

Statistical Model Checking of Opportunistic Network Protocols

Ankit Rathor

A Thesis Submitted to
Indian Institute of Technology Hyderabad
In Partial Fulfillment of the Requirements for
The Degree of Master of Technology



भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Under the guidance of
Dr. M.V.P. Rao

Department Of Computer Science and Engineering
I.I.T. Hyderabad

June 2016

Declaration

I, Ankit Rathor, declare that this thesis titled, 'Statistical Model Checking of Opportunistic Network' and the work presented in it are my own. I confirm that this work was done wholly or mainly while in candidature for a thesis research. Any part of this thesis has not previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated. I have consulted the published work of others, this is always clearly attributed. I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work. I have acknowledged all main sources of help. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Ankit

Signature:

ANKIT RATHOR

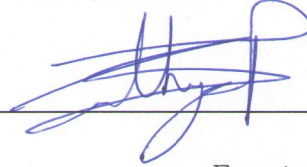
Name:

CS14MTECH 11001

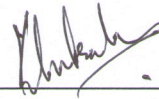
Roll No.:

Approval Sheet

This Thesis entitled SMC for Opportunistic Network by Ankit Rathor is approved for the degree of Master of Technology from IIT Hyderabad



Examiner Dept. of CSE IITH



Examiner Dept. of CSE IITH



(Dr. M.V.P. Rao) Adviser Dept. of CSE IITH



U. RAMAKRISHNA.

Chairman Dept. of CSE IITH

Acknowledgements

I would like to express my deepest appreciation to all those who provided me the possibility to complete my thesis. A special gratitude to my thesis adviser Dr. M.V.P. Rao for his constant encouragement, patience and immense knowledge. Without his guidance and persistent help, this dissertation would not have been possible. I would like to thank my colleague Shiraj Arora for her continued support and help in understanding the domain related concepts, specifically the discussions on Statistical Model Checking. I thank the Almighty, my family and friends for their support and constant encouragement.

Dedication

This work is dedicated to my parents and all of my friends without whom none of my success would be possible.

Abstract

Statistical model checking is one of the powerful methods, used to analyze any large system. Test bed experiments are used for analysis of routing algorithms in computer network. For more deeper penetration, we use statistical model checking to analyze properties and performance of opportunistic networks. In order to do so, we link a statistical model checker to a discrete event simulator for opportunistic network. This linking allows statistical model checking of several opportunistic network properties and protocols.

Contents

Declaration	i
Approval Sheet	ii
Acknowledgements	iii
Dedication	iv
Abstract	v
1 Introduction and Motivation	1
2 Preliminaries and Previous work	2
2.1 Opportunistic Networks	2
2.2 Routing Protocols in Opportunistic Networks	2
2.3 The ONE Simulator	4
2.4 Model Checking and MultiVeStA	4
2.4.1 MultiVeStA	5
3 Approach and Implementation	6
4 Results and Analysis	8
5 Propagation of Epidemics and Malware in Human and Computer Networks	12
5.0.1 Linking between DES and MultiVeStA	12
5.0.2 Results and analysis	14
6 Conclusion	17
7 Reference	18

Chapter 1

Introduction and Motivation

Nowadays, social networks are a part of everybody's daily life. In a social network, people form groups with the people whom they find alike or with whom they find something common like language, organization, place etc. which is also known as homophiles in sociology. Availability of different applications like WhatsApp, Facebook etc. make social networks stronger.

Wikipedia reports that the estimated number of mobile phone users are 3.3 billion worldwide, which is more than half of the world's population and most of them are smartphone users, which also helps social network grow more.

Opportunistic networks are delay tolerant networks. Human mobility is used to forward messages from source to destination. The movement model in an opportunistic network follows the human behavior, which allows researchers to predict the node behaviour. PROPHET[1] routing protocol is an example of how opportunistic networks depend on social networks.

Opportunistic network has large possibilities in research, which motivates us to work on this field and for that we study opportunistic networks and using formal methods like model checking, we explore various properties of opportunistic networks.

This thesis arranged as follows. Section 2 provides a quick introduction to the concepts, terminology and tools used in this thesis. Section 3 describes our approach and implementation. Section 4 discusses experimental results and analysis. Section 5 discusses another application of statistical model checking, wherein propagation of epidemics and malware in human and computer networks is analyzed using MultiVeStA. Section 6 concludes the thesis.

Chapter 2

Preliminaries and Previous work

2.1 Opportunistic Networks

Opportunistic networks are basically wireless networks. Nodes in these networks are typically handheld devices like smartphones carried by people. Opportunistic networks do not require any additional infrastructure, and are similar to mobile ad hoc networks. Many techniques from mobile ad hoc networks can be used in opportunistic networks.

