AntHocNet can be implemented which is always aware that the OuterNet satellites have links with each other. The initial idea proposed by the OuterNet team was to use only point-to-point and utilise the OuterNet relay satellites as data mules. This could maximise the throughput per pass.

Lastly it should be pointed out how easy it is to organise a network such as the OuterNet in the SatSim environment. A simple loop which adjusts the Mean Anomaly of the created satellite was added to the baseline script and simulation could then continue.

5.3.4.1 Effect of additional client Satellites

Additional client satellites can be added to make use of OuterNet's services, and they can be added in two main ways. Adding an additional client satellite with a sufficiently different Mean Anomaly will result in an an additional client making use of the OuterNet whilst having minimal effect on the other clients. However, when one places an additional client in a shifted orbit by manipulating the RAAN of the client satellite, it may have a far more significant effect on the network.

The preliminary unoptimized results determined for the OuterNet suggest that the original plan of using the OuterNet relay satellites as data mules may give better results overall as well as maximising the given throughput for a single point-to-point communication window.

Chapter 6

Conclusion and Recommendations for Future Work

6.1 Conclusion

The three objectives for this research were the expansion of SatSim as a full satellite network simulator, the analysis and implementation of two complex distributed MANET routing protocols and the evaluation of those two routing protocols in the same network scenario.

6.1.1 Expansion of SatSim

SatSim was first converted to an object orientated design and optimized as much as possible. The principle actors of a SatSim simulation, instances of simObjects, was expanded considerably in order to accommodate the more complex routing protocols and to be able to maintain more state information. Additional components for the SatSim environment were designed in order to allow additional functionality, such as table wizards and routing schedulers. It was attempted to maintain the main design drivers of user-friendliness, modularity and re-usability wherever possible but some concessions had to be made.

A major contribution to SatSim was the addition of a channel access method that was achieved through a partial implementation of a MAC layer. The objective of SatSim as a network simulator was achieved and verified, however there are still many additions possible based upon the unique needs of a given scenario.

6.1.2 Analysis and Implementation of Routing Protocols

AntHocNet and AODV were chosen as the two candidate routing protocols to be implemented on the SatSim environment. AODV was chosen since it had reputable results available for evaluation. AntHocNet was chosen since it is a complex routing protocol that covered most distributed routing functionality. Furthermore very limited research for AntHocNet in a satellite environment exists which allows this research to contribute more to the field. Basic scenarios were created and used to verify the initial functionality of the protocols while a full verification process was undertaken for AODV. AntHocNet was also studied for the same Iridium like network which lead to some observations regarding the performance of AntHocNet in a space-based and quasi-static environment. The implementation of the two protocols can be considered a success, although there is

still additional functionality that can be implemented as well as modifications which can be added for space-based applications.

6.1.3 Evaluation of Routing Protocols

AntHocNet and AODV's performance was then evaluated on a specific novel network called the OuterNet. The results generated suggested that a novel network protocol specific to the OuterNet would be better than using a general case routing protocol, due to the atypical nature of the OuterNet. A key finding of this research was that SatSim can be used to evaluate network performance for a specific network and can be used to create and test new routing protocols as well as contribute towards adapting the large wealth of information regarding terrestrial MANETs to space based applications.

6.2 Recommendations for Future Work

Since this research was very open-ended, a wide variety of new work is now available to be researched using the SatSim platform. This work can be divided into further expansion of the simulator, additional protocols to investigate and investigating a new strategy of routing that hasn't been extensively researched yet.

6.2.1 Expansion of SatSim

SatSim is designed to be very modular and flexible. A great deal of additional functionality can be added, that will allow additional unique scenarios to be simulated. However, care should be taken to ensure that memory consumption and execution time do not change significantly after these additions. One of the main design goals of SatSim is that additional functionality should be easy to add and can be disabled if not needed.

6.2.1.1 Antennas

SatSim currently has a functional antenna system but this can be improved to allow for beam shaping and more complex antenna patterns to be simulated. The current assumptions of omnidirectional antennas can also be replaced by modelled pointing, as well as more accurately simulating switching times. A system which read in and implement antenna patterns can also be highly beneficial.

