

# **Performance Analysis of the Well-Known DTN Routing Protocols**

**Mohamed Agleiwan**

Submitted to the  
Institute of Graduate Studies and Research  
in Partial Fulfillment of the Requirements for the degree of

Master of Science  
in  
Computer Engineering

Eastern Mediterranean University  
July 2017  
Gazimağusa, North Cyprus

## ABSTRACT

Delay /disruption-tolerant networks (DTNs) are described by an absence of persistent paths between nodes because of roaming of nodes, constrained information storage capacity of a few or the greater part of its nodes. To beat the successive separations, and obliged energy resources, the nodes in DTNs are demanding to carry and hold on information bundles until they get close to different nodes. Storing of this information might take a long time. Additionally, to build up the delivery likelihood, they spread numerous duplicates of a similar bundle through the network in order to be ensure that one of these copies will arrive to its final recipient. Due to, the constrained power sources and restricted storage of numerous hubs in this environment, so there is a big tradeoff between expanding the packet Delivery Ratio and storage capacity utilization. Therefore, this thesis concentrates the routing issue in DTNs with constrained resources and limited storage and study the performance of five well-know DTN routing protocols such as, MaxProp, PROPHET, Spray and Wait, Epidemic and Social Group-based Routing (SGBR). Additionally, we modified SGBR protocol by reformulating the main equation to maximize the packet Delivery Ratio while reducing network overhead. Next, we compare the protocol that we modified with current famous DTNs routing protocols using the ONE simulator. The simulation results demonstrate that the SGBR\_V2 protocol realizes a better Delivery Ratios and lower levels in terms of the network Overhead Ratio contrasted with the original SGBR protocol when the traffic load is high.

**Keywords:** Routing Protocols, Delay /disruption-tolerant networks (DTNs), Social network, performance evaluation.

## ÖZ

Gecikmeli / bozulmaya dayanıklı ağlar (DTNs), düğümlerin dolaşımı, düğümlerin bir kısmının kısıtlı bilgi depolama kapasitesi nedeniyle, düğümler arasındaki kalıcı yolların olmaması ile açıklanmaktadır. Ardışık ayrımları yenmek ve enerji kaynaklarını yüklemek için DTN'lerdeki düğümler, farklı düğümlere yaklaşıncaya kadar bilgi paketlerini tutup taşırlar. Bu bilgilerin saklanması uzun zaman alabilir. Ayrıca, dağıtım olasılığını artırmak için, paketleri çoğaltarak, bu kopyalardan birinin son alıcıya ulaşmasını sağlarlar. Sınırlandırılmış güç kaynakları ve bu ortamda sayısız hub'ların kısıtlı depolaması nedeniyle, Paket Teslim Oranı'nı ve depolama kapasitesi kullanımını genişletmek arasında büyük bir takas söz konusudur. Bu nedenle, bu tezde, sınırlı kaynakları ve sınırlı saklama alanlı DTN'lerdeki yönlendirme sorununa yoğunlaşmış olup, MaxProp, PROPHET, Spray ve Wait, Epidemic and Social Group-based Routing (SGBR) gibi iyi bilinen beş DTN yönlendirme protokolünün performansı incelenmektedir. Buna ek olarak, SGBR protokolünü, Paket Teslim Oranı'nı en yükseğe çıkarmak için ana denklemi yeniden formüle ederek değiştirip ağ yükü'nü de azalttık. Daha sonra, ONE simülatörünü kullanarak mevcut iyi bilinen DTN yönlendirme protokolleri ile değiştirdiğimiz protokolü karşılaştırdık. Simülasyon sonuçları, SGBR\_V2 protokolü orijinal SGBR ile karşılaştırıldığında, trafik yükü yüksek olduğunda, daha iyi bir Paket Teslim Oranı ve daha düşük seviyeli ağ Tavan Oranı gerçekleştirdiği görülmüştür.

**Anahtar Kelimeler:** Yönlendirme protokolleri, Gecikme / bozulmaya dayanıklı ağlar (DTNs), Sosyal ağ, Performans değerlendirmesi

## DEDICATION

This thesis work is dedicated to my parents, who have been a constant source of support and encouragement during the challenges of graduate school and life. I am truly thankful for having you in my life, who have always loved me unconditionally and whose good examples have taught me to work hard for the things that I aspire to achieve. I am grateful.

To my brothers and sisters, to the long nights you spent helping me getting through this thesis, the long hours you spent encouraging me that it will all be over, to the times when it was impossible to continue, you were all standing by my side with your encouraging words and love, this work would not have been complete with all of you. I am lucky to have you all in my life. Thank you.

Not least of all, I owe so much to my friends for their undying support, their unwavering belief that I can achieve so much. Unfortunately, I cannot thank everyone by name because it would take a lifetime but, I just want you all to know that you count so much. Had it not been for all your prayers and benedictions; were it not for your sincere love and help, I would never have completed this thesis. So thank you all.

## **ACKNOWLEDGMENT**

In the Name of Allah, the Most Merciful, the Most Compassionate all praise be to Allah, the Lord of the worlds; and prayers and peace be upon Mohamed His servant and messenger.

First and foremost, I must acknowledge my limitless thanks to Allah, the Ever Magnificent; the Ever-Thankful, for His helps and bless. I am totally sure that this work would have never become truth, without His guidance.

My sincere appreciation also goes to my supervisor Assoc. Prof. Dr. Gürcü Öz, whose contribution and constructive criticism has pushed me to expend the kind of efforts I have exerted to make this work as original as it can be. Thanks to her I have experienced true research and my knowledge on the subject matter has been broadened. I will never forget you lady.

A special thanks to my friends, for helping me through the process of finalizing this thesis, no words can express my gratitude to your efforts, thanks for your assistance and your kind words, you will be in my mind.

# TABLE OF CONTENTS

ABSTRACT .....	iii
ÖZ .....	iv
DEDICATION .....	v
ACKNOWLEDGMENT .....	vi
LIST OF FIGURES .....	x
LIST OF TABLES .....	xii
LIST OF ABBREVIATIONS .....	xiii
1 INTRODUCTION .....	1
1.1 Introduction .....	1
1.2 Summary of Contributions and Expected Outcome.....	2
1.3 Thesis Outline .....	3
2 LITERATURE REVIEW.....	4
2.1 Background.....	4
2.1.1 Key Properties of DTNs .....	4
2.2 Network Architecture.....	5
2.2.1 Naming, Addressing and Binding.....	6
2.2.2 Routing and Forward .....	8
2.2.3 Fragmentation and Reassembly .....	9
2.3 DTN Applications.....	10
2.3.1 Battlefield Communications .....	11
2.3.2 Disaster Rescue and Environment Monitoring Communications.....	12
2.3.3 Digital Communication for Rural Areas.....	13

2.3.4	Personal/Wildlife Communications .....	17
2.4	Routing in Delay Tolerant Networks .....	18
2.4.1	Strategy Properties .....	18
2.4.2	Carry, Store, and Forward Approach.....	20
3	DTN ROUTING PROTOCOLS .....	24
3.1	Epidemic .....	24
3.2	Spray and Wait.....	26
3.3	MaxProp.....	28
3.4	PROPHET .....	30
3.5	SGBR (Social Grouping-Based Routing).....	32
3.6	SGBR_V2 .....	34
3.6.1	Protocol Description .....	34
3.6.2	Protocol Design.....	34
3.7	Related work.....	35
4	SIMULATION STUDY.....	37
4.1	Implementation .....	37
4.2	Performance Metrics .....	37
4.3	Simulator Setup and Parameters .....	39
4.3.1	ONE Simulator .....	39
4.3.2	Simulation Setup.....	40
4.4	Simulation Results.....	41
4.4.1	Impact of Buffer Size.....	42

4.4.2 Impact of TTL (Time To Life) .....	47
4.4.3 Impact of TL (Traffic Load) .....	51
4.4.4 Impact of Packet Size .....	56
4.4.5 Impact of the Number of Nodes .....	60
5 CONCLUSION AND FUTURE WORK.....	65
REFERNCES .....	67
APPENDIXES .....	74
Appendix A: Algorithms .....	75
A1: Epidemic Routing Protocol Algorithm .....	75
A2: Spray and Wait Routing Protocol Algorithm .....	76
A3: MaxProp Routing Protocol Algorithm .....	77
A4: PROPHET Routing Protocol Algorithm .....	78
A5: SGBR Routing Protocol Algorithm.....	79
Appendix B: Main Code.....	81
B1: Code of SGBR Routing Protocol .....	81
B2: Code of SGBR_V2 Routing Protocol .....	93
Appendix C: Screenshots of the ONE Simulator .....	94
C1: GUI Interface .....	94
C2: Setting File .....	94
C3: Report File.....	101



# LIST OF FIGURES

Figure 2. 1: Bundle layer DTN network .....	6
Figure 2. 2: DTN application for airborne network .....	12
Figure 2. 3: DakNet topology .....	13
Figure 2. 4: TrainNet topology .....	15
Figure 2. 5: Carry, store and forward approach in DTN .....	21
Figure 2. 6: DTN routing classification .....	23
Figure 3. 1: Epidemic strategy .....	24
Figure 3. 2: The Epidemic routing protocol when two hosts, A and B, come within connectivity range to each other. ....	26
Figure 3. 3: Spray and Wait binary mode .....	27
Figure 3. 4: The MaxProp routing strategy.....	28
Figure 3. 5: MaxProp path cost calculation .....	30
Figure 3. 6: A network that divided into three social groups.....	33
Figure 4. 1: The ONE structure [41]. ....	39
Figure 4. 2: Helsinki map in ONE simulator with 10 nodes.....	42
Figure 4. 3: Impact of buffer size on Delivery Ratio in a small network. ....	43
Figure 4. 4: Impact of buffer size on Average Hop Count in a small network.....	45
Figure 4. 5: Impact of buffer size on Delivery Ratio in a larger network. ....	46
Figure 4. 6: Impact of buffer size on Average Hop Count in a large network.....	47
Figure 4. 7: Impact of TTL on Delivery Ratio in a small networks.....	48
Figure 4. 8: Impact of TTL on Total Dropped Packet in a small network.....	49
Figure 4. 9: Impact of TTL on Delivery Ratio in a large network.....	50
Figure 4. 10: Impact of TTL on Total Dropped Packet in a large network. ....	51

Figure 4. 11: Impact of TL on Delivery Ratio in a small network.....	53
Figure 4. 12: Impact of TL on Delivery Ratio in large network. ....	54
Figure 4. 13: Impact of TL on Overhead Ratio in a small network. ....	55
Figure 4. 14: Impact of TL on Overhead Ratio in a large network.....	56
Figure 4. 15: Impact of packet size on Delivery Ratio in a small network.....	57
Figure 4. 16: Impact of packet size on Delivery Ratio in a large network. ....	58
Figure 4. 17: Impact of packet size on Average Latency in a small network. ....	59
Figure 4. 18: Impact of packet size on Average Latency in a large network.....	60
Figure 4. 19: Impact on number of nodes on Delivery Ratio.....	62
Figure 4. 20: Impact on number of nodes on Total Dropped Packet. ....	63
Figure 4. 21: Impact on number of nodes on Overhead Ratio. ....	64
Figure C. 1: The ONE simulator GUI interface.....	94
Figure C. 2: ONE simulator report file .....	101

## LIST OF TABLES

Table 4. 1: Simulation parameters .....	41
Table 4. 2: Impact of buffer size on Delivery Ratio in a small network.....	43
Table 4. 3: Impact of buffer size on Average Hop Count in a small network. ....	45
Table 4. 4: Impact of buffer size on Delivery Ratio in a large network. ....	46
Table 4. 5: Impact of buffer size on Average Hop Count in a large network.....	47
Table 4. 6: Impact of TTL on Delivery Ratio in a small network. ....	48
Table 4. 7: Impact of TTL on Total Dropped Packet in small network.....	49
Table 4. 8: Impact of TTL on Delivery Ratio in a large network. ....	50
Table 4. 9: Impact of TTL on Total Dropped Packet in a large network.....	51
Table 4. 10: Impact of TL on Delivery Ratio in a small network.....	53
Table 4. 11: Impact of TL on Delivery Ratio in a large network. ....	54
Table 4. 12: Impact of TL on Overhead Ratio in a small network. ....	55
Table 4. 13: Impact of TL on Overhead Ratio in a large network. ....	56
Table 4. 14: Impact of packet size on Delivery Ratio in a small network. ....	57
Table 4. 15: Impact of packet size on Delivery Ratio in a large network.....	58
Table 4. 16: Impact of packet size on Average Latency in a small network. ....	59
Table 4. 17: Impact of packet size on Average Latency in a large network.....	60
Table 4. 18: Impact on number of nodes on Delivery Ratio.....	62
Table 4. 19: Impact on number of nodes on Total Dropped Packet. ....	63
Table 4. 20: Impact on number of nodes on Overhead Ratio. ....	64

## LIST OF ABBREVIATIONS

ADU	Application Data Units
AN	Airborne Network
AODV	Ad-hoc On-Demand Distance Vector
Del	Delay ratio
DoD	Department of Defense
DSR	Dynamic Source Routing
DTN	Delay Tolerant Network
DTNRG	Delay-Tolerant Networking Research Group
DTNSim	Delay Tolerant Network Simulator
DVC	Delivery Cost
EID	Endpoint Identifier
FSOC	Free Space Optical Communications
GUI	Graphical User Interface
ICMANET	Intermittently Connected Mobile ad-hoc Network
IPN	Interplanetary Networks
IRTF	Internet Research Task Force
MANET	Mobile Ad-hoc Network
MAPs	Mobile Access Points
MRG	Minimum Reception Group
NS-2	Network Simulator, 2000
OMNet	Operations and Maintain Network
ONE	Opportunistic Network Environment
OPNET	Operations Network

PDR	Probability of Delivery
POP	Point Of Presence
RF	Radio Frequency
SCF	Store, Carry and Forward
SeNDT	Sensor Network with Delay Tolerance
SGBR	Social Group-Based Routing
SnW	Spray and Wait
TTL	Time To Life
TL	Traffic Load
UAVs	Unmanned Aerial Vehicles
URIs	Uniform Resource Identifier
VANETs	Vehicular Ad-Hoc Networks
WBAN	Wireless Body Area Networks

# Chapter 1

## INTRODUCTION

### 1.1 Introduction

Mobile Ad-hoc Network (MANET) is a group of autonomous mobile nodes that are self-configuring infrastructure and connected to wireless medium. Here the nodes are arranged in a disseminated manner. Each device in an MANET is spare to relocation independently, and will consequently change its links to other devices frequently. The message travels through the network of the help of the in-between nodes to reach its destination, so each node must send on traffic unrelated to its own use, and thence be a router. Fitting out each device to continuously maintain the information demanded to properly route traffic is the primary challenge in the establishing of an MANET. In order to defeat this challenge, some MANET is restricted to a local area of wireless devices and operate by themselves while others may be connected to the larger internet [1].

MANETs led to disconnected MANETs that is known as Intermittently Connected Mobile ad-hoc Network (ICMANET), and it is also called Delay Tolerant Network (DTN). Since the limitation of the transmission range and mobility are the most common properties in this kind of MANETs, The DTNs are characterized by intermittent connectivity, long or variable delay, asymmetric data rate and high error rates [2]. For these challenging environments the traditional ad-hoc routing protocols such as Ad-hoc On-Demand Distance Vector (AODV) [3] or Dynamic Source Routing

(DSR) [4] do not operate comfortably in DTN because of fully connected path between source and destination is required for communication to be possible. In order to overcome this challenge DTN protocols applies “Store, Carry and Forward” (SCF) mechanism for routing messages where receive and store the messages in the middle hops buffer allow to keep these messages alive until reaching their destination [5]. The challenges in DTNs is dealing with constrained resources and limited storage in some or most DTN nodes to be able to apply SCF mechanism. DTNs have applications for various ad-hoc networking and data spreading operations, like battlefield, wildlife monitoring, transportation engineering.

## **1.2 Summary of Contributions and Expected Outcome**

The major contributions of this thesis are the main objectives in this research which are summarized as follows:

- Understanding algorithms of well-known DTNs routing protocols.
- Understanding usage of the ONE simulator
- A working implementation of the SGBR routing algorithm.
- An improvement in SGBR protocol by editing the main formula in this protocol.
- Investigation performance of DTNs routing protocols using an important performance metrics.
- Comparison the modified protocol with existing protocols using Delivery Ratio, Average Latency, Total Dropped Packet, Overhead Ratio and Average Hop Count.

The outcomes of this research will be performed based on the major goals and the partial objectives from the “contribution” section. The expected outcomes will be declared based on their type:

- A brief survey of five various DTN routing protocols.
- 5 Implemented network scenarios of different sizes that have various values of some important network parameters such as Buffer Size, Time To Life (TTL), Traffic Load (TL) in terms of packet generation time, packet size and number of nodes.
- Presenting the results of different simulation-runs of the designed network models and displaying it in forms of tables/ diagrams.
- Interpretation of the table and diagrams and providing the reader with the suitable conditions for using each of the protocols.

### **1.3 Thesis Outline**

The thesis is organized as follows:

- In Chapter 2, we review the DTNs architecture and routing strategies.
- In Chapter 3, we review the well-known DTNs routing protocols, we also present our modification of the existing SGBR protocol.
- In Chapter 4, we provide the simulation setup and results for Delivery Ratio, Average Latency, Overhead Ratio, Total Dropped Packet and Average Hop Count. Some results and explanation of network traffic load as well as a more detailed investigation in relation between packet size and Average Latency. We provide a performance comparison among social based routing protocols to selected well-known DTN routing protocols.
- In Chapter 5, we present some concluding comments and put forward an idea into possible future research.



## Chapter 2

### LITERATURE REVIEW

#### 2.1 Background

In the 1970s Interplanetary Networks (IPN) were invented to communicate between earth and mars. Here the idea of Delay Tolerant Network (DTN) was presented. However, during 2001 and 2002, IPN researchers investigated how they could adopt the IPN architecture to different situations in which end-to-end connections were subject to delays and disruptions [6]. The DTN is a wireless ad-hoc network which allows the intermittent connectivity. DTN is also defined as intermittently connected wireless ad-hoc network. By using store-carry-forward approach DTNs can tolerate the longer delays, and prevent data from being lost [7].

##### 2.1.1 Key Properties of DTNs

DTN is distinguished by some significant key properties as stated below:

- Longed queuing delay:

In DTNs, the queuing delay is exacerbated as a result of the disconnection more than the usual contrast with the ordinary networks. In that sense, the queuing delay could be considerably high where is in the worst cases may take some hours or days .Thus, the queuing delay dominates the end-to-end Average Latency of data delivery.

- Disconnection:

In most cases, unexpected fault as well as network partition led to the disconnection. Normally, disconnection is more usual than connection. Due to, it is impossible to have an end-to-end path.

- High Average Latency and Low Delivery Ratio:

Due to, the persistent and random roaming of the DTN nodes, there is no guarantee to find a persistent path between any two nodes for a long time. Hence, the Delivery Ratio (data rate) would remain at a low level. The Delivery Ratio may be extremely low and high asymmetrical with the high Average Latency of data delivery.

- Limited longevity:

In the battlefields, wildlife monitoring and disaster areas end nodes can be propagated. For instance, the sensor nodes used for military detection or disaster recovery are friable and brittle to be out of order by the horrible surroundings. Due to power consumption, hostile actions, or environmental dangers. The end-to-end delay from the sensor nodes to the destination sink is longer than the surviving time of the node itself, which stores the data temporarily.

- Limited resources:

Practically, the DTN nodes have limited resources, Due to, they are mobile and battery operated with wireless connection. For example, the node in DTNs needs to store the received data until the connectivity to the next node is available. Consequently, the lack of memory capability will restrict the data buffering [8].

## **2.2 Network Architecture**

DTN networks differ from other MANETs in terms of network architecture [9] where it has a new layer that called “bundle layer” as illustrated in Figure 2.1 which is between Transport and Application layers in order to support the communication in

challenging environment. The Internet Research Task Force (IRTF) Delay-Tolerant Networking Research Group (DTNRC) has proposed this new layer. In this new architecture the bundle layer offers the real end-to-end data reliability across a heterogeneous network. Whereas, transport protocol end-to-end features are confined to homogeneous network segments. Although different protocols may exist underneath, the appended part feeds a uniform view of the network.

One of the most challenging situations in DTN environment is disorderly or discontinuous connect. In the case DTN can provide end-to-end connection by the optional “custody transfer” mechanism: DTN packets, called “bundles”, are stored in local databases at intermediate DTN nodes until the next hop is reachable, Then they are delivered as soon as the connection makes it possible. Until receiver’s acknowledgment bundles will be maintained in databases [10].

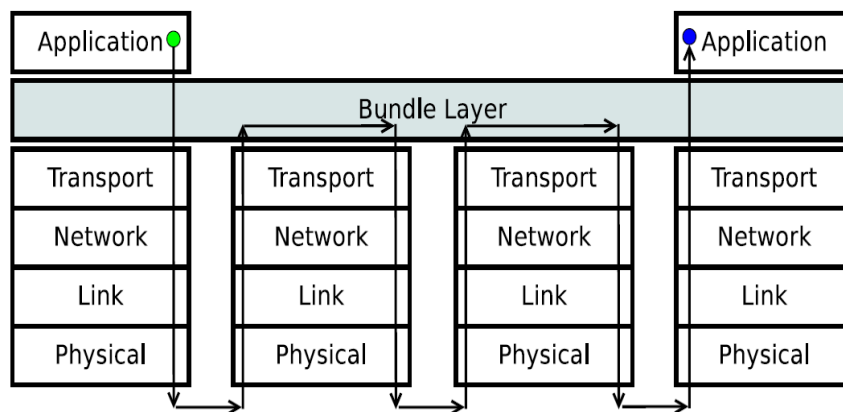


Figure 2. 1: Bundle layer DTN network

### 2.2.1 Naming, Addressing and Binding

In DTNs, the entire data about the condition of names, routers and addresses are impossible to be exist all time. Because of the services and nodes can move, appear, and disappear dynamically. In many cases, the destination node may have changed after creation of the bundle. That implies in order to locate nodes in such a changeable

circumference. Thus, utilize the location, sensed values and roles as name attributes of nodes is very important as well as Canonical DTN endpoint identifiers.

