TÉCNICO
LISBOA

Routing and Dropping Policies for Delay Tolerant Networks

Miguel Pinheiro Rodrigues

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor: Prof. Paulo Rogério Barreiros D' Almeida Pereira

Examination Committee

Chairperson: Prof. António Manuel Raminhos Cordeiro Grilo

Supervisor: Prof. Paulo Rogério Barreiros D' Almeida Pereira

Member of Committee: Prof. Miguel Nuno Dias Alves Pupo Correia

May 2017

To my family and girlfriend, for all the support and guidance

Acknowledgements

Firstly, I would like to thank my supervisor Professor Paulo Rogério Pereira, for all the support, patience and time dedicated. It would not have been possible to do this work if it was not for him, the dedication he puts in each student is outstanding, and definitely something that helped me a lot. Also, I would like to thank Professor Naércio Magaia, for making me feel integrated and as a part of INESC, and for his constant presence and help especially in frustrating coding matters. Both motivated me since the beginning, and I must acknowledge that I had all the support I ever needed. In addition, I would like to thank INESC, for all the resources given, such as the workspace and the server access, which allowed me to speed up my work.

Also, I would like to thank my mother, father, brother and sister, for providing me the education and conditions that allowed me to achieve my degree. They were the ones who guided me throughout my life, celebrating in the good moments, and advising me in the bad.

Furthermore, I would like to thank my girlfriend, Mafalda Gama, for celebrating with me at each moment of success, and for all the motivation and support given when things did not go as well. Her understanding and help, not only on this final phase of my master's degree, but since the beginning of my bachelor's degree were key to the course completion, and for that I thank her.

Finally, I would also like to thank all my great companions in my academic journey at IST, specifically: Alexandre Costa, António Mendes, Bruno Candeias, Daniel Sousa, Hugo Pereira, Jessy Neves, João Paquim, Luísa Aguilar, Miguel Monteiro, Pedro Figueiredo, Rúben Borralho, Tiago Leitão, Tomás Duro and Verónica Cortez. Not only for being great friends, but also for teaching me so much with their life experiences and advices. I hope to never lose touch with any of you.

Abstract

Delay Tolerant Networks (DTN) are characterized by a lack of end-to-end connectivity. As such, messages (called bundles) can be stored in buffers for a long time. Network congestion can result in poor delivery rates, as bundles are dropped before having a chance of reaching their destination. Some routing protocols, such as MaxProp and Probabilistic Routing Protocol using History of Encounters and Transitivity (PRoPHET), maintain estimations of delivery probabilities for each destination. In this thesis, a new drop policy called Largest Bundle's Hosts Deliverability (LBHD) is proposed that considers all the hosts that received a replica of the same bundle, and their respective delivery probability as estimated by a routing protocol. LBHD uses this additional information to better manage congestion. Simulation results show that LBHD consistently achieves the best delivery probability when paired with PRoPHET and compared with other drop policies proposed in the literature. Also, when paired with MaxProp, LBHD shows the most efficient performance among all the other state of the art policies considering performance metrics such as average delay, overhead ratio and bundle delivery rate. In addition, another drop policy called One Hop Delivery Estimation Drop (OHDED) is proposed. OHDED takes advantage of the encounter predictions of every node in the network stored in every node when using MaxProp. By accurately predicting the bundles that have the highest probability of being delivered directly or in two hops, the results show the best results in delivery rate and overhead ratio in high congestion scenarios.

Keywords

Delay Tolerant Networks, Drop Policies, LBHD, OHDED, Routing Protocols, Scheduling Policies.

Resumo

As Redes Tolerantes a Atrasos caracterizam-se por falta de conetividade ponto-a-ponto. As mensagens agrupadas (*bundles*) podem ser armazenadas por um longo período de tempo. O congestionamento da rede pode resultar em baixas taxas de entrega por os pacotes serem descartados antes de chegarem ao destino. Alguns protocolos de encaminhamento, como o *MaxProp* e o *Probabilistic Routing Protocol using History of Encounters and Transitivity* (*PRoPHET*), conservam estimativas de probabilidades de entrega por destino. Na presente tese, é proposta uma nova política de descarte denominada *Largest Bundle's Hosts Deliverability* (*LBHD*), considerando nós que receberam uma réplica do mesmo *bundle* e a respetiva probabilidade de entrega calculada por um protocolo de encaminhamento. Demostra-se útil para gerir eficientemente o congestionamento. Os resultados mostram consistentemente que a *LBHD* consegue o melhor rácio de entrega quando emparelhada com o *PRoPHET*, comparativamente com outras políticas de descarte propostas na literatura. Analogamente, quando emparelhada com o *MaxProp*, a *LBHD* mostra o desempenho mais eficiente entre a restantes políticas, relativamente a métricas de desempenho como atraso médio, rácio de *overhead* e taxa de entrega. Adicionalmente, propõe-se outra política de descarte designada *One Hop Delivery Estimation Drop* (*OHDED*). A política *OHDED* tira partido da previsão de encontros entre nós na rede, armazenada em cada nó pelo *MaxProp*. Prevendo com precisão os pacotes com maior probabilidade de entrega, diretamente ou em dois *hops*, os resultados mostram o melhor desempenho de todos, em termos de taxa de entrega e rácio de *overhead*, quando implementada em cenários de elevada congestão.

Palavras-Chave

LBHD, *OHDED,* Políticas de Agendamento, Políticas de Descarte, Redes Tolerantes a Atrasos, Protocolos de Encaminhamento.

Table of Contents

Acknowledgements ..... **v**

Abstract ..... **vii**

Resumo ..... **viii**

Table of Contents ..... **ix**

List of Figures ..... **xiv**

List of Tables ..... **xvi**

List of Acronyms ..... **xviii**

List of Software ..... **xx**

Chapter 1 Introduction **1**

1.1 Overview and Motivation 2

1.2 Objectives 3

1.3 Document's Structure 4

Chapter 2 State of the Art **5**

2.1 Fundamental Aspects 6

2.1.1 Delay Tolerant Networks basic concepts 6

2.1.2 Vehicular Delay Tolerant Networks 7

2.1.3 Bundle Protocol 7

2.2 Routing Protocols 9

2.2.1 Single copy 10

Direct Delivery *10*

First Contact *10*

2.2.2 Flooding and flooding based 10

Epidemic *11*

Spray and Wait *12*

2.2.3 Information based flooding 12

PRoPHET *12*

RAPID *13*

MaxProp *13*

2.3 Scheduling Policies 16

2.3.1 Basic Concepts 16

2.3.2 Elementary Scheduling Policies 16

Random *17*

FIFO *17*

RL *17*

SSmall *17*

2.3.3 Information Based Scheduling Policies 17

RC *18*

2.4 Drop Policies 20

2.4.1 Elementary Drop Policies 20

Random *20*

Drop Head *20*

Drop Last *20*

Drop Oldest *20*

Drop Youngest *21*

Drop-largest *21*

Drop Most Hops *21*

2.4.2 Information Based Drop Policies 21

RC-Drop *21*

MOFO *21*

N-Drop *22*

T-Drop *22*

E-Drop *22*

MDC *22*

Mean Drop *23*

2.4.3 Probabilistic Drop Policies 23
MOPR *23*
LEPR *23*
Small-Copies drop *23*
2.5 Congestion Control Mechanisms 27
2.5.1 Global knowledge 27
ANFER *27*
2.5.2 Neighbourhood and local knowledge 28
MACRE *28*
Congestion Avoidance Based on Buffer Space Advertisement *28*
Autonomous Congestion Control *29*
2.6 Tools and Simulator 30
2.6.1 The ONE 31
2.6.2 The ONE Simulation Scenarios 32
2.6.2.1 Node Capabilities *32*
2.6.2.2 Mobility Models *32*
2.6.2.3 Simulation Scenarios Parameters *33*
2.6.3 Performance Metrics 33
Chapter 3 Drop Policies Comparison 35
3.1 Elementary drop policies comparison 36
3.1.1 Simulation scenario 36
3.1.2 Simulation Results 38
3.1.2.1 Bundle Delivery Rate *38*
3.1.2.2 Average Delay *40*
3.1.2.3 Overhead Ratio *42*
3.1.3 Simulation conclusions 43
3.2 Comparison between Small-Copies and elementary drop policies 44
3.2.1 Simulation scenario 44
3.2.2 Simulation Results 44
3.2.2.1 Bundle Delivery Rate *44*

3.2.2.2 Average Delay ... *46*
3.2.2.3 Overhead ratio ... *47*
3.2.3 Simulation conclusions ... 48
Chapter 4 Largest Bundle's Host Deliverability ... 49
4.1 Major Points of Improvement ... 50
4.1.1 Only the nodes which receive a bundle should count ... 50
4.1.2 No Transitivity in MOPR's FP value ... 50
4.1.3 No update between hosts of the same bundle ... 50
4.2 LBHD – Algorithm Description ... 51
4.2.1 Only the nodes which receive a bundle should count ... 51
4.2.2 No Transitivity in MOPR's FP value ... 52
4.2.3 No update between hosts of the same bundle ... 52
4.3 LBHD and PRoPHET Simulation ... 52
4.3.1 Simulation scenario ... 52
4.3.2 Simulation Results ... 54
4.3.2.1 Bundle Delivery Rate ... *54*
4.3.2.2 Average Delay ... *56*
4.3.2.3 Overhead Ratio ... *58*
4.3.3 Simulation conclusions ... 59
4.4 One Hop Delivery Estimation Drop ... 59
4.5 LBHD and OHDED paired with MaxProp ... 61
4.5.1 LBHD Algorithm adaptation ... 61
4.5.2 Simulation scenario ... 61
4.5.3 Results ... 63
4.5.3.1 Bundle Delivery Rate ... *63*
4.5.3.2 Average Delay ... *64*
4.5.3.3 Overhead Ratio ... *66*
4.5.4 Simulation conclusions ... 68
Chapter 5 Conclusions ... 69

References 73

List of Figures

Figure 1 – City VDTN model (extracted from [3]). 2

Figure 2 – IoT: Number of connected devices worldwide from 2012 to 2020 (in billions) (extracted from [4]). 3

Figure 3 – Applications of DTNs. 6

Figure 4 – DTN network architecture layers (extracted from [15]). 8

Figure 5 – Routing in MANETs and VDTNs (adapted from [9]). 9

Figure 6 – Flooding multiple copy based routing. 11

Figure 7 – Illustration of a queuing, scheduling and dropping processes in a node (extracted from [25]). 16

Figure 8 – Illustration of the FIFO scheduling policy, the first incoming bundles are the first to be forwarded. 17

Figure 9 – Bundle replication illustration on a contact opportunity between nodes (*i* and *j*) for routing algorithms of the flooding type. 24

Figure 10 – Congestion Control based on buffer advertised availability scheme (extracted from [38]). 29

Figure 11 – Overview of the ONE simulation environment (extracted from [40]). 32

Figure 12 – Screenshot of the ONE simulator's GUI. 34

Figure 13 – Delivery probability for the elementary drop policies. 40

Figure 14 – Average delay for the elementary drop policies. 41

Figure 15 – Overhead ratio for the elementary drop policies. 43

Figure 16 – Bundle delivery rate comparison between SCop and the best performing elementary drop policies. 45

Figure 17 – Average delay comparison between SCop and the best performing elementary drop policies. 47

Figure 18 – Overhead ratio comparison between SCop and the best performing elementary drop policies. 48

Figure 19 – LBHD's HD metric values exchanged between nodes. 52

Figure 20 – Bundle delivery rate comparison, varying the TTL of the message, between LBHD and other drop policies when paired with the PRoPHET routing protocol. 56

Figure 21 – Average delay comparison in seconds, varying the TTL of the message, between LBHD and other drop policies when paired with the PRoPHET routing protocol. 57

Figure 22 – Overhead ratio, varying the TTL of the message, between LBHD and other drop policies when paired with PRoPHET routing protocol. 59

Figure 23 – Considered situations of a bundle delivery in OHDED drop policy algorithm: (a) direct delivery, 1 hop; (b) one intermediate node, 2 hops. 60

Figure 24 – Bundle delivery rate, varying the TTL of the message, between LBHD and other drop policies when paired with MaxProp routing protocol. 64

Figure 25 – Average delay in seconds, varying the TTL of the message, between LBHD and other drop policies when paired with MaxProp routing protocol. 66

Figure 26 – Overhead ratio, varying the TTL of the message, between LBHD and other drop policies when paired with MaxProp routing protocol. 67

List of Tables

Table 1 – Routing protocols brief table. 15

Table 2 – Scheduling policies brief table. 19

Table 3 – Drop policies brief table. 26

Table 4 – Elementary Drop Policies Simulation Scenario: General Configurations. 38

Table 5 – Elementary Drop Policies Simulation Scenario: Node Configurations. 38

Table 6 – Bundle delivery rate for elementary drop policies using Spray and Wait binary. 39

Table 7 – Bundle delivery rate 95% confidence intervals for elementary drop policies using Spray and Wait binary. 39

Table 8 – Average delay for elementary drop policies using Spray and Wait binary. 41

Table 9 – Average delay 95% confidence intervals for elementary drop policies using Spray and Wait binary. 41

Table 10 – Overhead ratio for the elementary Drop and Scheduling policies. 42

Table 11 – Overhead ratio 95% confidence intervals for the elementary Drop and Scheduling policies. 42

Table 12 – Bundle delivery rate for the best performing elementary drop policies and SCop. 45

Table 13 – Bundle delivery rate 95% confidence intervals for the best performing elementary drop policies and SCop. 45

Table 14 – Average delay for the best performing elementary drop policies and SCop. 46

Table 15 – Average delay 95% confidence intervals for the best performing elementary drop policies and SCop. 46

Table 16 – Overhead ratio for the best performing elementary drop policies and SCop. 47

Table 17 – Overhead ratio 95% confidence intervals for the best performing elementary drop policies and SCop. 48

Table 18 – LBHD and PRoPHET simulated scenario general configurations. 53

Table 19 – LBHD and PRoPHET simulated scenario node configurations. 53

Table 20 – Bundle Delivery Rate comparison between LBHD and the best performing state of the art policies when paired with the PRoPHET routing protocol. 55

Table 21 – Bundle Delivery Rate 95% confidence intervals for the comparison between LBHD and the best performing state of the art policies when paired with the PRoPHET routing protocol. 55

Table 22 – Average Delay comparison between LBHD and the best performing state of the art policies when paired with the PRoPHET routing protocol. 56

Table 23 –Average Delay 95% confidence intervals for the comparison between LBHD and the best performing state of the art policies when paired with the PRoPHET routing protocol. 57

Table 24 – Overhead ratio comparison between LBHD and the best performing state of the art policies when paired with the PRoPHET routing protocol. 58

Table 25 – Overhead ratio 95% confidence intervals for the comparison between LBHD and best performing state of the art policies when paired with the PRoPHET routing protocol. 58

Table 26 – LBHD/OHDED and MaxProp simulated scenario general configurations. 62

Table 27 – LBHD/OHDED and MaxProp simulated scenario node configurations. 62

Table 28 – Bundle delivery rate comparison between LBHD and other drop policies compatible with MaxProp routing protocol. 63

Table 29 – Bundle delivery rate 95% confidence intervals for the comparison between LBHD and other drop policies compatible with MaxProp routing protocol. 64

Table 30 – Average delay comparison, in seconds, between LBHD and other drop policies compatible with MaxProp routing protocol. 65

Table 31 – Average delay 95% confidence intervals for the comparison, in seconds, between LBHD and other drop policies compatible with MaxProp routing protocol. 65

Table 32 – Overhead ratio comparison between LBHD and other drop policies compatible with MaxProp routing protocol. 67

Table 33 – Overhead ratio 95% confidence intervals for the comparison between LBHD and other drop policies compatible with MaxProp routing protocol. 67

List of Acronyms

ANFER	Average Forwarding Number based on Epidemic Router
BDP	Bundle's Delivery Probability
BP	Bundle Protocol
BS	Bundle Size
CRAWDAD	Community Resource for Archiving Wireless Data At Dartmouth
CS	Congestion State
DD	Direct Delivery
DO	Drop Oldest
DTN	Delay Tolerant Networks
DY	Drop Youngest
E-Drop	Equal Drop
EID	End Point Identifier
FC	First Contact
FIFO	First In First Out
GUI	Graphic User Interface
ICMP	Internet Control Message Protocol
IoT	Internet of Things
IPN	Inter Planetary Networks
MACRE	Message Admission Control based on Rate Estimation
MANET	Mobile Ad hoc Network
MBM	Random Map-Based Movement
MDC	Message Drop Control
MOFO	Evict Most Forwarded First
MOPR	Evict Most Favourably Forwarded First
NF	Number of Forwards
OHDED	One Hop Delivery Estimation Drop

ONE	Opportunistic Network Environment
PRoPHET	Protocol using History of Encounters and Transitivity
RAPID	Resource Allocation Protocol for Intentional DTN
RC	Replicated Copies
RCAsc	Replicated Copies Ascending
RCDesc	Replicated Copies Descending
RC-Drop	Replicated Copies Drop
RC-DropAsc	Replicated Copies Drop Ascending
RC-DropDesc	Replicated Copies Drop Descending
RFC	Request for Comments
RL	Remaining Lifetime
RMBM	Routed Map-Based Movement
RTTL	Residual/Remaining TTL
SCop	Small Copies drop
SSmall	Schedule Small
SPMBM	Shortest Path Map-Based Movement
T-Drop	Threshold Drop
TCP	Transmission Control Protocol
TTL	Time To Live
UB	Upper Bound
VANET	Vehicular Ad hoc Network
VDTN	Vehicular Delay Tolerant Networks
VN	Vehicular Network
V2V	Vehicle-to-Vehicle
WDM	Working Day Movement

List of Software

Draw.io	Online images, schemes and flowchart development
Eclipse Neon 4.6.0	Java Integrated Development Environment
FileZilla	File Transfer Protocol client application
GitHub	Version control repository and Internet hosting service
Mendeley Desktop	Reference management tool
Microsoft Excel 2016	Spreadsheet and graph application
Microsoft Word 2016	Word processor
The ONE Simulator	Delay Tolerant Networks simulator

Chapter 1

Introduction

In this chapter the motivation for the thesis, its main goals and structure explanation is given. Also an overview on DTNs fundamental aspects and its evolution is provided.

1.1 Overview and Motivation

In the 1970s, researchers developed routing technologies for non-fixed computer locations. The development of the ad-hoc routing in the 1980s and the broader use of wireless protocols brought an interest on DTNs (Delay Tolerant Networks) [1].

The DTN appeared as way to provide inter-planetary connections for space communications. The development of vehicular networks, networks for emergency response, military operations, tracking and monitoring applications [2] raised the question of how to optimize the already proposed strategies for DTNs, networks with unpredictable frequent disconnections and possibly with high mobility.

Recently, there is a growing interest in Vehicular DTNs (VDTN). Companies such as Veniam[1] see a future where car communications are more automatic than ever. A future where cars must share location, speeds, road quality, weather conditions, road accidents, among others with the network and the cloud. Not only to improve our transportation methods but also for smart cities applications, which usually uses data that may not be sent in real time. As exemplified in Figure 1, VDTNs may be composed by trucks, cars, buses and trams which connect in a mesh providing the devices access to the network. Veniam has already established VDTNs in cities such as Oporto, New York and Singapore [3].

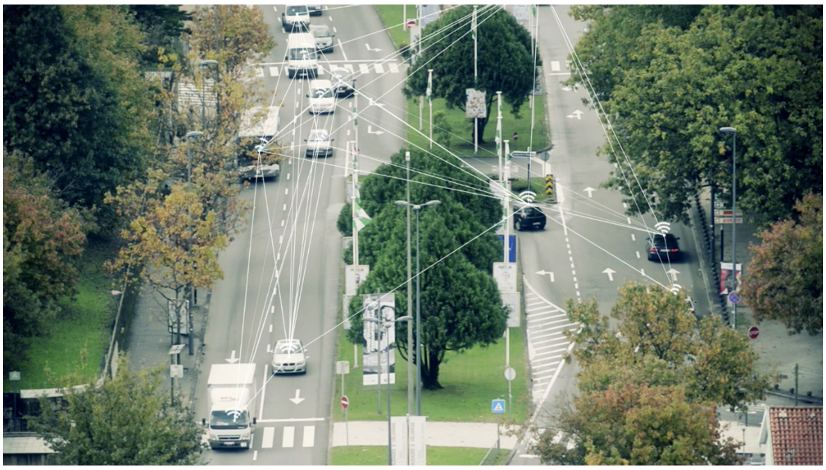


Figure 1 – City VDTN model (extracted from [3]).

With the broader utilization of the DTN architecture in mobile networks with intermittent connectivity, the need of passing information through store and carry methodologies arises. Since the major routing protocols and congestion mechanisms available were developed for continuous connectivity, the necessity of creating compatible protocols for DTNs became a reality.

[1] https://veniam.com

Since the connections are frequently disrupted, and the networks may have high mobility, the development of mechanisms that would properly manage the buffer space and schedule the packets to be forwarded and dropped became a challenge.

As visible in Figure 2, the number of connected devices has been augmenting in the past years and is expected to reach 50.1 billion devices in 2020. This is due to the Internet of Things (IoT) phenomenon, in which every person possesses multiple devices which are connected to the internet. These devices, could range from a smart plug to a toaster, fridge, or even agriculture applications such as water sensors for large corn fields. The data that these devices share with the network does not have to be seen in real time hence, the DTN architecture may provide a solution for transporting the sensor data.

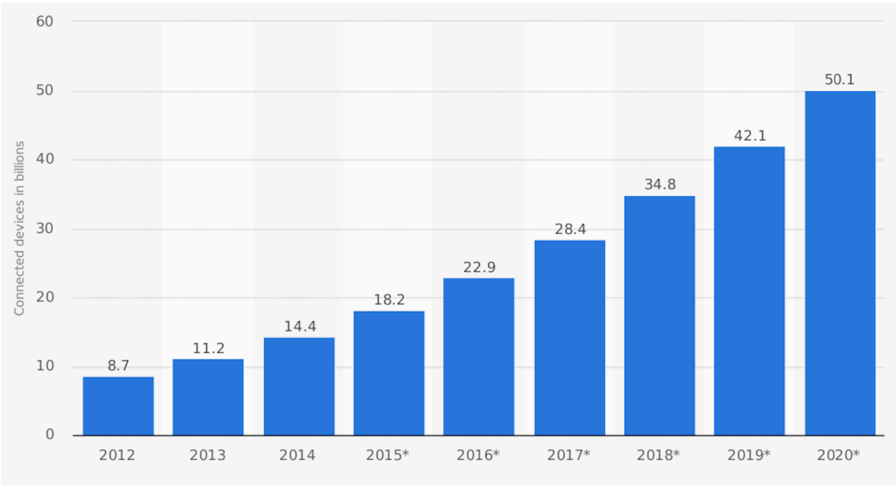


Figure 2 – IoT: Number of connected devices worldwide from 2012 to 2020 (in billions) (extracted from [4]).