6.2.1.2 Evaluation and Modification of the MAC layer

The use of different MAC layer techniques for MANETs has not been extensively studied or adapted for satellite constellations. Some observations have been that the inter frame wait times should be adjusted and the contention windows and backoff timers may have to be adjusted, which resulted in the addition of the MAC 802.11SAT configuration. However, while this configuration offered better results, it is by no means an optimal solution. Implementing a variable contention window such as suggested by [33] is one of many possible optimizations to the MAC 802.11 standard which can be investigated in SatSim.

The implementation of the MAC layer was also not satisfactory upon evaluation. It would be better to recreate the MAC layer as two separate classes for transmission and

receiving. Each transmitter and receiver would then have its own instance of a MAC layer, allowing communication to occur in different bands.

6.2.1.3 Addition of GUI and Animator

It will be beneficial to user-friendliness if an easy to use GUI is added to the SatSim environment. This GUI will have to be well designed in order to allow the user the same scope of control that scripting does.

The Animator class in the original SatSim suffered from cross-platform issues. An animator may improve user experience and allow some additional insights to be gained from the simulations. However, it should be crucial that this not affect the cross platform performance of SatSim.

6.2.2 Additional Protocols to Evaluate

Whilst AODV and AntHocNet were sufficient to verify the expansion of SatSim and allowed research to be carried out, the initial research suggested that investigation on two specific additional protocols may also yield interesting results.

6.2.2.1 ZRP

ZRP can be useful to work with some of the quasi-static satellite constellations that can be achieved by correctly choosing orbits. ZRP will be well suited to these networks by correctly choosing the zone areas.

6.2.2.2 Optimize Performance of the OuterNet

The results obtained from this thesis has suggested that both AODV and AntHocNet will produce suboptimal results for the OuterNet network. However, the OuterNet has a set of specific properties that can be used to create a novel custom routing protocol which utilises them directly. The constant links between relay satellites and knowledge of the the position of the ground station can easily be built into such a protocol. This protocol should compare its performance both with the baseline scenario provided here and with the alternative scenario of the OuterNet relay satellites acting as data mules.

An additional change which will help reduce collisions and competition for the same node, that can occur when a client and relay satellite want to transmit to the same relay satellite is to assign a custom bandwidth for the ISL.

6.2.2.3 Aggregated Network

SatSim currently assumes that networks will use a single, universally used routing protocol. AODV and other routing protocols do make provision for aggregated networks and there may be benefits to having specific satellites use their own specialised or optimized routing protocols. However, the OBC will have to be expanded in order to allow the functionality of multiple protocols to occur simultaneously.

6.2.3 Investigations

The two investigations detailed below do not directly tie in to the research performed in this thesis, but act rather as suggested uses for the SatSim platform.

6.2.3.1 Hierarchical Satellite Constellations

The use of hierarchical satellite constellations for quasi-centralised routing has been noted by this study but not undertaken, partially due to lack of information regarding that type of satellite links as well as there not being a clear indication of real world interest. However, it would still be advantageous to research this method of routing and the associated protocol to see how it compares to the methods outlined here for a satellite network.

6.2.3.2 Adaptation of Existing terrestrial WSN knowledge

The field of protocols and strategies for WSN on terrestrial networks is far more expansive and well studied than that of the same protocols executed on satellite networks. Furthermore as the issues with verifying AntHocNet showed, there exists little to no research for some protocols that has been conducted explicitly on Satellite constellations. With SatSim suitably expanded more work can be done to evaluate different protocols in satellite constellations as well as to see what procedures are best suited for the unique topology characteristics and what is needed to port these protocols and strategies.

This can be further improved by including additional terrestrial propagation models, that will allow additional routing protocol results to be reproduced, adding additional verification to the SatSim implementation of the routing protocol.

Appendices

Appendix A

SatSim Supplementary

The following Appendix Cover the supplementary information for the installation and use of SatSim. It also includes additional diagrams and information which will allow the user to easily configure SatSim for unique simulations and add their own protocols and additional functionality.