An Endpoint Identifier (EID) [9] is a name, communicated utilizing the general linguistic structure of Uniform Resource Identifier (URIs), that recognizes a DTN endpoint. Utilizing an EID, a hub is proficient to decide the Minimum Reception Group (MRG) of the DTN endpoint named by the EID. An EID may indicate to an endpoint comprising at least one DTN hubs, i.e. an EID may point to one node (unicast) one of a group of nodes (anycast) or all of the group of nodes (multicast and broadcast). At least one EID is required for each node to uniquely identify it. The canonic EID points to the EID of a bundle processing structure where it is able to receiving bundles oriented to that EID from other DTN nodes. The naming mechanism aims is to link the name attributes to the canonic EID.

Binding is the operation of expounding of an EID to select a next hop to which a bundle can be forwarded toward its destination. Due to, in DTNs the destination EID is possibly reinterpreted at each hop, binding might occur at the source, during transmission, or potentially at the destination. The latter two scenarios are attributed as late binding.

The naming system aims to enable service location and resource discovery and are as follows:

**Expressiveness:** To deal a vast set of devices and services, the naming system should be flexible in terms of the ability of applications to precise arbitrary service characterization and queries.

**Robustness:** The naming system must be elastic to conflicts in the internal state of the resolvers as well as name resolver and service failures.

**Easy configuration:** The resulting must propagate resolution queries among resolvers automatically. Manual registration of services must not be forced and the name resolvers must configure themselves with minimum manual interference.

**Responsiveness:** In common cases, network location of a service is changeable because of the performance fluctuations, the end-node and service mobility, and other factors. Therefore, the naming system must deal and adapt quickly to those circumstances [9].

### **2.2.2 Routing and Forward**

Due to, in a DTNs the nodes may connect by using many kinds of elemental network technology. The framework for routing and forwarding are provided at the bundle layer for unicast, anycast, and multicast messages in the DTN architecture. DTN architecture uses the bundle layer to provide routing and forwarding framework for messages, whether these messages are oriented to one node (unicast), one of a group of nodes (anycast) or all of the group of nodes (multicast and broadcast). A DTN network is defined theoretically utilizing a "multigraph" (where a vertices in a graph might be connected with many edges). The edges are, in general, time-varying with respect to their directional, delay and capacity. Because a prospect of one connected path in this graph between the edges. An edge is considered to not being connected when its capacity has zero value.

Since nodes in a DTN might allow extensive delay, it is significant to identify where time is measured while expressing delay or a node's capacity. For example, suppose  $B$  bits are placed in particular node at time  $t$ , they entirely reach by time:

$$t + D(t) + (1/C(t))*B \quad (2.1)$$

Equation (2.1) is provided in [9].

Where  $t$  is time that the source started to transmit  $B$  bits,  $D(t)$  is delay  $D$  at time  $t$  and  $C(t)$  is a node's capacity at time  $t$ . Assuming that,  $C(t)$  and  $D(t)$  do not change while the time is between  $[t, t + D(t)+(1/C(t))*B]$ .

### **2.2.3 Fragmentation and Reassembly**

Fragmentation and reassembly affect significantly on the bundle transfers. So, in DTN both are designed to improve the transmission rate by using fully contact volumes as well as avoiding any “partially-forwarded” [11] of bundles retransmission. DTN fragmentation and reassembly can be perform in two approaches:

### 1) Reactive fragmentation

In DTN a bundle may be transferred incompletely. For this situation, the nodes that sharing an edge in the DTN graph might cooperate to fragment this bundle. The incoming bundle will be modified by the receiving bundle layer in order to refer it is fragmented, and send it normally. Convergence-layer protocols may inform the previous hop sender that only a portion of the bundle was delivered to the next hop. Consequently, sending of the remaining portion(s) will be started when subsequent contacts become available (possibly to different next-hops if the path was changed). Since the fragmentation process comes after an attempted transmission occurs, this approach is called reactive fragmentation.

### 2) Proactive fragmentation

In this approach, a block of application data may split within numerous smaller blocks and convey them by the source node as a bundle independently. Here, the final recipient(s) will elicit and reassembling the smaller blocks as placed as in the original bundle, and eventually Application Data Units (ADU). This approach is used primarily when contact volumes are known or prior predicted. This approach is used foremost when transmit volumes are known or prior expected. Therefore, it called proactive fragmentation approach.

## **2.3 DTN Applications**

DTNs aim to transmit the data in challenging environments. Therefore, the applications that utilize this technology expect the delay and losing data during the communication [12]. To compensate this, these applications tend to send multiple

copies of their data and use the terminal points. DTNs technologies can be utilized in many fields as follow:

### **2.3.1 Battlefield Communications**

- **DTN for Military Missions**

In combat zone so as to bolster military errands and supply high limit transmission capacity they utilize Free Space Optical Communications (FSOC). In any case, there are a few burdens related with FSOC, for example, constriction and vacillation of optical flag at a recipient. Due to, FSOC joins are working outside in the climate where flick, hazes and environmental disturbance influence the execution of RF and any optical transmission. To enhance the correspondences effectiveness in front line a neighborhood approach is required and DTN advances can be used. Nichols et al. [13] built up a DTN based calculation to vanquish these burdens, where they attempt to travel data nearer to their goals on each hop by evaluation the topology controls prerequisite from FSOC, and in light of the fleeting casing build the bounce by-jump choices are utilized.

- **DTN for Airborne Networks**

An Airborne Network (AN) is "A system framework that offers connection operations through no less than one router which is implemented on a stand competent of flight" [14]. This includes planes jet and helicopters, Unmanned Aerial Vehicles (UAVs) and bombers. AN offers many service such as, weather information, mission updates, voice commands, tactical video and so on. Therefore, it becomes an essential element of the Department of Defense (DoD) in battlefield communication.



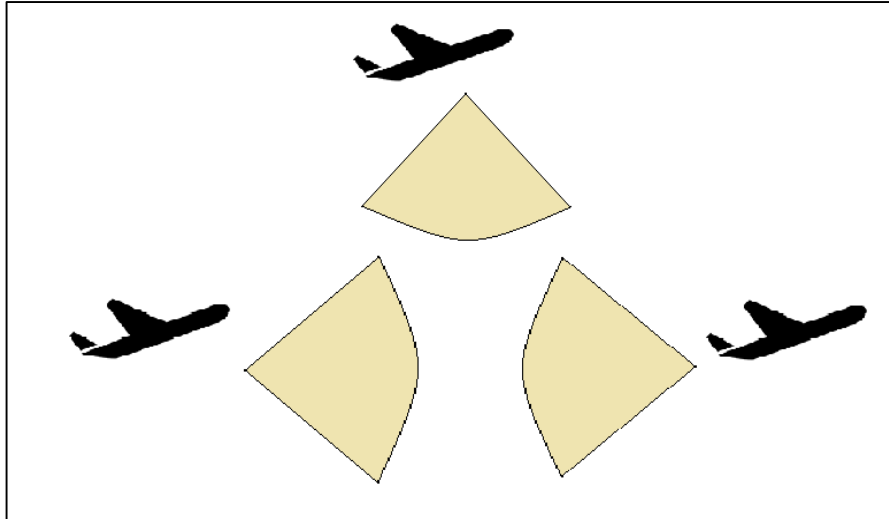


Figure 2. 2: DTN application for airborne network

Figure 2.2 illustrates the analogue waves disseminate phoneme correspondences between airplanes, where these airplanes should be in a particular waves range.

### 2.3.2 Disaster Rescue and Environment Monitoring Communications

- **DTN for Disaster Response Communications**

DTN technics are more suitable with the awful circumstances where it tolerates delay of connections. Therefore, DTNs are used in disaster rescue systems where such these systems need to communicate whether an Internet connection is available or not. Because of in most big disasters such as earthquakes cases the cellular towers become out of service. Thus, the critical issue in these cases is how to improve the reliability of communication platform in order to provide emergency reporting.

Fall et al proposed like this system in [15]. The proposed system that uses DTN technics aim to offer reliable backup service for Internet connection between mobile nodes in emergency cases when general communication platforms that not obtainable.

- Sensor Network with Delay Tolerance

Sensor networks are one of these networks that facing many challenges because of their harsh environments. Energy exhaustion, lack of storage capacity leads to losing data. Sensor Network with Delay Tolerance (SeNDT) [16] is developed to provide a robust sensor platform with the ability to survive for long periods in difficult environments. This platform is used to develop many applications in this field used to monitoring the noises and water quality in awful environments.

### 2.3.3 Digital Communication for Rural Areas

- **DakNet**

Massachusetts Institute of Technology (MIT) Media Lab developed the infrastructure communication in rural areas in terms of mobility and cover the huge areas at an extraordinarily low-cost[17]. This proposed System that called DakNet integrates physical transports with a wireless technology service to extend Internet association with a focal center point as internet service provider shops.

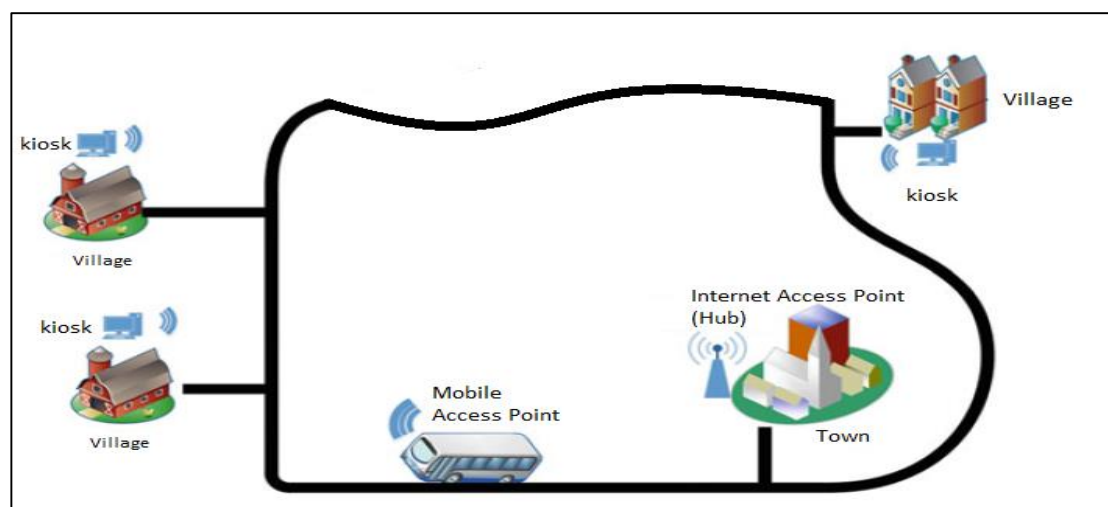


Figure 2. 3: DakNet topology

By using a present connections platform and transportation substructure to compose a DTN, DakNet supplies an unsynchronized digital communication on the Ad-hoc network platforms, as illustrated in Figure 2.3.

DakNet presented in outlying zones of India and Cambodia at a charge less contrasted with ordinary existing networks. Because of DakNet systems do not require to convey information for long distances, which is a favorable position since it leads to minify expenses and spare power extensively. Rather, information is transmitted through low cost end-to-end paths amongst kiosks and mobile stockpiling devices called Mobile Access Points (MAPs).

As appeared in Figure. 2.4, a MAP convey information among fixed stands such as, both state kiosks, Internet empowered stations and non-Internet reached to routers by utilizing portable generators that affixed on any kind of transportation.

Low-cost WiFi radio transceivers are used within high bandwidth point-to-point data transfers. Each time a MAP-prepared transportation enters inside an internet empowered station domain, it uncovers the WiFi automatically and afterward transfers download many megabytes of information. In contrast, when the MAP-equipped vehicle enters within a domain of an Internet provider stations, mechanically the created data from rustic kiosks will be synchronized. With only one a small MAP-equipped vehicle every day that it is enough to offer the workaday data services for a tiny hamlet. This application forms low cost DTN and asynchronous data reciprocation infrastructure.

- **TrainNet**

Zarafshan-Araki and Chin [18] proposed TrainNet to deliver non real-time data between cities if there is a train network connects them. They used the DTN technique with a mechanical backhaul based on a train which provides a high bandwidth link at a low cost. They used trains instead of other transportation systems since there are constant and deterministic movement timetables as well as trains have ability to cover a big spaces. It can also take high storage capacity in order to carry big quantities of data.

The TrainNet system requires to existence a hard disk as a rack of storages in each Train and station to store, carry and forward data. Thus, as illustrated in Figure 2.4 the supplier transfers non real-time data to the train via its point of presence (POP) that exists in each station.

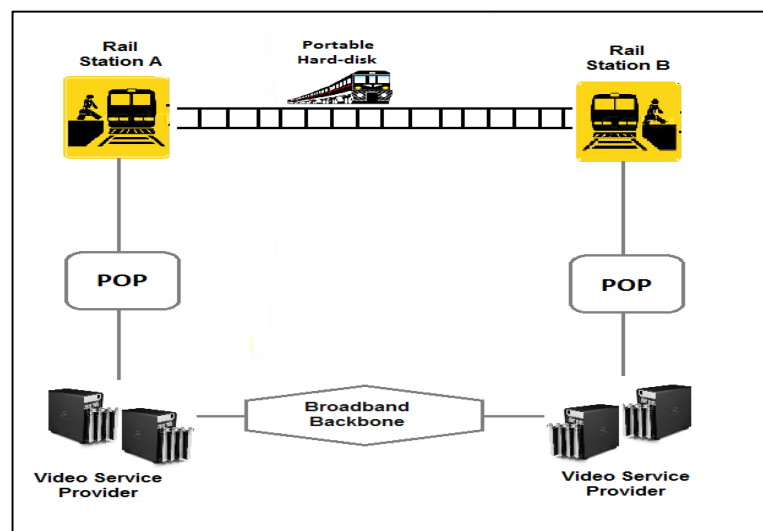


Figure 2. 4: TrainNet topology

- **KioskNet**

Kroeker et al. [17] expanded the idea from DakNet in order to provide a comprehensive approach in terms of “naming, addressing, forwarding and routing” which is called

KioskNet [19], a kiosk controller supplies memory devices, network flows and provide a function of client management. For regulation clients, like state officials, they deal with the kiosk comptroller as an internet access point to offer a DTN serve to the devices enabled Internet. Whereas, other ordinary clients connect to the kiosk comptroller in order to entree the accomplished applications. The major merit of this approach is the benefit of all kinds of transportation as a bridge to transfer data to the network access point that is internet enabled when they close to a kiosk range.

The ordinary methods to communicate with kiosks mostly based on dial up bloodline, wide ranges WiFi or the satellite stations, where all of these ways might endure some drawbacks. For instance the dial up line is moderate as well as inconsistent, satellite station is profoundly costly and the long range WiFi requires a lot of access points that deployment at large scale. Thus, KioskNet is cheaper than those ways because of, within only one connection period up to 5 min by any mechanical backhaul which could be any kind of vehicles or trains, around 50MB could be transmitted.

### **2.3.4 Personal/Wildlife Communications**

- **Pollen**

Natalie et al. proposed The Pollen network in [20]. Everybody in the network has an electronic type of pollen, such as a cellular phone. These devices can be shaped Pollen-ready by installing suitably programmed separate miniature computers, such as iButtons.

The considerable advantage of Pollen is the data commutation process does not rely on network platforms. The notion of pollinated a plant by insects pushes to release The Pollen network idea. In other words, an insect moves between flowers to collect nectar at the same time it picks up some pollen and distribute it between these flowers unintentionally. The same method is used in Pollen network, where a mobile device moves between different iButton groups, so, users can leave comments to these societies or carries bits of pollen to different portable devices. Therefore, Pollen network can provide connections amongst big counts of objects and devices, since they are high costly to be networked with the existing substructure. Nevertheless, the applications that demand a rapid reply or sure delivery are not suit for Pollen networks.

- **Body Area Networks**

Quwaider and Biswas developed Store-and-forward routing protocols that used in DTNs for Wireless Body Area Networks (WBAN) [21]. To enable a probabilistic and a distance vector on body routing protocol they introduce the notion of a stochastic link cost in the existence of postural mobility of the human body. To resolve the body bundles expelling defects in the existence of topologies division. WBAN uses sensor nodes that rely on low-power provided RF transceivers [22, 23] where the postural

body movements and clothing able to affect the signal transmission. this mechanism aims to reduce the end-to-end delay as well as guarantee low storage delay by sending a packet from its source to the its final recipient through various paths then the packet that delivered first refers to the minimum end-to-end storage delay.

## **2.4 Routing in Delay Tolerant Networks**

DTNs are characterized by the absence of end-to-end connectivity. Thus, the traditional routing protocols do not work well, because the timer and acknowledgement mechanisms of the TCP/IP protocol will fail here. In DTNs the mobility of the nodes increases the provoked of this problem as well, especially if its mobility pattern is unknown. As a result, that will be led to the problem of lack of knowledge about the current position of the node [24].

Reliability is the most important factor in the networks. Hence, many and many routing approaches have been developed to act well in the DTN environment where several issues should be taken into account, such as, increasing the Delivery Ratio, reducing the Average Delay, giving scalability, and improving resource utilization etc. Each of these approaches has its own merits and demerits and is suited in certain knowledge bases.

### **2.4.1 Strategy Properties**

There are two major properties are considered in DTN routing strategies. The first property, replication, refers to the number of the copies of the messages that will be used and how the strategy will choose from these copies as well as how does use them to deliver the original message to the final recipient. Whereas, the second property, knowledge, denotes to the amount of the information that will be used to route the messages between the hops till reach the end receiver.

## 1) Replication

DTNs is distinguished by disconnection is more common than the connection between their nodes because of the unreliable or unpredictable circumstances. To compensate for this, several routing strategies tend to send multiple copies of each message to increase the reliability that at least one copy will be delivered, or to minimize delivery Average Latency. This is an obvious trade-off between cost and performance. The intuition is that making multiple copies leads to increasing in the probability of the delivery, since, one of these copies will reach to the destination. However, this will affect the Total Overhead Ratio and the consumption of network resources. Although, the cheapest strategy is to make a single copy of the message, one failure means that this message will being lost. In contrast, the most reliable mechanism is to route one copy of the message to each node in the network. In this case, the message will being lost only if all the nodes in carrying it are fail to deliver it. Nevertheless, this consumes the network bandwidth and the capacity of storage resources proportional to the number of nodes.

## 2) Knowledge

Routing strategies are different in terms of the amount of information that DTN node needs about the network contacts in order to route its messages. Here, another tradeoff is that minimizing of information about the network schedule lead to more consumption of network resources. However, using a fully information about future schedule depletes the storage resources. A DTN node in these strategies that use zero knowledge can make decisions by using prior static rules that are configured when the strategy is designed as well as all nodes obey those rules. This leads to simple implementations that demand few configuration and control messages, since all the



rules are configured ahead of time. The impair point is that the strategy is not able to adjust to various networks or conditions, hence, it might not make optimum decision. In versa side of the spectrum, a node may require to know the full future schedule of every contact in the network. Provided that offering precise information. This permits to get very efficient use of network resources by route a message among the nodes along the best path. In between these two extremes there is a domain of values. For instance, in some strategies no need for prior information. However, they will learn it automatically. Or, partial information about the future nodes schedules might be exist [25].

#### **2.4.2 Carry, Store, and Forward Approach**

In traditional Internet routing incoming packets are stored in the current node buffer until the packets are routed and forwarded to the next hop with considering the routing decision. If the connection to the next hop is down, the packets might drop. Also the size of the buffer capacity is not very large, as packets are improbable to stay long time in the buffer. Whereas, a Store, Carry and Forward (SCF) technique is used by DTN nodes.

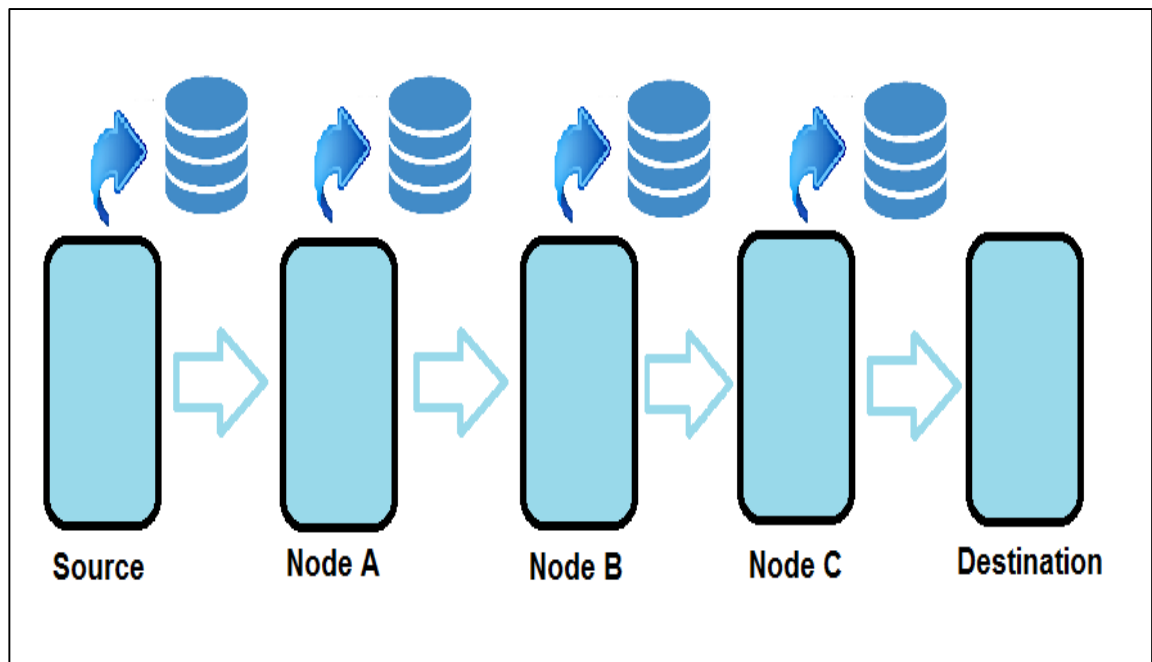


Figure 2. 5: Carry, store and forward approach in DTN

A DTN node operates to carry a bundle until it is either delivered to the destination or another DTN node which in turn carries it after coming into contact. The attribute of contact is used to realize a window pane of chance when it is possible to set up a connection with another DTN node. It should be mentioned that in many cases such a window of chance may be short. For example, in Vehicular Ad-Hoc Networks (VANETs) when a mobile device that in a car, motorcycle or train and comes closer to another device and has to transmit the bundle, before the other node goes away. Therefore, the bundle may take long time to move from a source or destination, because of intermittent connectivity and persistent storage in intermediate DTN nodes [26] as shown in Figure 2.5.