1.2 Objectives

In this section the several objectives of the present work will be listed. As already mentioned in the previous section, DTNs often have congestion problems due to their store and carry routing strategies. When the network is congested, the nodes' buffers are usually full, and must be able to choose which message to discard in order to store a new one. The algorithms that decide which bundle to drop are called drop policies. The algorithms that decide the order in which the stored messages will be forwarded are called scheduling policies.

The main objective of the present work is to propose two new drop policies with better performance than the other state of the art policies, when paired with routing protocols with node's delivery probability estimation capabilities. One of the policies is generic, thus can be applied in every routing protocol with delivery probability estimation, and the other is specific for the MaxProp routing protocol.

Also, this work aims to study, analyse and compare the different state of the art drop policies, scheduling policies, routing protocols and congestion control mechanisms, providing a small description of the underlying algorithms responsible to increase the delivery rate, decrease network delay and optimize the nodes' buffer space.

Moreover, it is also important to simulate the dropping and scheduling policies paired with different routing protocols, in order to understand which policies have the overall better performance.

1.3 Document's Structure

The present work is divided in five chapters:

- Chapter 1 – Introduction;
- Chapter 2 – State of the Art;
- Chapter 3 – Drop Policies Comparison;
- Chapter 4 – Largest Bundle's Host Deliverability;
- Chapter 5 – Conclusions.

In the present chapter, an introduction to the DTN's history and main applications is done. Also, the main goals of the work, and the structure of the document are also presented.

The second chapter provides an overview on DTNs and its fundamental concepts, a detailed enumeration and explanation of the most relevant drop policies, scheduling policies, routing protocols and congestion control mechanisms. Additionally, in the second chapter, an overview and main features of the used tools is given.

The third chapter, provides a performance analysis of the elementary drop and scheduling policies already presented in the second chapter. These policies are simulated in the same scenario and the results are analysed in order to conclude which characteristic is the most important when evaluating which message should be dropped first.

In the fourth chapter, the proposed drop policy, named Largest Bundle's Hosts Deliverability (LBHD) is explained in detail. In addition, various simulations are made comparing the proposed drop policy with several drop policies using different routing protocols as well as the results' discussion of those simulations. Furthermore, a paper describing LBHD was accepted for publication in [5]:

> LBHD: Drop Policy for Routing Protocols with Delivery Probability Estimation.
> Miguel Pinheiro Rodrigues, Naércio Magaia, Paulo Rogério Pereira.
> 12th Iberian Conference on Information Systems and Technologies, (CISTI'2017),
> 21-24 de Junho de 2017, Lisboa, Portugal.

Also, it is proposed in Chapter 4 the One Hop Delivery Estimation Drop (OHDED) drop policy, developed to be paired with MaxProp.

Eventually, in the fifth and final chapter, an analysis of the complete work is made, the conclusions are drawn, and the future work which is thought to improve the scientific research in this area is proposed.

Chapter 2

State of the Art

This chapter firstly provides a background on DTN fundamental concepts, main areas of application and the main challenges for the scientific community. Then, it is provided an analysis of all the drop, scheduling and routing polices used in the development and results. Also, a small overview on the main congestion control mechanisms and used tools is given.

2.1 Fundamental Aspects

In this section, an introduction on DTNs' and VDTNs' fundamental concepts is provided, including their major differences. The Bundle protocol and its relation to the DTN architecture is also explained.

2.1.1 Delay Tolerant Networks basic concepts

As several articles state ([6]–[8]), DTNs were historically originated as a proposal for Inter Planetary Networks (IPN) to provide communications between satellites, planets and base stations, this was a scenario in which high delays and frequent disconnections were very common, so the network should be able to take all of this into account, and use proper mechanisms to optimally manage this type of networks.

DTN is a very broad concept, and it is important to distinguish between the various types, depending on the specifications of each DTN. Cao in [9], as illustrated in Figure 3, distinguishes the many DTN applications into two different segments:

- Space Application, more specifically IPNs;
- Terrestrial Application:
    - UWNs – Under Water Networks;
    - PSNs – Pocket Switched Networks;
    - VANETs – Vehicular Ad hoc NETworks;
    - ANs – Airborne Networks;
    - Suburb Networks for developing region.

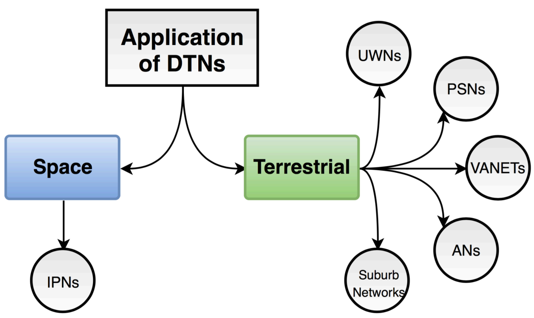


Figure 3 – Applications of DTNs.

From the distinction made in [9], and considering the one provided in [6], this distinction was made taking into account that IPN networks have connectivity patterns – easily known through predictable planet/satellite movement – contrarily to the others that exhibit a more chaotic behaviour due to the highly unpredictable mobility of each node.

In [6], the authors distinguish the problem in DTNs routing from the one in standard dynamic routing by listing a few major differences. For instance, not having end-to-end connectivity – in dynamic routing it is assumed that the topology is always connected – which causes problems in DTNs, such as flooding, drops, delays and others.

According to [7], another main difference is that in DTNs, when a message cannot be routed to its destination, it is not immediately discarded but stored and carried until a route is available. The drop of a packet, in this kind of networks, is always related to buffer management reasons or Time To Live (TTL) expiring.

2.1.2 Vehicular Delay Tolerant Networks

As proposed by [10], a Mobile Ad hoc Network (MANET) is a collection of mobile nodes that are dynamically and arbitrarily located in such a manner that the interconnections between nodes are capable of changing on a continual basis. Also, according to [11], this kind of wireless networks do not require any fixed infrastructure being preferred for applications where communication is required, although the deployment of a fixed infrastructure is impossible.

Vehicular Ad hoc Networks (VANETs) should not be confused with MANETs, although both are mobile, VANET nodes change more rapidly, and are not always connected since the receiver may not be reachable from the sender [7].

One of the main VANET application scenarios is the automobile industry [7]. Many wireless solutions continue to be developed with the main objective of improving safety through Vehicle-to-Vehicle (V2V) communication for collision avoidance and consequently also improving route planning and better control of traffic congestion. VANETs are often characterized by [11]:

- Trajectory-based movements;
- Varying number of vehicles with independent or correlated speeds;
- Fast time-varying channel (e.g. signal transmissions can be blocked by buildings);
- Lane-constrained mobility patterns (e.g., frequent topology partitioning due to high mobility).

In most of the literature about Vehicular Networks (VNs), a clear distinction between VDTNs and VANETs is not made. Benamar, in [12] provides a clear distinction between the two concepts, referring to VDTNs as a special type of VANETs, where the vehicular network is sparse, and direct paths between the sender and the receiver of the packet are often not available.

2.1.3 Bundle Protocol

The Bundle Protocol (BP) [13] was created to deal with the different characteristics of DTNs, since the traditional Internet protocols are not optimal for DTNs due to its frequent disconnections and high delays. As illustrated in Figure 4, the BP works on the application layer. In this way, it is possible to run the BP over the current Internet protocols as well as over the more specific protocols like the ones used in spacecraft. This allows for a different transport protocol when needed, since that in certain parts of the bundle path, Transmission Control Protocol (TCP) [14] and other mainstream protocols may not be optimal, as it is illustrated in Figure 4.

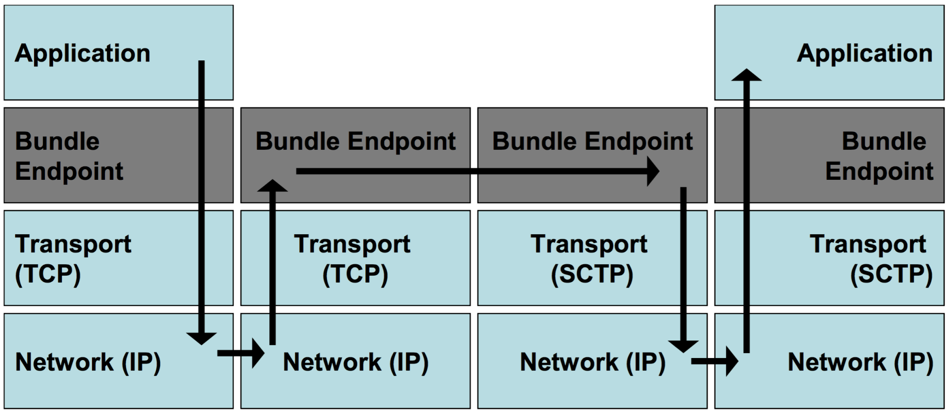


Figure 4 – DTN network architecture layers (extracted from [15]).

A Bundle is defined in this application as a stack of packets where each bundle is able to provide the application with the information needed for a transaction, since a single packet would not often provide sufficient information for the application to work properly [12].

Huang in [16] says that the bundle layer can store, carry and forward the entire bundles or bundle fragments. This retransmission mechanism works, as pointed by Farrell in [15], as a custody-based retransmission, and it is marked as one of the most important concepts to discuss regarding DTN routing. In custody-based retransmission, a bundle is only dropped – deleted from the node buffer – when there has been a successful transmission to another node that has taken custody of the bundle, and so assuming responsibility for the bundle delivery. Farrell in [15] introduces the term “custodian node” as the node that stores the bundles and carries them until a transmission opportunity arises, assuming the commitment to deliver the bundle to its destination node.

Silva in [8] explains that since there is usually no end-to-end connectivity, the final destinations of the bundles are endpoints associated with nodes and identified by End Point Identifiers (EIDs). Thus, the bundles are forwarded in a series of contacts – communication links that are established whenever nodes come in range of one another [8] – until they reach their destination node.

In 2007, the Requests for Comments (RFCs) in [13], [17] were created as a way to deliver a framework for the development of DTN applications. As referred in the RFC in [13], the key capabilities of BP include [9]:

- Custody-based retransmission;
- Ability to cope with intermittent connectivity;
- Ability to take advantage of scheduled, predicted, and opportunistic connectivity (in addition to continuous connectivity);
- Late binding of overlay network endpoint identifiers to constituent Internet addresses.

2.2 Routing Protocols

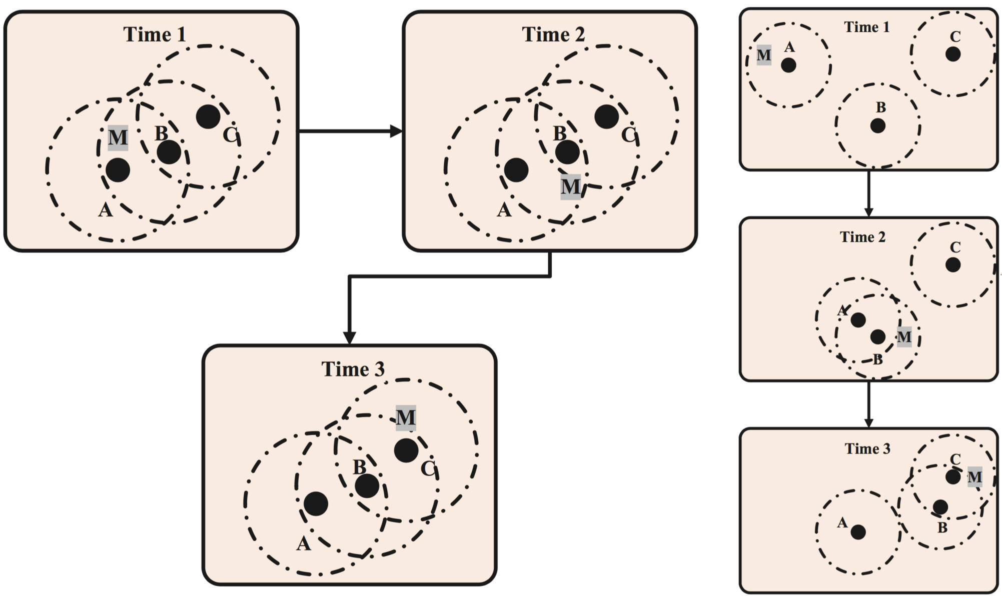
In this section, a detailed explanation of the most important routing protocols for DTNs will be provided. These will be divided in different sections, for each type of routing algorithm. At the end a summary table, which resumes this section, will be presented.

The routing paradigm in DTNs and MANETs is different since on the latter there is usually end-to-end connectivity and so the typical relay of the packets is a very effective approach, such does not apply in VDTNs since as mentioned before, the end-to-end connectivity is often not available and so the routing protocols must exploit the nodes mobility as pointed by Cao in [9].

In Figure 5 (a), a message is relayed from node *A* to node *B* and at last to node *C*, taking advantage of the end-to-end connection in MANETs. In Figure 5 (b), node *A* (Time 1) carries the message, until a connection opportunity arises (Time 2) relaying the carried message to node *B*, finally node *B* delivers the message to node *C* when both nodes are connected. Benamar in [12], also differentiates MANETs and VDTNs similarly, noting that the former aims to obtain an end-to-end connectivity and does not work as the latter in a carry and forward paradigm.

Many types of Routing Protocols are available, in this section a description of the various types and the most important examples is provided.

As previously mentioned, DTNs may have very different characteristics, thus routing protocols were developed that take advantage of the network topology's predictability and connectivity. One possible application scenario of these protocols would be for instance IPNs [6].



(a) – Routing in MANETs.　　(b) – Routing in VDTNs.

Figure 5 – Routing in MANETs and VDTNs (adapted from [9]).

Benamar in [12] explains that one of the main characteristics that allows the distinction between algorithms is if the algorithm is of the single copy type, or the replicative type – multiple copies. The main difference between these two types is that the single copy type never replicates a bundle, forwarding it until it reaches its destination, and the multiple copies type may replicate the messages in the right conditions, depending on the algorithm, having more than one copy of the original bundle. Although the latter may have more bundles copies in the network, increasing the delivery probability, this type of algorithms often come at a cost, usually flooding the network. In the following topics, a more in depth explanation will follow.

Random forwarding algorithms, also defined in [12] as stochastic DTN routing algorithms, are often applied in VDTNs scenarios where the position unpredictability of the nodes is substantial. The delivery is made hop-by-hop until the bundle reaches its intended destination. Since the contacts are unpredictable, there cannot be guarantees of delivery.

2.2.1 Single copy

In the present section, the most relevant single copy algorithms will be presented. In this type of algorithms, there is only one replica of the bundle in the whole network.

Direct Delivery

As explained in [18], Direct Delivery (DD) relies on a quite simple strategy, by having the source holding the data it has to deliver until a connection opportunity with the destination is available. DD reduces the number of hops to only one, reducing the congestion and also the number of messages carried by each node. Hence, this algorithm is classified as single message. A drawback is that if the source does not meet the destination, then the message is not delivered, which means that the delivery latency or delivery rate might not be good.

First Contact

Essentially, First Contact uses Random forwarding. A node carrying a bundle randomly chooses one of the neighbour nodes to send each bundle in its buffer to [6]. In this algorithm, no information about the network is necessary and so each intermediate node, at every contact opportunity, forwards the bundle randomly to one of the existing connections it may have, thus maintaining only one copy of the message in the network, and so it can be classified as a single copy algorithm.

The main disadvantage of this algorithm is its random character. By choosing the nodes randomly, the chosen node may not deliver the bundles to their destination, causing more drops and lower delivery ratios than if some criteria were used in the choosing process.

2.2.2 Flooding and flooding based

In the multiple copy routing protocols, there may be more than one copy of a certain bundle in the network. This results in the increase of the bundle delivery probability in networks as DTNs.

Flooding and flooding based algorithms also classify as multiple copy. In these strategies, as illustrated in Figure 6, from the source node, the bundles are replicated at every contact opportunity until one of the copies reaches the destination node [12].

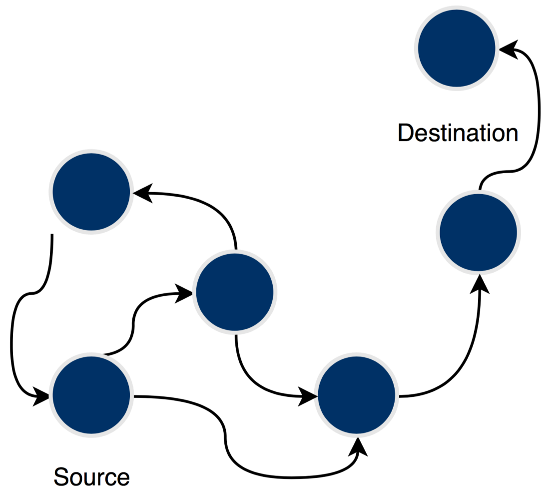


Figure 6 – Flooding multiple copy based routing.

In flooding algorithms, the delivery ratio is usually superior to the one in single copy algorithms as seen by Sukhbir in [19]. For these higher ratios to be obtained, a network must have certain characteristics:

- Increased storage capabilities;
- Increased bandwidth.

Some of the disadvantages, besides the increased capabilities requirements, are that the algorithms may cause congestion and often are not easily scalable.

Two common examples of flooding protocols are the Epidemic [12] and the Spray and Wait [20] protocols. Epidemic generates an unlimited number of copies of each bundle, while Spray and Wait limits the number of bundle replicas to a configurable parameter.

Epidemic

Besides being from the flooding category, Epidemic also classifies as a multiple copies algorithm [12]. This algorithm is especially useful in situations where there is not a constant end-to-end connection, the network is very sparse and there is usually not much information about the network. The Epidemic protocol does not require any information [18]. At every contact opportunity nodes exchange the bundles each one is missing from the other [12]. By wasting the network's storage and bandwidth, in a scenario with an infinite number of contact opportunities and buffer space, it achieves minimum delivery delay and maximum delivery ratio. In this algorithm, by dropping the older messages, the delivery ratio for low contact rate nodes declines. This is the main disadvantage of the Epidemic Protocol.

Spray and Wait

As explained by the authors in [20], the main aim of the Spray and Wait protocol is to:

- Perform fewer transfers than the Epidemic and other flooding based routing algorithms;
- Achieve close to optimal delivery delay;
- To be highly scalable i.e. maintain the performance despite a change in the network size;
- Require little knowledge about the network.

The algorithm consists in two phases [20]. Being Spray the first one, in this phase for every message originating at a source node, L message copies are initially spread – forwarded by the source and possibly other nodes receiving a copy – to L distinct "relays".

On the Wait phase, each of the L nodes carrying a message copy performs DD.

Beyond other variations, one of the most common is the Binary Spray and Wait, which consists on: every node which has more than one message hands over half of the message copies it possesses. Once a contact opportunity occurs, the node transfers half of the copies to the other node, keeping the other half to itself. This process is repeated until the number of copies on the node's buffer is unitary, at that time the node must perform DD.

Spray and Wait combines the speed of epidemic routing with the simplicity and thriftiness of direct transmission.

2.2.3 Information based flooding

In the flooding strategy, messages are replicated to enough nodes so that destination node must receive them, while in the information based flooding strategy, information about the network is used to select the best path to the destination. This type of protocols selects the best path to the destination using information from the network learned locally, estimating the probability of delivery of a bundle.

PRoPHET

By not eliminating message replications that would not improve the delivery ratio in the network, the Epidemic protocol is very resource hungry, since that in a real scenario the nodes encounters are not purely random. By taking advantage of this predictability, it is possible to improve the delivery ratio.

As explained in [12], one of the main routing protocols of this type is the Probabilistic Routing Protocol using History of Encounters and Transitivity (PRoPHET) protocol [21]. In this protocol, each of the nodes contains a vector with the delivery predictability of the other nodes of the network. The delivery predictability is calculated using the information provided from the past encounters [18].

In the PRoPHET protocol, the bundle carrying node only transfers the bundle to nodes with higher delivery probability than itself, managing to improve the redundancy and the delivery ratio, since as explained above, this is done using information of past encounters [12].

Sukhbir in [19] explains that since the PRoPHET mainly uses past information, in the initial state where there is an uniform probability distribution, the performance is at its minimum. PRoPHET++ was created with the intention of improving the main PRoPHET disadvantage.

Patel in [18] introduces the PRoPHET++ protocol. As a way to solve the problem listed above, the PRoPHET++ protocol, in the early stages behaves as the Epidemic protocol, as a way to initially disseminate the messages, preventing message dropping of the initial messages.

RAPID

As mentioned by Balasubramanian in [22], the Resource Allocation Protocol for Intentional DTN (RAPID) routing protocol is specifically made to work in DTNs environments where the bandwidth is limited and the contact opportunities are very short timed.

This algorithm works in four distinct phases, being the first one the Initialization, in which a node *Y* transfers to its neighbour node *X*, data about the stored bundles in its buffer:

- Average size of the past transfer opportunities;
- Expected meeting times with nodes;
- List of already delivered bundles;
- The delay estimate for each individual stored bundle;
- Information about other bundles.

Secondly, there is the Direct Delivery phase in which the packets are delivered prioritizing first the bundles with the biggest utility. The utility is measured using the following equation

$$U_i = \begin{cases} -D(i), & D(i) \geq D(j) \ \forall j \in S \\ 0, & otherwise \end{cases}. \tag{1}$$

In which *i* and *j* represent the bundles, *S* the total set of the bundles and *D* the expected delivery delay, estimated by RAPID, of a certain bundle. The expected delivery delay corresponds to the minimum expected time that any node, carrying a bundle replica, takes to deliver the bundle to its destination. Hence, all nodes need to know, for each bundle in their buffer, which other nodes are holding replicas, and when they expect each bundle replicas to be delivered [22]. Thus, in the direct delivery phase the prioritized bundles are those with minor expected delivery delays.

In the third phase, Replication, the marginal utility, i.e. the increase of the utility of the bundle, of the uncommon bundles between *X* and *Y* is computed. Afterwards the marginal utility U_i is divided by the size of the bundle s_i, and the bundles are forwarded in decreasing order of this ratio.

In the fourth and final phase, Termination, the *X* node stops the transmission of the bundle only when all the bundles were replicated or when it is out of the transmission range of the other node *Y*.

MaxProp

One of the proposed features of the MaxProp protocol is that once a bundle is delivered to its destination, the messages in the network are cleared [12]. By using this proper buffer management approach, MaxProp manages to lower the packet drop ratio. Such feature may be implemented in other routing protocols.

In the MaxProp routing protocol [23], every node in the network has a table of the probabilities of each node delivering a bundle to another. This table is updated at every encounter to maintain the values realistic. This delivery probability table is used to estimate the cost of delivery of a certain bundle by any node in the network. This cost is estimated through Dijkstra's algorithm [24], in order to optimize the probability calculation since the algorithm has to approximate the cost of delivery, with small error and without iterating through all the delivery probability table.