A.1 Installation

In order to allow easy cross platform use, SatSim makes use of custom functions wherever possible and only uses external libraries where absolutely needed. The following appendix details which packages have to be installed in order to use SatSim and the reasoning behind using them.

The core language SatSim is written in is Python, due to it being highly reliable. Most of the computationally expensive functions or high loop functions have been executed using C based procedures in NumPy or PyEphem.

A.1.1 PyEphem

PyEphem is used to calculate the positions of the satellites. The PropagationModels class makes use of this module to determine the position of the simulation nodes as needed. The use of the module was compared to the freely available SGP4 [39] module in 2015, but while similar results were generated PyEphem had far better execution times. Upon analysis it was confirmed this was due to SGP4 being written mostly in Python while PyEphem made extensive use of C.

A.1.2 NumPy

NumPy was extensively used due to it's popularity and fast execution time. All matrices in SatSim are NumPy matrices, optimized according to their likely or expected data type. Furthermore all NumPy matrices and their operations are highly optimized and should be fairly familiar to any user who often uses them in Python.

A.1.3 SimPy

SimPy as used for the discrete event driven simulation. This was due to it's use being established in previous versions of SatSim and it's extensive resources. The expansion

of SatSim to allow networks also created the need for far more timers and events, which SimPy was capable of easily facilitating.

A.1.4 cPickle

Used for the storage of the large amount of output data generated by the SatSim environment as well as the storage of network tables. cPickle is used since the stored data will be either single values or matrices rather than complex objects.

A.1.5 Summary of Dependencies

Table A.1: Summary of Dependencies

Dependency	Available at	Version
PyEphem	http://rhodesmill.org/pyephem/	0
NumPy	numpy.org	0
SimPy	https://simpy.readthedocs.io/en/latest/	0
cPickle	https://docs.python.org/2/library/pickle.html	0

A.2 Configuration and Use

The following appendix includes a list and brief summary of all the currently implemented configurations available to SatSim. SatSim is designed so that each new configurations can simply be added to it's corresponding module.

The following modules can be configured:

- Data Link Layer
- Mac Layer
- Network Layer
- Transport Layer
- Upper OSI Layers
- Modulation Scheme
- Strategy
- Memory

A.2.1 Physical Layer

The physical layer in SatSim is not directly configured by users, although some options can be set as necessary. The full physical layer comprises of the environment and rfchannel class in general, and an instance of the receiver and transmitter class for a specific link.

A.2.2 Data Link Configurations

The data link layer directly interacts with the physical layer and sends frames there. The functions of a data link layer will always be present in any connection, as it handles directly the sending of data between two nodes.

A.2.3 MAC Layer

The MAC configurations allows you to specify which configuration is to be used. Currently the following variations of the IEEE MAC 802.11 standard are available:

- MAC 802.11a
- MAC 802.11b
- MAC 802.11SAT(Custom)

A.2.4 Protocols

The following protocols have been implemented in the SatSim environment:

- AntHocNet
- AODV

A.2.5 SatSim Network Configuration

Table A.2: SatSim Network Configuration

Configuration	Unit	Mandatory
simTime	seconds	Yes
Number of Nodes	integer	Yes
Validation Mode	boolean	No
Mac Layer	String	No
Start Date Time	PyEphem Date	Yes
RunName	String	Yes

Appendix B

Routing Supplementary Information

This section briefly describes some of the issues inherent to routing in MANETS. These issues frequently occur while implementing routing protocols for MANETS and while most strategies cannot fully solve these problems but some forms of mitigation are possible.

B.1 Inherent Issues in MANET routing

This section only aims to outline such issues as they are mentioned and approached in this thesis and note how this may apply to satellite constellations and also which steps have been taken to mitigate or remove their effects. Some studies have referenced potential issues with AODV loop freedom (which can potentially lead to the counting to infinity problem), however the networks we implement are so small relative to terrestrial networks this is never likely to be a significant issue.