According to the routing strategy properties DTN routing protocols are grouped into two fundamental classifications, "Flooding protocols" group and "Forwarding protocols" group. The strategy used in flooding protocols is described as creating multiple copies of a message and propagate them through the network it to other nodes

so the message reaches its destination based on several parameters. Whereas, one copy of the message is used in a forwarding strategy that is travelling from the source to the destination via intermediate nodes.

Flooding strategy:

In this strategy, protocols do not require any knowledge about the network. To compensate, message replication is used in order to increase the chance of the message is delivered successfully to the final recipient. Therefore, many copies of the message will be created and send it to other DTN nodes called relay nodes, which carry and store it in the buffer until it reach the destination(s).

Forwarding strategy:

In this strategy, each node tries to convey a message through the network should has a knowledge about the network graph at that time in order to find the best path to reach the message's destination with low cost as possible,. In forwarding strategy no need for replication of data [27].

Figure 2.6 depicts the classification of DTN routing protocol based on the previous strategies, also each strategy includes sub grouped protocols which use different mechanisms as we will present in chapter three.

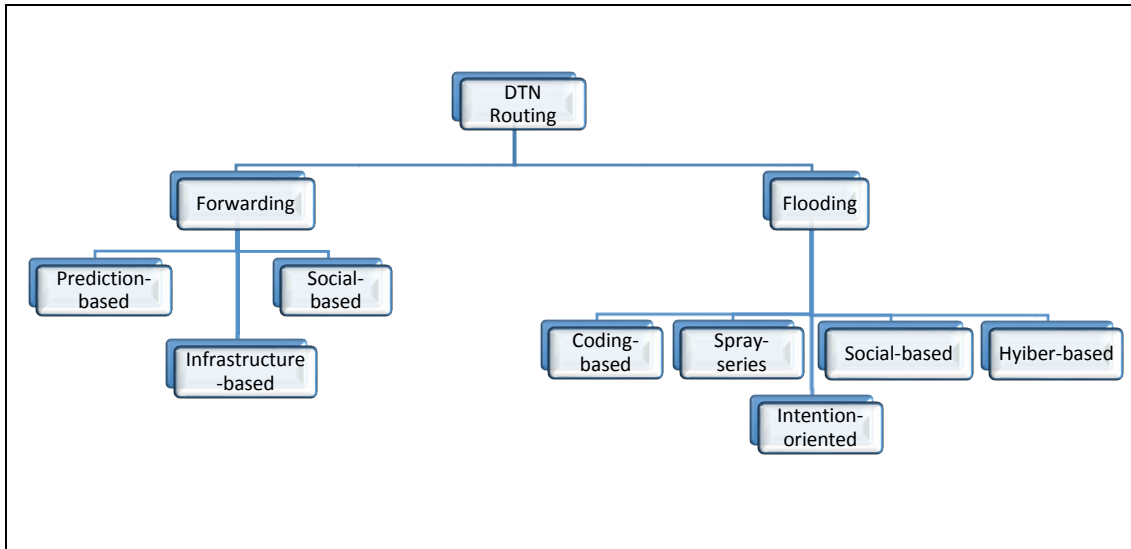


Figure 2. 6: DTN routing classification

According to this classification we will present two flooding-based protocols such as Epidemic and Spray and Wait, also we will present three different forwarding-based protocols such MaxProp, PROPHET and SGBR protocol.

## Chapter 3

### DTN ROUTING PROTOCOLS

#### 3.1 Epidemic

Epidemic Routing underpins the possible conveyance of messages to discretionary destinations with negligible suppositions in regards to the hidden topology and connectivity of the basic system. Truth be told, to guarantee possible message delivery just intermittent pair-wise connectivity is required. The protocol depends upon the transitive conveyance of messages out of ad-hoc networks, with messages reaching their final destination. Each host keeps up a buffer comprising of messages that it has emerged and additionally messages that it is buffered on behalf of the rest of the hosts.

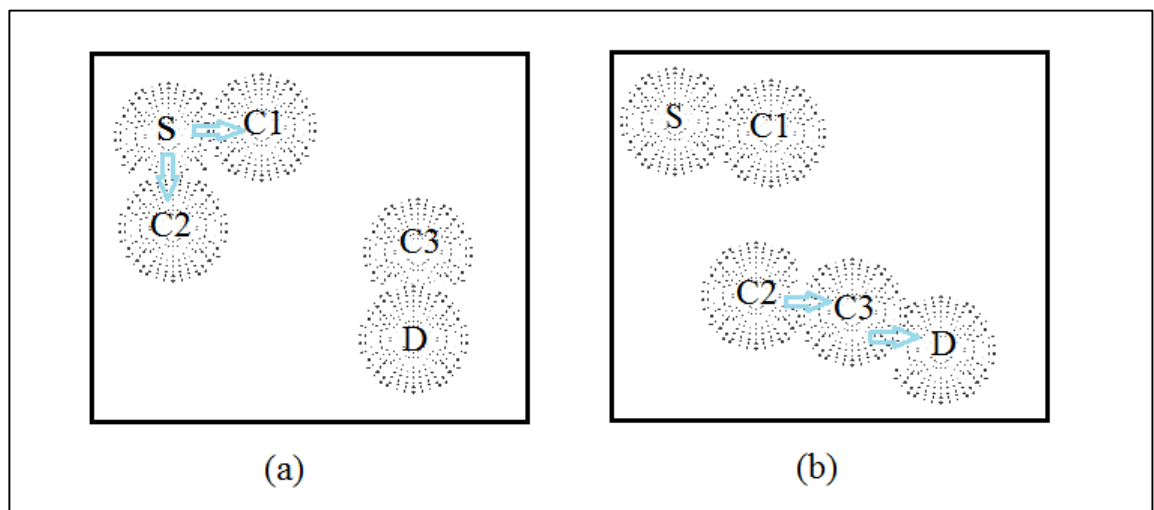


Figure 3. 1: Epidemic strategy

Through such technique for data transmission, packet has a high chance of inevitably arriving to its final recipient. Figure 3.1 delineates Epidemic Routing at a high level [28], in Figure 3.1(a), a source,  $S$ , interests to transmit a packet to certain recipient

which is  $D$ , yet no associated link is accessible from  $S$  to  $D$ .  $S$  sends its packets to the closest nodes which are,  $C1$  and  $C2$ , inside immediate connection area. Later, as appeared in Figure 3.1(b),  $C2$  entered within immediately connection area with different host,  $C3$ , then sends the packet to it.  $C3$  is in direct scope of  $D$  lastly routes the packet to its final recipient. For effectiveness, a hash table cross-indexes this list of packets, distinguished by an unparalleled identifier related to each packet. Each node stocks a bit vector that demonstrates which passages in their local hash tables are adjust this vector called the summary vector.

In order to get considerably lessen the space overhead connected with the summary vector "Blossom channel" [29, 30] is used. At the point while two hosts exist into connection scope of each other, the node that has littler identifier creates an anti-entropy session with the node with the bigger identifier.

Each node keeps up a reserve of hosts that it has talked with as of late to evade excess connections. Anti-entropy is not re-created with a distant node that has been connected within deterministic interval.

Amid anti-entropy, the two nodes trade their summary vectors to figure out which packets put away remotely have not been meet with local node. Thusly, each node at that time demands duplicates of packets that it has not yet observed. The recipients keeps up aggregate independence in choosing whether it will admit a packet. For instance, it might verify that it rejects to convey packets bigger than a given size or bound for specific nodes [31]. Figure 3.2 illustrates the summary vector exchange in the Epidemic Routing protocol. The Epidemic algorithm is provided in Appendix A.

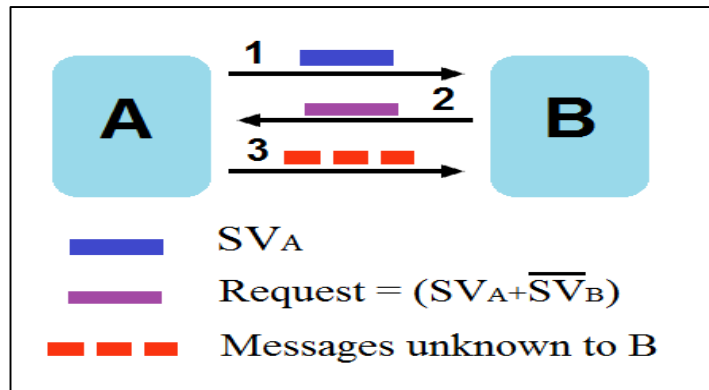


Figure 3. 2: The Epidemic routing protocol when two hosts, A and B, come within connectivity range to each other.

Node A send its summary vector ( $SV_A$ ) to node B, ( $SV_A$ ) is a summarized representation of all the packets being buffered at A. Then node B compare its summary vector ( $SV_B$ ) with ( $SV_A$ ) and send a request to node A that including ( $\overline{SV_B}$ ) that refers to the packets that node B need and ( $SV_A$ ). That is, B determines the set difference between the messages buffered at A and the messages buffered locally at B. It then transmits a vector requesting these messages from A. In step three, A transmits the requested messages to B.

### 3.2 Spray and Wait

Spray and Wait (SaW) routing uncouples the quantity of duplicates produced per message, and in this manner the quantity of transmissions performed will decrease, from the system size. By propagating a small number of copies each to a various relay.

This scheme comprises of two stages:

Spray stage: each node tries to send a message it propagates initially L message copies and probably some of those copies reaches to other nodes that will resend those copies again as a relays or receive one of copies as a destination.

Wait stage: in DTN the destination is not reachable for any time, so if it unreachable every node that has a copy of message carry out “Direct Transmission” in other words, it tries to send the message only to its final recipient.

SaW merges the velocity of Epidemic routing with provision of immediately conveyance. At first it “jump-starts” both SaW protocol and Epidemic protocol are distributing copies of each packet in the same technique. At the point, to ensure that no less than one of copies will reach to the destination rapidly a sufficient copies are distributed [32], and then it stops and allows other nodes that has a copy carry out direct transmission.

Spray and Wait has a pair of various models based on the number of message copies distributed. We have used Spray and Wait in a binary mode, which is explained here. In binary mode the source node starts with  $L$  copies of the message; any node A (source or relay) that carries  $n > 1$  message copies, and meets another node B which does not have any copy, hands over to B ( $n/2$ ) and maintains ( $n/2$ ) in its buffer, when it has just a single copy, it resorts to direct transmission. Although Spray and Wait protocols decouples the number of transmission performed, it requires a big buffer space in each node. Figure 3.3 depicts the binary mode mechanism when the source node S initiates  $L$  message copies and how it distributes the copies to other node, and then each relay node sends half the number of its copies. SaW algorithm is provided in Appendix A.

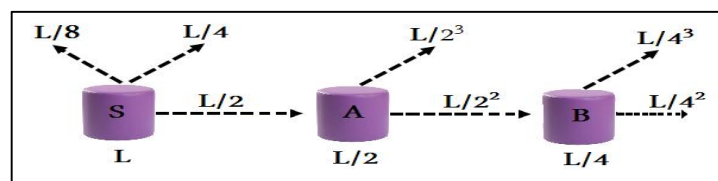


Figure 3. 3: Spray and Wait binary mode



### 3.3 MaxProp

MaxProp has proposed by John Burgess et al. [33] to addresses situations, such as, transmission time or restriction of storage capacity in the network by utilizing hop counts in messages as a measurement of networks resources, and uses network information that are spread through the network. MaxProp reserve a list of previous relay nodes to restrict data from spreading twice to the same node.

MaxProp utilizes the path probabilities to nodes based on historical knowledge, acknowledgments, lists of prior relay nodes and a head-start for new bundle. Those mechanisms that illustrated in Figure 3.4 are used to build the schedule of messages transmitted to other nodes and the schedule of messages to be dropped. MaxProp is based on prioritizing both these schedules to deliver the messages with a minimal transmit duration as well as low usage of storage resource.

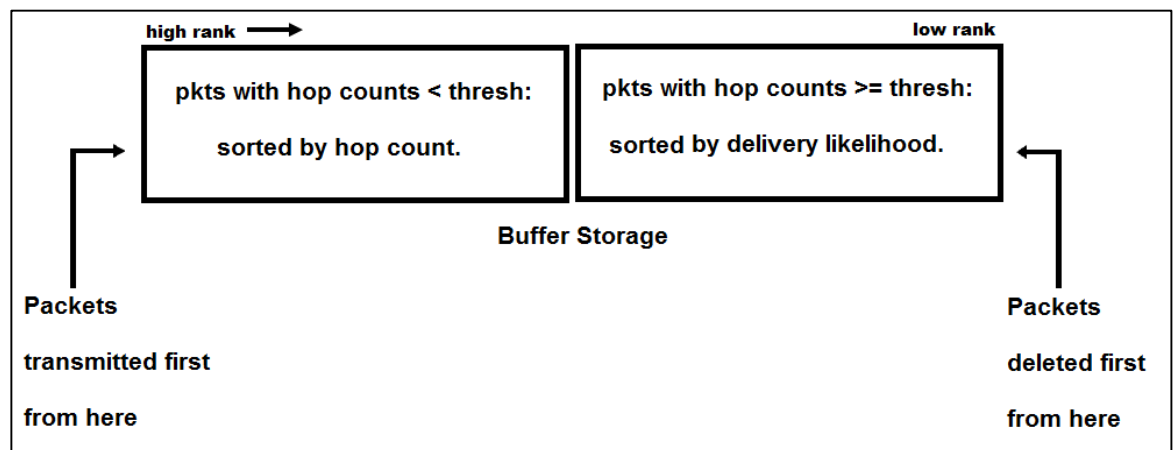


Figure 3. 4: The MaxProp routing strategy.

Based on a cost estimated to each destination, MaxProp protocol sorts the list of the node's stored packets. The cost is an appreciation of transmission probability. Furthermore, the information are used in order to inform all nodes when the packet is

delivered. In MaxProp protocol the new packets are granted a higher priority than older packets, and it also tries to impede receiving two copies for the same packet.

- Estimating Delivery Likelihood

MaxProp assigns weights for the paths that connect nodes as follows:

Each node belongs to the network has a probability to meet the other nodes  $P_{ij}$ . Initially, this probability equals to 1 divided to the number of the rest nodes, assume that there are five nodes the probability for each one to meet other node  $P_{ij}= 0.25$ . This probability will incremented by 1 in each time that node  $i$  and node  $j$  are encountered, and then all probabilities are normalized by the same method. A current node evaluates the path costs to each other nodes that knows their probabilities, the cost is calculated for every potential path to the destination as follow  $c(i, i+1, \dots, \text{Destination})$ , up to the number of hops in middle.

The estimated path cost is one minus a value of the probabilities that each connection does happen.

$$c(i, i + 1, \dots, d) = \sum_{x=i}^{d-1} [1 - (P_{xx} + 1)] \quad (3.1)$$

Equation (3.1) is provided in [33]

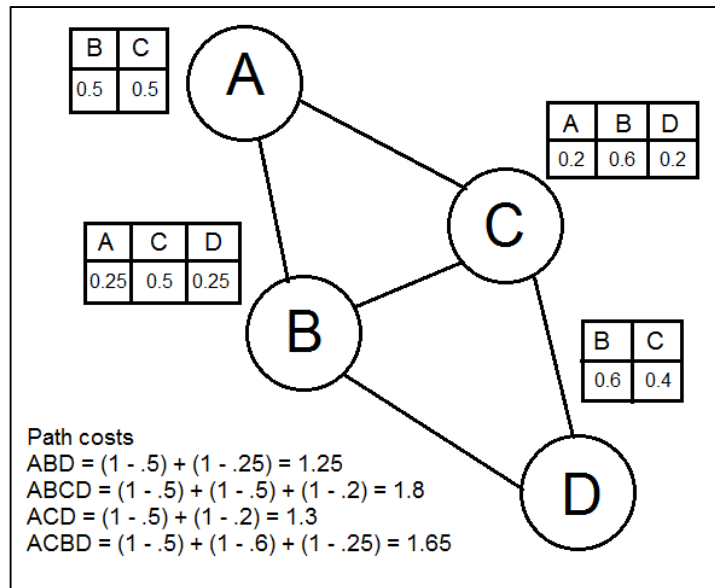


Figure 3. 5: MaxProp path cost calculation

The destination's cost is determined as the path which has cheapest cost from all available paths. Figure 3.5 depicts the path that had cheapest cost which is 1.25 is selected as the favorable path from *A* to *D*. MaxProp algorithm is provided in Appendix A.

### 3.4 PROPHET

PROPHET is forwarding probabilistic-based protocol that proposed by Anders Lindgren et al. [34]. PROPHET uses the history of peer encounters and transitivity to increase the likelihood of delivery message. To achieve that PROPHET relies on a delivery predictability,  $P(a, b) \in [0, 1]$  as a probabilistic metric. This refers to how probability it is that this node (*a*) will be able to convey a message to its destination (*b*). When two hosts meet the PROPHET and Epidemic protocol behaviors are same, where the summary vectors are exchanged, including the delivery predictability acknowledgement preserved at the hosts in order to update the internal delivery predictability vector, also to decide which messages are requested from the node at the other end.

The delivery predictabilities calculation has three phases:

- Updating the delivery predictabilities

Each time the node is encountered the predictability metric will update as follows where  $P_{init} \in [0, 1]$  is an initialization constant.

$$P(a, b) = P(a, b)_{old} + (1 - P(a, b)_{old}) \times P_{init} \quad (3.2)$$

Equation (3.2) is provided in [34]

- Aging

When two nodes do not meet each other a long, they have less chance to be good relays of messages to each other, hence, the PROPHET protocol reduces the delivery predictability values by aging this value, as in below equation.

$$P(a, b) = P(a, b)_{old} \times \gamma^k \quad (3.3)$$

Equation (3.3) is provided in [34].

Where  $\gamma$  is a  $\in [0, 1)$  is the aging constant, and  $k$  is the quantity of time units that elapsed since the last time they met. The time unit have to assigned based on the application and the predictable delays in the targeted system.

Updating transitivity

If node  $b$  encounters node  $a$  frequently, and node  $a$  encounters node  $c$  every now and again, then node  $c$  is a good relay node to convey messages oriented for node  $b$  to.

This transitive property affects the delivery predictability as follow in the equation of updating transitivity.

$$P(a, c) = P(a, c)_{old} + (1 - P(a, c)_{old}) \times P(a, b) \times P(b, a) \times \beta \quad (3.4)$$

Equation (3.4) is provided in [34]

Where  $\beta \in [0, 1)$  is a scaling constant that determines the extent of the effect of the transitivity on the delivery predictability. The algorithm for PROPHET protocol is provided in Appendix A.

### **3.5 SGBR (Social Grouping-Based Routing)**

Tamer Abdelkader et al. proposed a social protocol “SGBR” [35] based on the social relation among the nodes to restrict repetitive duplicating of packets which is reducing the network overhead while increasing the message Delivery Ratio. SGBR aims to distribute a minimal number of message copies to minimize the cost of delivery, by routing the message copies using a nearby social data to convey it into their destination. SGBR considers the nodes which meet regularly, they belong to the same social group also they are predicted to encounter each other again considerably. Those nodes often have approximately the similar relation with different hubs in the network. Thus, each member in one group might view itself as an agent of the group to spread messages to the rest of the groups. Thus, each hub that carries a message oriented to different hubs prefers to route the message to different social groups. SGBR does not tend to keep many duplicates of a similar message inside one social group. Figure 3.6 illustrates the DTN network that divided into three social groups according to the related between its nodes.

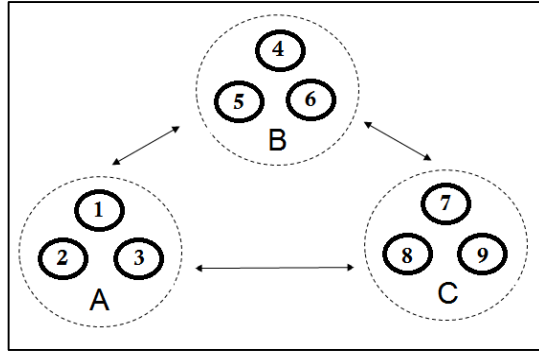


Figure 3. 6: A network that divided into three social groups.

In SGBR protocol each node has to know which nodes are belong to its group and which nodes are not. To assess whether two nodes ( $a, b$ ) are in the same group or not SGBR calculates the strength of connection ( $\Gamma_{ab}$ ) between them, where it increased by repeated encounters among each couple of nodes and decreased as long as pass a long time since previous encountered. The following equation used to update their value of connection strength:

$$\Gamma_{ab} = (\Gamma_{ab})_{old} \gamma^k + (1 - (\Gamma_{ab})_{old}) \alpha. \quad (3.5)$$

Equation (3.5) is provided in [35].

Where  $\alpha \in (0, 1]$  is the updating factor,  $\gamma \in (0, 1]$  is a constant for aging and  $k$  is the quantity of time that two nodes didn't meet since last meeting.

When two nodes meet SGBR operates like the Epidemic Routing, where the summary vectors are exchanged based on the value of connection strength to determine which packets should be transmit and which packet copies should be keep in its source if both nodes belong to the same group. SGBR algorithm is provided in Appendix A.

## **3.6 SGBR\_V2**

We attempted to improve the SGBR protocol in terms of increment the Delivery Ratio and additionally diminish the overhead proportion by changing the main equation in SGBR algorithm.

### **3.6.1 Protocol Description**

SGBR considers that when some nodes meet regularly they build some kind of relation. This relation is evaluated by calculating the connectivity strength between nodes based on how many times they met recently. In that sense, the network may divide into a number of groups where each group has those nodes that encountered each other more than other nodes in different group. Therefore, each member in one group might view itself as a delegate of other nodes in its group to spread messages to the rest of the groups. Thus, each hub that carries a message oriented to different hubs prefers to route the message to different social groups. SGBR does not tend to keep many duplicates of a similar message inside one social group. In SGBR\_V2 we used the same idea with modifying how the node decides if other node is located in the same group or not.

Our modification aims to reduce the mathematical operations that used to assess the strength of connectivity between two nodes to determine whether they will exchange their messages or not.