This cost is computed calculating the probability of a delivery for a certain location. It is also able to send messages in specific order that considers message hop counts and delivery probabilities based on previous encounters.

This protocol does not require the knowledge of [12]:

- Network connectivity (present nor future);
- Node's location (present nor future);
- Stationary relay nodes.

The main disadvantages of the protocol are:

- Decreased time for message exchange: By exchanging the tables, the contact time used to transfer the requested messages is shorter;
- Not suited for sparse networks: It does not provide a connected graph.

In Table 1 is presented the summary of the present section, which classifies, summarizes and presents each routing protocol major characteristics, advantages and disadvantages.

Table 1 – Routing protocols brief table.

Routing Protocols	Main Routing Protocols Categories				Main metric	Summary	Advantages	Disadvantages
	Single Copy (SC) or Multiple copies (MC)	Random (R)	Pure Flooding (PF) or Flooding Based (FB)	Information Based Flooding (IBF)				
Direct Delivery (DD)	SC	-	-	-	Bundle's destination address	The source node holding the data, only delivers its carried bundle when a connection opportunity with the destination node is available.	More buffer space available, less bandwidth used and small overhead, when compared with other policies.	Not suited for large sparse networks.
First Contact (FC)	SC	R	-	-	-	Each node forwards the bundles randomly to one of the various connections it may have, maintaining only one copy of each bundle in the network.	Less dropped bundles and smaller buffer space occupancy and overhead, when compared with flooding policies.	May cause large latency values.
Epidemic	MC	R	PF	-	-	At every contact opportunity nodes exchange the bundles each one is missing from the other, spoiling the network's storage and bandwidth.	Increased delivery rate, when compared with non-flooding.	Large count of dropped bundles and overhead. Increased bandwidth usage.
Spray and Wait	MC	R	FB	-	Number of times a bundle has been replicated.	On the Spray phase, *L* bundle copies are initially spread among connections. On the Wait phase, if the destination was not reached, each of the *L* nodes carrying a bundle replica performs DD.	Smaller network congestion and better bundle delivery rate when compared with epidemic.	Large dropped bundles number. No prioritization in bundle order.
PRoPHET	MC	-	FB	IBF	Delivery probability prediction	Each node keeps a vector with the delivery predictability of all the other nodes. The delivery predictability is calculated using the information provided in the past encounters. The node only transfers to nodes with higher delivery probability than itself.	Shared information on the network, which implies better forwarding, in terms of the order of the bundles. Performs better in terms of bundle delivery rate and overhead when compared with Spray and Wait and Epidemic.	Not suited for random movement models. Has a convergence time.
RAPID	MC	-	FB	IBF	Bundle Utility	On initialization phase, a node *Y* transfers to the node *X* data about the stored bundles in *X's* buffer. On the DD phase, the packets are delivered prioritizing first, the bundles with the biggest utility. On the Replication phase, the increased utility is divided by the size of the bundle, and the bundles are forwarded in decreasing order.	Better delivery rate when compared to PRoPHET, with less flooding and good bundle prioritization.	Not suited for random movement models.
MaxProp	MC	-	FB	IBF	Encounter probability prediction	Every node has a table of the probabilities of each node delivering a bundle to another. This table is updated at every encounter to maintain the values realistic. At an encounter, the nodes transfer the bundles ordered by the bundle's deliverability probability of the new node.	Once a bundle is delivered, the information is spread on the network with the intent to drop the delivered bundle's replicas.	Decreased time for bundle exchange. Probability values for isolated nodes may not be accurate.

2.3 Scheduling Policies

In the following sections, some basic concepts about scheduling and dropping policies will be provided. Also, the most relevant scheduling policies for the developed work will be introduced and explained in detail in the present section.

2.3.1 Basic Concepts

Scheduling and dropping policies were created as a way to properly manage the node's buffer. In Figure 7, a representation of both policies is presented.

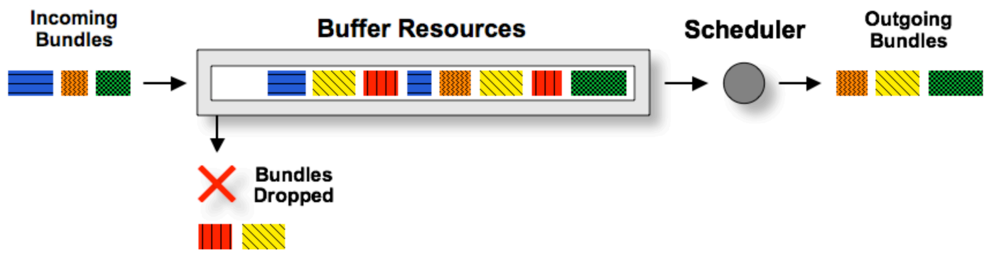


Figure 7 – Illustration of a queuing, scheduling and dropping processes in a node (extracted from [25]).

Drop policies are defined as the way to select the bundles to be deleted from a node's buffer, when it is full and must make room for incoming bundles.

In the scheduling policies, the Scheduler takes the role of choosing the order in which the bundles are forwarded at the next contact opportunity.

The scheduling policies should not be confused with the previously mentioned routing protocols, whose main function is to decide to which node the bundles – as ordered by the scheduler – should be forwarded to.

Also, a brief table summarizing and listing the advantages and disadvantages of the addressed scheduling polices will be provided at the end of the section.

2.3.2 Elementary Scheduling Policies

The use of very complex protocols, in some cases, may not be beneficial since these often require information about the network, making the implementation harder. For small networks that do not have a hefty bundle influx, First In First Out (FIFO) and Random policies are some of the most elementary examples of simple scheduling policies with an acceptable cost-benefit.

One of the main concepts that should be explained is TTL, which is a timeout value: the time a bundle has before it is no longer considered useful, being discarded once this time has passed. This bundle header field was created with the intention of avoiding forwarding bundles no longer needed, which are wasting the network resources and can be deleted.

Random

In the Random policy, as the name would suggest, the forwarded bundle is chosen randomly from the node's buffer. Since this policy does not take any parameter into account, in a congested network the results in terms of delivery and latency are far from optimal, as illustrated by Soares in [25].

FIFO

In FIFO policy, as illustrated in Figure 8 and mentioned by Soares in [25], the first scheduled bundle from the Node is the first incoming one, and so the priority in the bundle queue is such that the first bundle to arrive is the first to be forwarded.

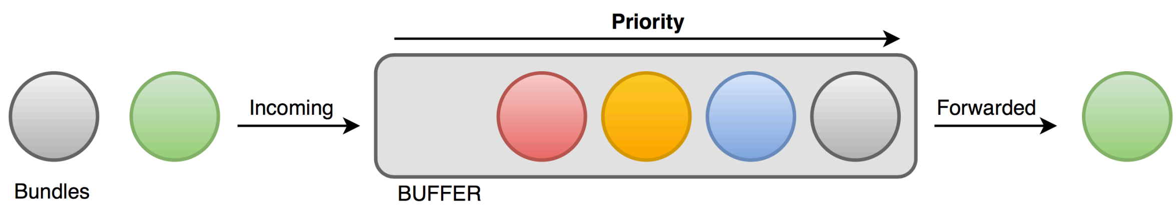


Figure 8 – Illustration of the FIFO scheduling policy, the first incoming bundles are the first to be forwarded.

RL

The Remaining Lifetime (RL) policy, as explained by Soares in [25], relies mainly on the TTL value of each bundle. This scheduling policy orders the bundles based on their Remaining/Residual TTL (RTTL). The main variations of the protocol are RL Descending, in which the ones with larger TTL are prioritized, and RL Ascending, which works on the opposite way, prioritizing the bundles with the smaller TTLs to be sent. When a tie happens, usually a more elementary policy such as FIFO or Random is used.

SSmall

In the Schedule Small (SSmall) scheduling policy, the prioritized bundles are the ones whose size is the smallest. The forwarding queue is ordered in increasing sizes i.e. the smallest sized bundles are forwarded first.

2.3.3 Information Based Scheduling Policies

As a way to optimize large bundle influx and/or frequently disconnected scenarios as it is the case with VDTNs, the creation of policies that take advantage of the information provided in the bundles and in the contacted nodes is an important research problem. One of the main examples of Information Based Scheduling Policies is listed and explained in this section.

RC

If the nodes keep track of the number of times each bundle has been replicated, and pass that information to every node that carries one of the bundles' copies, this information can be used for scheduling.

The Replicated Copies (RC) scheduling policy, as explained by Soares in [25], relies on the number of copies of each of its buffered bundles, using this number to create a priority queue. The two variations are the RC Descending, where the prioritized bundles are the most replicated ones, and the RC Ascending where the prioritized bundles are the least replicated ones. When a tie happens, usually a more elementary policy such as FIFO or Random is used.

This protocol may be used in any routing protocol if the number of copies information is provided in some way. One of the routing protocols that is able to provide such information is Spray and Wait.

In Table 2, it is presented the summary of the present section, which shows the compatibility, summarizes and presents for each scheduling policy, its major advantages and disadvantages.

Table 2 – Scheduling policies brief table.

	Scheduling Policies	Main Parameter of the used metric	Routing Protocols Compatibility	Summary	Advantages	Disadvantages
Elementary	Random	None	All	Schedules a randomly chosen bundle.	Adequate for small networks with low traffic.	Its random nature.
	FIFO	Time of arrival	All	Schedules the first incoming bundle.	Good for networks with low traffic.	Does not take advantage of the routing protocol metric.
	RL					
	Ascending	RTTL	All	Schedules the bundle with the smallest RTTL.	None	Lower delivery rate when compared to random.
	Descending			Schedules the bundle with the largest RTTL.	Schedules the youngest bundles in the network.	Does not take into account BS nor the destination of the node.
	Ssmall	Size	All	Schedules the smallest sized bundle.	In congested networks, achieves the highest average node's available buffer space.	Bad in networks where the bundle size is fixed.
Information Based	RC			Using the stored number of replicas of each bundle:		
	Ascending	Number of copies	Limited: The information on the number of copies of each bundle must be given.	Schedules the bundle with the lowest number of replicas.	Good for congested networks.	Requires the information of the number of replicas in the network.
	Descending			Schedules the bundle with the highest number of replicas.	None	Lower delivery rate when compared to random.

2.4 Drop Policies

As a way of increasing the delivery ratio in VDTNs, the nodes carry as many bundles in the buffer as possible. Consequently, the necessity of proper buffer management appeared. As previously mentioned in 2.3, and also illustrated in Figure 7, the drop policy is responsible to decide which of the bundles should be dropped first, in order to make room, when a fully occupied buffer has an incoming bundle that needs to be stored.

The various types of drop policies may be grouped into different classifications. In this section a description of the main examples is provided.

Moreover, at the end of the present section, a brief table containing a summary, advantages and disadvantages of the addressed drop policies will also be provided.

2.4.1 Elementary Drop Policies

The following policies rely on the information present in the bundle at the time it is stored in the buffer. These are the easiest to implement and are usually reliable, although the drop policies which try to infer information about the network have better results in terms of delivery rate, overhead ratio and average latency, as will be analysed in sections 2.4.2 and 2.4.3.

Random

For an incoming bundle that has to be stored in a full node's buffer, in the Random policy, as the name would suggest, the dropped bundle is chosen randomly among every stored bundle [8].

Drop Head

In the Drop Head policy, as mentioned by Soares in [25], when the node's buffer is full, the first dropped bundle from the Node is the first incoming one, and so the priority in the bundle queue is such that the first bundle to arrive is the first to be dropped. Of course, the information of the time the bundle was received has to be kept.

Drop Last

Contrarily to Drop Head policy, in the Drop Last policy, the first bundle to be dropped is the last received bundle in the buffer.

Drop Oldest

In Drop Oldest (DO) [26], the analysis is based on the RTTL of the bundle. The bundle with the shortest RTTL is the dropped one.

Drop Youngest

The Drop Youngest (DY) algorithm functions as an inverse of the DO algorithm. In this case the chosen bundle to be dropped is the one with the largest RTTL.

Drop-largest

In the Drop-largest algorithm, the dropped bundle is the one that occupies the most memory in the node's buffer.

Drop Most Hops

The number of hops corresponds to the number of nodes a bundle was passed through until it reached the current node. In the Drop Most Hops policy, the chosen bundle to be dropped is the one with the most hops [27].

2.4.2 Information Based Drop Policies

The following algorithms rely on learned information of the bundle after it has been buffered, as main sources of information to choose which bundle to drop. Some also rely on specific algorithms that compare the bundle to be stored and its characteristics to the stored bundles, to choose the dropped bundle.

RC-Drop

The Replicated Copies Drop (RC-Drop) policy, also explained by Soares in [25], relies on the number of copies of each of its buffered bundles in the network. The algorithm works optimally for Spray and Wait routing protocol since it is possible to infer the number of copies of each bundle in the network with only local knowledge. The main variations are the RC-Drop Descending (RC-DropDesc), where the dropped bundles are the most replicated ones, and RC-Drop Ascending (RC-DropAsc) where the dropped bundles are the least replicated ones. When a tie happens, other elementary drop policy such as Drop Head or Random is used.

MOFO

Evict Most Forwarded First (MOFO) [21] used by Rani in [28] chooses the bundle which has been forwarded the largest number of times to be dropped.

In this policy, every bundle has a corresponding forward counter variable – stored separately in each node – named Number of Forwards (NF), which starts at 0 and is incremented every time the bundle is forwarded. Each time a bundle is replicated the NF counter is updated through the equation,

$$NF = NF_{old} + 1. \qquad (2)$$

In which NF corresponds to the number of times the bundle has been forwarded in the present moment and NF_{old} to the previously stored NF value.

N-Drop

Just as with MOFO, N-Drop [29] relies on the NF value of each bundle. In this algorithm, a maximum N value is given to the NF. When a new bundle must be stored, the algorithm drops the bundle whose NF is larger than N in order to make room for the incoming bundle. If every bundle's NF is smaller than N, as standard, the algorithm uses Drop Last policy to choose which bundle to drop.

T-Drop

In the Threshold Drop (T-Drop) policy, described by Ayub in [30], when the node's buffer is full, for an incoming bundle that has to be stored, the algorithm searches the buffer for bundles with sizes in the threshold range T, deleting the first encountered. This process is repeated until the needed space for the incoming bundle is freed.

For instance, if the range of threshold T is between 100 kb and 200 kb, and the stored bundles' sizes do not fall within the specified range, the node will not drop any bundle.

According to the authors in [30], this technique results in less message drops, overhead and hop count, with high delivery rate.

E-Drop

The Equal Drop (E-Drop) algorithm approaches a congested node that must buffer a new bundle, by checking every Bundle Size (BS) on the node's buffer, dropping the first bundle i in which

$$BS_i \geq BS_{new}, \tag{3}$$

in which BS_i corresponds to the BS of the bundle i, and BS_{new} corresponds to the incoming bundle BS. Freeing the necessary space for the new incoming bundle. If condition (3) fails, the algorithm will drop the necessary number of bundles, dropping the bundles from the biggest to the smallest, until the freed buffer space is greater or equal to the incoming bundle's size [26].

By considering the BS instead of the NF in this algorithm, the number of drops is smaller since the space for the incoming bundle will be freed using the minimum drops possible. Rashid in [26], argues that this way the smaller messages will be prioritized, reducing the message drop ratio, and increasing the delivery probability in comparison with the MOFO algorithm.

Of course, this method is not optimal for scenarios where the BS fluctuation across the network is flat, since that in this scenario the algorithm will have a behaviour close to FIFO. In this case, MOFO algorithm would have better results.

MDC

Ayub in [31], introduces the Message Drop Control (MDC) policy. This policy was created to work with FC routing protocol. This algorithm works similarly to the E-Drop algorithm with the introduction of the Upper Bound (UB) variable.

In MDC, if the total occupied memory is larger than the previously defined UB, while the freed space on the buffer is smaller than the incoming bundle size, the algorithm searches the largest available bundle in the buffer, dropping it, until the available buffer size is larger than the incoming bundle size.

Mean Drop

The Mean Drop policy, described in [32] by Rashid, when it has to drop bundles in order to store a new one, it starts by computing the bundle size mean, afterwards it drops every equal or bigger sized bundle, until the freed space is sufficient to accommodate the newly received node.

Mean Drop has good performance in scenarios where PRoPHET and Epidemic routing protocols are working. According to [32], in these kind of applications this algorithm outperforms MOFO and DO in message relay, drop, overhead and also enhances the bundle delivery rate.

2.4.3 Probabilistic Drop Policies

The estimations of the delivery probability of the bundle emerged to increase the bundle delivery rate and improve the overall performance of the network. In the present section, some of the most relevant probabilistic drop policies will be explained. MOPR and LEPR policies use metrics present in the routing protocols which they were developed to work in. On the other hand, Small-copies drop uses its own proprietary metric, that may be used with every flooding or flooding based routing protocol.

MOPR

Evict Most Favourably Forwarded First (MOPR), is a weighted version of MOFO. Instead of incrementing the Forwarding Predictability (FP) by one each time the bundle is forwarded, this algorithm increments FP with the delivery predictability of the receiving node. Of course, as in MOFO, the FP is initialized at 0. As illustrated by Lindgren in [33], MOPR policy follows the following equation:

$$FP = FP_{old} + P. \tag{4}$$

Being FP_{old} the value of the bundle before it was forwarded, FP the updated value and P the deliverability predictability the receiving node has for the message.

LEPR

Evict least probable first (LEPR), as described in [33], is a policy in which is dropped the least probable bundle to be delivered by the bundle carrying node. This policy can only function when paired with routing protocols in which the delivery probability is estimated e.g. PRoPHET and MaxProp.

For each bundle, considering the bundles' destination, a delivery probability is estimated by the routing protocol. Afterwards, the bundle which the delivery probability is the lowest is chosen to be dropped.

Small-Copies drop

Drop Head and DO algorithms in scenarios with Epidemic Routing, show better delivery ratio and latency when compared respectively, to Drop Last and DY as is mentioned by Kim in [34].

Based on these algorithms, Kim in [34], concludes that a drop of a more replicated bundle has less impact on the delivery ratio than a drop of a less replicated one. Small-Copies drop algorithm (SCop), developed in [34], tries to calculate how many copies of a bundle are in the network, and drops the most replicated message.

In order to specify the algorithm, it might be useful to define the variables:

- *T*: specified time;
- λ: average number of encounters of new nodes in the time *T*;
- *R*: bundle's RTTL;
- *i* and *j*: nodes;
- *M*: bundle.

As illustrated in Figure 9, at a contact opportunity, the node *i* replicates the bundle *M* to *j*, and calculates the following value:

$$P(\text{"}j\ is\ unable\ to\ replicate\ the\ bundle\text{"}). \tag{5}$$

Which is equal to the probability of *j* not encountering other nodes to replicate *M*. Through (5), and assuming the node encounter probability follows a Poisson distribution it is possible to calculate the following probability

$$P(\text{"}M\ cannot\ be\ relayed\ from\ j\text{"}|\text{"}j\ receives\ M\ from\ i\text{"}) = e^{-\lambda_j R_{ij}}. \tag{6}$$

In which λ_j is the average number of contacts that node *j* has in the specified time *T* and R_{ij}, which is the remaining life at the transfer time of *M* from *i* to *j*.

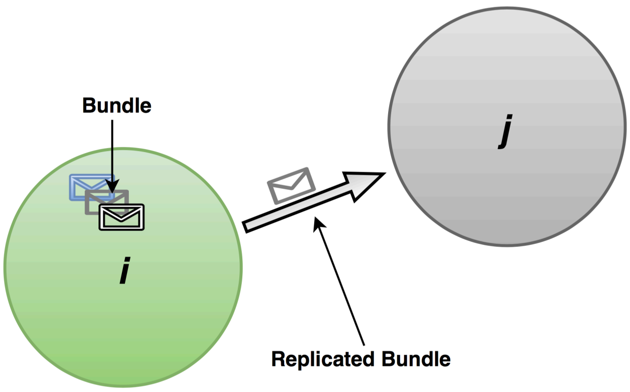


Figure 9 – Bundle replication illustration on a contact opportunity between nodes (*i* and *j*) for routing algorithms of the flooding type.

The probability is initialized at the node *i* as

$$P(M) = e^{-\lambda_i R}, \tag{7}$$

where λ_i is the average number of contact opportunities at the node *i*. At each replication of the bundle *M* to a node *j*, this value gets updated to

$$P(M) = e^{-(\lambda_i R + \lambda_j R_{ij})}. \tag{8}$$

Basically, this algorithm searches for the bundle with the maximum value *P(M)* and drops it, successfully managing to only drop the most replicated bundles in the network.

This algorithm, according to Kim in [34], outperforms DY, DO, Drop Front and Drop Last policies in terms of delivery ratio. Of course, in a scenario where the node's encounters are not independent from each other, thus not following a Poisson distribution, this algorithm is sub optimal.

In Table 3, it is presented a summary of the discussed drop policies, which shows the compatibility, summarizes and presents for each drop policy its major advantages and disadvantages as well as its compatibility with the routing protocols already addressed in section 2.2 and Table 1.

Table 3 – Drop policies brief table.