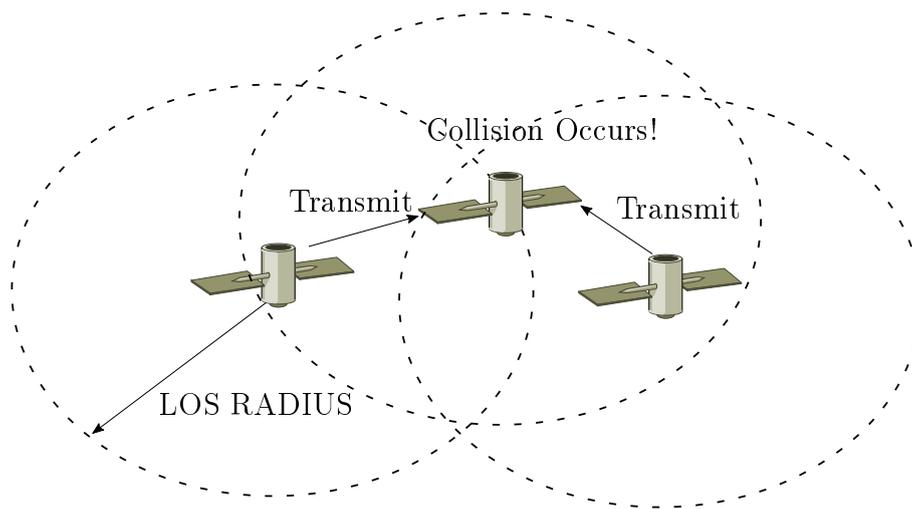
B.1.1 Counting to Infinity

The count-to-infinity problem occurs due an inherent problem in sharing routing information, specifically when a node informs another node of a path it has to some destination.

Tanenbaum [40] states the problem as follows: When node x tells node y that it has a path to node z , node y has no way of knowing whether it itself is on the path. So if node y detects a broken link to a third node z , but x says it has a valid path to z , node Y will assume node x does still have a path to node z , assigning node x as the next hop towards node z . Since that path x assumed is through Y , node x and node y will start updating each in a loop which eventually reaches infinity. While this is a pertinent issue, it is unlikely to occur in a satellite environment and some sanity check have been added to prevent it if it does occur.

B.1.2 The Hidden Terminal Problem

The hidden terminal problem is intrinsic to any single channel contention scheme. It occurs when two or more nodes with no vision of each other wishes to transmit to a third node. Due to both nodes attempting to transmit to a node at the same time, collisions occur with no clear way to avoid them. This scenario is illustrated by figure B.1. In this research a custom MAC layer implementation was added in order to mitigate the effects of the hidden terminal problem.