### **3.6.2 Protocol Design**

Each node have to know which nodes are belong to its group and which nodes are not. To assess whether two nodes (a, b) are in the same group or not. We changed the SGBR equation that calculates the strength of connection ( $\Gamma_{ab}$ ) between them, where it increased by repeated encounters among each couple of nodes and decreased as long

as pass a long time since previous encountered. The following equation used to update their value of connection strength:

In SGBR protocol

$$\Gamma_{ab} = (\Gamma_{ab})_{old} \gamma^k + (1 - (\Gamma_{ab})_{old} \gamma^k) \alpha. \quad (3.5)$$

Equation (3.5) is provided in [33]

Where  $\alpha \in (0, 1]$  is the updating factor,  $\gamma \in (0, 1]$  is aging constant, and  $k$  is the quantity of time units that elapsed since the last time they met.

- In SGBR\_V2

We calculate proportion of how many times that node ( $a$ ) meet with node ( $b$ ) to total connection that node ( $a$ ) has done.

$$\Gamma_{ab} = \frac{\sum C_{a,b}}{\sum_{i=1}^N C_{a,i}} \quad (3.6)$$

Where  $C_{a,b}$  counts the connection between ( $a$ ) and ( $b$ ),  $N$  the total of the nodes that node ( $a$ ) encountered. If the strength of connection ( $\Gamma_{ab}$ ) up to the threshold two are in one social group otherwise they are in a different group, where the value of threshold depend on network environment. SGBR\_V2 algorithm is provided in Appendix A.

### 3.7 Related work

Spacious research work has been presented in the area of DTN routing protocols. Different kind of simulators were used to simulate the behavior of various routing



protocols. In our work we will review briefly some recent research publications, about evaluation of DTN routing protocols performance. In this thesis work we use the ONE [41] simulator 1.5.1 to simulate five DTN routing protocols such as Epidemic, SaW, MaxProp, PROPHET and SGBR against three metrics such as network Overhead Ratio, Average Latency and Delivery Ratio. The protocols that show best result in the network Delivery Ratio must be the stellar in the network throughput. We will discuss the performance evaluation of different DTN routing protocols.

The results given in [36] analyze Epidemic, Spray and Wait, PROPHET, and MaxProp show that SnW and PROPHET are more efficiency in delivery cost, while MaxProp is better in Delivery Ratio and Average Delay. In [37] the given results illustrate that the Epidemic protocol outperforms in Delivery Ratio and Average Delay.

In reference [38] the author conclude that under the considered scenario the SaW routing protocol shows best results for delivery ratio and Overhead Ratio. Another paper in reference [39] stated that the simulation results demonstrate that Epidemic routing and PROPHET outperform in delivery ratio, but with a very high Overhead Ratio. Whereas, MaxProp and Spray and Wait have lower delivery ratio, but outperform in Overhead Ratio. For other protocols in [40] the author wrote a different kind of conclusions as follow:

SaW outperforms given the high delivery ratio as well as low Overhead Ratio. MaxProp is better in term of Average Latency. Epidemic has the worst Overhead Ratio and the highest Average Hop Count.

## Chapter 4

### SIMULATION STUDY

#### 4.1 Implementation

The implementation of the ONE simulator and the protocols such as, Epidemic protocol, MaxProp, PROPHET protocol and SaW are available in form of “javadocs” on [https://www.netlab.tkk.fi/tutkimus/dtn/theone/javadoc\\_v141/](https://www.netlab.tkk.fi/tutkimus/dtn/theone/javadoc_v141/). Whereas, SGBR is not available on the internet networks. Therefore, we wrote the code of SGBR based on its algorithm that existing in [35]. SGBR code is provided in Appendix B along with the modification in SGBR\_V2.

#### 4.2 Performance Metrics

- **Delivery Ratio**

The proportion of the total delivered packets to the total created packets.

$$Delivery\ Ratio = \frac{\sum_i Packets\ Delivered}{\sum_i Packets\ Sent} \quad (4.1)$$

Equation (4.1) is provided in [35].

- **Average Hop Count**

It is defined as the ratio of the total number of every message copy’s overall hops to the sum of created messages.

$$Average\ Hop\ Count = \frac{\sum_{i=1}^N Ph_i}{N} \quad (4.2)$$

Where  $Ph$  is a number of hop count for each delivered packet and  $N$  is the

- **Overhead Ratio**

It shows the amount of the utilization of communication resources that needed to deliver one packet to final recipient and is defined by the ONE simulator as:

$$\text{Overhead Ratio} = (P_r(t) - P_d(t)) / P_d(t) \quad (4.3)$$

Where  $P_r$  is the total number of packets relayed by time  $t$  and  $P_d$  is the total number of packets delivered by time  $t$ .

- **Average Latency**

It refers to the average amount of time takes all delivered packets to move from source to destination.

$$\text{Average Latency} = \frac{\sum_{n=1}^N T_{del_n} - T_{init_n}}{N} \quad (4.4)$$

Where  $T_{init}$  is time of creation of packet  $n$ ,  $T_{del}$  is time that node  $n$  is delivered to its destination and  $N$  is total of delivered packets.

- **Total Dropped Packet**

The summation of dropped packets for each created packet.

$$\text{Total Dropped Packet} = \sum_{n=1}^N Dr_n \quad (4.5)$$

Where  $Dr$  is dropped packet for each created packet and  $N$  is total of created packets.

Equations (4.3), (4.4) and (4.5) are provided in Javadoc documentation that exist on the ONE simulator page <https://www.netlab.tkk.fi/tutkimus/dtn/theone>.

## 4.3 Simulator Setup and Parameters

### 4.3.1 ONE Simulator

AT Helsinki University of Technology they proposed the Opportunistic Network Environment simulator [41]. It is an agent-based discrete event simulator. Ari Keranen who presents the ONE simulator utilizes time slicing approach [42] to make it adequate and sufficiently productive for simultaneous routing and movement simulation. The ONE is a java-based software which supplies DTN protocol simulation abilities within a single framework. Figure 4.1 shows the elements of ONE simulator and their interaction.

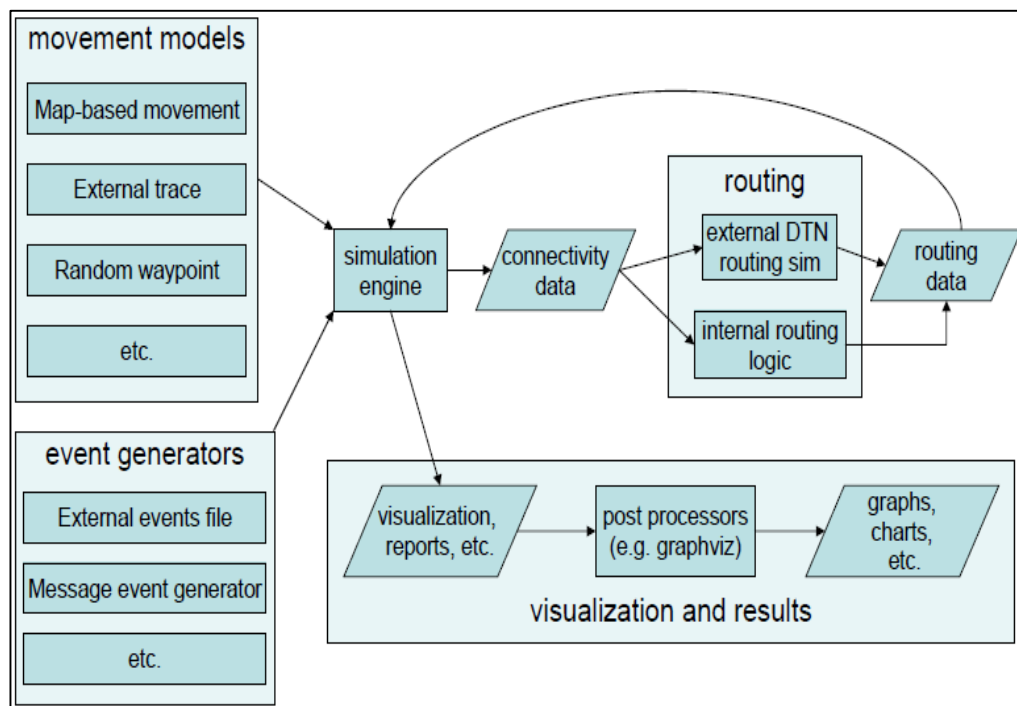


Figure 4. 1: The ONE structure [41].

- **Why ONE simulator**

For DTN environment, there are several simulators along with the ONE simulator available such as, OMNet++, OPNET, DTNSim (Delay Tolerant Network Simulator), and NS-2 (Network Simulator, 2000). The ONE is preferable comparing to other the

simulators, since OPNET and OMNet++ are designed to specific research requirements. Thus, have limited support for existing DTN routing protocols. The NS-2 simulator only supports Epidemic routing so it does not support entire DTN features while there are shortage in movement models in DTNSim.

- **ONE simulation setup**

As we stated previously the ONE simulator is java-based so it only requires any java environment for instance, Eclipse to work well. The project of ONE simulator is available on (<https://akeranen.github.io/the-one>).

- **Running simulations**

The ONE can be operated in two distinct modes: GUI and batch. Batch mode can be utilized for running a large number of simulations with various sets of parameters and the GUI mode is particularly valuable for testing, investigating and exhibition purposes. Both modes can comprise any number of report forms which create and provide statistics of the simulation. These statistics are analyzed and summarized as charts and plots.

#### **4.3.2 Simulation Setup**

There are a lot of parameters that affect the routing performance. The table below consists of the most parameters that we used in our simulation. Some of these parameters are fixed such as, world size, simulation time and transmission speed. Whereas, there are changeable parameters like buffer size, Time To Life (TTL) and number of nodes also, there are some parameter values are different since we use two types of nodes which are pedestrians, P and vehicles, V.

Table 4. 1: Simulation parameters

Parameters	Network size			
	Small		Large	
Node type	P	V	P	V
World Size (meter * meter)	4500 * 3400			
No of nodes	5	5	40	20
Node movement speed (m/second)	0.5 - 1.5	2.7 - 13.9	0.5 - 1.5	2.7 - 13.9
Buffer size (MB)	2,4,6,8,10		5,10,15,20,25	
Packet inter-arrival time (second)	10,30,60, 300 and 600			
Packet Time To Life TTL(Hour)	2,4,6,8 and10			
Transmission speed (MB / second)	5			
Node movement model	Shortest Path Map-Based Movement (SPMBM)			
Packet size (KB)	250-500		500-1024	
Simulation time (Hour)	12			

#### 4.4 Simulation Results

In this section we exhibit results from two main scenario simulations in term of the network size: small network that has only 10 nodes (5 pedestrians (P) and 5 vehicles (V)) and large network that consists 60 nodes (40 pedestrians (P) and 20 vehicles (V)).

We concentrated on the performance metrics: Delivery Ratio, Average Latency, Average Hop Count, Total Dropped Packet and Overhead Ratio by varying some of parameters such as buffer size, packet TTL, packet inter-arrival time, packet size and number of nodes to investigate the impact of these parameters on the performance. We applied all of scenarios on Helsinki city map in map-based model that exist in ONE simulator, also we run each sub scenario eleven times for each protocols, and then we presented the average of values.

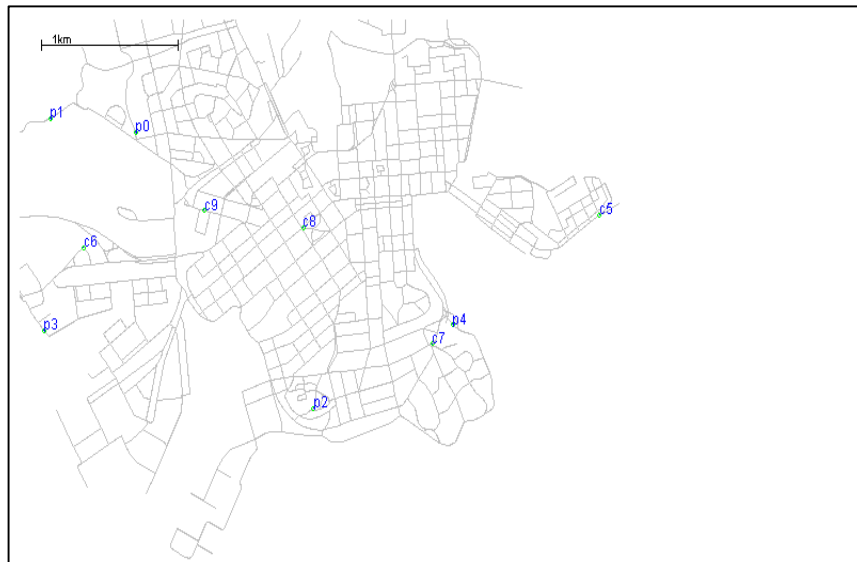


Figure 4. 2: Helsinki map in ONE simulator with 10 nodes

#### 4.4.1 Impact of Buffer Size

We used two different sets of buffer size values to evaluate protocols performance in terms of Delivery Ratio and Average Hop Count, in small network (with 10 nodes) the values of set are (2MB, 4MB, 6MB, 8MB and 10MB). Whereas, in large network (with 60 nodes) the values of set are (5MB, 10MB, 15MB, 20MB and 25MB). Other parameters are fixed such as, TTL is 4 Hours, TL= 300 seconds and Packet Size is 250KB-500KB in a small network or 500KB-1MB in a large network.

- Small Network

For Delivery Ratio both SGBR and SGBR\_V2 a outperformed in the scenario that uses the smallest buffer capacity since they do not require a large buffer size as well as MaxProp exhibits high performance. The protocols that based-knowledge, such as, SGBR, SGBR\_V2 and MaxProp do not tend to distribute a high number of packet copies. Therefore, the varied of buffer size affect slightly their Delivery Ratio. Whereas, it affects considerably the performance of other protocols like SaW, PROPHET and Epidemic. Table 4.2 and Figure 4.3 show the impact of buffer size on the Delivery Ratio.

Table 4. 2: Impact of buffer size on Delivery Ratio in a small network.

Protocol	Buffer Size (MB)				
	2	4	6	8	10
SGBR	0.72	0.87	0.88	0.88	0.88
SGBR_V2	0.72	0.87	0.88	0.88	0.88
MaxProp	0.69	0.86	0.89	0.89	0.89
PROPHET	0.54	0.68	0.75	0.78	0.81
SaW	0.64	0.78	0.83	0.85	0.86
Epidemic	0.46	0.60	0.70	0.77	0.80

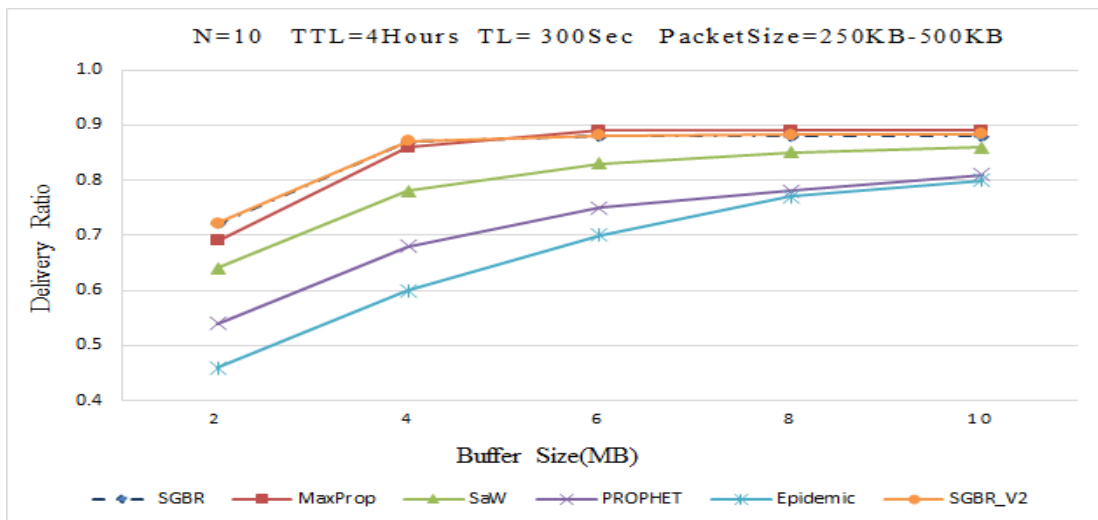


Figure 4. 3: Impact of buffer size on Delivery Ratio in a small network.



For Average Hop Count Table 4.3 and Figure 4.4 illustrates that SaW has the lowest average because of it does not take into account the cost of delivery, so it just distributes packet copies to all nodes that it encounters as Epidemic. However, SaW restricts the number of packet copies while Epidemic does not. MaxProp and PROPHET show that their averages are increased by increasing the buffer size. The reason is increasing the buffer size means that each node can carry more packet so the drop ratio of these packets will increase. Thus, the Average Hop Count will rise. The Average Hop Count of social protocols are dropped slowly by increasing the buffer size, because they propagate a fixed number of packet copies so dropped of packet only depends on TTL. Therefore, each node can travel with their packets until reaching it to their destination. Finally, since Epidemic is blind protocol, i.e. it sends packets to all nodes that encounters, so its Average Hop Count curve is raised by increasing the buffer size unless this buffer is enough to store all packets that receive.

Table 4. 3: Impact of buffer size on Average Hop Count in a small network.

Protocol	Buffer Size (MB)				
	2	4	6	8	10
SaW	1.94	2.02	2.02	2.03	2.04
MaxProp	2.04	2.17	2.20	2.20	2.20
SGBR	2.45	2.28	2.18	2.18	2.18
SGBR_V2	2.44	2.30	2.20	2.20	2.20
PROPHET	2.08	2.34	2.42	2.37	2.34
Epidemic	2.94	3.49	3.55	3.60	3.26

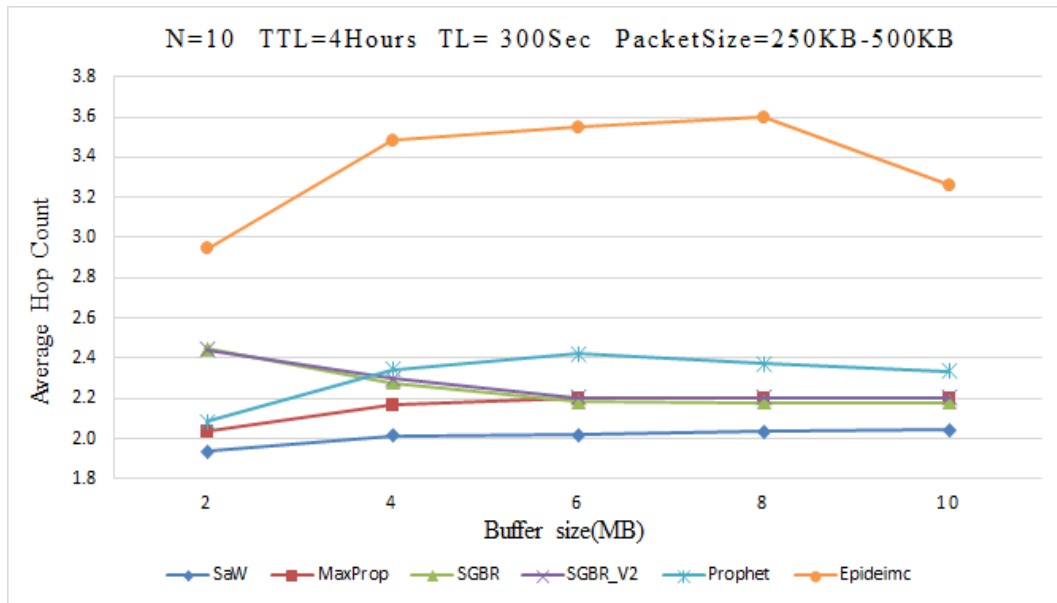


Figure 4. 4: Impact of buffer size on Average Hop Count in a small network.

- Large network

MaxProp, SGBR, SGBR\_V2 and SaW show the same behavior in terms of Delivery Ratio and Average Hop Count that they not influenced by varying the buffer size in large network. While, the Delivery Ratio of PROPHET and Epidemic is enhanced rapidly by increasing the buffer size, because the greater the number of packet copies, the greater the chance of packet delivery as in Table 4.4 and Figure 4.5. Also it decrements the average of Average Hop Count slightly as in Table 4.5 and Figure 4.6.

Table 4. 4: Impact of buffer size on Delivery Ratio in a large network.

Protocol	Buffer Size(MB)				
	5	10	15	20	25
MaxProp	0.96	0.96	0.96	0.96	0.96
SGBR	0.93	0.93	0.93	0.93	0.93
SGBR_V2	0.90	0.91	0.91	0.91	0.91
PROPHET	0.55	0.67	0.75	0.82	0.88
SaW	0.90	0.91	0.91	0.91	0.91
Epidemic	0.34	0.52	0.67	0.77	0.85

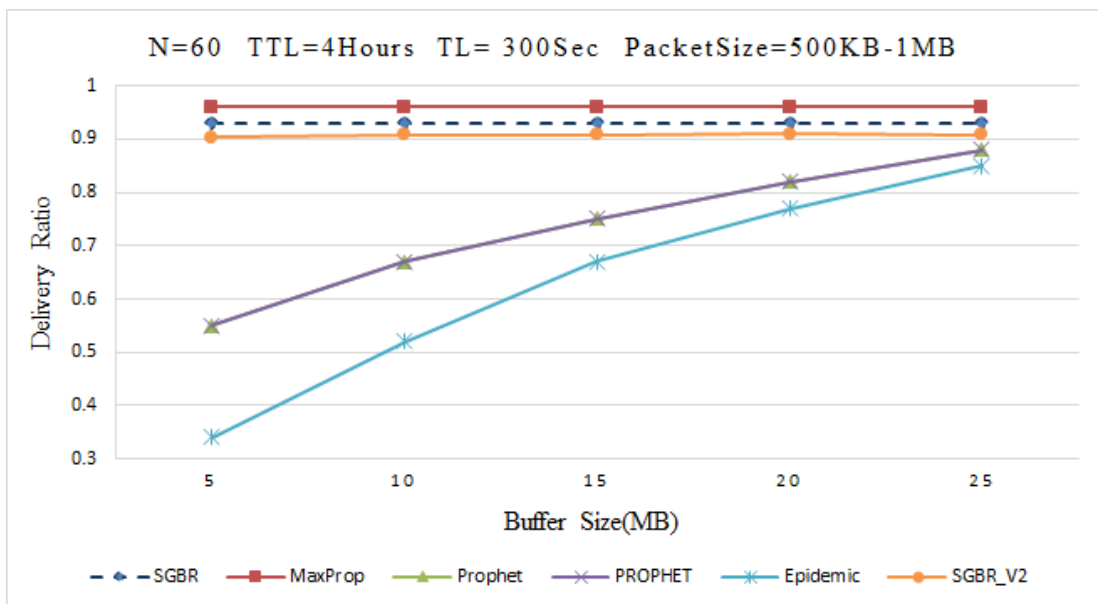


Figure 4. 5: Impact of buffer size on Delivery Ratio in a large network.