	Drop Policies	Main Parameter of the used metric	Routing Protocols Compatibility	Summary	Advantages	Disadvantages
Elementary	Random	None	All	Drops a randomly chosen bundle.	Adequate for small networks with low traffic.	Its random nature.
	Drop Head	Time of arrival	All	Drops the first incoming bundle.	Good for networks with low traffic.	Does not take advantage of the routing protocol metric.
	Drop Last	Time of arrival	All	Drops the last incoming bundle.	None	Lower delivery rate when compared to random.
	Drop Oldest	RTTL	All	Drops the bundle with the smallest RTTL.	Drops the bundles which would be dropped in a near future, increasing the delivery probability for younger bundles.	Does not take into account BS nor the destination of the node.
	Drop Youngest	RTTL	All	Drops the bundle with the largest RTTL.	None	Lower delivery rate when compared to random.
	RC-Drop:			Using the stored number of replicas of each bundle:		
	Ascending	Number of copies	Spray and Wait	Drops the bundle with the lowest number of replicas.	Good for congested networks.	Requires the information of the number of replicas in the network.
	Descending			Drops the bundle with the highest number of replicas.	None	Lower delivery rate when compared to random.
	Drop-largest	Size	All	Drops the largest sized bundle.	In congested networks, achieves the highest average node's available buffer space.	Bad in networks where the bundle size is fixed.
	Drop Most Hops	Number of hops	All	Drops the bundle with the most hops.	Good performance in congested networks.	Bad performance in very sparse networks.
Information Based	MOFO	Number of forwards	All	Drops the bundle with the largest NF value.	Good for detecting bundles which have already been replicated, and therefore do not need to be prioritized.	Does not update NF values between same bundle carriers.
	N-Drop	Number of forwards	All	Drops every bundle with a NF larger than a certain threshold N.	Good for detecting bundles which have already been replicated, and therefore do not need to be prioritized.	Bad in sparse networks.
	T-Drop	Number of forwards	All	Drops bundles with a similar size to the incoming one.	Only drops bundles with similar sizes to the incoming one, keeping delivery opportunity independent of the BS.	Bad in networks where the bundle size is fixed.
	E-Drop	Number of forwards	All	Drops larger bundles than the incoming bundle.	Only drops the needed space to accommodate the new bundle.	Bad in networks where the bundle size is fixed.
	MDC	Number of forwards	FC	If a certain level of occupancy is reached, the algorithm searches a larger bundle than the incoming one to be dropped.	Increases the homogeneity of the occupied buffer space.	Bad in networks where the bundle size is fixed.
	Mean Drop	Number of forwards	PRoPHET, Epidemic	It computes the bundle size mean, and drops every bundle which value is larger than the mean until enough memory is available.	Prioritizes smaller bundles which have a small cost in terms of buffer space.	Bad in networks where the bundle size is fixed.
Probabilistic	Small-Copies Drop	Proprietary metric	Only flooding/flooding based	Drops the bundle which has been in the most "popular" nodes.	Good in scenarios where the movement is close to random.	Uses as premises that the node encounters have a random behaviour, which is not true in human movement.
	MOPR	PRoPHET metric	PRoPHET	Drops the bundle which has been in the most probable to deliver nodes.	Prioritizes bundles which have not been forwarded to nodes with a large delivery probability. Good performance in congested networks.	Does not update nor transfer the FP values with other nodes.
	LEPR	PRoPHET metric	PRoPHET, MaxProp	Drops the least probable bundle according to the protocol metric.	Good in congested networks, where a large heterogeneity exists in terms of delivery probability between nodes.	Does not take into account the RTTL.

2.5 Congestion Control Mechanisms

In the internet, the availability of end-to-end connectivity is assumed, thus from router-to-source, the Congestion Control mechanisms are based on two options: explicit and implicit. The explicit mechanisms work by sending information of the router congestion to the traffic source, e.g. by sending an Internet Control Message Protocol (ICMP) [35] source quench packet. In the implicit mechanisms, the congestion control works exploiting TCP operation [14], by simply discarding the datagrams as the need comes, the TCP acknowledgement will not be received by the source, which will result in the source reducing the transmission rate. Of course in DTNs, this type of mechanisms cannot be applied due to its frequent disconnections and the absence of an end-to-end connection [36].

In DTNs, there are many types of congestion control mechanisms, which aim to control the network and prevent it from reaching the Congestion State (CS). These can be classified in open or closed loop. The open loop congestion mechanisms are often based on dropping bundles, whereas closed loop relies on the feedback information of network parameters such as the NF and average buffer memory occupation. As referred by Silva in [8], the vast majority of the mechanisms (over 80%) classify as closed loop mechanisms, besides their harder implementation in scenarios such as DTNs.

Also, it is possible to classify the mechanisms as proactive, reactive or hybrid. While proactive mechanisms aim to avoid the CS, the reactive mechanisms try to respond when the network is in the CS. The hybrid type uses both proactive and reactive congestion mechanisms to avoid the CS. According to [8], the majority of the mechanisms (52%) fall into the Hybrid approach.

However, since that Silva also separates the various mechanisms per amount of network knowledge they require, to separate the mechanisms, a similar approach was followed.

Since that in DTNs in general, global knowledge is hard to acquire, the algorithms were separated in the global knowledge, and the neighbourhood and local knowledge types, being that for DTN' applications the most useful ones are the neighbourhood and local knowledge mechanisms, since they do not require a global network information access to function.

2.5.1 Global knowledge

In this section, one of the main global knowledge mechanisms is explained. This classification comes from this type of mechanisms relying on global network knowledge, which is very hard to acquire especially in DTNs.

ANFER

The Average Forwarding Number based on Epidemic Router (ANFER), as the name would suggest, is used for Epidemic Routing and falls into the open loop category.

When a node with a full buffer has to accommodate a new bundle, the mechanism randomly drops a bundle in which the NF is greater than the average number of forwards of every bundle in the network. This is achieved by keeping track of how many times a bundle has been forwarded in the network which is equal, in the Epidemic routing protocol, to the number of copies of the bundle throughout the network [8].

This algorithm is classified as a global knowledge algorithm, since the information of a bundle's global number of forwards requires the access to every node in the network, otherwise only estimation techniques based on other parameters could be used. Thus, this technique is not very useful in DTNs.

2.5.2 Neighbourhood and local knowledge

The neighbourhood and local knowledge mechanisms, as the name suggests, are easier to implement in scenarios where global knowledge is very hard to acquire. The following mechanisms only require local and neighbourhood acquired knowledge about the network and so, are very interesting for DTN applications. Some require information on the neighbourhood level to perform well, thus being able to achieve good results with fairly easily obtained knowledge. The vast majority of the available Congestion Control Mechanisms fits in this category, beyond being the more balanced ones, they allow for good performance results in DTN scenarios.

MACRE

An in [37] introduces the Message Admission Control based on Rate Estimation (MACRE) mechanism, which uses local information (particularly the data rates) to make decisions about accepting new bundles or not.

For a bundle of size *N*, when a node has a full buffer or its R_{out} – the historical statistical value of the data output rate – multiplied by the $RTTL$ of the bundle is smaller than *N*, the incoming bundle is discarded. Not being discarded, if either the free space of the buffer FB is larger than a defined constant (between 0 and 1) multiplied by the total buffer size, or the following condition

$$(R_{in} - R_{out}) \times RTTL < FB \tag{9}$$

is respected, being R_{in} the historical statistical value of the data input rate, the bundle is received. Otherwise, the bundle is discarded.

In conclusion, this technique aims to choose the bundles in which the RTTL is sufficient for them to be forwarded, based on the previous input and output rates of the node.

Congestion Avoidance Based on Buffer Space Advertisement

Introduced by Lakkakorpi in [38], the Congestion Avoidance Based on Buffer Space Advertisement is a congestion mechanism in which each node provides information to the connected nodes about its buffer availability. The information is provided inside the *Hello* messages already periodically shared between the nodes.

As illustrated in Figure 10, the buffer advertised availability B_a is smaller than the true free space in the buffer B_s. The B_a is calculated through the expression

$$B_a = T_C \times B_s - B_o, \quad (10)$$

in which T_C is the congestion threshold and B_o is the buffer occupancy. The congestion threshold is a number between 0 and 1, and serves the purpose of providing a size adjusted safety margin, which is important since this mechanism works with heterogeneous buffers, and also because concurrent transmissions from adjacent nodes could lead to message drops.

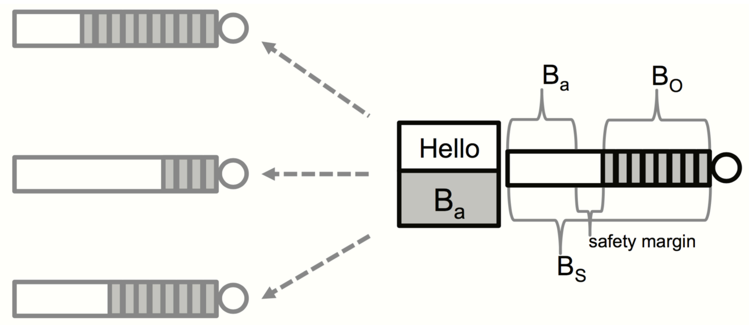


Figure 10 – Congestion Control based on buffer advertised availability scheme (extracted from [38]).

T_C is also adjusted with the number of drops. For example, if there is a message drop between *Hello* messages, T_C is reduced. On the other hand, if there are no drops, T_C is incremented and the safety margin is minor.

The mechanism can work in any DTN routing scheme, and does not rely on specific routing information, which eases the implementation process and also simplifies it for use in DTNs where the information about the network is very hard to obtain.

Autonomous Congestion Control

Introduced by Burleigh in [36], this control mechanism was made using an analogy in which a node is a capital banker in which its buffer capacity is the capital the banker has available to work. The mechanism works based on three rules.

Using the variables:

- Q – Buffer occupied memory (in bytes);
- N – Size of the incoming bundle (in bytes);
- K – Node's buffer capacity (in bytes).

In the first proposed rule, for an incoming bundle, it is refused if the following rule is not respected:

$$(Q + N) \leq K. \quad (11)$$

If the first rule is true for the incoming bundle, by analysing the $RTTL$ of the bundle and the mean net growth per second in a period of time T, M_T, which is the sum of the sizes of the accepted bundles minus the sum of the sizes of the forwarded bundles during T, it is possible to calculate the expected net growth, G_{RTTL}, during the $RTTL$ of the incoming bundle using the equation:

$$G_{RTTL} = M_{RTTL} \times RTTL. \tag{12}$$

The calculation in (12) is used to calculate the worst-case scenario of the amount of dropped bytes during the incoming bundle $RTTL$:

$$D = (G_{RTTL} + Q + N) - K. \tag{13}$$

After computing the D value, the second rule may be applied, in which if $D > 0$, the bundle acceptance will probably have a cost, thus if $D \leq 0$ the bundle is accepted, otherwise the acceptance risk of the bundle must be analysed.

The node is incentivized to carry high priority bundles. Through the priority information provided by the application, a *fee*, F_q, is calculated, which is proportional to the previously defined priority of the bundle by the application. The following equation computes the total cost of the bundle delivery, P:

$$P = N \times F_q. \tag{14}$$

The value, V, of accepting a bundle is related to the *fee*, being the most valuable bundles the ones with the highest priority. Assuming that the total value of the bundles received during the period T, V_T, is calculated and the risk, R, of accepting a new bundle is calculated through the following equation:

$$R = N \times RTTL. \tag{15}$$

It is possible to compute the mean of the risk of the incoming bundles during the period T, R_T.

The risk rate of a bundle is R/V also, being the mean risk rate R_T/V_T the third rule is:

$$\frac{R}{V} < \frac{R_T}{V_T}. \tag{16}$$

If the above rule is not respected, the bundle is refused, otherwise it is accepted.

Having only local information of the node, the mechanism minimizes data loss in the network; which eliminates communications overheads as may happen with mechanisms that request neighbourhood information.

2.6 Tools and Simulator

In the following section, a small introduction to the main simulation tool used in the work is provided. An overview is provided, some of the capabilities and some of the already implemented scenarios and routing protocols are described.

2.6.1 The ONE

The performance of routing and other application protocols is affected by the underlying characteristics of the network such as the mobility and characteristics of the different nodes. The Opportunistic Network Environment (ONE) simulator[2] is able to simulate through the defined parameters the specified scenario and measure its performance through the generated reports and data that may be post processed.

Other alternatives to measure the performance of these routing protocols would be to simulate them in real world scenarios by using real life traces like those provided by the Community Resource for Archiving Wireless Data At Dartmouth (CRAWDAD)[3]. These traces are obtained from real world environment movement using mobile phones. Using these traces, the performance of routing protocols, dropping policies, scheduling policies and congestion control mechanisms can be tested in a simulator that is able to read the traces to have a more realistic result. This type of methodologies may validate mobility and connectivity characteristics obtained by synthetic models. Although, as referred by Keränen in [39], the CRAWDAD platform has some drawbacks such as the small population and the reduced node capabilities – in order not to drain the phones battery – which may result in unreliable results.

Some of the main functionalities of the ONE include being able to simulate node movement, inter node contacts and also routing and bundle handling, as explained by Keränen in [40].

As illustrated in Figure 11, the node movement is simulated using movement models, which may be synthetic, such as random movement patterns, or based on existing collected movement traces. The connectivity is based on node location, communication range and bit-rate. Routing is implemented by some available protocol models such as Spray and Wait, Epidemic, RAPID, FC, MaxProp, DD and PRoPHET, already explained in 2.1. It is possible to add other routing protocols to the ONE through the creation of a new routing module. The routing modules decide which bundles to forward over the existing contact opportunities. The messages are generated through event generators.

Beyond the simulation, the ONE collects the simulation data metrics and illustrates them through the visualization (graphs, message paths) and reports generation. The data may also be processed using other tools.

---

[2] https://akeranen.github.io/the-one

[3] https://crawdad.cs.dartmouth.edu

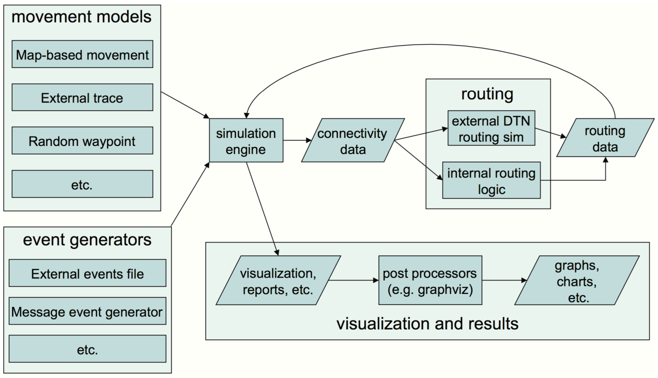


Figure 11 – Overview of the ONE simulation environment (extracted from [40]).

2.6.2 The ONE Simulation Scenarios

In this section, it is explained how the configuration process takes place, the configurable parameters and also a brief explanation of the node capabilities, and the mobility models.

2.6.2.1 Node Capabilities

Nodes act as a store-and-carry router, which may be a pedestrian, a car or a tram, with different velocities and available paths. As explained in [40], some of the node capabilities include radio interface, persistent storage, movement, energy consumption and message routing.

Some of the capabilities like radio interface and persistent storage depend on parameters such as the communication range, bitrate, storage capacity and peer-scanning interval.

Others such as movement and message routing work through modules, which implement a particular behaviour, since different mobility models are available.

Dynamic links and context awareness configuration mechanisms depending on the node's context, considering the surroundings and the distance between peers, can adjust range and the bit-rate thus, providing more realistic results.

The node's energy consumption is based on the provided energy budget. Through other modules, it is possible to adjust the scanning frequency of the surroundings and the transmission power, as a way to save energy.

2.6.2.2 Mobility Models

As referred in [39], in order to have a scalable and flexible simulation, model based mobility and Map-Based mobility are available to use in the simulations. These mobility models may be Random Map-Based Movement (MBM) which has the nodes randomly navigating through the available paths in the map; Shortest Path Map-Based Movement (SPMBM) which is more realistic since the nodes navigate from one point in the map, through the shortest path available, to random points in the map or previously chosen points of interest; or even Routed Map-Based Movement (RMBM) which has the nodes navigating in predetermined routes, this may be useful for modelling the movement of vehicles such as trams and buses.

Although the previously mentioned models are intuitive and easily implemented, they do not match real movement traces as the Working Day Movement (WDM) does. By modelling typical human routines during working weeks, bearing in mind sleeping hours at home, working hours at the office, friends gathering at the end of the day and predictable group movements, Keränen in [39] states that the model achieves more realistic movement patterns when compared to the simpler non-social models such as MBM, SPMBM and RMBM which do not consider communities and relationships among the nodes.

The ONE is able to simulate different nodes in different mobility models, which is beneficial since the various types of nodes may have movement patterns better simulated by different models.

2.6.2.3 Simulation Scenarios Parameters

The simulation scenarios are created by changing the nodes characteristics like the storage capacity, bit-rate, range, among others. Also, by choosing the type of simulated movement and the routing model to use. The simulation duration and granularity should also be defined, as referred in [39].

The customization of the simulation is done through simple text files, containing the simulation scenarios, user interface, reporting parameters, etc. Through these simple text files, it was possible to configure the various simulations to be executed and analysed in the present work.

2.6.3 Performance Metrics

Through the reports modules, the ONE provides information about the performance of the routing protocol or scheduling and drop policy being tested, the visualization of the simulation and its results may be done through the Graphic User Interface (GUI), or through post produced images, graphs and tables.

In the ONE's GUI, as illustrated in Figure 12, it is possible to see node locations, connections, number of carried bundles, paths, etc. It is also possible to adjust the simulation speed, pause it and on the map zoom in and zoom out.

The reports information may be used with tools like Graphviz[4] to generate graph files to see nodes connections during the simulation. Message statistics reports gathers statistics of the overall performance such as [39]: the number of created messages; delivery ratio; how long bundles stay in the buffer; etc..

---

[4] http://www.graphviz.org

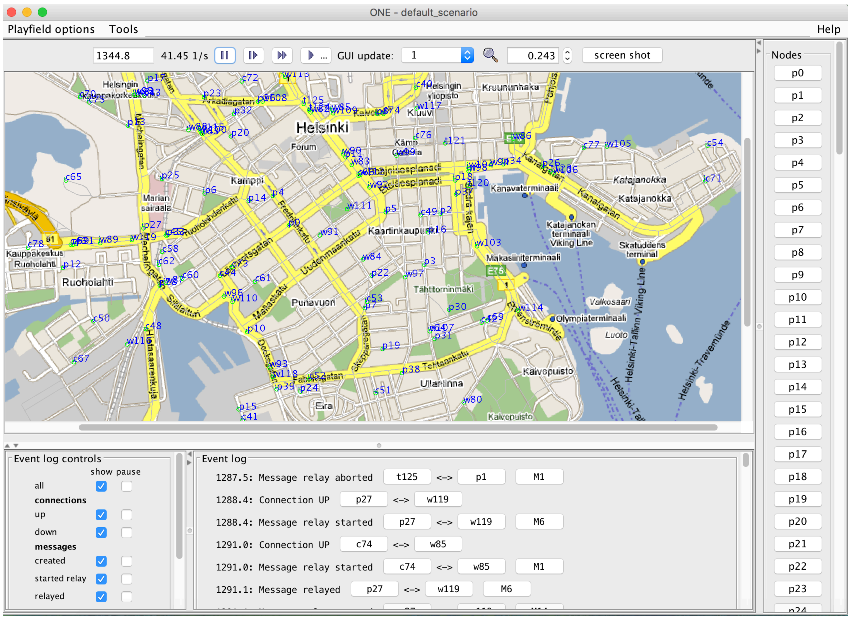


Figure 12 – Screenshot of the ONE simulator's GUI.

Through all provided information in the ONE after each simulation, in the present work the used performance metrics to evaluate the performance of the different routing protocols, drop and scheduling polices combinations are:

- Bundle delivery rate – Ratio of the successfully delivered bundles over the total number of bundles generated;
- Average delay – Average time the bundles take to be delivered;
- Overhead – Ratio of the difference between the relayed and delivered messages (R_M and D_M respectively), over the number of delivered messages, as illustrated below:

$$Overhead = \frac{R_M - D_M}{D_M}. \qquad (17)$$

Considering the ONE's open configuration capabilities in the implementation of protocols, policies, scenarios and reports, it was seen as an excellent tool to implement and test a proposed policy.

Chapter 3

Drop Policies Comparison

In this chapter, an analysis in terms of the bundle delivery rate, average delay and overhead ratio of every relevant state of the art drop policy is made. Simulations were made in different scenarios so that the results are as accurate as possible and prove consistency when compared with related works in similar scenarios. This study is made to understand which attributes of a node, such as its size, time of arrival, RTTL, number of copies, etc. are the most important when evaluating which bundle should be dropped.

3.1 Elementary drop policies comparison

In this section, the elementary drop policies described in section 2.4.1 will be compared in terms of delivery probability, average delay and overhead ratio.

After the analysis of each policy, to understand which standard bundle variables (such as the TTL, size, time of arrival, number of copies, etc.) have more impact in the performance metrics presented in section 2.6.3, when simulated in the same scenario, some conclusions will be drawn.

In this comparison, the drop policies have been joined with the corresponding scheduling policies because of their simpler nature. In this way, it is easier to infer which characteristics cause more impact in the network performance.

3.1.1 Simulation scenario

With the intent of comparing the most elementary policies, it was simulated a simple scenario in the ONE simulator to conclude which features of the policies are the most important. In the present section, the scenario parameters and settings will be given in detail.

The chosen routing protocol was the Spray and Wait protocol, because with this protocol [8], it is possible to infer locally, without the exchange of information between nodes the number of replicas of a certain bundle in the network. As explained in section 2.1, when using the binary variation of the Spray and Wait, when the number of replicas is counted, knowing the initial number of replicas of the bundle when it was created, if subtracted to the initial number of replicas the number of replicas in the buffer, it is possible to learn, in total, the number of replicas in other network nodes.

Through the equation below

$$N_{forwards} = log_2\big(N_{buf}(t_0)\big) - log_2\left(N_{buf}(t)\right), \tag{18}$$

in which $N_{forwards}$ corresponds to an estimation of the number of times the bundle was replicated to other nodes, N_{buf} the number of replicas in the node's buffer, t_0 to the time of creation of the bundle and t to the present moment, it is possible to estimate the number of times a message has been forwarded.

The estimated number of replicas of a certain bundle in the network, N_{net}, corresponds to

$$N_{net} = N_{buf}(t_0) - N_{buf}(t). \tag{19}$$

For instance, if at a certain moment a node has 8 replicas of a certain bundle which started with 32 replicas at the time of its creation, using the equations above it is possible to understand that the bundle was replicated 2 times to reach this particular node and the total number of replicas in other network nodes is 24.

The general scenario parameters and settings are defined in detail below in Table 4, and the specific node configuration parameters are specified in Table 5.

To truly test the drop policies the used scenario should be one in which the network is congested, since in the case of a low congestion scenario, there is no need to drop packets often and so the results with all the drop policies are similar for the performance metrics proposed in section 2.6.3. Hence, the chosen message creation interval was from 15 to 30 seconds, and the buffer size for most of the nodes was 5 MB as visible in Table 4 and Table 5.

The chosen message size interval is between 250 kB and 2 MB. One way to create congestion is by increasing the ratio between the message size and the buffer size.

The Bluetooth interface data speeds chosen were based on Bluetooth speeds and range, and the high-speed interface values are the standard values in the ONE.

The chosen movement model was the Shortest Path Map Based Movement (SPMBM). As explained in [41] by Keranen, SPMBM is based on the simple random map-based model, in which the nodes move to randomly determined positions on the map following the roads defined by the map data. Although, in SPMBM model the nodes do not wander randomly around the map, instead the Dijkstra's shortest path algorithm is used to calculate the shortest paths from the current location to a randomly selected destination. The bundle's source and destination is chosen randomly, using a random number generator with a seed derived from the nodes' group prefix. The different seeds guarantee that each simulation is unique, since that the nodes have different movement patterns for different seeds.

Influenced by Soares in [25], the simulations were run varying the TTL of the bundle, which was varied from 30 to 180 minutes.

In the interest of gathering the most precise and exact results of the performance metrics, 12 different seeds were used for each TTL and drop policy pair. The main objective was to compute 95% confidence values for each 12 simulations. This number of seeds managed to deliver values with narrow 95% confidence intervals. Other values such as the number of nodes, speeds, etc. were dimensioned to create a realistic scenario in terms of movement.