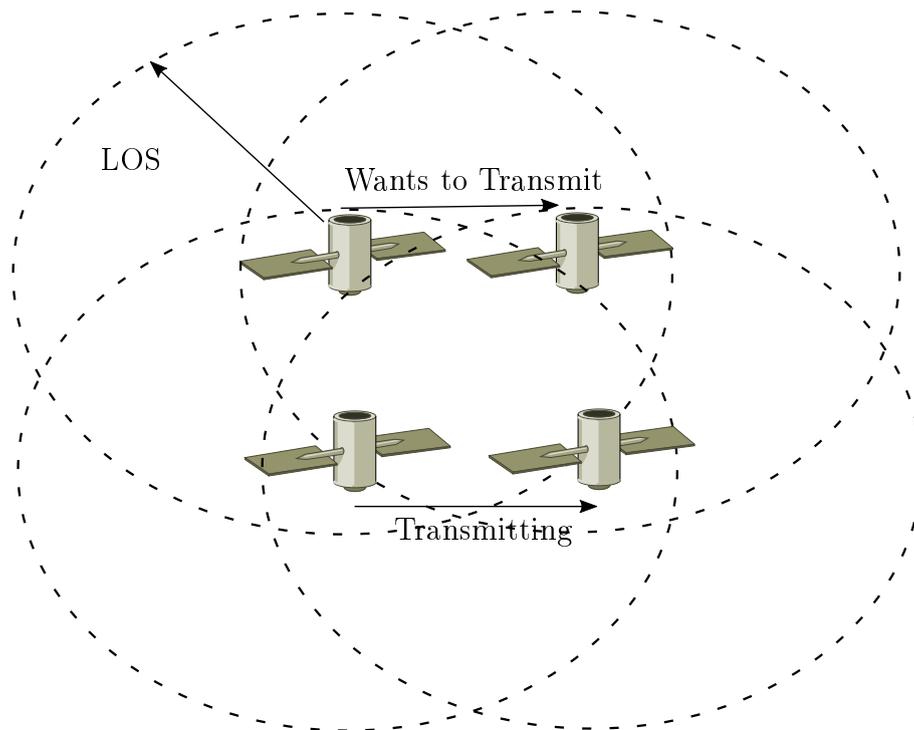


Transmitting Nodes have no vision of each other

Figure B.1: The Hidden Terminal Problem

B.1.3 The Exposed Terminal Problem

The exposed terminal problem is more or less the inverse of the hidden terminal problem. This occurs when a node detects activity on a channel does not know which node is currently busy. It can then incorrectly assume that a free node is busy, wasting time when the channel was available to it. This scenario is shown by figure B.2. In this research a custom MAC layer implementation was added in order to mitigate the effects of the hidden terminal problem.



All nodes have vision of each other

Figure B.2: The Exposed Terminal Problem

Glossary

- AppID** A unique identifier which groups data packets as belong to a specific stream. 58
- backoff** A set amount of time a node waits before reattempting some failed transmission or to use the channel again. 57, 78
- broadcast** A Data transmission from a node which is sent to every node which has vision of it. Can lead to flooding. 16, 20, 25, 38, 41, 44, 54–56
- collision** Occurs when a node receives data from multiple sources at once, resulting in the relevant packets being dropped. 31
- DataManager** The SatSim class which is used for the storing and analysing simulation results.. 9
- flood** A process where broadcasting or spamming leads to a single message overwhelming and saturating a network. 38
- geostationary** Refers to a satellite at approximately 35786km orbit. It is so called because it appears stationary relative to the earth. 23
- Hello Message** A short message sent to neighbouring nodes, usually to determine which nodes are currently visible from the node. 16, 17, 20
- HEO** A High Earth Orbit(HEO) satellite has an orbit above 35786 kilometres. 15
- Hybrid Protocol** A protocol which makes use of both proactive and reactive elements in its functionality. 42, 71
- Jitter** Jitter is the difference in time between the arrival of packets. Also called inter-packet delay.. 10
- LEO** A Low Earth Orbit(LEO) satellite is a satellite whose orbit is lower than 2000 kilometres. 15, 23
- Lifetime** Refers to the expected amount of time for which routing data is valid. 37
- MEO** A Medium Earth Orbit(MEO) Satellite stays in orbit in an altitude between 2000 and 35786 kilometres. 15
- Neighbour** A node which currently has line of sight with the current node. 13, 20, 44
- OuterNet** A proposed ring of equatorial satellites which will act as an effective relay network for sun synchronous satellites by providing additional access windows.. 2, 6, 27, 67, 72–76, 78, 79

- precursor** In AODV, refers to all the nodes which may use or be dependant of a specific routing entry. 40
- Reactive Protocol** A protocol which only updates routing information (and therefore generates overhead when explicitly needed for transmissions. Often called on-demand.. 37, 70
- RFChannel** The class in SatSim which is used to simulated communication links between two nodes. 56
- Route Discovery** The process a node undertakes to find a routing path towards a specific destination. 19
- Routing Freedom** A term which refers to a routing protocols ability to prevent messages from looping indefinitely in the network. 21
- SatSim** A Satellite Simulator developed at Stellenbosch University. 3–8, 11, 16–19, 21, 26–29, 31, 33, 34, 37, 42, 43, 50, 51, 53, 54, 56–59, 63, 67, 69–72, 75–80, 82, 83
- SimObject** A node in the satellite environment which will either be a Ground Station or a Satellite. 53, 58, 59
- simTime** Refers to the amount of time(usually in seconds) which has elapsed in the the simulation. 34, 53, 59
- slotTime** The size of the intervals the MAC layer waits in before sensing the channel again. 31, 32, 36, 62
- switching times** Time it takes for a node to go from receiving mode to transmit mode. 78
- Telecommand** A communications process which is used to control a satellite. 3
- Telemetry** A communications process by which measured data is transmitted from a source to a destination. 3
- Throughput** The Rate at which data is send from an source node to a destination node. 9
- unicast** A Data transmission from a node which is sent to only a single visible node. 39