Table 4. 5: Impact of buffer size on Average Hop Count in a large network.

Protocol	Buffer Size(MB)				
	5	10	15	20	25
SaW	2.34	2.33	2.33	2.33	2.33
SGBR	2.64	2.61	2.61	2.60	2.60
SGBR_V2	2.65	2.63	2.62	2.63	2.64
MaxProp	3.60	3.63	3.63	3.63	3.63
PROPHET	3.35	3.26	3.13	3.12	2.97
Epidemic	5.44	5.58	5.20	4.99	4.59

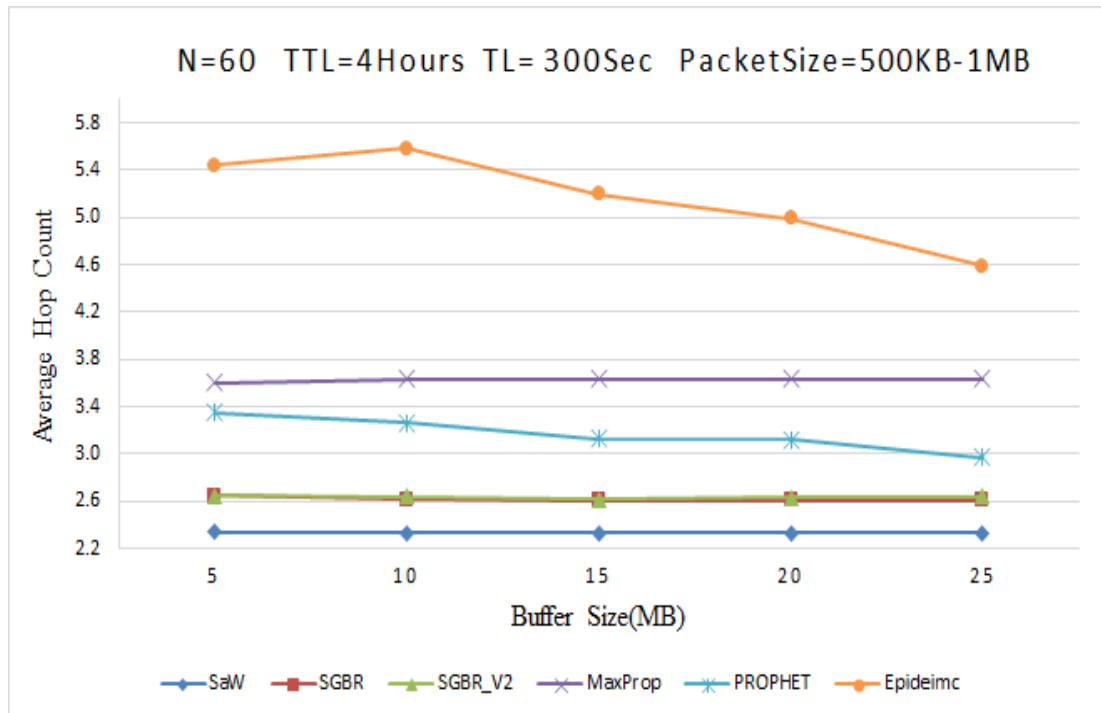


Figure 4. 6: Impact of buffer size on Average Hop Count in a large network.

#### 4.4.2 Impact of TTL (Time To Life)

We applied a set of TTL values to evaluate the impact of TTL on protocol performance in terms of Delivery Ratio and Total Dropped Packets. In both small and large networks scenarios use the same values of the set which are (2, 4, 6, 8 and 10) Hours.

Other parameters are fixed as follows:

Buffer Size is 10MB, TL is 300 second and packet size is 250KB-500KB in a small network or 500KB-1MB in a large network.

- Small network

For Delivery Ratio, all protocols reach to their top performance when TTL is 4 Hours. After that they demonstrate two distinct behaviors: 1) MaxProp, SGBR, SGBR\_V2 and SaW Respectively that remain stable without any effected by changing of TTL as in Table 4.6 and Figure 4.7.

Table 4. 6: Impact of TTL on Delivery Ratio in a small network.

Protocol	TTL(Hours)				
	2	4	6	8	10
MaxProp	0.78	0.89	0.89	0.89	0.89
SGBR_V2	0.77	0.88	0.89	0.89	0.89
SGBR	0.77	0.88	0.89	0.89	0.89
SaW	0.73	0.86	0.86	0.86	0.86
Proohet	0.74	0.82	0.79	0.77	0.76
Epidemic	0.77	0.80	0.76	0.72	0.70

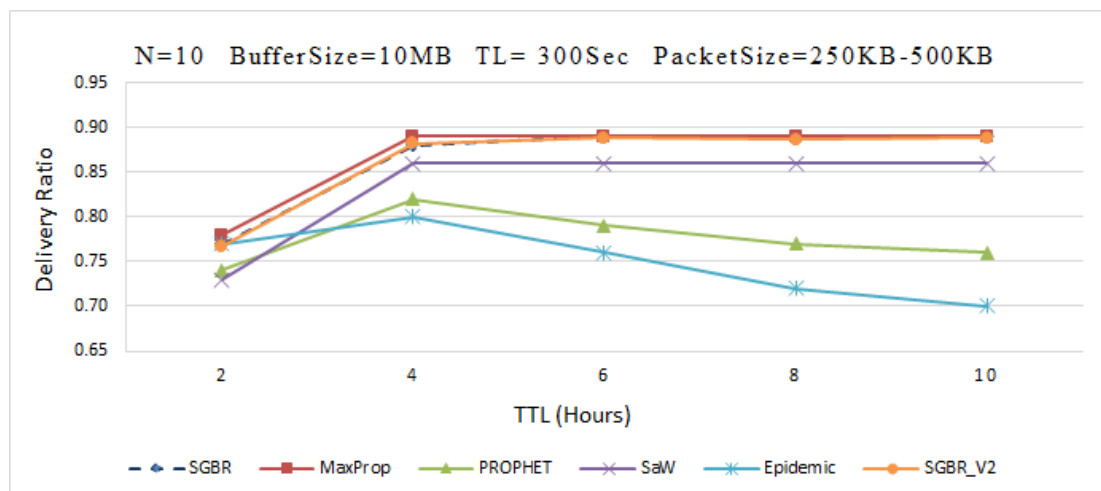


Figure 4. 7: Impact of TTL on Delivery Ratio in a small network.

Due to they send a limited number of packet copies, so when TTL is increment the Total Dropped Packets will decrement until reaches to zero as in Table 4.7 and illustrated in Figure 4.8. Thus, the Delivery Ratios for these protocols will be high as long as TTL is increasing; 2) PROPHET and Epidemic they show that while TTL is increasing their Delivery Ratios are reduced as a result of a considerable number of dropped packets as shown in Table 4.7 and Figure 4.8.

Table 4. 7: Impact of TTL on Total Dropped Packet in small network.

Protocol	TTL(Hours)				
	2	4	6	8	10
SGBR	176.6	28.4	1.6	0.2	0.0
SGBR_V2	178.1	27.5	1.7	0.2	0.0
sMaxProp	238.7	26.9	1.6	0.2	0.0
SaW	478.5	424.3	385.9	361.3	343.7
PROPHET	602.8	3063.0	4339.5	4886.6	5093.4
Epidemic	912.2	6138.8	7372.1	7496.6	7653.0

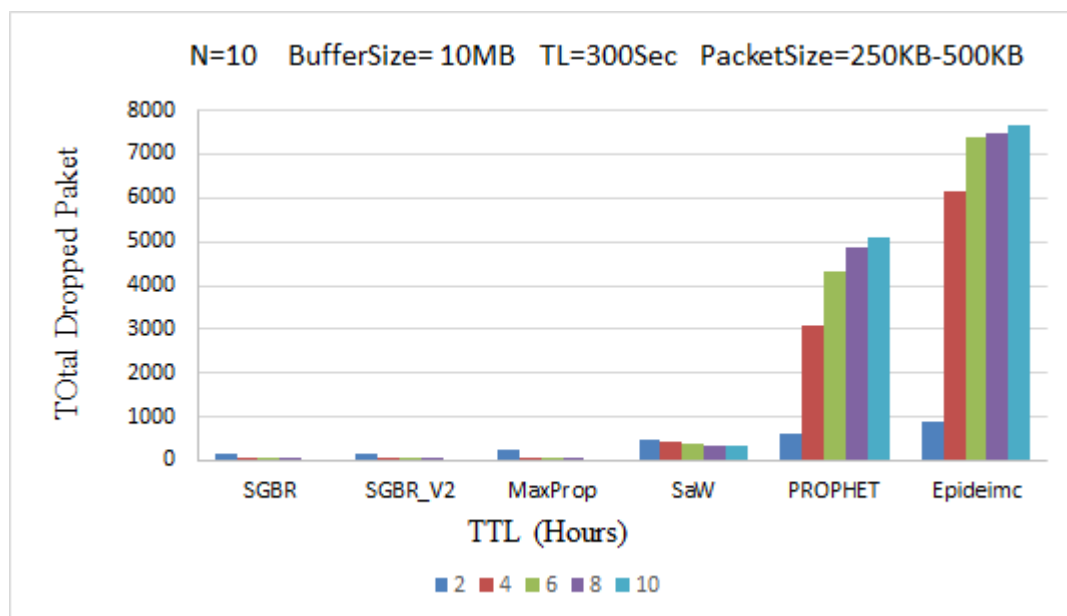


Figure 4. 8: Impact of TTL on Total Dropped Packet in a small network.

- Large network

In a large network, protocols shown two different attitudes too: 1) MaxProp, SGBR, SGBR\_V2 and SaW act similar to small network in terms of Delivery Ratio and Total Dropped Packet as in Figures 4.9 and 4.10; 2) PROPHET and Epidemic they instead of TTL is 4H in small network scenario they achieve the top performance when TTL is 2H then their performance decrement when increment TTL value as in Table 4.8 and Table 4.9, also they illustrated in Figures 4.9 and 4.10.

Table 4. 8: Impact of TTL on Delivery Ratio in a large network.

Protocol	TTL(Hours)				
	2	4	6	8	10
MaxProp	0.93	0.94	0.94	0.94	0.94
SGBR	0.83	0.91	0.91	0.91	0.91
SGBR_V2	0.82	0.91	0.91	0.91	0.91
SaW	0.76	0.87	0.89	0.89	0.89
PROPHET	0.90	0.81	0.73	0.67	0.62
Epidemic	0.93	0.75	0.65	0.60	0.56

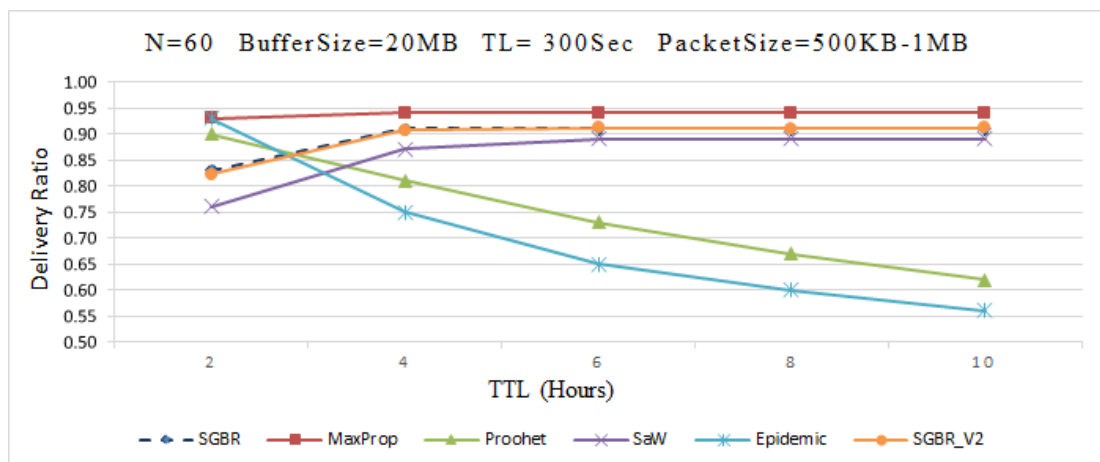


Figure 4. 9: Impact of TTL on Delivery Ratio in a large network

Table 4. 9: Impact of TTL on Total Dropped Packet in a large network.

Protocol	TTL(Hours)				
	2	4	6	8	10
SGBR	246.6	24.5	0.2	0.0	0.0
SGBR_V2	255.4	29.5	0.7	0.3	0.3
MaxProp	316.8	1.0	0.0	0.0	0.0
SaW	587.1	469.5	352.9	240.7	119.5
PROPHET	4533.1	80360.3	99573.2	106594.3	113723.0
Epidemic	7114.6	123001.4	136742.9	139439.9	141058.5

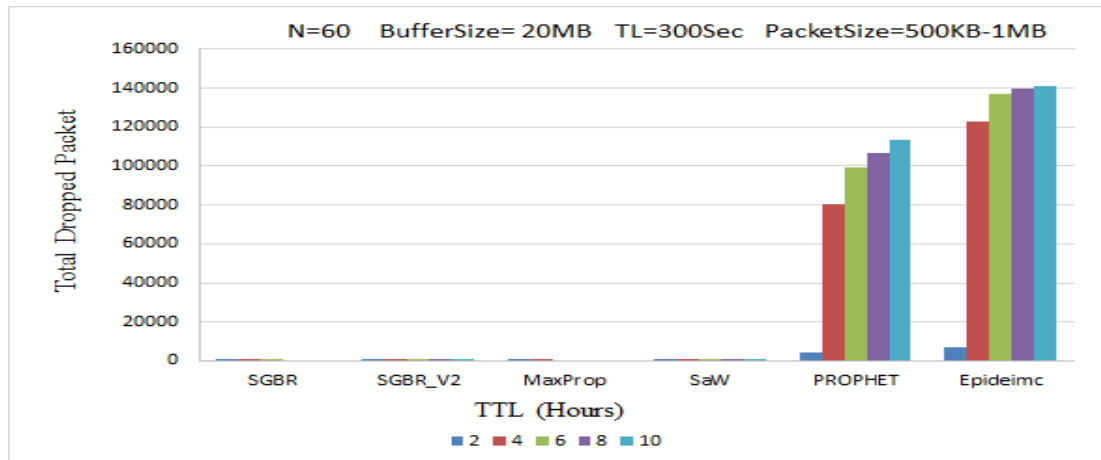


Figure 4. 10: Impact of TTL on Total Dropped Packet in a large network.

#### 4.4.3 Impact of TL (Traffic Load)

We applied five different packet generation rates in this experiment to assess the performance in terms of Delivery Ratio and Overhead Ratio. The average packet generation rates as follows 379, 121, 66, 12, and 6 packets/hour, these values are found from the intervals: (8–12, 25–35, 55–65, 250–350, and 550–650) sec, respectively. The intervals determine how many packets will be created per hour during the simulation. So, it refers to the Network Traffic Load (TL) for each scenario. For instance, if we run the simulation for 12 Hours and the interval is 550-650 seconds, so the total of generation packets during whole simulation is:  $12 * 6 = 72$  packets. Other parameters are fixed as follows:



Buffer size is 10MB, TTL is 4 Hours, packet size is 250KB-500KB in a small network or 500KB-1MB in a large network and number of nodes is 10 in a small network or 60 in a large network.

Expanding the traffic load in a network with a settled number of nodes causes the overburdening of buffers and builds up the dropping rate. In this manner, the Delivery Ratio drops considerably. Hence, the Delivery Ratio for all protocols are decreased by increment the packet generation rate, i.e. the greater of network traffic means the lesser of Delivery Ratios. Table 4.10, Table 4.11, Figure 4.11 and 4.12 show that SGBR\_V2, SGBR, MaxProp and SaW outperformed PROPHET and Epidemic in both scenarios.

Table 4. 10: Impact of TL on Delivery Ratio in a small network.

Protocol	TL(Packet/Hour)				
	6	12	66	121	379
SGBR_V2	0.90	0.91	0.91	0.78	0.42
SaW	0.87	0.87	0.80	0.68	0.42
SGBR	0.90	0.91	0.91	0.75	0.40
MaxProp	0.95	0.94	0.83	0.65	0.38
PROPHET	0.78	0.68	0.47	0.39	0.26
Epidemic	0.71	0.53	0.32	0.26	0.18

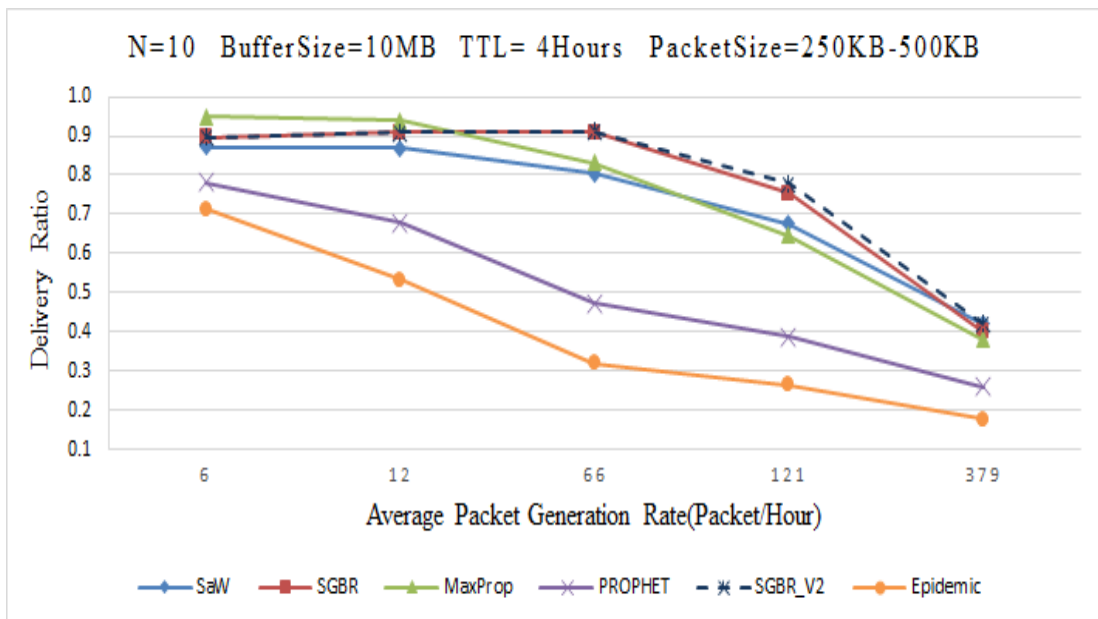


Figure 4. 11: Impact of TL on Delivery Ratio in a small network.

Table 4. 11: Impact of TL on Delivery Ratio in a large network.

Protocol	TL(Packet/Hour)				
	6	12	66	121	379
SaW	0.85	0.86	0.68	0.54	0.31
MaxProp	0.88	0.89	0.76	0.56	0.30
SGBR_V2	0.87	0.88	0.78	0.57	0.30
SGBR	0.87	0.88	0.78	0.56	0.30
PROPHET	0.85	0.82	0.61	0.49	0.29
Epidemic	0.88	0.80	0.51	0.39	0.24

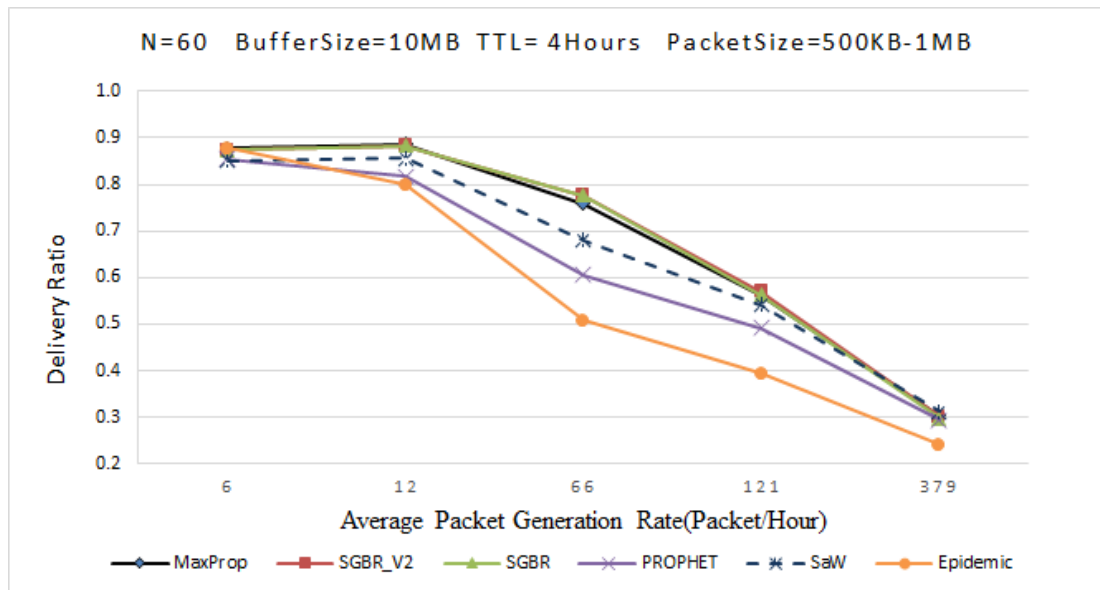


Figure 4. 12: Impact of TL on Delivery Ratio in large network.

The protocols show three different attitudes related to Overhead Ratio: 1) SaW, SGBR\_V2 and SGBR that presented the lowest Overhead Ratio, since they propagate a limited amount of packet copies through the network. For these protocols the Overhead Ratio grows slowly when traffic load increased; 2) MaxProp shows medium values for Overhead Ratio. Although, it distributes a low number of packet copies, it runs a lot of operations to find the closest path for each packet; 3) PROPHET and SaW have the highest Overhead Ratio since they send unlimited number of packet copies in order to increase the probability of delivering each packet. Those protocols have a reverse ratio with traffic load, i.e. when the traffic load increased their Overhead Ratio

decreased, due to when a network generate a lot of original packets with limited buffers that means increases of dropped packets and reduce of delivery ratio that effect on Overhead Ratio. Table 4.12, Table 4.13, Figure 4.13 and 4.14 demonstrates the impact of TL on Overhead Ratio in small and large network, respectively.

Table 4. 12: Impact of TL on Overhead Ratio in a small network.