Considering that these elementary policies do not need to estimate any parameter, the chosen simulation time was 12 hours, since the results would not differ for longer periods.

The warmup time, is a setting in the ONE simulator which prevents it to consider events during the warmup time. The warmup time setting is usually used when a protocol estimates specific values which take time to converge, and so the events in the beginning of the simulation may not be of interest. Since that in the present simulation, none of the protocols or policies must estimate any values, no warmup time was set.

Table 4 – Elementary Drop Policies Simulation Scenario: General Configurations.

Elementary Drop Policies Simulation Scenario		
General Configurations		
Routing Protocol Initial number of replicas	Spray and Wait Binary 12	
Movement Model	Shortest Path Map Based Movement	
Number of Seeds	12	
Time	12 hours	
Message Size interval	250kB to 2MB	
Message Creation Interval	15 to 30 sec.	
Normal Bluetooth Interface	Data Speed Range	4.5 Mbps 30 m
Highspeed Interface	Data Speed Range	40 Mbps 10 m

Table 5 – Elementary Drop Policies Simulation Scenario: Node Configurations.

Node Configurations		
Pedestrian	**Number of hosts**	**100**
	Wait Time	5 to 15 min.
	Node Speed	1.9 to 5.6 km/h
	Buffer Size	5 MB
	Blutooth Interfaces	1 Normal
Car	**Number of hosts**	**50**
	Wait Time	10 to 30 min.
	Node Speed	10 to 50 km/h
	Buffer Size	5 MB
	Blutooth Interfaces	1 Normal
Tram 1	**Number of hosts**	**4**
	Wait Time	10 to 30 sec.
	Node Speed	25 to 35 km/h
	Buffer Size	50 MB
	Blutooth Interfaces	1 Normal
Tram 2	**Number of hosts**	**2**
	Wait Time	10 to 30 sec.
	Node Speed	25 to 35 km/h
	Buffer Size	50 MB
	Blutooth Interfaces	1 Normal 1 Highspeed

3.1.2 Simulation Results

After the simulation, the results were processed and organized in the tables and figures below. In this section, the results of the proposed performance metrics described in 2.6.3 will be displayed and analysed.

3.1.2.1 Bundle Delivery Rate

In Table 6 it is visible that the policies with the best overall performance are Drop Oldest, Drop Head and Drop-largest – the table values are represented through a graph in Figure 13. One of the reasons for the better performance in terms of bundle delivery rate, is the ordering of the messages by their age. In the case of the Drop Oldest policy, the RTTL is used to order the messages, dropping the one with the lowest RTTL.

In addition, as visible in Table 7, and also in the illustrated intervals for each policy measurement in Figure 13, the results in this performance metric show high reliability, since the 95% confidence intervals are narrow.

Table 6 – Bundle delivery rate for elementary drop policies using Spray and Wait binary.

TTL [min.]	Bundle Delivery Rate						
	FIFO and Drop Head	Random and Random	RCAsc and RC-DropDesc	RCDesc and RC-DropAsc	RLDesc and DO	RLAsc and DY	SSmall and Drop-largest
30	24.44%	23.54%	24.00%	22.98%	24.66%	22.58%	24.30%
60	37.56%	35.82%	36.24%	33.33%	37.66%	33.00%	37.59%
90	42.59%	40.82%	40.95%	36.83%	42.43%	37.87%	43.13%
120	44.74%	42.86%	43.03%	37.77%	44.54%	39.47%	45.46%
150	46.01%	44.62%	44.02%	37.97%	45.81%	40.34%	46.04%
180	46.93%	45.43%	44.84%	38.12%	46.73%	40.11%	46.37%

Table 7 – Bundle delivery rate 95% confidence intervals for elementary drop policies using Spray and Wait binary.

TTL [min.]	Bundle Delivery Rate 95% confidence intervals						
	FIFO and Drop Head	Random and Random	RCAsc and RC-DropDesc	RCDesc and RC-DropAsc	RLDesc and DO	RLAsc and DY	SSmall and Drop-largest
30	0.43%	0.32%	0.43%	0.35%	0.44%	0.27%	0.34%
60	0.57%	0.48%	0.52%	0.47%	0.63%	0.42%	0.59%
90	0.68%	0.72%	0.69%	0.61%	0.70%	0.56%	0.68%
120	0.68%	0.83%	0.73%	0.50%	0.71%	0.60%	0.56%
150	0.67%	0.63%	0.69%	0.49%	0.71%	0.65%	0.67%
180	0.69%	0.57%	0.73%	0.39%	0.68%	0.52%	0.74%

In terms of Drop Head and Drop-largest policies, the good results are due to the efficient buffer management, by dropping the messages which were stored in the buffer for longer periods and the ones with larger sizes, respectively, the buffer can give the opportunity to other messages to be relayed instead of prioritizing messages that either have had opportunity to be relayed already or take too much space in the buffer, respectively.

It is also important to refer that the Drop-largest policy has also achieved good results due to bearing in consideration the size of the message, which validates the theory of the policy – to increase the bundle delivery rate, it is better to drop less messages to be able to buffer more messages.

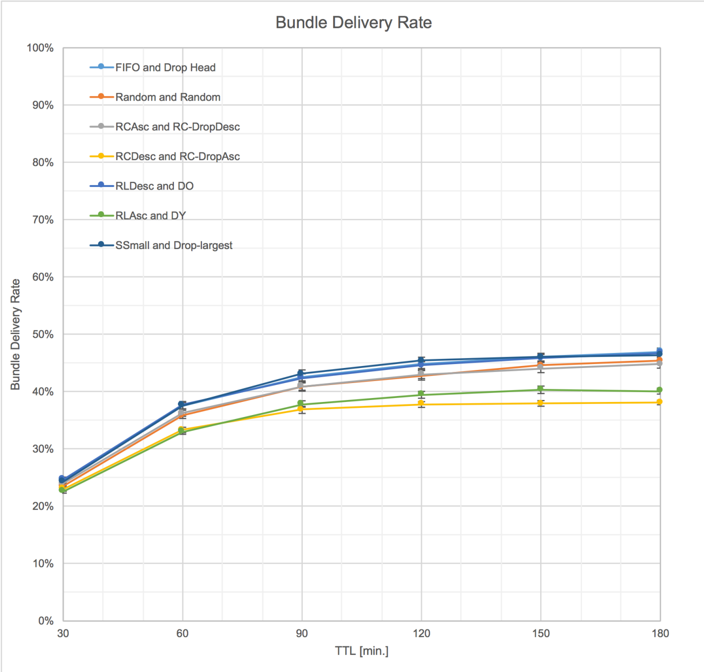


Figure 13 – Delivery probability for the elementary drop policies.

In the larger TTL values, the policies start to differ from each other, since the better the drop policy behaves, the better it can manage the bundles in the network whose number increases for larger TTLs. Of course, this does not mean the bundle delivery rate will decrease with an increase of the TTL. Since for larger TTLs the messages have more time to be delivered, only the average delay of the bundle is expected to increase.

3.1.2.2 Average Delay

As expected, the Drop Oldest policy achieves a good result in terms of average delay. This is due to its algorithm that prioritizes messages with larger RTTL. If an old message (with a small RTTL) has still not reached its destination, it will not be kept in the buffer for long and so, messages with a small RTTL will not be delivered, lowering the average delay.

As visible in Figure 14 and Table 8, the Drop Head policy also shows a very close behaviour to the Drop Oldest policy. Of course, this is related with discarding the bundles stored earlier. Tendentiously, bundles which have been in the network for longer, have also small RTTLs, which explains the similarity between DO and Drop Head.

Withal, as visible in Table 9, and also in the illustrated confidence intervals for each policy measurement in Figure 14, the results in this performance metric show high reliability, since that the 95% confidence intervals are small.

Table 8 – Average delay for elementary drop policies using Spray and Wait binary.

TTL [min.]	Average Delay [s]						
	FIFO and Drop Head	Random and Random	RCAsc and RC-DropDesc	RCDesc and RC-DropAsc	RLDesc and DO	RLAsc and DY	SSmall and Drop-largest
30	899.24	909.64	908.27	913.22	895.55	928.87	916.78
60	1479.30	1559.18	1554.73	1584.28	1453.97	1658.75	1612.23
90	1825.60	2022.80	2014.85	2071.05	1784.78	2275.49	2151.37
120	2041.28	2298.65	2348.33	2430.35	1998.48	2747.75	2542.35
150	2210.72	2558.97	2600.47	2697.11	2166.68	3120.33	2844.33
180	2362.95	2727.50	2824.11	2930.68	2321.91	3462.83	3104.29

Table 9 – Average delay 95% confidence intervals for elementary drop policies using Spray and Wait binary.

TTL [min.]	Average Delay 95% confidence intervals [s]						
	FIFO and Drop Head	Random and Random	RCAsc and RC-DropDesc	RCDesc and RC-DropAsc	RLDesc and DO	RLAsc and DY	SSmall and Drop-largest
30	9.01	8.02	11.55	11.81	10.40	8.25	9.88
60	13.93	15.94	17.27	15.72	14.13	18.42	16.09
90	12.67	19.06	19.73	23.17	11.83	14.34	26.24
120	18.03	29.58	29.45	43.64	19.11	26.86	22.90
150	21.35	23.95	30.99	44.10	26.41	35.88	36.65
180	27.59	34.43	35.22	57.56	34.38	38.02	24.63

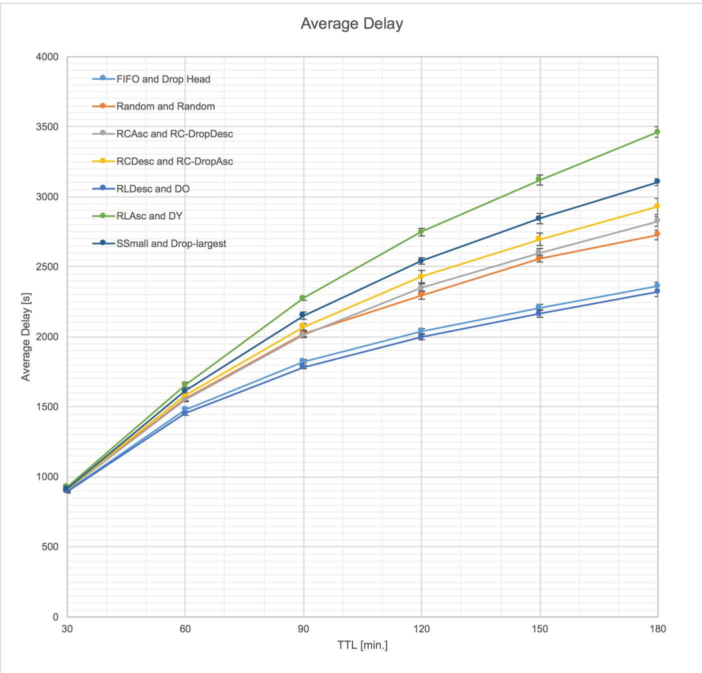


Figure 14 – Average delay for the elementary drop policies.

3.1.2.3 Overhead Ratio

In terms of overhead ratio, as visible in Table 10 and illustrated in Figure 15, the Drop-largest policy obtained the smallest overhead. Since that the largest bundles are the first ones to be dropped when needed, which lowers the average bundle size in the network, a situation is created where the average number of bundles in a node's buffer is larger, which increases the number of delivered bundles – as already verified in section 3.1.2.1 – decreasing the overhead ratio.

Also, as visible in Table 11, and also illustrated by the confidence intervals in Figure 15, the results in this performance metric are reliable, since the 95% confidence intervals are small.

Table 10 – Overhead ratio for the elementary Drop and Scheduling policies.

TTL [min.]	Overhead Ratio							
	FIFO and Drop Head	Random and Random	RCAsc and RC-DropDesc	RCDesc and RC-DropAsc	RLDesc and DO	RLAsc and DY	SSmall and Drop-largest	Small Copies
30	19.00	18.95	19.43	18.43	18.76	18.82	18.40	18.38
60	12.82	12.36	13.37	11.95	12.66	12.17	11.72	11.89
90	11.32	10.67	11.86	10.44	11.25	10.22	9.95	10.38
120	10.78	10.14	11.29	10.01	10.72	9.68	9.26	9.95
150	10.48	9.71	11.03	9.88	10.42	9.35	9.04	9.79
180	10.27	9.52	10.83	9.79	10.21	9.33	8.90	9.73

Table 11 – Overhead ratio 95% confidence intervals for the elementary Drop and Scheduling policies.

TTL [min.]	Overhead Ratio 95% confidence intervals						
	FIFO and Drop Head	Random and Random	RCAsc and RC-DropDesc	RCDesc and RC-DropAsc	RLDesc and DO	RLAsc and DY	SSmall and Drop-largest
30	0.33	0.26	0.36	0.27	0.34	0.22	0.25
60	0.19	0.16	0.19	0.18	0.22	0.16	0.18
90	0.17	0.20	0.20	0.17	0.19	0.14	0.15
120	0.15	0.20	0.19	0.14	0.17	0.14	0.10
150	0.14	0.14	0.17	0.11	0.16	0.14	0.12
180	0.14	0.12	0.17	0.09	0.15	0.11	0.14

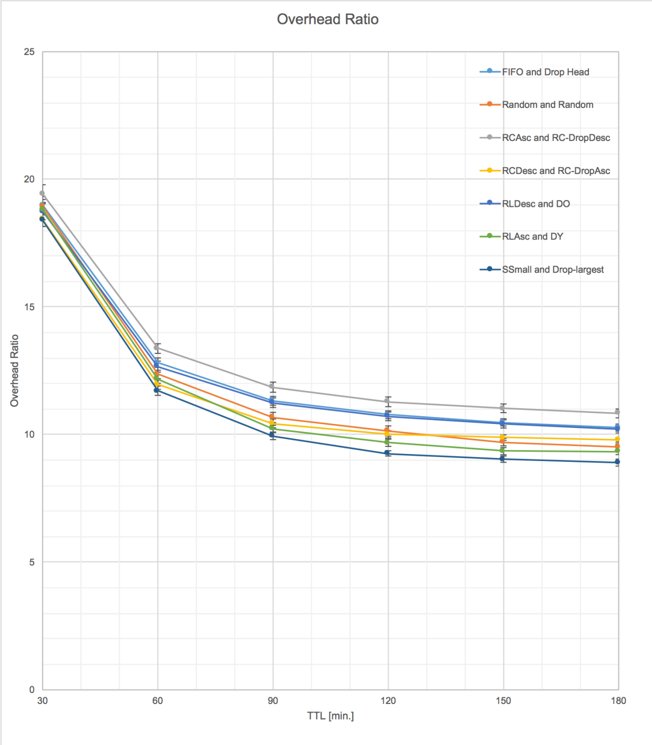


Figure 15 – Overhead ratio for the elementary drop policies.

3.1.3 Simulation conclusions

Generally, every policy which tries to measure the age of a message indirectly, through a related characteristic, or directly, has good results in the performance metrics analysed since an older message in the buffer, generally is expected to be either already in the destination or at least in the buffer of many different nodes and so its drop will not have a major impact.

It is worth mentioning that the receive time of a message is also clearly a parameter to consider since a message that has been in the buffer for a long time has had more opportunities to be replicated and delivered. The size of the bundle has proven to be also a good factor to consider in order to minimize the drop count in every node.

To conclude, it is possible to infer that the most relevant local characteristics of a message for dropping purposes are its RTTL, size and receive time.

3.2 Comparison between Small-Copies and elementary drop policies

In this section, the previously studied policies in section 3.1, will be compared with the Small-Copies (SCop) drop policy. It was thought that the study of a probabilistic estimation approach would be relevant. Hence, in this section, the SCop policy will be compared in the same scenarios with the best performing elementary drop policies.

3.2.1 Simulation scenario

The used settings and parameters were the same as the ones used in 3.1, as specified in Table 4 and Table 5.

To be coherent with the rest of the scenario, a scheduling policy which uses the same metric as the SCop was implemented. In the other elementary drop policies, the respective scheduling policies would prioritize in the sending queue, the opposite type of bundles the drop policy would discard. For instance, Drop-largest would discard the largest sized bundles and SSmall would send first the smallest ones.

The implemented Small Copies scheduling policy is the exact inverse as its drop policy i.e. it firstly sends the bundle with the smallest *P(M)* in the queue.

3.2.2 Simulation Results

After the simulation, the results were processed and organized in the tables and figures below. In this section, the results of the proposed performance metrics described in 2.6.3 will be displayed and analysed.

3.2.2.1 Bundle Delivery Rate

Contrarily to what would be intuitive, SCop did not perform as expected, having a very poor performance in terms of bundle delivery rate when compared with Drop Head, DO and Drop-largest as illustrated in Table 12 and Figure 16.

Furthermore, as visible in Table 13, and also illustrated in the confidence intervals in Figure 16, the results in this performance metric are reliable, since the 95% confidence intervals are quite small.

It would be expected that the SCop behaviour would surpass the DO policy since it considers the RTTL of the bundle and the average number of encounters of every node which holds a replica.

Table 12 – Bundle delivery rate for the best performing elementary drop policies and SCop.

TTL [min.]	Bundle Delivery Rate			
	FIFO and Drop Head	RLDesc and DO	SSmall and Drop-largest	Small Copies
30	24.44%	24.66%	24.30%	23.06%
60	37.56%	37.66%	37.59%	33.65%
90	42.59%	42.43%	43.13%	37.16%
120	44.74%	44.54%	45.46%	38.16%
150	46.01%	45.81%	46.04%	38.48%
180	46.93%	46.73%	46.37%	38.57%

Table 13 – Bundle delivery rate 95% confidence intervals for the best performing elementary drop policies and SCop.

TTL [min.]	Bundle Delivery Rate 95% confidence intervals			
	FIFO and Drop Head	RLDesc and DO	SSmall and Drop-largest	Small Copies
30	0.43%	0.44%	0.34%	0.25%
60	0.57%	0.63%	0.59%	0.63%
90	0.68%	0.70%	0.68%	0.50%
120	0.68%	0.71%	0.56%	0.51%
150	0.67%	0.71%	0.67%	0.58%
180	0.69%	0.68%	0.74%	0.57%

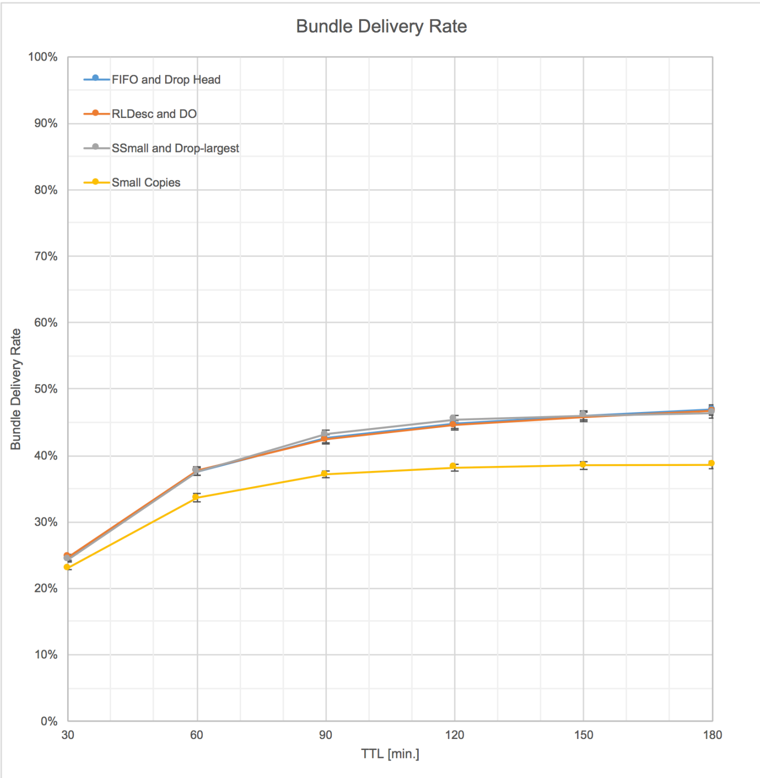


Figure 16 – Bundle delivery rate comparison between SCop and the best performing elementary drop policies.

3.2.2.2 Average Delay

In the case of the average delay performance metric, as visible in Table 14 and Figure 17, the results for SCop were also below the expectations since that it would be expected that by taking into account the RTTL of the bundle the SCop algorithm would be capable to exclude older bundles from its buffer, allowing the younger ones to have an opportunity to be delivered.

Nonetheless, as visible in Table 15, and in the confidence intervals in Figure 17, the results in terms of average delay for each policy are reliable, presenting small 95% confidence intervals.

Table 14 – Average delay for the best performing elementary drop policies and SCop.

TTL [min.]	Average Delay [s]			
	FIFO and Drop Head	RLDesc and DO	SSmall and Drop-largest	Small Copies
30	899.24	895.55	916.78	913.25
60	1479.30	1453.97	1612.23	1586.19
90	1825.60	1784.78	2151.37	2095.07
120	2041.28	1998.48	2542.35	2455.17
150	2210.72	2166.68	2844.33	2730.42
180	2362.95	2321.91	3104.29	2990.66

Table 15 – Average delay 95% confidence intervals for the best performing elementary drop policies and SCop.

TTL [min.]	Average Delay 95% confidence intervals [s]			
	FIFO and Drop Head	RLDesc and DO	SSmall and Drop-largest	Small Copies
30	9.01	10.40	9.88	10.91
60	13.93	14.13	16.09	13.21
90	12.67	11.83	26.24	22.26
120	18.03	19.11	22.90	43.99
150	21.35	26.41	36.65	27.43
180	27.59	34.38	24.63	43.16

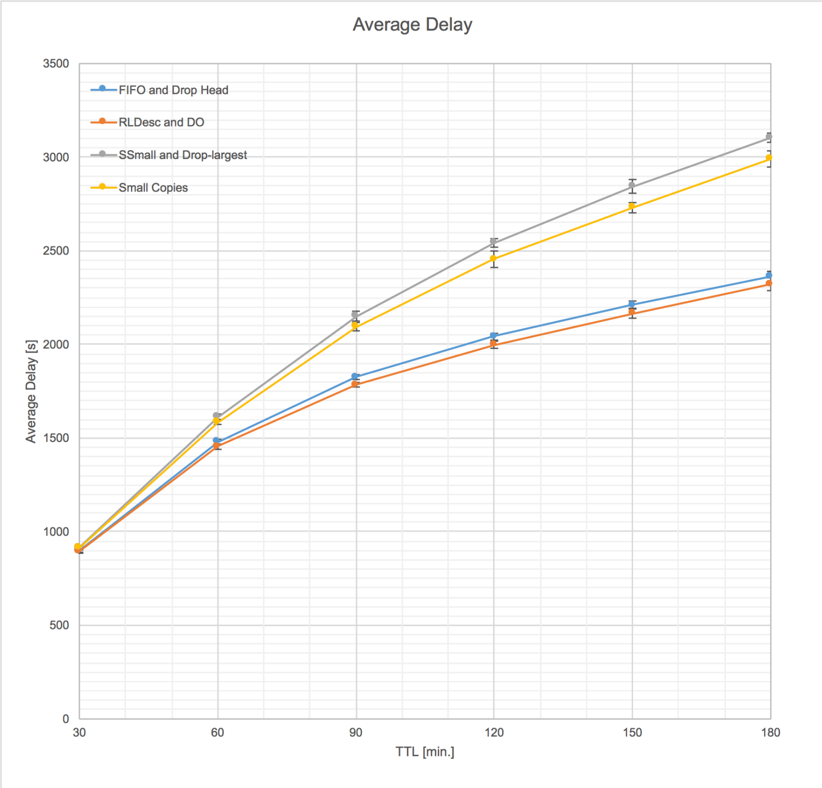


Figure 17 – Average delay comparison between SCop and the best performing elementary drop policies.