Bibliography

- [1] Kearney, M., Botma, P., Jordaan, H., Gerber, J., Thesnaar, E., Nolte, F., Erlank, A., Groenewald, C. and Barnard, A.: The outernet: A novel satellite communication relay constellation. 2013.
- [2] Al, D.M.E.: Qb50 is an european fp7 project for facilitating access to space. 2016. Available at: <https://www.qb50.eu/index.php>
- [3] Laboratory, E.S.: Sunsat. 2003. Available at: <http://research.ee.sun.ac.za/sunsat/>
- [4] N2YO: Sumbandila. Available at: <http://www.n2yo.com/satellite/?s=35870>
- [5] Le Roux, J.-H.: *Development of a satellite network simulator tool and simulation of AX. 25, FX. 25 and a hybrid protocol for nano-satellite communications*. Master's thesis, Stellenbosch: Stellenbosch University, 2014.
- [6] Bezuidenhout, Q.: *Satellite Communications Strategy selection for optimal LEO satellite communication*. Master's thesis, Stellenbosch: Stellenbosch University, 2012.
- [7] Liu, X., Gokhale, D., Mizas, T. and Springston, P.: Analysis of ip routing approaches for leo/meo satellite networks. 2010. Available at <http://enu.kz/repository/2010/AIAA-2010-8685.pdf> , 28th AIAA International Communications Satellite Systems Conference , 2010.
- [8] Corson, S. and Macker, J.: Mobile ad hoc networking (manet): Routing protocol performance issues and evaluation considerations. Tech. Rep., University of Maryland , Naval Research Laboratory, 1998.
- [9] Johnson, D.B. and Maltz, D.A.: Dynamic source routing in ad hoc wireless networks. In: *Mobile computing*, pp. 153–181. Springer, 1996.
- [10] Johansson, P., Larsson, T., Hedman, N., Mielczarek, B. and Degermark, M.: Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In: *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pp. 195–206. ACM, 1999.
- [11] Hoebeke, J., Moerman, I., Dhoedt, B. and Demeester, P.: An overview of mobile ad hoc networks: applications and challenges. *Journal-Communications Network*, vol. 3, no. 3, pp. 60–66, 2004.
- [12] Heinzelman, W.R., Chandrakasan, A. and Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: *System sciences, 2000. Proceedings of the 33rd annual Hawaii international conference on*, pp. 10–pp. IEEE, 2000.
- [13] Chen, C. and Ekici, E.: A routing protocol for hierarchical leo/meo satellite ip networks. *Wireless Networks*, vol. 11, no. 4, pp. 507–521, 2005.