Protocol	TL(Packet/Hour)				
	6	12	66	121	379
SaW	3.7	3.5	4.5	4.9	5.4
SGBR_V2	4.3	4.3	10.6	10.4	7.7
SGBR	4.3	4.3	11.2	11.3	8.1
MaxProp	5.0	4.7	24.8	22.6	13.7
PROPHET	5.3	26.7	17.4	11.8	6.6
Epidemic	8.1	54.3	30.3	21.1	11.7

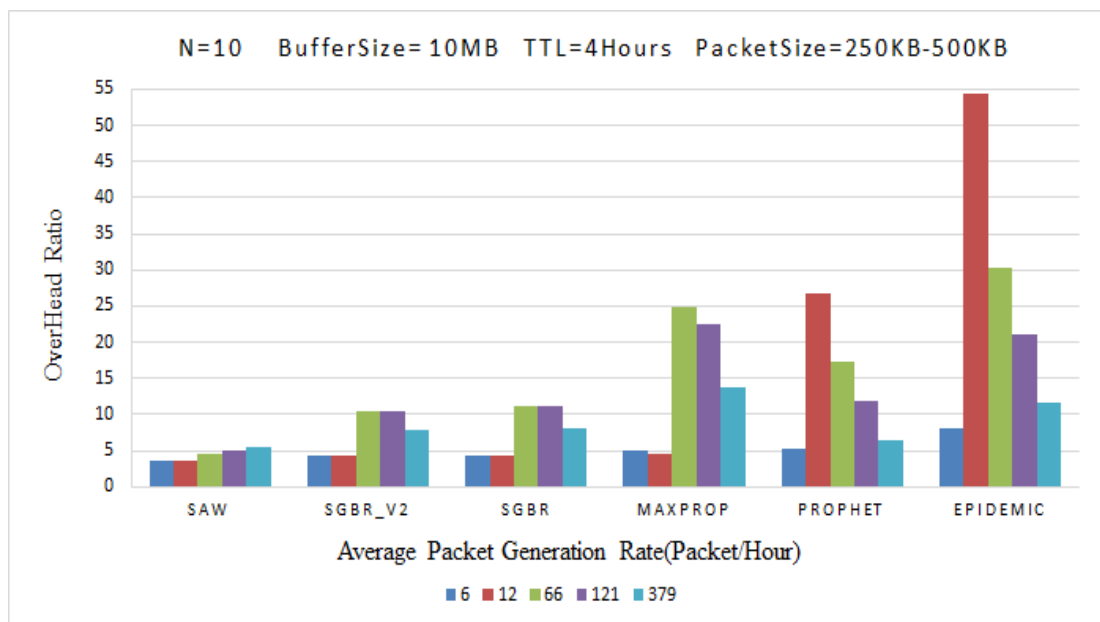


Figure 4. 13: Impact of TL on Overhead Ratio in a small network.

Table 4. 13: Impact of TL on Overhead Ratio in a large network.

Protocol	TL(Packet/Hour)				
	6	12	66	121	379
SaW	4.4	4.3	4.7	5.6	8.5
SGBR_V2	6.1	8.0	16.4	17.8	21.9
SGBR	6.3	8.7	18.7	21.3	29.0
MaxProp	38.1	39.0	359.7	273.5	159.3
PROPHET	653.3	649.8	275.6	180.8	95.3
Epidemic	855.3	1051.3	496.2	317.0	162.4

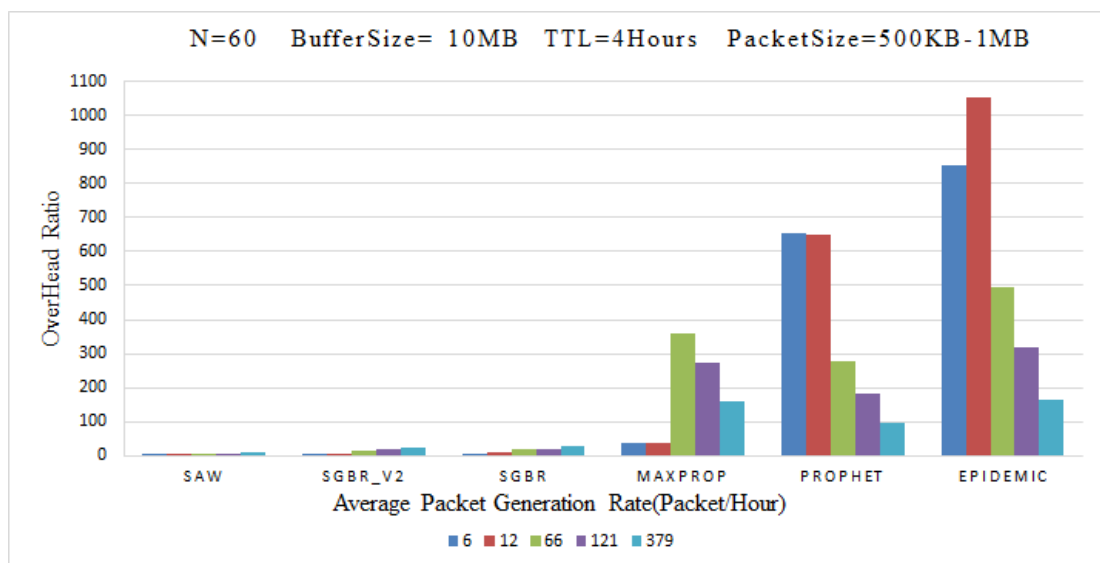


Figure 4. 14: Impact of TL on Overhead Ratio in a large network.

#### 4.4.4 Impact of Packet Size

We used two sets of various packet sizes to evaluate average of Average Latency for each protocol. the first set has a fixed packet size (250KB, 500KB, 1MB, 2MB), while the second one has changeable packet size (250KB-500KB, 500KB-1MB, 1MB-2MB). Whether in small or large network the performance of each protocol is same.

Other parameters are fixed as follows:

Buffer size is 10MB, TTL is 4 Hours and number of nodes is 10 in a small network or 60 in a large network.

For Delivery Ratio, all protocols have same behavior which is when the packet size is increment the Delivery Ratio is going down, due to when increase the packet size with limited buffers that means each node cannot carry a lot of packets, so it causes surging from of the dropped packet rate and reducing the chance of packet delivery as in Tables 4.14 and 4.15, also they figured in Figure 4.15 for small network and Figure 4.16 for large network.

Table 4. 14: Impact of packet size on Delivery Ratio in a small network.

Protocol	Packet Size (Bytes)						
	250K	500K	1M	2M	250K-500K	500K-1M	1M-2M
SGBR	0.90	0.90	0.89	0.75	0.91	0.91	0.90
MaxProp	0.90	0.90	0.89	0.74	0.94	0.94	0.93
PROPHET	0.87	0.80	0.70	0.58	0.81	0.67	0.56
SaW	0.87	0.86	0.80	0.66	0.87	0.87	0.86
Epidemic	0.89	0.78	0.62	0.49	0.73	0.53	0.37
SGBR_V2	0.90	0.90	0.89	0.77	0.91	0.91	0.91

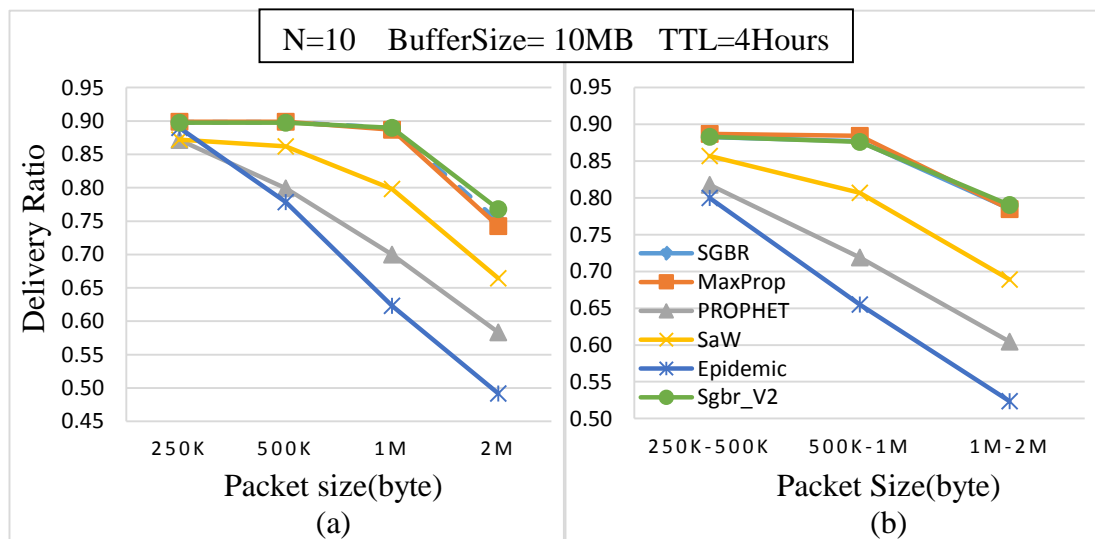


Figure 4. 15: Impact of packet size on Delivery Ratio in a small network.

Table 4. 15: Impact of packet size on Delivery Ratio in a large network.

Protocol	Packet Size (Bytes)						
	250k	500k	1M	2M	250k-500k	500k-1M	1M-2M
SGBR	0.93	0.93	0.93	0.92	0.91	0.91	0.90
MaxProp	0.96	0.96	0.96	0.95	0.94	0.94	0.93
PROPHET	0.92	0.76	0.64	0.53	0.81	0.67	0.56
SaW	0.88	0.88	0.87	0.85	0.87	0.87	0.86
Epidemic	0.92	0.65	0.47	0.36	0.73	0.53	0.37
SGBR_V2	0.92	0.92	0.92	0.92	0.91	0.91	0.91

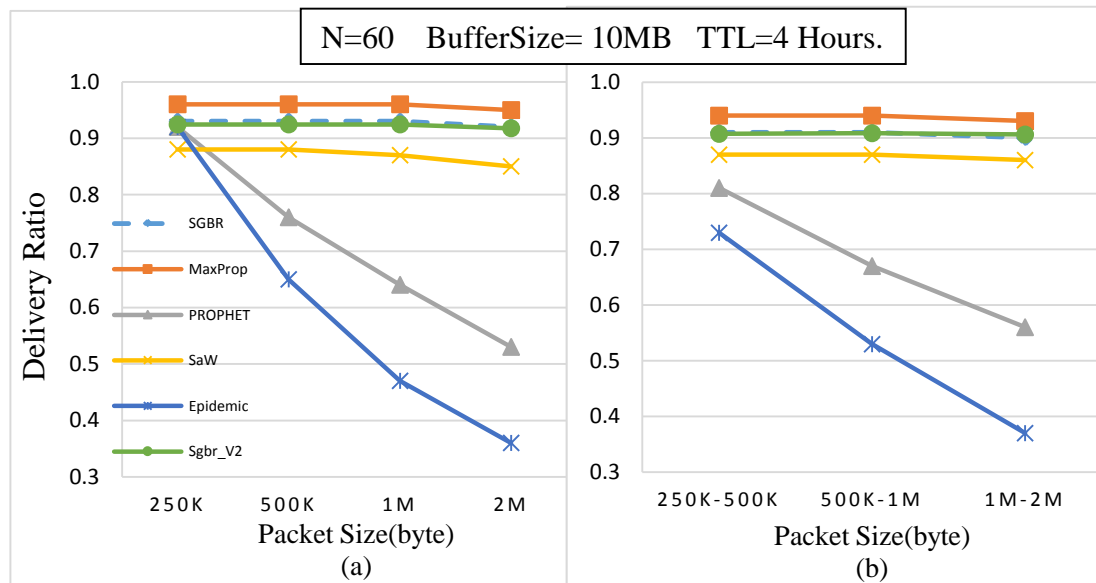


Figure 4. 16: Impact of packet size on Delivery Ratio in a large network.

For Average Latency, all protocols in small network simulation show that by rising packet size the average of the Average Latency is reduced as in Table 4.16 and Figure 4.17. Whereas, in a large network scenario in Table 4.17 and Figure 4.18 all protocols presented that average of the Average Latency increased if the packet size is increment decreased except SaW since it always send the same number of packet copies.

Table 4. 16: Impact of packet size on Average Latency in a small network.

Protocol	Packet Size (Bytes)						
	250K	500K	1M	2M	250K-500K	500K-1M	1M-2M
SaW	3298	3205	2802	2223	3322	2948	2381
Epidemic	3138	3116	2830	2396	3166	2945	2582
SGBR	3151	3151	3177	2808	3264	3272	3072
SGBR_V2	3170	3162	3182	2872	3261	3248	3078
PROPHET	3390	3375	3312	2905	3540	3455	3115
MaxProp	3109	3109	3106	2950	3192	3249	3164

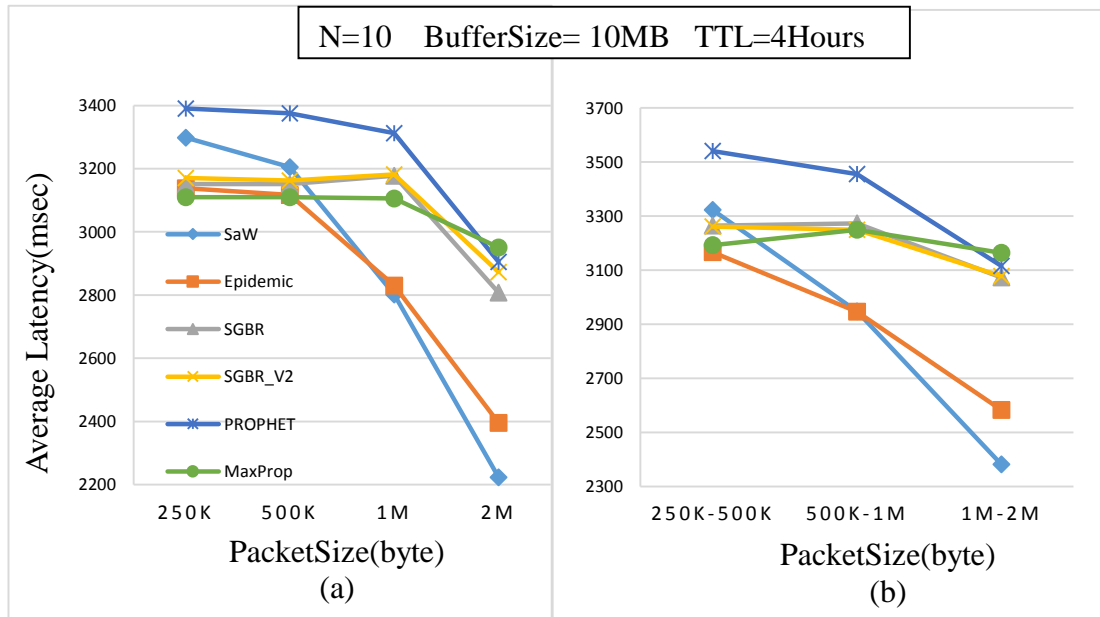


Figure 4. 17: Impact of packet size on Average Latency in a small network.



Table 4. 17: Impact of packet size on Average Latency in a large network.

Protocol	Packet Size (Bytes)						
	250K	500K	1M	2M	250K-500K	500K-1M	1M-2M
MaxProp	1868	1868	1870	2010	1921	1924	2012
Epidemic	2026	2656	2897	2990	2507	2756	2772
SaW	3169	3169	3162	3024	3362	3357	3264
SGBR	2990	2984	2971	3081	3034	3023	3090
SGBR_V2	3025	3024	3031	3085	3084	3097	3152
PROPHET	2508	3350	3732	3731	3065	3710	3874

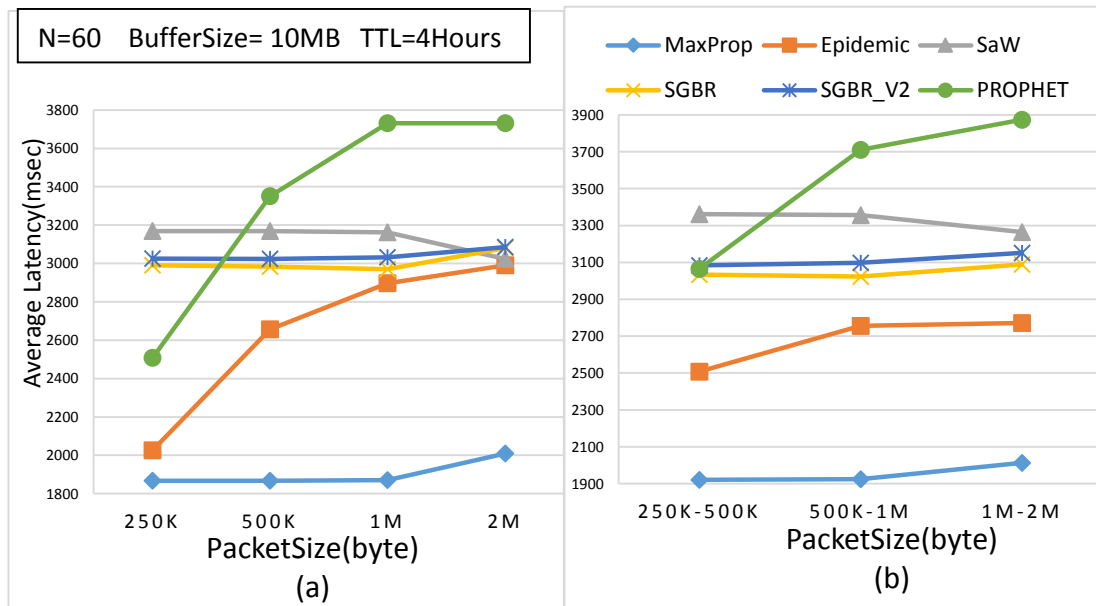


Figure 4. 18: Impact of packet size on Average Latency in a large network.

#### 4.4.5 Impact of the Number of Nodes

we run five scenarios with different values in terms of number of nodes as follows (20, 40, 60, 80 and 100). In this experiment, we try to study the impact of the number of nodes on three metrics that are Delivery Ratio, Total Dropped Packets and Overhead Ratio. Other parameters are fixed as follows:

Buffer size is 10MB, TTL is 4Hours, TL is 300 seconds and packet size is 500KB-1MB.

For those three metrics, the protocols showed two distinct attitudes: 1) the protocols as, MaxProp, SGBR, SGBR\_V2 and SaW that distribute a limited number of packet copies. They haven't affected substantially by changing the number of nodes and they keep a high performance of Delivery Ratio as in Table 4.18 and Figure 4.19, also low level in total of the dropped packet as in Table 4.19 and Figure 4.20 and Overhead Ratio as showed in Table 4.20 and Figure 4.21. ; 2) PROPHET and Epidimc that use an unrestrained number of packet copies, so by increasing the number of nodes the traffic load will grow up and causes increment the dropped packet rate as well as increasing the Overhead Ratio as in Table 4.19 and Table 4.20, also they depicted in Figure 4.20 and Figure 4.21 respectively. Therefore, the Delivery Ratio of these protocols will decrement while increasing the number of nodes as shown in Table 4.18 and 4.19.

Table 4. 18: Impact on number of nodes on Delivery Ratio.

Protocol	Number of Nodes				
	20	40	60	80	100
SGBR	0.93	0.95	0.91	0.93	0.91
MaxProp	0.94	0.96	0.94	0.97	0.96
Proohet	0.74	0.72	0.68	0.62	0.61
SaW	0.89	0.91	0.87	0.87	0.86
Epidemic	0.64	0.60	0.53	0.53	0.51
SGBR_V2	0.93	0.94	0.91	0.92	0.90

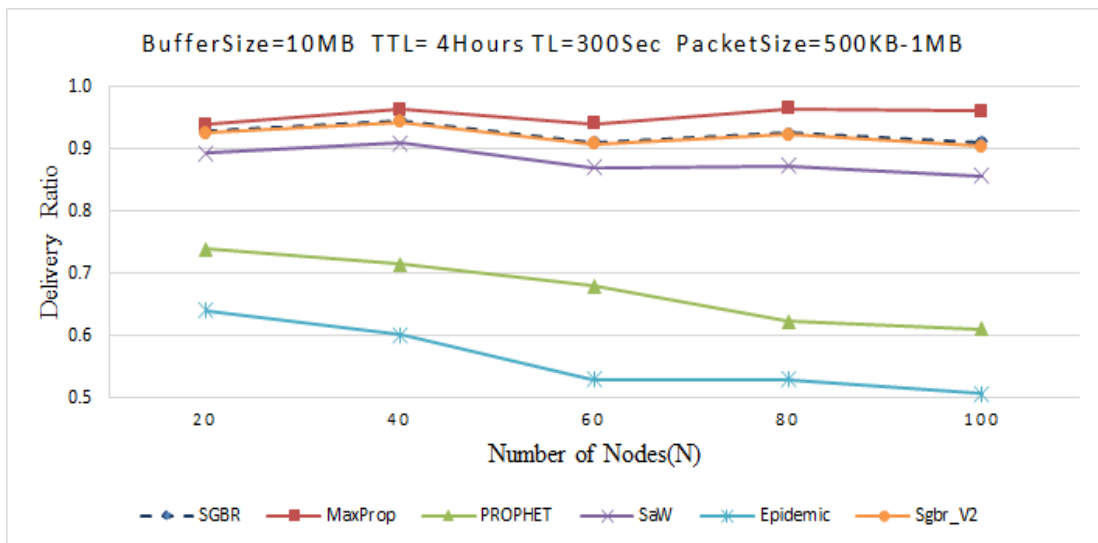


Figure 4. 19: Impact on number of nodes on Delivery Ratio.