3.2.2.3 Overhead ratio

In the case of the overhead ratio performance metric, the results were good when compared to the best performing elementary drop policies as shown in Table 16 and illustrated in Figure 18. This is due to the fact that the bundle delivery rate was small, which implied that the number of delivered bundles was also small, decreasing the overhead ratio.

Moreover, as visible in Table 17, and also in the illustrated intervals for each policy measurement in Figure 18, the results in this performance metric present a high confidence, since the 95% confidence intervals are narrow.

Table 16 – Overhead ratio for the best performing elementary drop policies and SCop.

TTL [min.]	Overhead Ratio			
	FIFO and Drop Head	RLDesc and DO	SSmall and Drop-largest	Small Copies
30	19.00	18.76	18.40	18.38
60	12.82	12.66	11.72	11.89
90	11.32	11.25	9.95	10.38
120	10.78	10.72	9.26	9.95
150	10.48	10.42	9.04	9.79
180	10.27	10.21	8.90	9.73

Table 17 – Overhead ratio 95% confidence intervals for the best performing elementary drop policies and SCop.

TTL [min.]	Overhead Ratio 95% confidence intervals			
	FIFO and Drop Head	RLDesc and DO	SSmall and Drop-largest	Small Copies
30	0.33	0.34	0.25	0.19
60	0.19	0.22	0.18	0.21
90	0.17	0.19	0.15	0.14
120	0.15	0.17	0.10	0.14
150	0.14	0.16	0.12	0.15
180	0.14	0.15	0.14	0.14

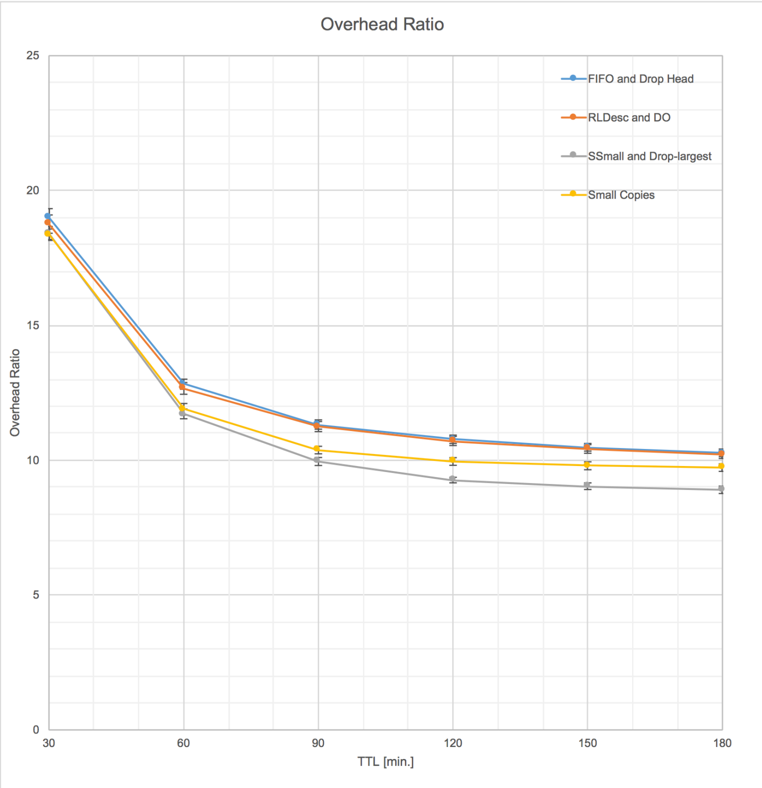


Figure 18 – Overhead ratio comparison between SCop and the best performing elementary drop policies.

3.2.3 Simulation conclusions

Of course, since the difference between SCop and the other policies was not considerable in terms of overhead ratio, it is not optimal in this kind of scenario. Since the results for the SCop, in bundle delivery rate and average delay, were worse in both average delay and bundle delivery rate, it is concluded that the SCop metric does not predict realistically the behaviour of the nodes.

As referred by Kim in [34], the metric used in Scop assumes a Poisson distribution for node encounters, which implies that the encounters should have a random behaviour. This premise does not hold true for an urban movement, in which it is possible to predict the position of a node. This predictability is used in some drop policies such as LEPR and MOPR.

Chapter 4

Largest Bundle's Host Deliverability

In this chapter a description of the main points of improvement of previously analysed policies in Chapter 2 is provided. Afterwards, the Largest Bundle's Hosts Deliverability (LBHD) drop policy algorithm and its main features will be explained in detail, concluding the chapter with simulation results analysis. Moreover, an additional drop policy called One Hop Delivery Estimation Drop (OHDED) will also be explained and tested in detail.

4.1 Major Points of Improvement

LBHD was developed through the major faults of other drop policies. After the analysis of the MOPR and LEPR drop policies, major points of improvement were identified. In this chapter, the proposed drop policy is described. As a start, the major points of improvement of the best performing policies in this category will be listed. Later in section 4.2, the proposed policy will be described in detail, listing the main features of the policy and how the proposed algorithm aims to surpass the points of improvement described in the section 4.1.

4.1.1 Only the nodes which receive a bundle should count

In the LEPR policy, the estimation of the delivery probability of the bundle's hosting node, calculated by PRoPHET, does not consider the number of times a bundle has been forwarded nor the probability of delivery of the new hosts.

Since the information may not be very recent, and the network changes with time, a node may be prioritizing bundles which have been replicated to other nodes with high delivery probability instead of dropping them. Seeing that, tendentiously, a much-replicated bundle has a large possibility to be delivered, and one which was not as much replicated has a lower delivery probability, it would make sense to account only the delivery probability of the nodes which have received the bundle so far.

4.1.2 No Transitivity in MOPR's FP value

In the MOPR drop policy there is no transitivity of the FP value, for instance, when a node receives a new bundle in its buffer, the FP for that bundle is 0, being increased every time this node relays the bundle for other nodes, independently of how many copies the previous nodes have created before.

This effect may be minimized by transferring the FP value with the bundle when it is forwarded to another node.

4.1.3 No update between hosts of the same bundle

In the MOPR drop policy, every node has a FP value for every bundle stored in the buffer. The FP is not updated with information from other nodes that carry the same bundle.

To have the most updated value of a bundle's FP, which may have been more replicated by a node than another, when an encounter occurs, the nodes should trade the FP values for the respective stored bundles, and in the case, that both the nodes carry the same bundle, the bundle's FP value of both nodes should be updated to the largest value between the two.

4.2 LBHD – Algorithm Description

Largest Bundle's Hosts Deliverability (LBHD) is the proposed drop policy, which is intended to have better performance than all the drop policies in section 2.4. It chooses the bundle to be dropped through the delivery probability of all the hosts of a certain bundle replica, updating the metric at every encounter opportunity. This policy was created to be used with routing protocols with delivery probability estimation.

Through some key ideas of LEPR and MOPR, described in [11], such as the use of the estimated delivery probabilities and the sum of the delivery probabilities of all the nodes, respectively, a new metric was developed. By adding the delivery probability of all the bundle hosting nodes and discarding the bundle with the largest sum, we can assure that the discarded bundle has been replied to nodes which have a high probability to deliver the bundle to its destination. The new metric is called Hosts Deliverability (HD) and is stored in a new field in every bundle. The HD is calculated using:

$$HD(t) = HD(t-1) + P_{i \to dest}\,, \qquad (20)$$

in which the $HD(t)$ corresponds to the HD value in the present moment, $HD(t-1)$ to the value previously stored in the buffer, and $P_{i \to dest}$ to the probability of delivery – estimated by the routing protocol – of node i to the destination of the bundle. Node i corresponds to a node that carries the bundle, i.e. every time a bundle is replicated, its HD value is incremented with the delivery probability of the new carrier.

Since it is not possible to infer the delivery probability of a node i, which is going to carry the bundle, to the bundle's destination, this information must be sent by node i – the new carrier – to the sender node, as soon as node i receives the bundle.

It is also important to refer that through (20), as soon as a node receives a new bundle, it updates the HD metric of the bundle with its own delivery probability to the bundle's destination. In this way, when a bundle is replicated, the information of the delivery probability of the sender node is already included in the HD metric.

When there is congestion and there is no buffer space left, the first bundle to be dropped is the one with the highest HD value. Since a bundle with a large HD value has been already replicated to other nodes with good delivery probabilities of that bundle, it is not important to keep this bundle in the buffer. The new bundles in the network and the ones created by the carrying node, which tendentiously have lower HD values, should be prioritized so that these types of bundles have a fair chance of being delivered.

4.2.1 Only the nodes which receive a bundle should count

Using the HD metric, the first point of improvement of 4.1 is addressed, since that only the probabilities of the nodes which actually receive the bundle are considered in the HD metric.

4.2.2 No Transitivity in MOPR's FP value

To improve on the second point listed in section 4.1, it was decided that the HD value should be included in the bundle itself instead of storing it in the node, which would cause the next host to lose information about the previous replication events of the bundle in other hosts.

4.2.3 No update between hosts of the same bundle

Addressing the third and final point of improvement, the nodes must exchange their bundles' HD values to get the most updated HD value for each bundle, and in case there is a match – both nodes have the same bundle buffered – the HD value of both bundles is updated to the largest value between the two, as illustrated in Figure 19, where only bundle B_3 is held by both nodes and the corresponding metric exchanged.

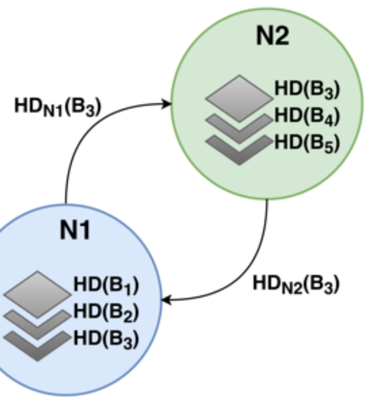


Figure 19 – LBHD's HD metric values exchanged between nodes.

4.3 LBHD and PRoPHET Simulation

In the present section, the LBHD drop policy performance will be tested in the proposed performance metrics when paired with PRoPHET routing protocol. Furthermore, the simulation scenario will be described, and the results will be presented and analysed.

4.3.1 Simulation scenario

The main goal of the scenario was that the DTN would be between some load and congestion. This could only be achieved by using a small buffer space in the nodes, large messages and short message creation interval. The chosen scenario of the simulation is illustrated in Table 18 and Table 19, for the general parameters in Table 18 and the more specific node parameters in Table 19.

Influenced by Soares in [25], the simulations were run varying the TTL of the bundle, which was varied from 30 to 300 minutes in increments of 30 minutes. This allows varying the network load, as a larger TTL results in messages circulating for longer time and a higher network load.

Several simulations with different seeds were performed and the results averaged for statistical confidence. Other values such as the number of nodes, speeds, etc. were dimensioned to create a realistic scenario in terms of movement. There is a total of 156 nodes classified in 4 different groups. The chosen movement model was the Shortest Path Map Based Movement (SPMBM).

Table 18 – LBHD and PRoPHET simulated scenario general configurations.

General Configurations		
Routing Protocol	PRoPHET	
Movement Model	Shortest Path Map Based Movement	
Number of Seeds	12	
Total Time	24 hours	
Warmup Time	10 000 sec.	
Message Size interval	250 kB to 2 MB	
Message Creation Interval	15 to 30 sec.	
Long range radio	Data Speed	4.5 Mbps
	Range	30 m
Highspeed radio	Data Speed	40 Mbps
	Range	10 m

Table 19 – LBHD and PRoPHET simulated scenario node configurations.

Node Configurations		
Car 1	**Number of hosts**	**100**
	Wait Time	5 to 15 min.
	Node Speed	30 to 50 km/h
	Buffer Size	8 MB
	Radio Interfaces	1 Long range radio
Car 2	**Number of hosts**	**50**
	Wait Time	5 to 15 min.
	Node Speed	10 to 50 km/h
	Buffer Size	5 MB
	Radio Interfaces	1 Long range radio
Tram 1	**Number of hosts**	**4**
	Wait Time	10 to 30 sec.
	Node Speed	25 to 35 km/h
	Buffer Size	50 MB
	Radio Interfaces	1 Long range radio
Tram 2	**Number of hosts**	**2**
	Wait Time	10 to 30 sec.
	Node Speed	25 to 35 km/h
	Buffer Size	50 MB
	Radio Interfaces	1 Long range radio 1 Highspeed radio

As explained in [41] by Keranen, SPMBM is based on the simple random map-based model, in which the nodes move to randomly determined positions on the map following the roads defined by the map data. Although, in SPMBM model the nodes do not wander randomly around the map, instead the Dijkstra’s shortest path algorithm is used to calculate shortest paths from the current location to a randomly selected destination.

4.3.2 Simulation Results

After the simulation, the results were processed and organized in the tables and figures below. In this section, the results of the proposed performance metrics described in section 2.6.3 will be displayed and analysed.

4.3.2.1 Bundle Delivery Rate

The results in terms of the bundle delivery rate are presented in the Table 20 and also displayed in Figure 20. Furthermore, by using 12 different seeds in the simulation scenario, as visible in Table 21, and also by the illustrated intervals for each policy measurement in Figure 20, the results in terms of bundle delivery rate may be considered reliable, considering that the 95% confidence intervals are narrow.

For low TTL values, most bundles are dropped before they have a chance of reaching the destination. When the TTL increases, the chances of having enough time to reach the destination increase, improving the delivery rate. However, as TTL increases, bundles also occupy buffer space for longer time, resulting in network congestion and smaller improvements on the delivery rate, or even a drop in the delivery rate.

The DO policy has a good delivery rate performance. By dropping the messages which have the lowest RTTL in the buffer, DO is dropping bundles that had the smallest time to reach their destinations and consequently a low delivery probability.

It is important to refer that the Drop Head, DY and Random policies have poor results for TTLs larger than 60 minutes. In the case of Random for dropping bundles without any criteria; DY for prioritizing older bundles in the network instead of the ones which have recently been created; and Drop Head for dropping only based on the receive time of the bundle, which accomplishes superior results to Random and DY, but is not ideal since other variables may have more impact for a congested network, as is the case of the RTTL, used by DO policy.

The LBHD policy is the one that achieves the highest delivery rates, surpassing Drop Head (standard drop policy in PRoPHET), DO, LEPR and MOPR, by an average of 37%, 3.9%, 14% and 17%, respectively. This is due to the added features and functionalities that none of the other studied policies have.

By trading the information of the probability of delivery of the relayed replicas of the bundles, and updating its value whenever two hosts of the same bundle have a contact opportunity, this policy can better manage a full buffer in congestion scenarios.

As expected, when congestion increases, LBHD delivery rate decreases since it becomes harder to manage the stored bundles.

Table 20 – Bundle Delivery Rate comparison between LBHD and the best performing state of the art policies when paired with the PRoPHET routing protocol.

TTL [min.]	Bundle Delivery Rate						
	Random	Drop Head	DO	DY	MOPR	LEPR	LBHD
30	40.84%	41.44%	45.67%	36.93%	43.08%	42.25%	44.66%
60	46.18%	47.70%	59.64%	36.90%	54.78%	54.69%	61.54%
90	41.25%	43.38%	60.86%	29.82%	57.11%	57.48%	65.64%
120	36.48%	38.92%	60.85%	24.66%	56.90%	58.84%	65.97%
150	32.94%	35.96%	60.83%	21.11%	56.78%	59.47%	65.77%
180	30.67%	33.83%	60.80%	19.13%	55.64%	58.62%	64.25%
210	28.79%	32.42%	60.80%	17.27%	53.67%	56.64%	63.43%
240	27.60%	31.48%	60.80%	15.79%	52.10%	53.73%	62.12%
270	26.51%	30.78%	60.80%	14.35%	49.63%	50.34%	60.93%
300	25.81%	29.76%	60.80%	13.46%	47.18%	46.40%	60.34%

Table 21 – Bundle Delivery Rate 95% confidence intervals for the comparison between LBHD and the best performing state of the art policies when paired with the PRoPHET routing protocol.

TTL [min.]	Bundle Delivery Rate 95% confidence intervals						
	Random	Drop Head	DO	DY	MOPR	LEPR	LBHD
30	0.47%	0.45%	0.54%	0.45%	0.51%	0.47%	0.55%
60	0.42%	0.35%	0.33%	0.36%	0.56%	0.55%	0.65%
90	0.27%	0.27%	0.34%	0.43%	0.24%	0.33%	0.31%
120	0.25%	0.33%	0.40%	0.38%	0.50%	0.32%	0.34%
150	0.27%	0.32%	0.40%	0.29%	0.47%	0.35%	0.31%
180	0.21%	0.35%	0.39%	0.39%	0.42%	0.36%	0.49%
210	0.24%	0.30%	0.39%	0.30%	0.55%	0.35%	0.48%
240	0.22%	0.20%	0.39%	0.19%	0.37%	0.33%	0.51%
270	0.23%	0.23%	0.39%	0.21%	0.54%	0.29%	0.42%
300	0.33%	0.31%	0.39%	0.21%	0.54%	0.35%	0.50%

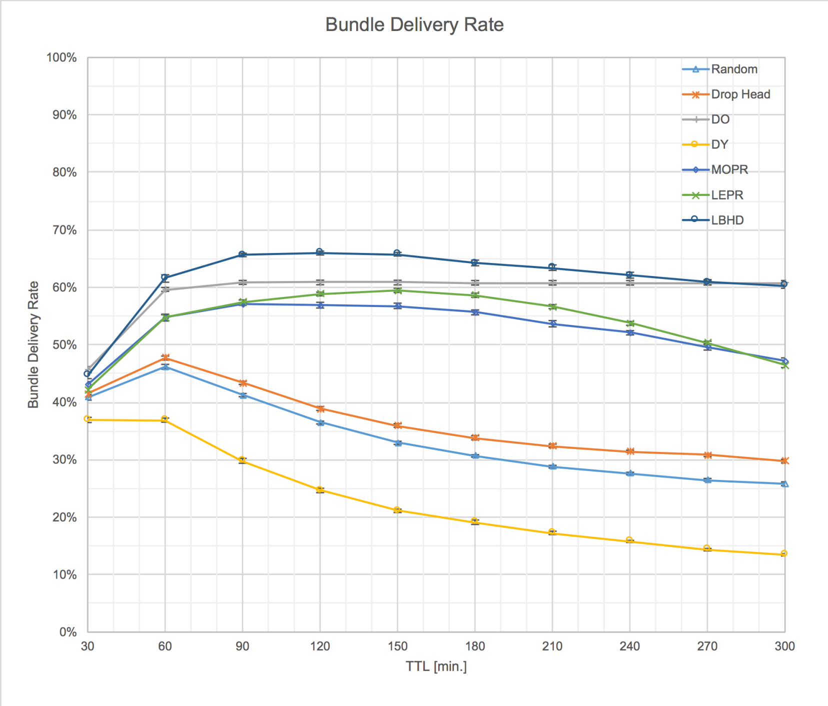


Figure 20 – Bundle delivery rate comparison, varying the TTL of the message, between LBHD and other drop policies when paired with the PRoPHET routing protocol.

4.3.2.2 Average Delay

The average delay is shown in Table 22 and also in Figure 21. As stated in Table 23, and also in the illustrated intervals for each policy measurement in Figure 21, the results in terms of average delay show reliability, by the reason of presenting short 95% confidence intervals.

Table 22 – Average Delay comparison between LBHD and the best performing state of the art policies when paired with the PRoPHET routing protocol.

TTL [min.]	Average Delay [s]						
	Random	Drop Head	DO	DY	MOPR	LEPR	LBHD
30	1083.53	1074.75	1044.50	1114.85	1102.50	1080.36	1075.79
60	1862.78	1830.54	1654.98	1949.20	2039.83	1946.26	1916.40
90	2349.99	2198.55	1773.76	2410.85	2867.00	2650.99	2528.67
120	2601.59	2346.72	1783.84	2699.73	3647.22	3297.35	2981.31
150	2660.37	2377.27	1781.23	2921.82	4424.47	3926.74	3337.76
180	2709.89	2366.55	1782.03	2921.04	5170.77	4547.00	3578.15
210	2687.94	2379.16	1782.03	2923.17	5959.85	5180.69	3783.23
240	2695.13	2339.63	1782.03	2833.82	6664.20	5803.71	3905.71
270	2678.41	2289.54	1782.03	2737.90	7425.15	6439.50	3993.96
300	2611.50	2290.11	1782.03	2642.21	8213.77	7061.09	4096.80

The proposed LBHD policy performs better than LEPR and MOPR, by an average of 26% and 34%, respectively. The default Drop Head policy performs an average of 21% better than LBHD. The best policy, in terms of latency, is DO, achieving an average delay reduction of 45% in comparison with LBHD. This is due to the fact that the oldest bundles of each buffer are discarded and so, the probability of a bundle being delivered in the end of its life is small, which lowers the average delay in the network.

On the other hand, LBHD has a higher Delivery Rate, since it manages to hold older bundles which may be delivered in the end of their life increasing the average delay. This is the price to pay for a better delivery rate.

Table 23 –Average Delay 95% confidence intervals for the comparison between LBHD and the best performing state of the art policies when paired with the PRoPHET routing protocol.