- [14] Lee, J. and Kang, S.: Satellite over satellite (sos) network: A novel architecture for satellite network. In: *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 1, pp. 315–321. IEEE, 2000.
- [15] Boukerche, A., Turgut, B., Aydin, N., Ahmad, M.Z., Bölöni, L. and Turgut, D.: Routing protocols in ad hoc networks: A survey. *Computer networks*, vol. 55, no. 13, pp. 3032–3080, 2011.
- [16] Marina, M.K. and Das, S.R.: On-demand multipath distance vector routing in ad hoc networks. In: *Network Protocols, 2001. Ninth International Conference on*, pp. 14–23. IEEE, 2001.
- [17] Perkins, C.E. and Bhagwat, P.: Highly dynamic destination-sequenced distance-vector routing (dsv) for mobile computers. In: *ACM SIGCOMM computer communication review*, vol. 24, pp. 234–244. ACM, 1994.
- [18] Rajagopalan, S. and Shen, C.-C.: Anisi: A swarm intelligence-based unicast routing protocol for hybrid ad hoc networks. *Journal of Systems Architecture*, vol. 52, no. 8, pp. 485–504, 2006.
- [19] Ducatelle, F.: *Adaptive routing in ad hoc wireless multi-hop networks*. Ph.D. thesis, Università della Svizzera italiana, 2007.
- [20] Perkins, C., Belding-Royer, E. and Das, S.: Ad hoc on-demand distance vector (aodv) routing. Tech. Rep., University of California, University of Cincinnati, 2003.
- [21] Johnson, D.B., Maltz, D.A., Broch, J. *et al.*: Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks. *Ad hoc networking*, vol. 5, pp. 139–172, 2001.
- [22] Clausen, T. and Jacquet, P.: Optimized link state routing protocol (olsr). Tech. Rep., Network Working Group, 2003.
- [23] Haas, Z.J., Pearlman, M.R. and Samar, P.: The zone routing protocol (zrp) for ad hoc networks. *Not Specified*, 2002.
- [24] NASA: The afternoon constellation. 2016.
Available at: <http://atrain.nasa.gov/>
- [25] Inc., I.C.: Iridium next:the bold future of satellite communications. 2016.
Available at: <https://www.iridium.com/network/iridiumnext>
- [26] Globalstar: Global star: Be heard. 2016.
Available at: <http://www.globalstar.com/en/>
- [27] plc, I.: Inmarsat website. 2016.
Available at: <http://www.inmarsat.com/>
- [28] Orbcomm: Orbcomm is a leading global provider of machine-to-machine (m2m) and internet of things (iot) communication solutions that remotely track, monitor, and control fixed and mobile assets. . . . 2016.
Available at: <http://www.orbcomm.com/>
- [29] Rhodes, B.: Pyephem: Basic astronomical computations for python. 2016.
Available at: <http://rhodesmill.org/pyephem/>
- [30] Group, O.: Outernet: Radio for the information age. 2016.
Available at: <https://outernet.is/>

- [31] Schildt, H.: *Java the Complete Reference 7E*. McGraw-Hill, 2007.
- [32] Marszalek, M., Rummelhagen, M. and Schramm, F.: Potentials and limitations of iee 802.11 for satellite swarms. In: *2014 IEEE Aerospace Conference*, pp. 1–9. IEEE, 2014.
- [33] Ambartani, P.R., Patil, P. and Devi, U.: Analysis and simulation of networking protocols in constellation flying system for inter-satellite communication.
- [34] Society, I.C.: Part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications. 2012.
Available at: <http://standards.ieee.org/about/get/802/802.11.html>
- [35] Van Glabbeek, R., Höfner, P., Tan, W.L. and Portmann, M.: Sequence numbers do not guarantee loop freedom: Aodv can yield routing loops. In: *Proceedings of the 16th ACM international conference on Modeling, analysis & simulation of wireless and mobile systems*, pp. 91–100. ACM, 2013.
- [36] Di Caro, G., Ducatelle, F. and Gambardella, L.M.: Anthocnet: an adaptive nature-inspired algorithm for routing in mobile ad hoc networks. *European Transactions on Telecommunications*, vol. 16, no. 5, pp. 443–455, 2005.
- [37] Zimmermann, H.: Osi reference model-the iso model of architecture for open systems interconnection. *IEEE Transactions on communications*, vol. 28, no. 4, pp. 425–432, 1980.
- [38] AJ, M. and A, B.: Simulating manets:a study using satellites with aodv and anthocnet, 2016. Submitted.
- [39] et al, V.: Track earth satellite tle orbits using up-to-date 2010 version of sgp4. 2010.
Available at: <https://pypi.python.org/pypi/sgp4/>
- [40] Holter, K.: Count to infinity. 2005.
Available at: <http://folk.uio.no/kenneho/studies/essay/node19.html>