Table 4. 19: Impact on number of nodes on Total Dropped Packet.

protocol	Number of Nodes				
	20	40	60	80	100
MaxProp	17	2	1	0	0
SGBR	24	16	0	23	39
SGBR_V2	27	27	30	26	40
SaW	494	466	202	468	473
PROPHET	12580	33951	80023	99213	148554
Epidemic	17880	45627	94496	136049	191940

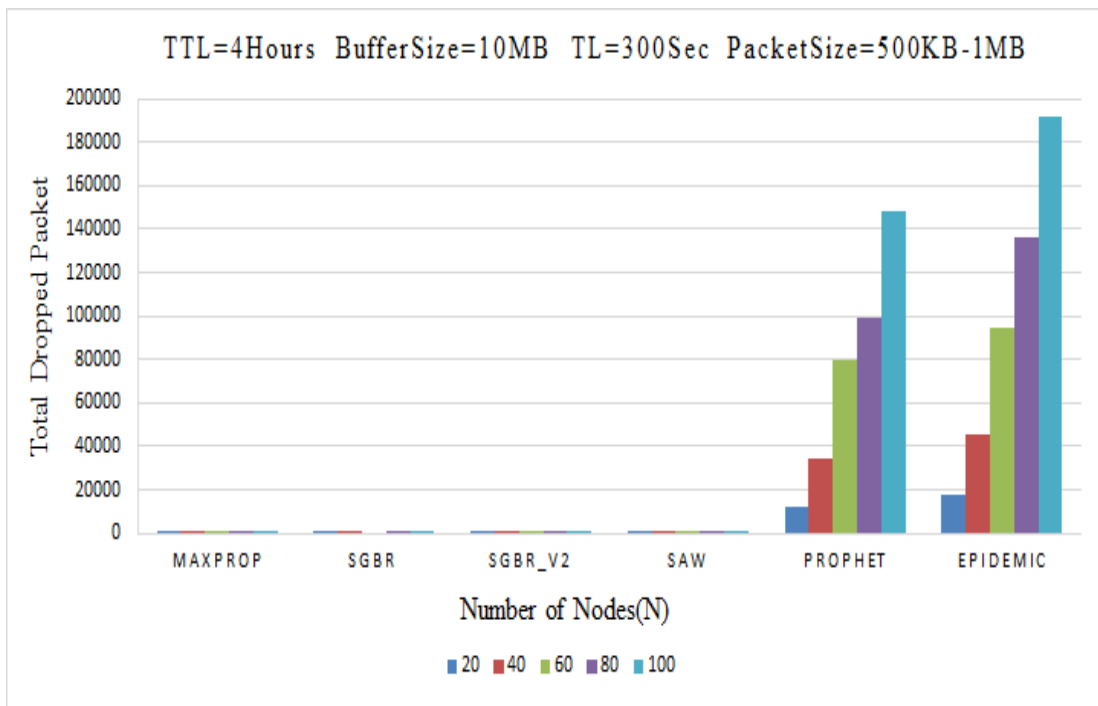


Figure 4. 20: Impact on number of nodes on Total Dropped Packet.

Table 4. 20: Figure 4.21: Impact on number of nodes on Overhead Ratio.

protocol	Number of Nodes				
	20	40	60	80	100
SaW	4	4	4	4	4
SGBR_V2	6	8	8	7	7
SGBR	7	8	8	8	8
MaxProp	10	24	35	53	70
PROPHET	118	336	1114	1138	1693
Epidemic	195	563	1637	1833	2631

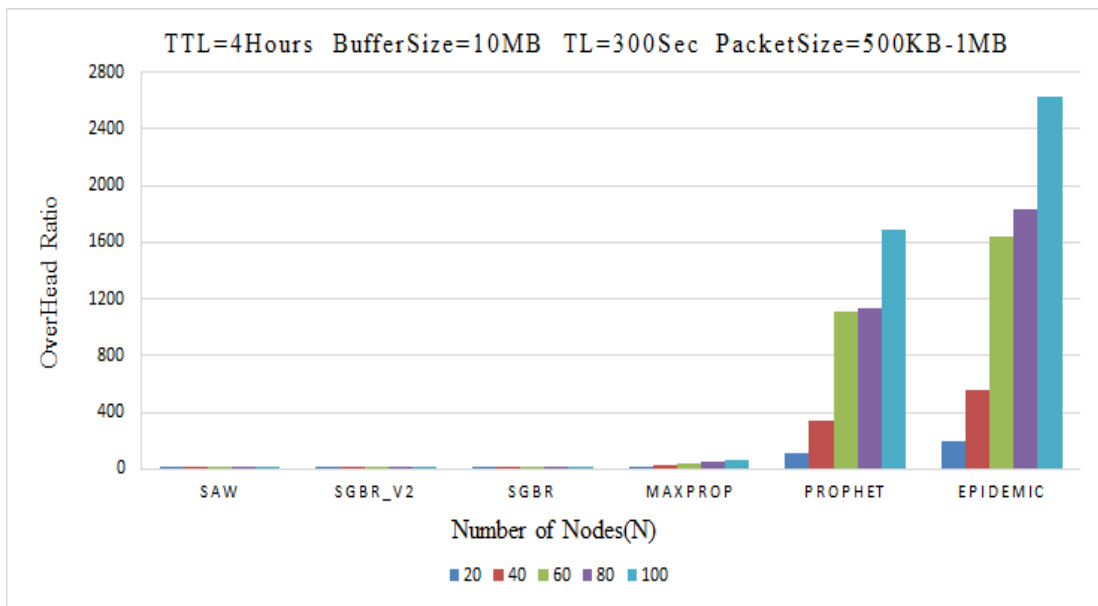


Figure 4. 21: Impact on number of nodes on Overhead Ratio.

## Chapter 5

### CONCLUSION AND FUTURE WORK

DTN, is known as absence of continual end-to-end connections between sources and destination. The nodes in DTNs are demanding to carry and hold on information bundles until they get close to different nodes. Storing of this information might take a long time that disregards one of the fundamental presumptions of ordinary routing algorithms, and excite the presenting of novel ideas.

In this search, we modified SGBR protocol and provided SGBR\_V2 that uses the social relations between nodes by changing the mechanism of spread packets through the network to minimize of distributing duplicate copies. In addition, we provided survey about the impact of packet size that haven't studied previously and other network parameters like, TTL and number of nodes on the most important performance metrics such as, Delivery Ratios and Overhead Ratio. We compared SGBR protocol and SGBR\_V2 protocol to four other protocols: MaxProp, SnW, PROPHET, and Epidemic protocol.

We used the ONE simulator, where the nodes travel in a network according to the Helsinki map that given in the simulator. Simulation results presented that the protocols that use network information to deliver the data through networks such as, SGBR protocol, SGBR\_V2 and MaxProp protocol significantly restrict redundant copies that minimizes the Network Overhead Ratio and the Total Dropped Packet, while keeping higher Delivery Ratio. Moreover, the results showed that SGBR\_V2

outperformed slightly the other protocols in high traffic load environments (up to 24 percent that of the Epidemic protocol in terms of Delivery Ratio and reach to 20 percent that of the Epidemic in average of Average Latency).

Our future work will be aiming implementation of more robust and reliable routing protocol in order to addressing the disadvantages of the existing DTN routing protocols, we will use more social features to reduce the number of packet copies that roam networks and maximizing the Delivery Ratio.

## REFERNCES

- [1] Archana, R. & Gracy, T. (2015, September). Survey on Attacks in MANET. *International Journal of Advanced Research in Computer and Communication Engineering*, 4, (9), 125-31.
- [2] Ramesh, S., & Indira, R. (2016). Ant Routing Protocol with Location Services in Intermittently Connected MANETs, *Circuits and Systems*, 7(07), 1087.
- [3] Perkins, C., Belding-Royer, E., & Das, S. (2003). Ad Hoc On-Demand Distance Vector (Aodv) Routing (RFC 3561).
- [4] Johnson, D., Hu, Y. C., & Maltz, D. (2007). The Dynamic Source Routing (DSR) Protocol For Mobile Ad Hoc Networks for Ipv4 (RFC 4728).
- [5] Chawla, M. (2013). Comparing Delay Tolerant Network Routing Protocols for Optimizing L-Copies in Spray and Wait Routing for Minimum Delay. *Atlantis Press*, 239-244.
- [6] McMahan, A., & Farrell, S. (2009). Delay-and Disruption-Tolerant Networking. *IEEE Internet Computing*, 13(6), 82-87.
- [7] Mangrulkar, R. S., & Atique, M. (2010, October). Routing Protocol for Delay Tolerant Network: A Survey and Comparison. In Communication Control and Computing Technologies (ICCCCT), *IEEE International Conference*, 210-215.



- [8] Shen, J., Moh, S., & Chung, I. (2008). Routing Protocols In Delay Tolerant Networks: A Comparative Survey. In the 23rd International Technical Conference on Circuits/Systems, *Computers and Communications, (ITC-CSCC 2008)*, 6-9.
- [9] Fall, K., Scott, K. L., Burleigh, S. C., Torgerson, L., Hooke, A. J., Weiss, H. S., ... & Cerf, V. (2007). Delay-tolerant networking architecture, 15-18.
- [10] Caini, C., Cornice, P., Firrincieli, R., & Lacamera, D. (2008). A DTN Approach to Satellite Communications. *IEEE Journal on Selected Areas in Communications*, 26(5), 820-827.
- [11] Wang, S., & Ma, R. (2013). NAME: A Naming Mechanism for Delay/Disruption-Tolerant Network. *International Journal of Computer Networks & Communications*, 5(6), 231-241.
- [12] Gao, L., Yu, S., Luan, T. H., & Zhou, W. (2015). Delay Tolerant Networks. *Springer International Publishing*, 9-16.
- [13] Nichols, R. A., Hammons, A. R., Tebben, D. J., & Dwivedi, A. (2007). Delay Tolerant Networking for Free-Space Optical Communication Systems. In Sarnoff Symposium, 2007 *IEEE*, 1-5.
- [14] Chruscicki, M. C., & Hall, F. (2007, October). Airborne Networking Component Architecture and Simulation Environment (AN-CASE). In Military Communications Conference, 2007. MILCOM 2007. *IEEE*, 1-7.

- [15] Fall, K., Iannaccone, G., Kannan, J., Silveira, F., & Taft, N. (2010). A Disruption-Tolerant Architecture for Secure and Efficient Disaster Response Communications. *Proceedings of ISCRAM*, Seattle 2010, 1-5.
- [16] McDonald, P., Geraghty, D., Humphreys, I., Farrell, S., & Cahill, V. (2007). Sensor Network with Delay Tolerance (SeNDT). In *Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on*. 1333-1338.
- [17] Pentland, A., Fletcher, R., & Hasson, A. (2004). Daknet: Rethinking Connectivity in Developing Nations. *Computer*, 37(1), 78-83.
- [18] Zarafshan-Araki, M., & Chin, K. W. (2010). TrainNet: A Transport System for Delivering Non Real-Time Data. *Computer Communications*, 33(15), 1850-1863.
- [19] Seth, A., Kroeker, D., Zaharia, M., Guo, S., & Keshav, S. (2006). Low Cost Communication for Rural Internet Kiosks Using Mechanical Backhaul. In *Proceedings of the 12th Annual International Conference, In Computer Communications and Networks ACM*, 334-345.
- [20] Glance, N., Snowdon, D., & Meunier, J. L. (2001). Pollen: Using People as a Communication Medium. *Computer Networks*, 35(4), 429-442.
- [21] Quwaider, M., & Biswas, S. (2010). DTN Routing in Body Sensor Networks with Dynamic Postural Partitioning. *Ad Hoc Networks*, 8(8), 824-841.

- [22] Qiao, J., Shen, X. S., Mark, J. W., Shen, Q., He, Y., & Lei, L. (2015). Enabling Device-To-Device Communications in Millimeter-Wave 5G Cellular Networks. *IEEE Communications Magazine*, 53(1), 209-215.
- [23] Cheng, M. X., Ye, Q., & Cai, L. (2015). Rate-adaptive Concurrent Transmission Scheduling Schemes for WPANS with Directional Antennas. *IEEE Transactions on Vehicular Technology*, 64(9), 4113-4123.
- [24] D'souza, R. J., & Jose, J. (2010). Routing Approaches in Delay Tolerant Networks: A Survey. *International Journal of Computer Applications*, 1(17), 8-14.
- [25] Jones, E. P., & Ward, P. A. (2006). Routing Strategies for Delay-Tolerant Networks. *Submitted to ACM Computer Communication Review (CCR)*, 1-10.
- [26] Benamar, N., Singh, K. D., Benamar, M., El Ouadghiri, D., & Bonnin, J. M. (2014). Routing Protocols in Vehicular Delay Tolerant Networks: A Comprehensive Survey. *Computer Communications*, 48, 141-158.
- [27] Liu, M., Yang, Y., & Qin, Z. (2011). A Survey of Routing Protocols and Simulations in Delay-Tolerant Networks. *Wireless Algorithms, Systems, and Applications*, 243-253.
- [28] Vahdat, A., & Becker, D. (2000). Epidemic Routing For Partially Connected Ad Hoc Networks. Duke University, Durham, NC 27708.

- [29] Bloom, B. H. (1970). Space/Time Trade-Offs In Hash Coding With Allowable Errors. *Communications of the ACM*, 13(7), 422-426.
- [30] Fan, L., Cao, P., Almeida, J., & Broder, A. Z. (1998, October). Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol. *In ACM SIGCOMM Computer Communication Review ACM*, 28(4), 254-265.
- [31] Terry, D. B., Theimer, M. M., Petersen, K., Demers, A. J., Spreitzer, M. J., & Hauser, C. H. (1995). Managing Update Conflicts In Bayou, A Weakly Connected Replicated Storage System. *In ACM SIGOPS Operating Systems Review ACM*, 29(5) 172-182.
- [32] Spyropoulos, T., Psounis, K., & Raghavendra, C. S. (2005). Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks. *In Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking ACM*, 252-259.
- [33] Burgess, J., Gallagher, B., Jensen, D. D., & Levine, B. N. (2006). MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks. In Infocom. Massachusetts University.
- [34] Lindgren, A., Doria, A., & Schelén, O. (2003). Probabilistic Routing In Intermittently Connected Networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 7(3), 19-20.

- [35] Abdelkader, T., Naik, K., Nayak, A., Goel, N., & Srivastava, V. (2013). SGBR: A Routing Protocol for Delay Tolerant Networks Using Social Grouping. *IEEE Transactions on Parallel and Distributed Systems*, 24(12), 2472-2481.
- [36] Abdelkader, T., Naik, K., Nayak, A., Goel, N., & Srivastava, V. (2016). A Performance Comparison of Delay-Tolerant Network Routing Protocols. *IEEE Network*, 30(2), 46-53.
- [37] Cuka, M., Shinko, I., Spaho, E., Oda, T., Ikeda, M., & Barolli, L. (2017). A Simulation System Based On ONE And SUMO Simulators: Performance Evaluation Of Different Vehicular DTN Routing Protocols. *Journal of High Speed Networks*, 23(1), 59-66.
- [38] Samyal, V. K., & Sharma, Y. K. (2017). Performance Evaluation of Delay Tolerant Networks Routing Protocols under varying Time to Live. *International Journal*, 8(1), 299-302.
- [39] Alajeely, M., Doss, R., & Ahmad, A. A. (2017). Routing Protocols in Opportunistic Networks—A Survey. *IETE Technical Review*, ISSN: 0256-4602, 1-19.
- [40] Spaho, E., Bylykbashi, K., Barolli, L., Kolic, V., & Lala, A. (2016). Evaluation of Different DTN Routing Protocols in an Opportunistic Network Considering Many-To-One Communication Scenario. In *Network-Based Information Systems (NBIS), 2016 19th International Conference on IEEE*, 64-69.

- [41] Keranen, A. (2008). Opportunistic Network Environment Simulator. Special Assignment report, Helsinki University of Technology, Department of Communications and Networking.
- [42] P. Ball. (1996). Introduction to Discrete Event Simulation. University of Strathclyde.

## **APPENDIXES**

## Appendix A: Algorithms

### A1: Epidemic Routing Protocol Algorithm

1: Procedure Name: OnContact

2: Input: node a, node b, integer ContactDuration

3: DropExpiredPackets(a,b) /\* Drop packets with their lifetime expired in both nodes

\*/

4: ExchangeSummaryVector(a,b)

5: if ContactDuration > 0 then

6:     pkt=GetPacket(a)

7:     if pkt then

8:         if NotReceivedBefore(pkt,b) then

9:             if IsDestination(pkt,b) then

10:                 SendPacket(pkt,a)

11:                 ConsumePacket(pkt,b)

12:             else

13:                 SendPacket(pkt,a)

14:                 StorePacket(pkt,b)

15:             end if

16:             ContactDuration=ContactDuration-size(pkt)

17:     end if



18: end if

19: end if

## **A2: Spray and Wait Routing Protocol Algorithm**

Procedure Name: OnContact

2: Input: node a, node b, integer ContactDuration

3: DropExpiredPackets(a,b) /\* Drop packets with their lifetime expired in both nodes

\*/

4: ExchangeSummaryVector(a,b)

5: if ContactDuration > 0 then

6:     pkt=GetPacket(a)

7:     if pkt then

8:         if NotReceivedBefore(pkt,b) then

9:             if IsDestination(pkt,b) then

10:                 SendPacket(pkt,a)

11:                 ConsumePacket(pkt,b)

12:             else

13:                 NrOfCopies=GetNrOfCopies(pkt,a)

14:                 if NrOfCopies > 1 then

15:                     SendPacket(pkt,a)

16:                     StorePacket(pkt,b)

```

17:                               SetNrOfCopies(pkt,a,NrOfCopies/2)

18:                               SetNrOfCopies(pkt,b,NrOfCopies/2)

19:                               endif

20:                               endif

21:                               ContactDuration=ContactDuration-size(pkt)

22:                               endif

23:                               endif

24: endif

```

### **A3: MaxProp Routing Protocol Algorithm**

```

1: Procedure Name: OnContact
2: Input: node a, node b, integer ContactDuration
3: DropExpiredPackets(a,b) /* Drop packets with their lifetime expired in both nodes
*/
4: ExchangeSummaryVector(a,b)
5: UpdateDeliveryPredictability()
6: SortPackets() /* Using MAXPROP sorting criteria */
7: if ContactDuration > 0 then
8:     pkt=GetPacket(a)
9:     /* pkt is the packet with the minimum hop count, or higher delivery
predictability */
10:    if pkt then
11:        if NotReceivedBefore(pkt,b) then
12:            if IsDestination(pkt,b) then
13:                SendPacket(pkt,a)
14:                ConsumePacket(pkt,b)
15:            else

```

```

16:             SendPacket(pkt,a)
17:             StorePacket(pkt,b)
18:         endif
19:         ContactDuration=ContactDuration-size(pkt)
20:     endif
21: endif
22: endif

```

#### **A4: PROPHET Routing Protocol Algorithm**

```

1: Procedure Name: OnContact
2: Input: node a, node b, integer ContactDuration
3: DropExpiredPackets(a,b) /* Drop packets with their lifetime expired in both nodes
*/
4: ExchangeSummaryVector(a,b)
5: UpdateDeliveryPredictability()
6: if ContactDuration > 0 then
7:     pkt=GetPacket(a)
8:     if pkt then
9:         if NotReceivedBefore(pkt,b) then
10:            if IsDestination(pkt,b) then
11:                SendPacket(pkt,a)
12:                ConsumePacket(pkt,b)
13:            else
14:                DPn1=DeliveryPredictability(pkt,a)
15:                DPn2=DeliveryPredictability(pkt,b)
16:                if DPn2 > DPn1 then
17:                    SendPacket(pkt,a)

```

```

18:                                     StorePacket(pkt,b)
19:                                     endif
20:                                 endif
21:                                 ContactDuration=ContactDuration-size(pkt)
22:                             endif
23:     endif
24: endif

```

All previous algorithms are provided in [36]

### **A5: SGBR Routing Protocol Algorithm**

```

1: Procedure Name: OnContact
2: Input: a,b,ContactDuration
3: DropExpiredPackets(a,b) /*Drop packets with their lifetime expired in both
nodes*/
4: ExchangeSummaryVector(a,b)
5: Age = CurrentTime - LastMeetingTime(a; b)
6:  $\Gamma_{ab} = (\Gamma_{ab})_{old} \gamma^k + (1 - (\Gamma_{ab})_{old} \gamma^k) \alpha$ . /* In SGBR_v2 the equation is

$$\Gamma_{ab} = \sum C_{a,b} / \sum_{i=n}^N C_{a,i} \quad */$$

7: if ContactDuration > 0 then
8:     /* Sort the packets so that those with minimum hops come first */
9:     SortPackets(a)
10:    if pkt=GetPacket(a) then
11:        if NotReceivedBefore(pkt,b) then

```

```

12:     if IsDestination(pkt,b) then
13:         SendPacket(pkt,a)
14:         ConsumePacket(pkt,b)
15:     else
16:         /* Forward a copy under protocol conditions*/
17:         NrCopies=GetNrOfCopies(pkt,a)
18:         if  $\Gamma_{ab} < C_{th}$  and NrCopies > 1 then
19:             SendPacket(pkt,a)
20:             StorePacket(pkt,b)
21:             SetNrOfCopies(pkt,a,NrCopies=2)
22:             SetNrOfCopies(pkt,b,NrCopies=2)
23:             if  $\Gamma_{ab} > D_{th}$  then
24:                 DropPacket(pkt,a)
25:             end if
26:         end if
27:     end if
28:     ContactDuration=ContactDuration-size(pkt)
29: end if
30: end if
31: end if

```

SGBR algorithm is provided in [35].