TTL [min.]	Average Delay 95% confidence intervals [s]						
	Random	Drop Head	DO	DY	MOPR	LEPR	LBHD
30	4.78	6.70	6.84	6.51	7.19	6.68	5.48
60	14.78	17.89	11.71	16.43	15.24	15.81	10.25
90	23.27	23.73	15.35	34.31	25.09	13.59	14.66
120	27.86	32.76	12.85	62.04	27.79	21.56	18.31
150	24.83	22.87	13.15	55.87	23.89	28.67	22.37
180	31.36	32.72	13.99	76.45	34.53	33.88	34.69
210	38.06	23.72	13.99	80.62	52.50	40.25	41.47
240	45.61	51.63	13.99	94.05	62.32	56.60	27.67
270	53.58	41.46	13.99	76.58	44.48	78.22	37.54
300	37.82	31.46	13.99	89.91	68.38	78.85	51.62

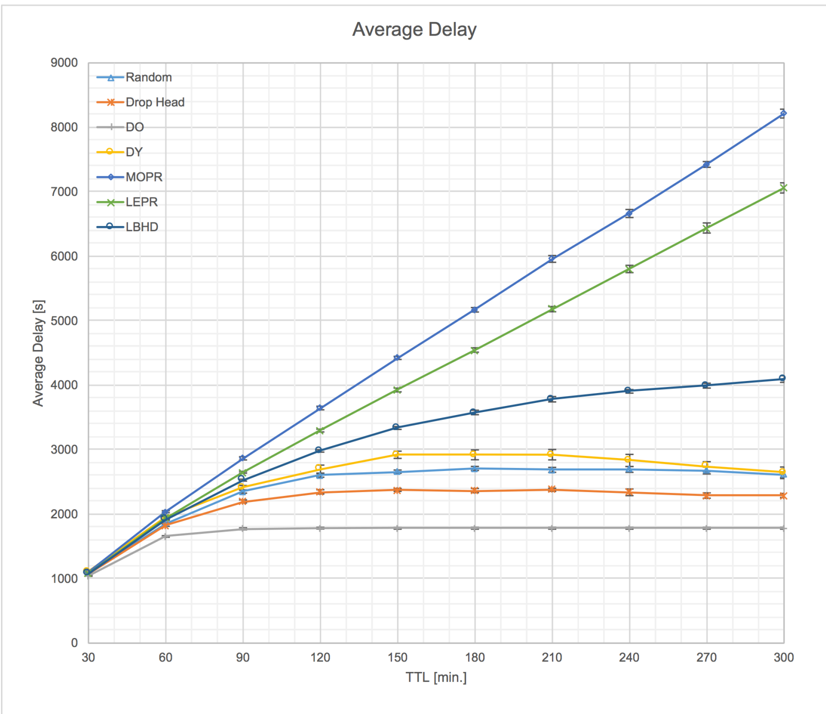


Figure 21 – Average delay comparison in seconds, varying the TTL of the message, between LBHD and other drop policies when paired with the PRoPHET routing protocol.

4.3.2.3 Overhead Ratio

The results for the overhead ratio are presented in Table 24 and in Figure 22. Moreover, as visible in Table 25, and also illustrated in Figure 22 by the interval bars for each point, the results in terms of overhead ratio are considered reliable, due to presenting narrow 95% confidence intervals.

LBHD behaves well for large TTL values, although the results of MOPR throughout all the simulated TTLs are better.

For smaller TTL values, since the congestion is not significant, LBHD allows more replicas of the bundles to be spread throughout the network.

For larger TTL values, since the network is highly congested, LBHD constrains the created number of replicas of the bundles, since it tendentiously prioritizes bundles with lower number of replicas. Thus, the overhead does not increase significantly for larger TTL values.

Table 24 – Overhead ratio comparison between LBHD and the best performing state of the art policies when paired with the PRoPHET routing protocol.

TTL [min.]	Overhead Ratio						
	Random	Drop Head	DO	DY	MOPR	LEPR	LBHD
30	63.20	60.38	97.75	72.28	55.39	87.03	81.02
60	86.55	79.43	162.79	122.17	64.36	148.56	123.43
90	112.81	99.33	171.04	182.03	65.76	174.44	136.92
120	136.28	118.27	171.22	236.30	60.47	179.48	144.00
150	157.32	132.24	171.32	281.43	53.67	176.53	146.77
180	174.69	145.21	171.34	311.24	48.31	173.21	151.33
210	190.91	155.83	171.34	343.71	42.38	169.67	153.49
240	202.48	161.29	171.34	365.66	37.76	165.53	156.60
270	211.02	165.28	171.34	379.76	33.86	161.58	159.18
300	217.79	171.26	171.34	390.26	30.17	159.57	160.89

Table 25 – Overhead ratio 95% confidence intervals for the comparison between LBHD and best performing state of the art policies when paired with the PRoPHET routing protocol.

TTL [min.]	Overhead Ratio 95% confidence intervals						
	Random	Drop Head	DO	DY	MOPR	LEPR	LBHD
30	0.82	0.81	1.91	1.14	0.74	1.41	1.41
60	0.76	0.75	1.46	1.66	0.91	1.96	1.50
90	1.34	0.90	1.81	4.70	0.96	1.44	1.26
120	1.70	2.01	2.03	4.84	0.78	1.93	1.47
150	1.43	1.96	2.36	5.63	1.27	1.83	1.89
180	2.34	2.44	2.06	6.33	0.76	2.26	2.03
210	3.67	2.76	2.06	7.34	0.75	2.03	2.04
240	2.79	1.78	2.06	10.72	0.56	1.92	1.52
270	2.06	2.38	2.06	11.73	0.47	1.48	2.44
300	3.02	2.67	2.06	10.73	0.45	1.64	1.83

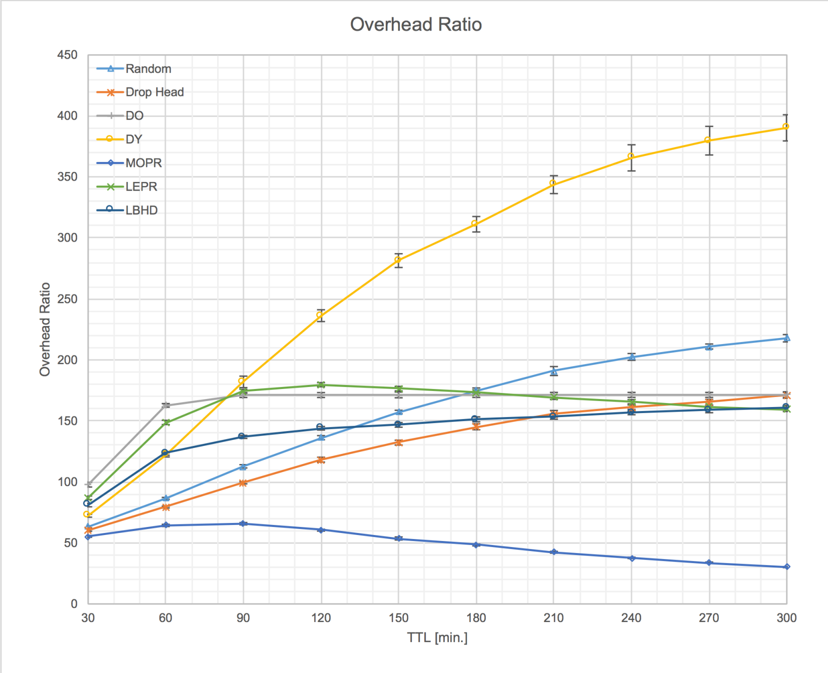


Figure 22 – Overhead ratio, varying the TTL of the message, between LBHD and other drop policies when paired with PRoPHET routing protocol.

4.3.3 Simulation conclusions

This thesis proposed a new LBHD drop policy that in congestion situations drops the bundles with the highest estimated delivery probability, since the probability of another carrying node delivering the bundle is high.

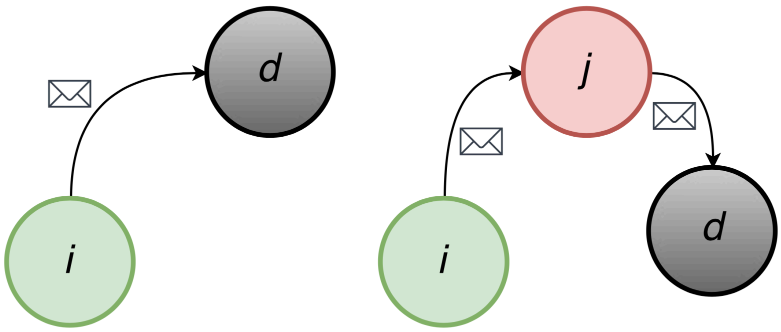
The LBHD policy achieved the best delivery rates, as compared with other existing drop policies, for a wide range of network loads, ranging from low congestion to high congestion. For low congestion, the overhead slightly increases, as more bundles are allowed to exist in the network. But as congestion increases, the overhead increase is minimal, as LBHD effectively controls congestion. As regards the average delay, the improvement of the delivery rate under congestion comes with the price of an increased average delay, as the additional bundles that are delivered take the longest time to reach their destinations due to congestion and would be dropped with other dropping policies.

4.4 One Hop Delivery Estimation Drop

The One Hop Delivery Estimation Drop (OHDED) policy was also developed to be used with the MaxProp routing protocol, since it exploits its table of encounter probabilities. The OHDED policy was developed as an improvement to the used Drop Max Hops in MaxProp.

Since that in the PRoPHET protocol, each node possesses a table with the encounter probability between every node in the network, it was thought that this information would be interesting to use to predict which bundles should be dropped, and which ones should be prioritized.

Once a buffer is full, the OHDED algorithm computes the Bundle's Delivery Probability (BDP) metric which takes into consideration the scenarios represented in Figure 23, in which it is considered the probability of direct delivery of the bundle to its destination node and also the probability of its delivery to one certain bundle in the network which finally delivers it to its destination node.



(a) (b)

Figure 23 – Considered situations of a bundle delivery in OHDED drop policy algorithm: (a) direct delivery, 1 hop; (b) one intermediate node, 2 hops.

As represented in the equation

$$BDP = P_{i \to d} + \sum_{\substack{j=1 \\ j \neq i,d}}^{N} P_{i \to j} P_{j \to d} \,, \tag{21}$$

in which BDP represents the BDP metric, i the present node with its buffer full, d the bundle's destination node, j one node in the network different than j and d, and N the total number of nodes in the network. The variables: $P_{i \to d}, P_{i \to j}$ and $P_{j \to d}$, correspond, respectively, to the encounter probability, estimated by MaxProp: of the present node i with the destination node d; of the present node i with other node in the network j; and of the node j with the bundle destination node d.

After the BDP is computed the first chosen bundle to be dropped is the one whose BDP is the lowest. In this way, the node will only maintain in the buffer: bundles which it has a substantial probability to deliver directly; or bundles to replicate to another node, which the present node has a large encounter probability, and the other node has also a high encounter probability with the bundle's destination.

4.5 LBHD and OHDED paired with MaxProp

In this section, the LBHD and OHDED drop policies will be simulated with the MaxProp routing protocol. Some adaptations had to be made in order to create compatibility between LBHD and MaxProp. In the following section, it will be explained in detail every adaptation made to the LBHD policy. Afterwards, the simulation settings and parameters will be discussed and analysed.

It was decided to simulate the LBHD policy using the MaxProp routing protocol, as specified in Table 26, to test its applicability in terms of performance in another delivery probability prediction routing protocols.

4.5.1 LBHD Algorithm adaptation

For the MaxProp algorithm an adaptation of the used metric was made. Since that LBHD's HD metric depends on the probability of delivery of a node calculated by the PRoPHET routing protocol, an adapted approach for the MaxProp routing protocol was used.

In the MaxProp protocol, the cost of a bundle being delivered by any node in the network is computed, as previously mentioned. Since that every node can compute the cost of a node delivering a bundle to its destination, the HD metric in (20) was updated to

$$HD(t) = HD(t-1) + Cost_{i \rightarrow dest} \,, \tag{22}$$

in which the $HD(t)$ corresponds to the HD value in the present moment, $HD(t-1)$ to the value previously stored in the buffer, and $Cost_{i \rightarrow dest}$ to the cost of delivery – estimated by the MaxProp routing protocol – of node i to the destination of the bundle. Node i corresponds to a node that carries the bundle, i.e. every time a bundle is replicated, its HD value is incremented with the cost of delivery of the new carrier.

In this way, by dropping the bundle with the largest HD value, the dropped bundle is one that has several replicas already in the network, which has low impact in the delivery probability.

The rest of the LBHD drop policy features were maintained.

4.5.2 Simulation scenario

The chosen simulation scenario is similar to the one described in section 4.3.1. The chosen general parameters and settings of the simulation scenario are described in Table 26, and the more specific node configurations set is described in Table 27.

As previously mentioned, congestion was forced through the buffer size, message size interval and message creation interval parameters, to create a scenario where the drop policies have an intensive use and consequently affect the performance metrics' results.

The chosen drop policies to be compared with the LBHD drop policy, in this scenario, were Drop Head, Drop Oldest, Drop Most Hops and OHDED. Since that the Drop Most Hops policy is the default drop policy in the MaxProp routing protocol, it was logical to compare the proposed drop policy with the standard one to understand if the LBHD can improve the performance of MaxProp as it can with PRoPHET.

Table 26 – LBHD/OHDED and MaxProp simulated scenario general configurations.

General Configurations		
Routing Protocol	MaxProp	
Movement Model	Shortest Path Map Based Movement	
Number of Seeds	12	
Total Time	24 hours	
Warmup Time	10 000 sec.	
Message Size interval	250 kB to 2 MB	
Message Creation Interval	15 to 30 sec.	
Long range radio	Data Speed	4.5 Mbps
	Range	30 m
Highspeed radio	Data Speed	40 Mbps
	Range	10 m

Table 27 – LBHD/OHDED and MaxProp simulated scenario node configurations.

Node Configurations		
Car 1	**Number of hosts**	**100**
	Wait Time	5 to 15 min.
	Node Speed	30 to 50 km/h
	Buffer Size	8 MB
	Radio Interfaces	1 Long range radio
Car 2	**Number of hosts**	**50**
	Wait Time	5 to 15 min.
	Node Speed	10 to 50 km/h
	Buffer Size	5 MB
	Radio Interfaces	1 Long range radio
Tram 1	**Number of hosts**	**4**
	Wait Time	10 to 30 sec.
	Node Speed	25 to 35 km/h
	Buffer Size	50 MB
	Radio Interfaces	1 Long range radio
Tram 2	**Number of hosts**	**2**
	Wait Time	10 to 30 sec.
	Node Speed	25 to 35 km/h
	Buffer Size	50 MB
	Radio Interfaces	1 Long range radio 1 Highspeed radio

4.5.3 Results

Having the settings and simulation parameters set, the results of the performance metrics proposed in section 2.6.3 will be displayed and analysed.

4.5.3.1 Bundle Delivery Rate

In terms of the bundle delivery rate performance metric, as visible in Table 28 and illustrated in Figure 24, the LBHD policy had the best performance in the lower TTL values, being surpassed by OHDED and Drop Most Hops policy for larger TTL values. The OHDED policy performed better than Drop Head, DO, Drop Most Hops and LBHD by an average of 17%, 11%, 1% and 2%, respectively. Although the difference between the Drop Most Hops, OHDED and LBHD in this performance metric is not considerable for TTLs larger than 90 minutes, the best performing policy, for larger TTLs is OHDED.

The LBHD policy performs better than the Drop Head policy (default drop policy in most of the routing protocols), by an average of 15%. Also, LBHD has a gain of 9%, 0% and –2% when compared with DO, Drop Most Hops and OHDED, respectively. The results show that LBHD has a similar performance to the standard drop policy in MaxProp routing protocol – Drop Most Hops, while having superior results when compared with the other state of the art policies. Also, the Drop Most Hops policy good performance is related to the fact that by eliminating bundles – which have been replicated many times already – beyond tendentiously having a smaller RTTL, have already had multiple opportunities of being replicated thus, will not have a large impact in the bundle delivery rate.

The remaining two drop policies – Drop Head (FIFO) and DO – had the worst results since both policies do not use any of the delivery probabilities estimated by MaxProp. DO had a better performance than Drop Head since that, as already stated previously, the RTTL of the bundle is a very important parameter, which is not considered in the Drop Head drop policy.

In addition, as shown in Table 29 and illustrated in Figure 24, the results in terms of bundle delivery rate may be taken as reliable, considering the very small 95% confidence intervals shown.

Table 28 – Bundle delivery rate comparison between LBHD and other drop policies compatible with MaxProp routing protocol.

TTL [min.]	Bundle delivery rate				
	Drop Head	DO	Drop Most Hops	OHDED	LBHD
30	55.24%	59.36%	60.12%	59.55%	62.21%
60	65.81%	69.92%	77.53%	77.80%	79.47%
90	68.83%	74.65%	84.51%	85.83%	85.07%
120	70.16%	76.78%	88.01%	89.83%	87.73%
150	70.91%	77.97%	89.99%	92.32%	88.27%
180	71.50%	78.62%	91.05%	93.29%	89.04%
210	71.70%	78.72%	92.05%	94.18%	89.52%

Table 29 – Bundle delivery rate 95% confidence intervals for the comparison between LBHD and other drop policies compatible with MaxProp routing protocol.

TTL [min.]	Bundle delivery rate 95% confidence intervals				
	Drop Head	DO	Drop Most Hops	OHDED	LBHD
30	0.50%	0.52%	0.54%	0.60%	0.38%
60	0.41%	0.41%	0.50%	0.43%	0.45%
90	0.49%	0.43%	0.45%	0.43%	0.28%
120	0.31%	0.52%	0.33%	0.33%	0.50%
150	0.50%	0.57%	0.43%	0.32%	0.72%
180	0.51%	0.66%	0.26%	0.27%	0.45%
210	0.56%	0.54%	0.22%	0.28%	0.43%

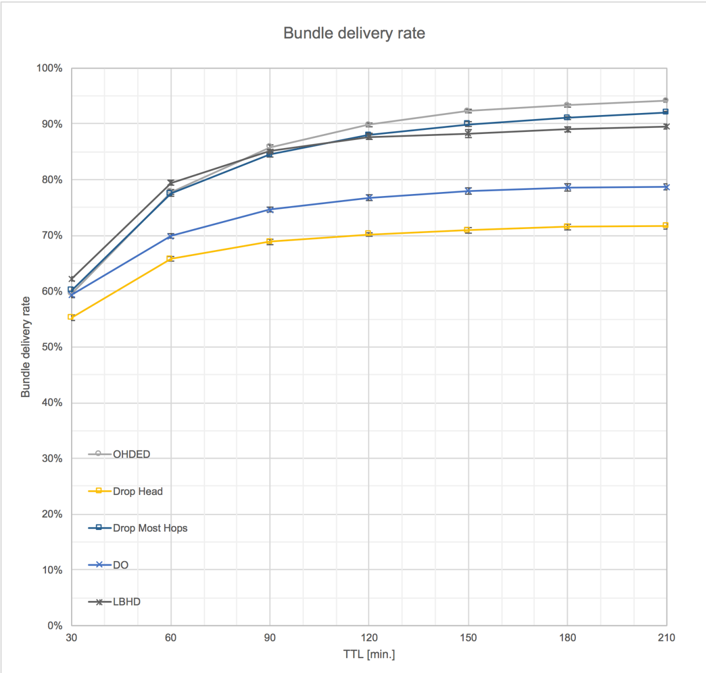


Figure 24 – Bundle delivery rate, varying the TTL of the message, between LBHD and other drop policies when paired with MaxProp routing protocol.

4.5.3.2 Average Delay

Analysing the performance in terms of average delay, detailed in Table 30 and illustrated in Figure 25, it was inferred that the best performing policy was the DO. The good performance in terms of average delay of the DO policy is related to its algorithm, in which the smaller RTTL bundles are the first dropped ones, which decreases the probability of delivering a bundle which has been in the network for a long time hence, the average delay decreases, but the delivery rate also decreases.

The Drop Head and LBHD drop policies also had a good performance in terms of average delay. By dropping the firstly stored bundle, tendentiously, the dropped bundles will also have a small RTTL.

In terms of average delay, LBHD drop policy performed better than Drop Head, DO, Drop Most Hops and OHDED policies by an average of –7%, –22%, 25% and 27%, respectively. Since that, in LBHD every time a bundle is replicated, the HD metric is incremented, being dropped the bundle which has the largest value, implicitly the bundles which have been in the network longest, having lower RTTLs, will be dropped.

The Drop Most Hops and OHDED drop policies show the worst performance in the simulated scenario, this is related to the fact that neither of the policies consider nor measure the RTTL of the bundle, which increases the probability of delivering a bundle in the end of its lifecycle.

Also, as visible in Table 31, and shown by the narrow 95% confidence intervals for each drop policy measurement in Figure 25, the results in terms of average delay may be considered reliable.

Table 30 – Average delay comparison, in seconds, between LBHD and other drop policies compatible with MaxProp routing protocol.

TTL [min.]	Average Delay [s]				
	Drop Head	DO	Drop Most Hops	OHDED	LBHD
30	972.96	869.49	1027.38	1029.52	982.37
60	1416.94	1148.73	1724.62	1732.48	1472.95
90	1606.95	1365.63	2184.65	2217.35	1709.37
120	1727.86	1521.46	2505.05	2597.04	1854.63
150	1812.05	1620.11	2741.93	2862.58	1968.77
180	1853.41	1674.86	2914.42	3068.51	2047.56
210	1894.44	1712.51	3003.05	3169.84	2085.88

Table 31 – Average delay 95% confidence intervals for the comparison, in seconds, between LBHD and other drop policies compatible with MaxProp routing protocol.

TTL [min.]	Average Delay 95% confidence intervals [s]				
	Drop Head	DO	Drop Most Hops	OHDED	LBHD
30	6.94	4.93	5.28	5.89	4.67
60	14.56	8.99	10.16	17.04	7.06
90	13.69	13.84	23.84	22.55	14.36
120	19.92	21.87	26.42	24.99	17.49
150	13.03	22.50	21.60	37.14	21.55
180	20.08	23.87	34.42	52.02	24.24
210	30.23	25.12	39.16	56.36	23.91

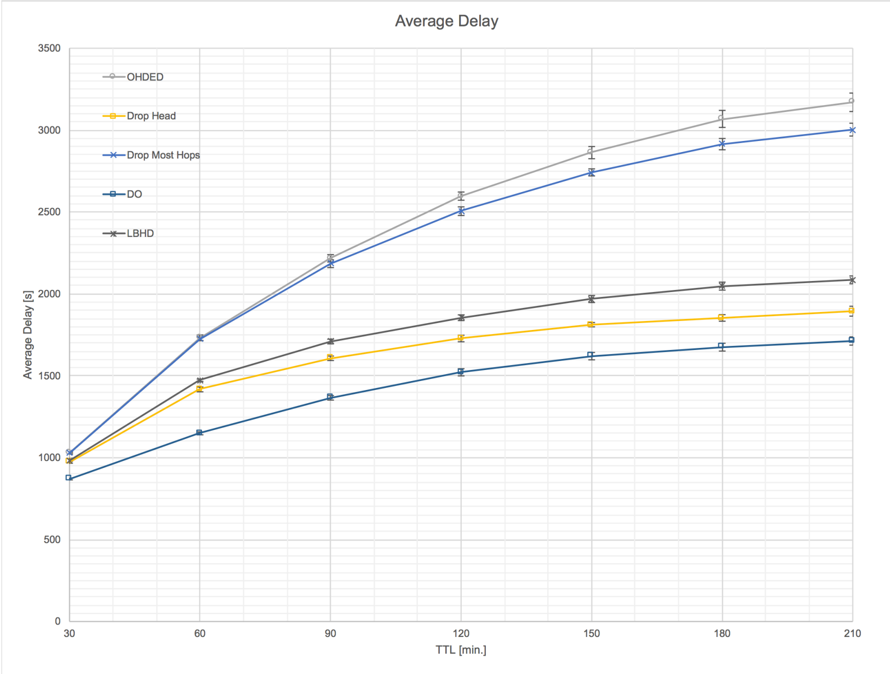


Figure 25 – Average delay in seconds, varying the TTL of the message, between LBHD and other drop policies when paired with MaxProp routing protocol.