## Appendix B: Main Code

### B1: Code of SGBR Routing Protocol

```
package routing;

import java.util.ArrayList;

import java.util.Collection;

import java.util.Collections;

import java.util.Comparator;

import java.util.HashMap;

import java.util.HashSet;

import java.util.List;

import java.util.Map;

import java.util.Set;

import routing.util.RoutingInfo;

import util.Tuple;

import core.Connection;

import core.DTNHost;

import core.Message;

import core.Settings;

import core.SimClock;

/**

Implementation of SgbrV1Router router as described in

<I>Probabilistic routing in intermittently connected networks</I> by

Anders Lindgren et al.

*/
```

```

public class SgbrRouter extends ActiveRouter{

/** identifier for the initial number of copies setting ({ @value})*

public static final String NROF_COPIES = "nrofCopies";

/** updating factor initialization constant*/

public static final double U_INIT =0.45;

/** aging constant default value */

public static final double DEFAULT_Aging = 0.98;

/** Sgbr router's setting namespace ({ @value})*

public static final String SgbrRouter_NS = "SgbrRouter";

/** connectivity threshold default value */

public static final double Con_T = 0.5;

/** dropping threshold default value */

public static final double Drp_T = 0.5;

public static final String MSG_COUNT_PROPERTY = SgbrRouter_NS + "." +
"copies";

public double t1;

boolean drop;

private Map<DTNHost, Set<String>> sentMessages;

private Set<String> ackedMessageIds;

/** The default value for alpha */

/**

aging constant (gamma) -setting id ({ @value}).

Default value for setting is { @link #DEFAULT_Aging}. */

public static final String GAMMA_S = "gamma";

```

```

/** value of gamma setting */

private double gamma;

private Map<DTNHost, Double> aging;

/** last delivery predictability update (sim)time */

private double lastMeetingTime;

public static final String BINARY_MODE = "binaryMode";

protected int initialNrofCopies;

protected boolean isBinary;

/**

Constructor. Creates a new message router based on the settings in
the given Settings object.

@param s The settings object */

public SgbrRouter(Settings s) {

super(s);

Settings SgbrRouterSettings = new Settings(SgbrRouter_NS);

initialNrofCopies = SgbrRouterSettings.getInt(NROF_COPIES);

isBinary = SgbrRouterSettings.getBoolean( BINARY_MODE);

if (SgbrRouterSettings.contains(GAMMA_S)) {

gamma = SgbrRouterSettings.getDouble(GAMMA_S); }

else {

gamma = DEFAULT_Aging; }

initaging(); }

/**

```



### Copyconstructor.

@param r The router prototype where setting values are copied from \*/

```
protected SgbrRouter(SgbrRouter r) {  
  
    super(r);  
  
    this.gamma = r.gamma;  
  
    this.initialNrofCopies = r.initialNrofCopies;  
  
    this.isBinary = r.isBinary;  
  
    this.ackedMessageIds = new HashSet<String>();  
  
    this.sentMessages = new HashMap<DTNHost, Set<String>>();  
  
    initaging(); }  
  
/**  
  
    Initializes aging hash  
  
    */  
  
    private void initaging() {  
  
        this.aging = new HashMap<DTNHost, Double>(); }  
  
    @Override  
  
    public void changedConnection(Connection con) {  
  
        super.changedConnection(con);  
  
        if (con.isUp()) { // new connection  
  
            DTNHost other = con.getOtherNode(getHost());  
  
            updateDgreeConnectivityFor(other);  
  
            //updateDgreeConnectivityFor(getHost());  
  
            if (con.isInitiator(getHost())) {  
  
                /* initiator performs all the actions on behalf of the
```

```

other node too (so that the meeting probs are updated
for both before exchanging them) */

DTNHost otherHost = con.getOtherNode(getHost());
MessageRouter mRouter = otherHost.getRouter();
assert mRouter instanceof SgbrRouter : "Sgbr only works "+ " with other routers of
same type";

SgbrRouter otherRouter = (SgbrRouter)mRouter;

/* exchange ACKed message data */

this.ackedMessageIds.addAll(otherRouter.ackedMessageIds);
otherRouter.ackedMessageIds.addAll(this.ackedMessageIds);

deleteAckedMessages();

otherRouter.deleteAckedMessages();

/* update both meeting probabilities */

}} }

/**
Updates dgree of connectivity for a host.
<CODE>P(a,b) = P(a,b)_old + (1 - P(a,b)_old) * U_INIT</CODE>
@param host The host we just met */

private void updateDgreeConnectivityFor(DTNHost host) {
double oldValue = getAgingFor(host);
double newValue = oldValue + (1 - oldValue) * U_INIT;
aging.put(host, newValue); }

```

```

/**
Returns the current aging (P) value for a host or 0 if entry for
the host doesn't exist.

@param host The host to look the P for
@return the current P value
*/

public double getAgingFor(DTNHost host) {
    dgreeConnectivityAging(); // make sure aging are updated before getting
    if (aging.containsKey(host)) {
        return aging.get(host); }

    else {
        return 0; }}

/**
Ages all entries in the dgree of connectivity.
<CODE>P(a,b) = P(a,b)_old * (GAMMA ^ k)</CODE>, where k is number of
time units that have elapsed since the last time the metric was aged.

@see #SECONDS_IN_UNIT_S */

private void dgreeConnectivityAging() {
    double timeDiff = (SimClock.getTime() - this.lastMeetingTime);
    if (timeDiff == 0) {
        return; }

    double mult = Math.pow(gamma, timeDiff);

```

```

for (Map.Entry<DTNHost, Double> e : aging.entrySet()) {
    e.setValue(e.getValue()*mult); }

this.lastMeetingTime = SimClock.getTime(); }

/**
Deletes the messages from the message buffer that are known to be ACKed
*/

private void deleteAkedMessages() {
    for (String id : this.ackedMessageIds) {
        if (this.hasMessage(id) && !isSending(id)) {
            this.deleteMessage(id, false); }} }

@Override

public int receiveMessage(Message m, DTNHost from) {
    return super.receiveMessage(m, from); }

@Override

public Message messageTransferred(String id, DTNHost from) {
    Message msg = super.messageTransferred(id, from);

    Integer nrofCopies = (Integer)msg.getProperty(MSG_COUNT_PROPERTY);

    assert nrofCopies != null : "Not a SnW message: " + msg;

    if (isBinary) {

        /* in binary S'n'W the receiving node gets ceil(n/2) copies */

        nrofCopies = (int)Math.ceil(nrofCopies/2.0); }

    else {

        /* in standard S'n'W the receiving node gets only single copy */

        nrofCopies = 1; }

```

```

msg.updateProperty(MSG_COUNT_PROPERTY, nrofCopies);

if (isDeliveredMessage(msg)) {

this.ackedMessageIds.add(id); }

return msg; }

@Override

public boolean createNewMessage(Message msg) {

makeRoomForNewMessage(msg.getSize());

msg.setTtl(this.msgTtl);

msg.addProperty(MSG_COUNT_PROPERTY, new Integer(initialNrofCopies));

addToMessages(msg, true);

return true; }

public void update() {

super.update();

if (!canStartTransfer() || isTransferring()) {

return; // nothing to transfer or is currently transferring }

/* try messages that could be delivered to final recipient */

if (exchangeDeliverableMessages() != null) {

return; }

tryOtherMessages(); }

protected List<Message> getMessagesWithCopiesLeft() {

List<Message> list = new ArrayList<Message>();

for (Message m : getMessageCollection()) {

Integer nrofCopies = (Integer)m.getProperty(MSG_COUNT_PROPERTY);

assert nrofCopies != null : "SnW message " + m + " didn't have " + "nrof copies

property!";

```

```

if (nrofCopies > 1) {
list.add(m); } }
return list; }

/**
Called just before a transfer is finalized (by
{@link ActiveRouter#update()}).

Reduces the number of copies we have left for a message.

In binary Spray and Wait, sending host is left with floor(n/2) copies,
but in standard mode, nrof copies left is reduced by one.

*/

@Override
protected void transferDone(Connection con) {
Integer nrofCopies;

String msgId = con.getMessage().getId();

/* get this router's copy of the message */
Message msg = getMessage(msgId);

if (msg == null) { // message has been dropped from the buffer after..
return; // ..start of transfer -> no need to reduce amount of copies }

/* reduce the amount of copies left */
nrofCopies = (Integer)msg.getProperty(MSG_COUNT_PROPERTY);

if (isBinary) {
nrofCopies /= 2; }

else {
nrofCopies--; }

```

```

msg.updateProperty(MSG_COUNT_PROPERTY, nrofCopies);

DTNHost recipient = con.getOtherNode(getHost());

Set<String> sentMsgIds = this.sentMessages.get(recipient);

/* was the message delivered to the final recipient? */

if (msg.getTo() == recipient) {

this.ackedMessageIds.add(msg.getId()); // yes, add to ACKed messages

this.deleteMessage(msg.getId(), false); // delete from buffer }

/* update the map of where each message is already sent */

if (sentMsgIds == null) {

sentMsgIds = new HashSet<String>();

this.sentMessages.put(recipient, sentMsgIds); }

sentMsgIds.add(msgId); }

/**

Tries to send all other messages to all connected hosts ordered by

hop counts and their delivery probability

@return The return value of { @link #tryMessagesForConnected(List) } */

private Tuple<Message, Connection> tryOtherMessages() {

List<Tuple<Message, Connection>> messages =

new ArrayList<Tuple<Message, Connection>>();

Collection<Message> msgCollection = getMessageCollection();

/* create a list of SgbrMessages that have copies left to distribute */

@SuppressWarnings(value = "unchecked")

List<Message> copiesLeft = sortByQueueMode(getMessagesWithCopiesLeft());

```

```

/* for all connected hosts that are not transferring at the moment, * collect all the
messages that could be sent */

for (Connection con : getConnections()) {

DTNHost other = con.getOtherNode(getHost());

SgbrRouter othRouter = (SgbrRouter)other.getRouter();

Set<String> sentMsgIds = this.sentMessages.get(other);

if (othRouter.isTransferring()) {

continue; // skip hosts that are transferring }

for (Message m : msgCollection) {

/* skip messages that the other host has or that have * passed the other host */

if(getAgingFor(getHost())>Drp_T){

deleteMessage(m.getId(), drop);

continue; }

if (othRouter.hasMessage(m.getId())) {

continue; }

/* skip message if this host has already sent it to the other
host (regardless of if the other host still has it) */

if (copiesLeft.size() > 0 && getAgingFor(other)< Con_T) {

messages.add(new Tuple<Message, Connection>(m,con)); } }

if (messages.size() == 0) {

return null; }

/* sort the message-connection tuples according to the number of Hops
defined in SgbrTupleComparator */

Collections.sort(messages, new TupleComparator());

```



```

return tryMessagesForConnected(messages); }

/**
Comparator for Message-Connection-Tuples that orders the tuples by
their delivery probability by the host on the other side of the
connection (GRTRMax)
*/

private class TupleComparator implements Comparator
<Tuple<Message, Connection>> {
public int compare(Tuple<Message, Connection> tuple1,
Tuple<Message, Connection> tuple2) {
// tuple1's message with tuple1's connection
int p1 = tuple1.getKey().getHopCount();
// "-" tuple2...
int p2 = tuple2.getKey().getHopCount();
// bigger probability should come first
if (p1 < p2 ) {
return -1; // message1 should be first }
else if (p2 > p1) {
return 1; // message2 "-"
}
if (p1 == p2) {
return compareByQueueMode(tuple1.getKey(), tuple2.getKey()); }
else {
return p1 - p2; }}}

```

```

@Override

public RoutingInfo getRoutingInfo() {

dgreeConnectivityAging();

RoutingInfo top = super.getRoutingInfo();

RoutingInfo ri = new RoutingInfo(aging.size() + " delivery prediction(s)");

for (Map.Entry<DTNHost, Double> e : aging.entrySet()) {

DTNHost host = e.getKey();

Double value = e.getValue();

ri.addMoreInfo(new RoutingInfo(String.format("%s : %.6f", host, value))); }

top.addMoreInfo(ri);

return top; }

@Override

public MessageRouter replicate() {

SgbrRouter r = new SgbrRouter(this);

return r; } }

```

## **B2: Code of SGBR\_V2 Routing Protocol**

```

/**

* Updates dgree of connectivity for a host.

* <CODE>  $\Gamma(a,b) = \Gamma_{ab} = \sum C_{a,b} / \sum_{i=n}^N C_{a,i}$ 

* @param host The host we just met

*/

private void updateDgreeConnectivityFor(DTNHost host) {

totalCon=totalCon+1;

```

```

double y = getPredFor(host);

double x = (y/totalCon);

double newValue = (1-x) ;

aging.put(host, newValue);

}

```

## Appendix C: Screenshots of the ONE Simulator

### C1: GUI Interface

We can run the ONE simulator in GUI mode by run its patch file which is “one.bat” that exist in the simulator folder. Figure C.1 shows the interface of the ONE simulator.

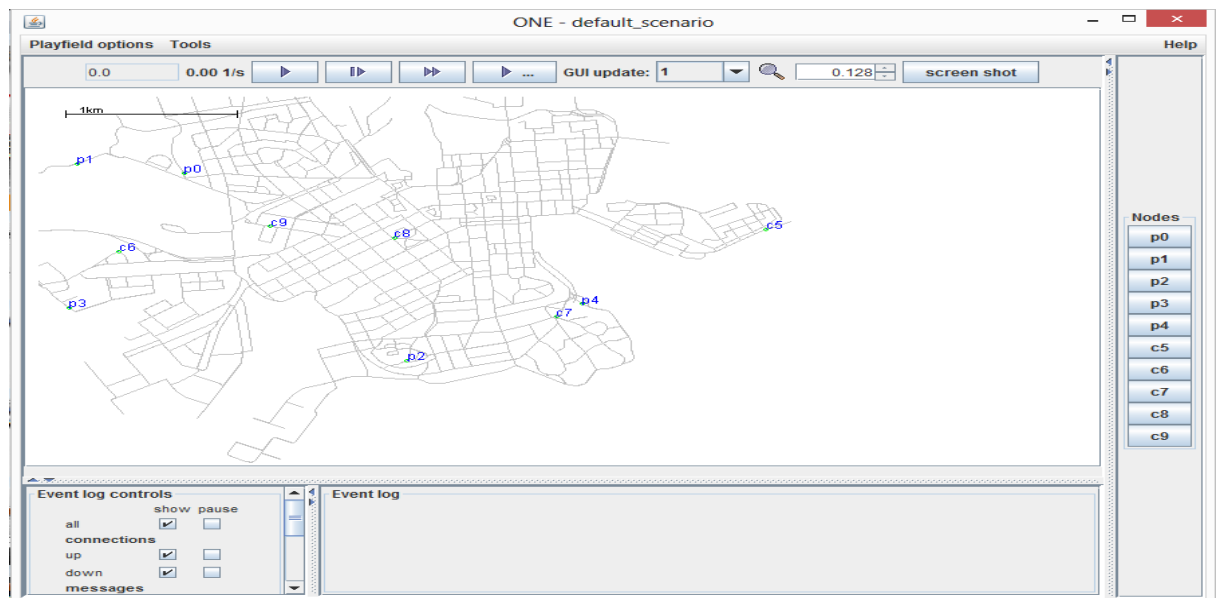


Figure C. 1: The ONE simulator GUI interface

### C2: Setting File

The ONE simulator takes the setting of the network parameters for each scenario form a text file with “txt” extinction that must be saved in the same folder with the patch

file of the simulator. This files includes a scenarios names and the path of the results reports along with the values of each parameter as follow:

```
# Default settings for the simulation
```

```
## Scenario settings
```

```
Scenario.name=default_scenario_%%Group.router%%_%%MovementModel.rngSeed%%_%%Group2.nrofHosts%%+%%Group.nrofHosts%%
```

```
Scenario.simulateConnections = true
```

```
Scenario.updateInterval = 0.1
```

```
# 43200s == 12h
```

```
Scenario.endTime = 43200
```

```
## Interface-specific settings:
```

```
# type : which interface class the interface belongs to
```

```
# For different types, the sub-parameters are interface-specific
```

```
# For SimpleBroadcastInterface, the parameters are:
```

```
# transmitSpeed : transmit speed of the interface (bytes per second)
```

```
# transmitRange : range of the interface (meters)
```

```
# "Bluetooth" interface for all nodes
```

```
btInterface.type = SimpleBroadcastInterface
```

```
# Transmit speed of 2 Mbps = 250kBps
```

```
btInterface.transmitSpeed = 5M
```

```

btInterface.transmitRange = 10

Scenario.nrofHostGroups = 2

## Group-specific settings:

# groupID : Group's identifier. Used as the prefix of host names

# nrofHosts: number of hosts in the group

# movementModel: movement model of the hosts (valid class name from movement
package)

# waitTime: minimum and maximum wait times (seconds) after reaching destination

# speed: minimum and maximum speeds (m/s) when moving on a path

# bufferSize: size of the message buffer (bytes)

# router: router used to route messages (valid class name from routing package)

# activeTimes: Time intervals when the nodes in the group are active (start1, end1,
start2, end2, ...)

# msgTtl : TTL (minutes) of the messages created by this host group, default=infinite

## Group and movement model specific settings

# pois: Points Of Interest indexes and probabilities (poiIndex1, poiProb1, poiIndex2,
poiProb2, ... )

#   for ShortestPathMapBasedMovement

# okMaps : which map nodes are OK for the group (map file indexes), default=all

#   for all MapBasedMovement models

```

```
# routeFile: route's file path - for MapRouteMovement

# routeType: route's type - for MapRouteMovement

# Common settings for all groups

Group.movementModel = ShortestPathMapBasedMovement

Group.router = [SgbrRouter; MaxPropRouter; EpidemicRouter; ProphetRouter;
SprayAndWaitRouter]

Group.bufferSize = 10M

Group.waitTime = 0, 120

# All nodes have the bluetooth interface

Group.nrofInterfaces = 1

Group.interface1 = btInterface

# Walking speeds

Group.speed = 0.5, 1.5

# Message TTL of 300 minutes (5 hours)

Group.msgTtl = 240

Group.nrofHosts = 10

# group1 (pedestrians) specific settings

Group1.groupID = p

#Group1.okMaps = 1

# group2 specific settings
```

```
Group2.groupID = c

Group2.nrofHosts = 10

# cars can drive only on roads

Group2.okMaps = 2

# 10-50 km/h

Group2.speed = 2.7, 13.9

Group2.transmitRange= 100

## Message creation parameters

# How many event generators

Events.nrof = 1

# Class of the first event generator

Events1.class = MessageEventGenerator

# (following settings are specific for the MessageEventGenerator class)

# Creation interval in seconds (one new message every 25 to 35 seconds)

Events1.interval = 25,350

# Message sizes (500kB - 1MB)

Events1.size = 500k,1M

# range of message source/destination addresses

Events1.hosts = 0,19

# Message ID prefix
```

```
Events1.prefix = M

## Movement model settings

# seed for movement models' pseudo random number generator (default = 0)

MovementModel.rngSeed = [0;1;2;3;4;5;6;7;8;9;10]

#MovementModel.rngSeed = 1

# World's size for Movement Models without implicit size (width, height; meters)

MovementModel.worldSize = 4500, 3400

# How long time to move hosts in the world before real simulation

MovementModel.warmup = 1000

## Map based movement -movement model specific settings

MapBasedMovement.nrofMapFiles = 4

MapBasedMovement.mapFile1 = data/roads.wkt

MapBasedMovement.mapFile2 = data/main_roads.wkt

MapBasedMovement.mapFile3 = data/pedestrian_paths.wkt

MapBasedMovement.mapFile4 = data/shops.wkt

## Reports - all report names have to be valid report classes

# how many reports to load

Report.nrofReports = 1

# length of the warm up period (simulated seconds)

Report.warmup = 0
```



```
# default directory of reports (can be overridden per Report with output setting)

Report.reportDir = reports/reports/NrOfNode is varied/20/240

# Report classes to load

Report.report1 = MessageStatsReport

## Default settings for some routers settings

#ProphetRouter.SECONDS_IN_UNIT_S="secondsInTimeUnit"

ProphetRouter.secondsInTimeUnit = 30

SprayAndWaitRouter.nrofCopies = 5

SprayAndWaitRouter.binaryMode = true

SgbrRouter.nrofCopies = 5

SgbrRouter.binaryMode = true

## Optimization settings -- these affect the speed of the simulation

## see World class for details.

Optimization.cellSizeMult = 5

Optimization.randomizeUpdateOrder = true

## GUI settings

# GUI underlay image settings

GUI.UnderlayImage.fileName = data/helsinki_underlay.png

# Image offset in pixels (x, y)

GUI.UnderlayImage.offset = 64, 20
```

```
# Scaling factor for the image
```

```
GUI.UnderlayImage.scale = 4.75
```

```
# Image rotation (radians)
```

```
GUI.UnderlayImage.rotate = -0.015
```

```
# how many events to show in the log panel (default = 30)
```

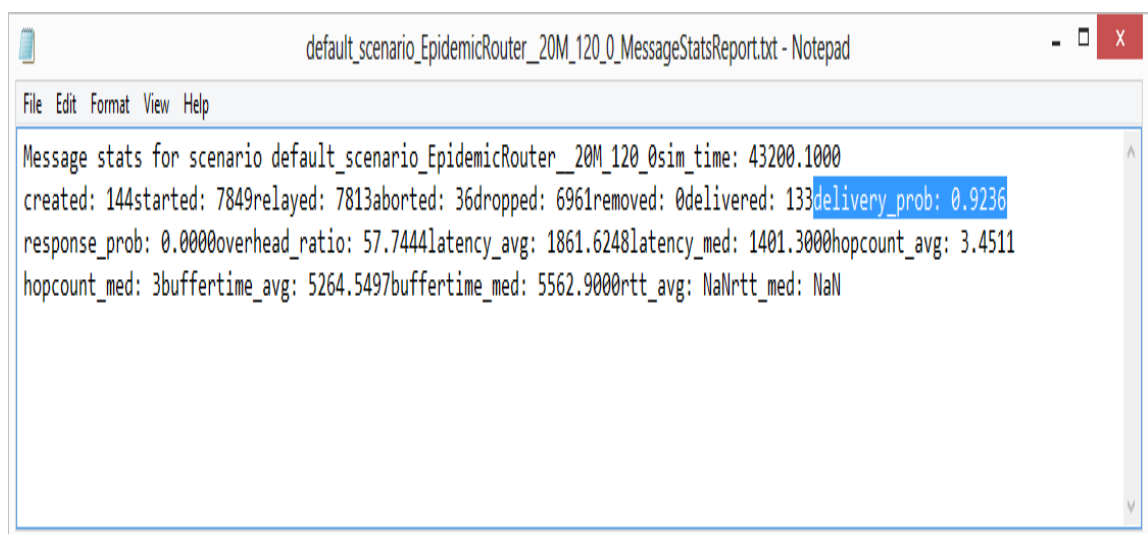
```
GUI.EventLogPanel.nrofEvents = 100
```

```
# Regular Expression log filter (see Pattern-class from the Java API for RE-matching  
details)
```

```
#GUI.EventLogPanel.REfilter = .*p[1-9]<->p[1-9]$
```

### C3: Report File

The ONE simulator creates a report file that contain the results for performance metrics and the results are shown as in Figure C.2 where each file named based on the protocol name and the scenario name.



```
default_scenario_EpidemicRouter_20M_120_0_MessageStatsReport.txt - Notepad
File Edit Format View Help
Message stats for scenario default_scenario_EpidemicRouter_20M_120_0sim_time: 43200.1000
created: 144started: 7849relayed: 7813aborted: 36dropped: 6961removed: 0delivered: 133delivery_prob: 0.9236
response_prob: 0.0000overhead_ratio: 57.7444latency_avg: 1861.6248latency_med: 1401.3000hopcount_avg: 3.4511
hopcount_med: 3buffertime_avg: 5264.5497buffertime_med: 5562.9000rtt_avg: NaNrtt_med: NaN
```

Figure C. 2: ONE simulator report file