4.5.3.3 Overhead Ratio

In what regards the third and last performance metric, overhead ratio, the OHDED, Drop Most Hops and LBHD policy had the best performance in the simulation as visible through Table 32, and illustrated in Figure 26.

As regards the average gains of LBHD policy when compared to Drop Head, DO, Drop Most Hops and OHDED these were of 20%, 11%, 0% and –2%, respectively. Hence, we may conclude that the performance of LBHD was equivalent to the Drop Most Hops policy. Since the network is highly congested, LBHD constrains the created number of replicas of the bundles, since it tendentiously prioritizes bundles with lower number of replicas which explains the good performance in terms of overhead.

Furthermore, since that for TTLs larger than 90 minutes, both OHDED and Drop Most Hops achieve better performance results in terms of bundle delivery rate when compared to LBHD, it was expected that – bearing in mind equation (17) – they both would have better performance in terms of the overhead ratio. OHDED drop policy achieves an average gain of 21%, 13%, 2% and 2% in performance when compared to Drop Head, DO, Drop Most Hops and LBHD, respectively. Which implies that OHDED achieved the best performance in terms of overhead ratio.

Additionally, 95% confidence intervals measurements were presented in Table 33, and illustrated in Figure 26. Since the intervals show low values, the results in terms of overhead ratio are reliable.

Table 32 – Overhead ratio comparison between LBHD and other drop policies compatible with MaxProp routing protocol.

TTL [min.]	Overhead Ratio				
	Drop Head	DO	Drop Most Hops	OHDED	LBHD
30	77.15	71.29	70.91	71.22	67.75
60	66.18	61.28	55.64	55.31	53.62
90	63.15	57.27	50.40	49.39	49.87
120	61.62	55.59	48.01	46.72	48.30
150	60.98	54.78	46.67	45.28	47.99
180	60.44	54.23	45.94	44.42	47.58
210	60.34	54.23	45.36	43.97	47.28

Table 33 – Overhead ratio 95% confidence intervals for the comparison between LBHD and other drop policies compatible with MaxProp routing protocol.

TTL [min.]	Overhead Ratio 95% confidence intervals				
	Drop Head	DO	Drop Most Hops	OHDED	LBHD
30	0.83	0.67	0.70	0.83	0.38
60	0.62	0.47	0.44	0.36	0.35
90	0.61	0.43	0.32	0.33	0.17
120	0.49	0.48	0.33	0.23	0.35
150	0.65	0.49	0.30	0.19	0.48
180	0.52	0.57	0.29	0.19	0.24
210	0.62	0.48	0.22	0.13	0.32

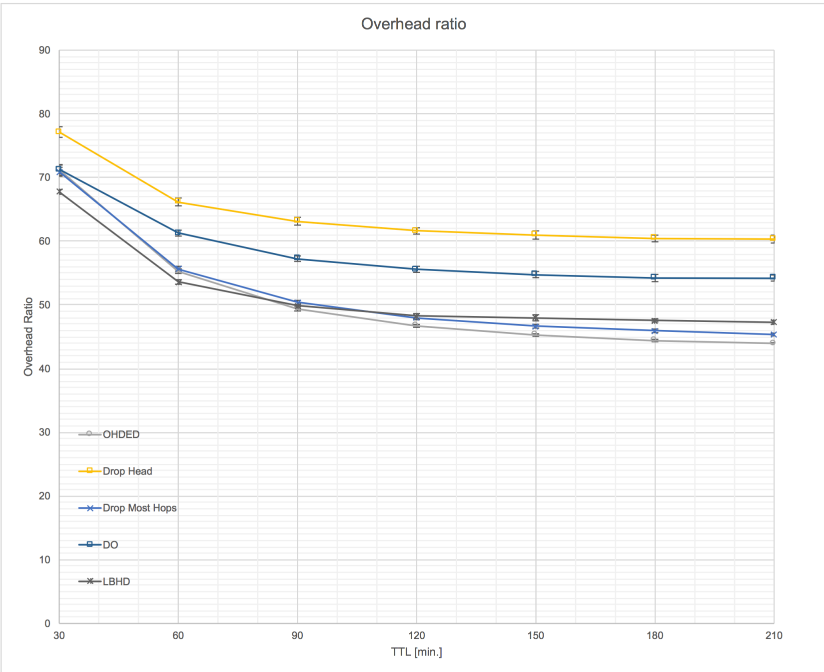


Figure 26 – Overhead ratio, varying the TTL of the message, between LBHD and other drop policies when paired with MaxProp routing protocol.

4.5.4 Simulation conclusions

For the proposed LBHD and OHDED drop policies, various simulations were run, in which they both had a good performance.

Since OHDED drop policy tends to drop the least probable bundle to be delivered by its carrying node either in direct delivery or doing two hops, it had the most successful results in terms of bundle delivery rate and overhead ratio, for larger TTLs. Considering that the number of delivered bundles is accounted in equation (17), by achieving the largest delivery rates, OHDED is able to decrease the overhead ratio.

Nevertheless, LBHD obtained the best results in terms of bundle delivery rate and overhead ratio in lower congestion scenarios – lower TTLs – also, in terms of average delay, its performance was better than the performance of the two other best policies (Drop Most Hops and OHDED) in all the simulated scenarios. Bearing in mind its results in terms of bundle delivery rate and overhead ratio for larger TTLs, and taking in consideration the excellent performance in terms of average delay, when compared to Drop Most Hops and OHDED, we may classify LBHD as the most efficient policy in all the performance metrics. Since the average gain in performance in average delay, when compared to Drop Most Hops, was of 25% and in the other performance metrics the gain was of 0%, it is concluded that the LBHD policy improves the Drop Most Hops policy – the standard drop policy in MaxProp routing protocol.

Chapter 5

Conclusions

In this chapter, a summary of the main conclusions of this master thesis is provided, as well as a few aspects to be developed in future work.

The main goal of this master thesis was to develop a new drop policy to be paired with delivery prediction routing protocols, taking advantage of the delivery prediction feature in order to achieve, in congestion scenarios, an increase in the bundle delivery rate and a decrease in the average delay and overhead ratio. In the direction of the proposed goal, the work of the various chapters will be shortly briefed.

In the first chapter, it is provided the master thesis' background to introduce the reader to the main DTN applications, research history and purpose. Moreover, it is also provided the motivation of the written work as well as a small description of the document's structure.

Chapter 2 provides a background on DTN fundamental concepts, main areas of application and the main challenges for the scientific community. Additionally, it is provided the analysis of all the routing protocols, dropping and scheduling polices used in the work's development and results. Also, a small overview on the main congestion control mechanisms and used tools is given. It was provided also a detailed explanation of the mainly used routing protocols in the simulations, such as MaxProp and PRoPHET. This is due to the work's main goal, which was to develop a drop policy that would have a good performance in this type of protocols. Furthermore, there was a detailed explanation on the main congestion mechanisms and scheduling policies which were used either by default in the routing protocols or imposed for each simulation. Furthermore, the mainly used drop policies in the state of the art were also explained in detail according to each one's potentialities in terms of results. Policies such as Drop Head, DO, Drop-largest, MOPR, LEPR and SCop should be highlighted, since they were the main focus of the work in terms of results comparison.

Also in the second chapter, a small overview on the main congestion control mechanisms and used tools is made. The ONE simulator was the chosen simulation tool, for its open configuration capabilities in the implementation of protocols, policies, and also scenarios and reports.

In Chapter 3, the preliminary tests were made. These tests aimed to simulate and test the elementary polices in similar conditions, in order to obtain intuition of which of the main bundle's characteristics such as the time it was buffered, its RTTL, its size and its number of replicas in the network are the most important when choosing a bundle to be dropped.

After simulating, the main conclusions were that policies such as DO or policies which, using a related characteristic such as the time the bundle was buffered, which also measures the bundle's RTTL, have good results in terms of bundle delivery rate. Bearing in mind that an older bundle in the buffer is generally expected to be either already in its destination, or at least in the buffer of many nodes, and so its drop will not have a major impact.

Another bundle characteristic which was found to have an impact in the chosen performance metrics, was the bundle's size. As seen, Drop-largest policy simulation results showed an increase in the number of deliveries. On the other hand, since that in Drop-largest drop policy, the prioritized bundles are the smallest, this creates a network in which the smaller bundles have a larger probability to be delivered when compared to the larger bundles, which is not ideal in a real network environment.

All in all, it was possible to infer that the most relevant local characteristics of a message for dropping purposes are its RTTL, size and receive time.

Furthermore, it was also studied the performance of the SCop drop policy, which was thought to deliver better performance when compared with the best elementary policies. However, for all the measured performance metrics, the SCop policy achieved worse performance than DO, Drop Head and Drop-largest, which implied that the SCop's metric did not accurately predict the node's behaviour, which is due to the Poisson distribution use for the calculation, which implied that the encounters had a random behaviour.

In Chapter 4 was where the LBHD policy was introduced and tested against the previously described drop policies. Firstly, the major points of improvement in policies such as LEPR and MOPR were listed.

In the LEPR policy, the bundles are ordered by the delivery probability estimation done in protocols such as PRoPHET. Since this probability does not consider the number of nodes the bundles were forwarded to and their probability of delivering the bundle, it may end up in a situation where a bundle, created by the carrying node, is dropped before ever being replicated.

In addition, although the MOPR drop policy took into account the number of times a bundle was replicated in a node, and the delivery probability of each of the new bundle holders have, it would not send this information for the bundle's receiving nodes. This creates situations where newly received several times replicated bundles are prioritized in relation to the already stored bundles, even if the stored ones were less replicated, or replicated to nodes with a small delivery probability. Also, it was seen that it would be logical to update the MOPR's FP metric among other nodes in the network.

The LBHD drop policy algorithm chooses to drop the bundle with the largest HD value, stored in a bundle's field – the HD's initial value is 0 – and it is incremented with the probability of delivery of a new carrier node, every time the bundle is replicated. Note that, as soon as a node receives a new bundle, the HD value is incremented with the node's delivery probability.

The LBHD policy, mainly achieved all the described points of improvement. In the HD metric, only nodes which receive the bundle are considered since that HD is incremented with the probability of all the nodes which the bundle was replicated to. Furthermore, since the HD metric is one field in the bundle itself, when a node receives a bundle, its HD value is updated with the delivery probabilities of all the nodes to which the bundle was replicated to in the past node. This achieves the second point of improvement. Additionally, when an encounter occurs, if the nodes have replicas of the same bundle stored in the buffer, they exchange the HD fields of the common bundles. If one node has a larger HD value than the other for the same bundle, these values are updated to the largest HD between the two, increasing the knowledge of the bundle's history in the network.

The LBHD drop policy when paired with the PRoPHET routing protocol, achieved the best delivery rates, when compared with other existing drop policies. For lower congestion tests, the overhead slightly increased, since a larger number of bundles were allowed to exist in the network. However, for larger congestion scenarios, the overhead increase is not significant, allowing the conclusion that LBHD effectively controls congestion.

In terms of average delay, the improvement of the delivery rate in congested scenarios, comes with the price of an increased average delay, considering that the additional bundles that are delivered, take longer to reach their destinations due to congestion and would be dropped with other dropping policies. The LBHD policy achieved an average delivery rate 37% better than the default Drop Head (FIFO) policy, and an average of 3.9%, 14% and 17% better than DO, LEPR and MOPR, respectively.

In the Chapter 4, the OHDED drop policy was also proposed, this policy considers either the delivery probability of a bundle being directly delivered to its destination and of the bundle being delivered in two hops. By removing firstly, the bundles which have the least delivery probability, we are prioritizing bundles which have the largest probability of being delivered.

Also, an adaptation of the LBHD's algorithm was made, in order to take advantage of the MaxProp routing protocol, in which the metric would use the cost of delivering a bundle instead of the delivery probability, taking advantage of the additional information present in a node.

When paired with MaxProp, both proposed drop policies – OHDED and LBHD – had an excellent performance. It is important to refer that LBHD was the best performing policy especially in low congestion scenarios, having the best results in all the performance metrics. For larger TTLs, in terms of bundle delivery rate and overhead ratio, OHDED achieved the best results among all the simulated drop policies. However, taking also into account the average delay performance metric, the most consistent policy in all the performance metrics was LBHD. Bearing in mind that LBHD had a similar performance to the standard MaxProp drop policy (Drop Most Hops) in terms of bundle delivery rate and overhead ratio, and since that LBHD has an average gain of 25% when compared to Drop Most Hops, the proposed policy may be considered the most efficient one.

In brief, this thesis proposes a new LBHD drop policy, which drops the bundles with the highest estimated delivery probability in congestion situations, as the probability of another carrying node delivering the bundle is high. It is also proposed an OHDED drop policy that when paired with MaxProp routing protocol, achieves not only the lowest overhead ratios but also the highest bundle delivery rates in high congestion scenarios.

In what regards the future work, it is proposed a further simulation of the policy in other delivery probability estimation routing protocols, and with different movement models. Also, it is proposed to add a new feature that would allow the bundle's holders information to be spread throughout the network, i.e. by transmitting at each encounter, the information of to which of the nodes each bundle has been replicated to – as it is done in MaxProp routing protocol with the bundle delivery messages. Moreover, it is proposed to relate the RTTL of each bundle to LBHD's HD metric to further improve the average delay. For congestion situations, incoming bundles could be rejected in some cases, instead of dropping existing bundles from the node's buffer, managing to decrease the overhead ratio while maintaining or increasing the delivery rate.

References

[1] K. Fall, “A Delay Tolerant Network Architecture for Challenged Internets,” *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM ’03)*. Karlsruhe, Germany, pp. 27–34, 2003.

[2] A. E. Al-Fagih and H. S. Hassanein, “Routing Schemes for Delay - Tolerant Networks - An Applications Perspective,” Kingston, Ontario, Canada, 2010.

[3] “Veniam.com.” [Online]. Available: https://veniam.com/. [Accessed: 25-Mar-2017].

[4] Hotel News Resource, “Internet of Things (IoT): number of connected devices worldwide from 2012 to 2020 (in billions).” [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. [Accessed: 26-Mar-2017].

[5] M. P. Rodrigues, N. Magaia, and P. R. Pereira, “LBHD : Drop Policy for Routing Protocols with Delivery Probability Estimation,” *12th Iberian Conference on Information Systems and Technologies, (CISTI’2017)*. IEEE, Lisbon, Portugal, 2017.

[6] S. Jain, K. Fall, and R. Patra, “Routing in a delay tolerant network,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 4, p. 145, 2004.

[7] S. M. Tornell, C. T. Calafate, J. C. Cano, and P. Manzoni, “DTN protocols for vehicular networks: An application oriented overview,” *IEEE Commun. Surv. Tutorials*, vol. 17, no. 2, pp. 868–887, 2015.

[8] A. P. Silva, S. Burleigh, C. M. Hirata, and K. Obraczka, “A survey on congestion control for delay and disruption tolerant networks,” *Ad Hoc Networks*, vol. 25, no. PB, pp. 480–494, 2015.

[9] Y. Cao and Z. Sun, “Routing in delay/disruption tolerant networks: A taxonomy, survey and challenges,” *IEEE Commun. Surv. Tutorials*, vol. 15, no. 2, pp. 654–677, 2013.

[10] D. Ismail and M. Jaafar, “Mobile ad hoc network overview,” *2007 AsiaPacific Conf. Appl. Electromagn.*, 2007.

[11] F. J. Martinez, C. K. Toh, J.-C. Cano, C. T. Calafate, and Pietro Manzoni, “A survey and comparative study of simulators for vehicular ad hoc networks (VANETs),” *Wirel. Commun. Mob. Comput.*, no. 11, pp. 813–828, 2011.

[12] N. Benamar, K. D. Singh, M. Benamar, D. El Ouadghiri, and J. M. Bonnin, “Routing protocols in Vehicular Delay Tolerant Networks: A comprehensive survey,” *Comput. Commun.*, vol. 48, pp. 141–158, 2014.

[13] K. Scott and S. Burleigh, “Bundle Protocol Specification,” *RFC 5050*. The IETF Trust, pp. 1–47,

2007.

[14] J. Postel, “Transmission Control Protocol,” *RFC 793*, no. September, pp. 1–85, Sep-1981.

[15] S. Farrell, “A Delay and Disruption Tolerant Transport Layer Protocol.” Ph.D. dissertation, Trinity College, University of Dublin, Dublin, Ireland, 2008.

[16] C. M. Huang, K. C. Lan, and C. Z. Tsai, “A survey of opportunistic networks,” *Proc. - Int. Conf. Adv. Inf. Netw. Appl. AINA*, pp. 1672–1677, 2008.

[17] V. Cerf, S. Burleigh, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss, “Delay-Tolerant Networking Architecture,” *RFC 4838*. The IETF Trust, pp. 1–35, 2007.

[18] D. Patel and R. Shah, “A Survey on Improved PROPHET Routing Protocol in DTN,” *Int. Res. J. Eng. Technol.*, vol. 2, no. 8, pp. 1237–1240, 2015.

[19] Sukhbir and R. Singh, “Effective routing protocols for delay tolerant network,” *Int. J. Mod. Eng. Res.*, vol. 2, no. 4, pp. 1732–1735, 2012.

[20] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, “Spray and Wait : An Efficient Routing Scheme for Intermittently Connected Mobile Networks,” *Proc. 2005 ACM SIGCOMM Work. Delay-tolerant Netw.*, pp. 252–259, 2005.

[21] A. Lindgren, A. Doria, E. Davies, and S. Grasic, “Probabilistic Routing Protocol for Intermittently Connected Networks,” *RFC 6693*. IETF, 2012.

[22] A. Balasubramanian, B. Levine, and A. Venkataramani, “DTN routing as a resource allocation problem,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, p. 373, 2007.

[23] J. Burgess, B. Gallagher, D. Jensen, and B. N. Levine, “MaxProp: Routing for vehicle-based disruption-tolerant networks,” *Proc. IEEE Infocom*, 2006.

[24] J. Chen, “Dijkstra’s shortest path algorithm,” *J. Formaliz. Math.*, vol. 15, no. 2, pp. 1–9, 2003.

[25] V. N. D. G. D. J. Soares and J. J. P. C. Rodrigues, “Performance Analysis of Scheduling and Dropping Policies in Vehicular Delay-Tolerant Networks,” *Int. J. Internet Technol.*, vol. 3, no. 1/2, pp. 1–9, 2010.

[26] R. Sulma, A. Qaisar, S. M. Z. M., and A. A.Hanan, “E-DROP An Effective Drop Buffer Management Policy for DTN Routing Protocols,” *Int. J. Comput. Appl.*, vol. 13, no. 7, pp. 8–13, 2011.

[27] Y. Harrati and A. Abdali, “MaxHopCount: A New Drop Policy to Optimize Messages Delivery Rate in Delay Tolerant Networks,” *Int. J. Interact. Multimed. Artif. Intell.*, vol. 4, no. 1, pp. 37–41, 2016.

[28] A. Rani, S. Rani, and H. S. Bindra, “Performance Evaluation of Mofo Buffer Management Technique With Different Routing Protocols in Dtn Under Variable Message Buffer Size,” vol. 3, no. 2. International Journal of Engineering Research & Technology (IJERT), Sirsa, Haryana, India, pp. 82–86, 2014.

[29] Y. Li, L. Zhao, Z. Liu, and Q. Liu, “N-Drop: congestion control strategy under epidemic routing

in DTN," *Proc. 2009 Int. Conf. Wirel. Commun. Mob. Comput. Connect. World Wirelessly - IWCMC '09*, p. 457, 2009.

[30] Q. Ayub and S. Rashid, "T-Drop : An optimal buffer management policy to improve QOS in DTN routing protocols," *J. Comput.*, vol. 2, no. 10, pp. 46–50, 2010.

[31] Q. Ayub, S. Rashid, and M. S. M. Zahid, "Buffer Scheduling Policy for Opportunitic Networks," *Int. J. Sci. Eng. Reserach*, vol. 2, no. 7, pp. 1–6, 2011.

[32] S. Rashid, A. H. Abdullah, M. S. M. Zahid, and Q. Ayub, "Mean Drop an Effectual Buffer Management Policy for Delay Tolerant Network," *Eur. J. Sci. Res.*, vol. 70, no. 3, pp. 396–407, 2012.

[33] A. Lindgren and K. S. Phanse, "Evaluation of Queueing Policies and Forwarding Strategies for Routing in Intermittently Connected Networks," *2006 1st Int. Conf. Commun. Syst. Softw. Middlew.*, pp. 1–10, 2006.

[34] D. Kim, H. Park, and I. Yeom, "Minimizing the Impact of Buffer Overflow in DTN," *3rd Int. Conf. Futur. Internet Technol.*, p. 2, 2008.

[35] J. Postel, "Internet Control Message Protocol," *RFC 792*, pp. 1–21, Sep-1981.

[36] S. Burleigh, E. Jennings, and J. Schollcraft, "Autonomous Congestion Control in Delay-Tolerant Networks," *SpaceOps*, pp. 1–10, 2006.

[37] Y. An and X. Luo, "MACRE: a Novel Distributed Congestion Control Algorithm in DTN," *Adv. Eng. Forum*, vol. 1, pp. 71–75, 2011.

[38] J. Lakkakorpi, M. Pitkänen, and J. Ott, "Using buffer space advertisements to avoid congestion in mobile opportunistic DTNs," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6649 LNCS, pp. 386–397, 2011.

[39] A. Keränen, T. Kärkkäinen, and J. Ott, "Simulating mobility and DTNs with the ONE," *J. Commun.*, vol. 5, no. 2, pp. 92–105, 2010.

[40] A. Keränen, J. Ott, and T. Kärkkäinen, "The ONE Simulator for DTN Protocol Evaluation," *Proc. Second Int. ICST Conf. Simul. Tools Tech.*, pp. 55–64, 2009.

[41] A. Keränen and J. Ott, "Increasing reality for DTN protocol simulations," *Helsinki Univ. Technol. Tech. Rep*, 2007.