Mobile ad hoc networks need real time routing[9] and it assumes that every node in the network is willing to route the traffic. It uses multi-hop communication. Opportunistic networks are basically delay tolerant networks which use human mobility[10] to transfer information. Opportunistic networks face numerous challenges because of disruptions, delays and connectivity. Privacy[2] is one of the main issue in opportunistic networks because it is difficult to maintain trust between nodes.

Opportunistic networks work on *store carry forward* paradigm. Assume that a message is *stored* in some portable device like a mobile phone. Then a person *carries* the data with him while walking or travelling around. In *forwarding* phase, the device sends data to another device based on some criteria which brings message closer to the destination.

2.2 Routing Protocols in Opportunistic Networks

The main problem in opportunistic networks is to forward the message to the next hop. Since in opportunistic networks route is calculated at every hop, so strictly dynamic routing is used in opportunistic networks.

Routing in opportunistic networks is based on two functions:

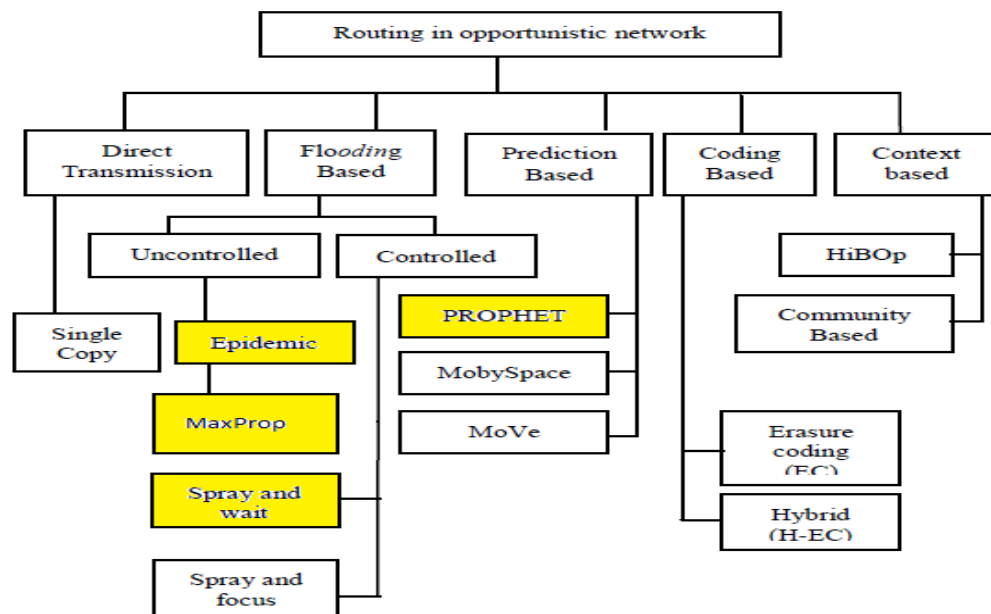
1. **To find a path to the destination:**

- There is no fixed path between source and destination in an opportunistic network. Intermediate nodes are used to form the paths dynamically.

2. **Selection of next hop forwarder:**

- If multiple paths are available, than find a intermediary node which can forward data to destination as soon as possible.

The classification of various routing algorithm[3], which can be used in opportunistic network are show in FIGURE 2.1 below



Every Routing has its own unique advantages and disadvantages. We discuss a brief overview of all routing algorithms in this section.

- **Direct Transmission:** - source directly sends data to the destination. This method is simple but delivery delay is high.
- **Flooding based:** - Flooding based routing algorithms are basically of two types – controlled and uncontrolled. In uncontrolled method, we use epidemic models, which have high delivery rate but use large bandwidths. In controlled method, we try to avoid usage of unnecessary bandwidth which causes more delay in message delivery, when compared to uncontrolled methods.

- **Prediction based:** - Prediction based routing algorithms basically use probability to predict the next best hop. If prediction is successful, then it gives best results but if prediction fails then message delay will be high.
- **Context based:** - In some cases, prediction based techniques may fail and reduce delivery ratio. In order to improve delivery ratio, prediction based approach is further refined by utilizing context information.
- **Coding based:** - In coding based routing schemes, a message is transformed into another format prior to transmission. Limited amount of blocks can be used to regenerate original message.

We use *Epidemic*, *Spray and Wait*, *Maxprop* and *Prophet* routing protocols in our experiments. Prophet is a prediction based routing protocol and rest three are flooding based protocols. ONE Simulator described in next section can be used to simulate opportunistic networks.

2.3 The ONE Simulator

Opportunistic Network Environment(ONE) is an open source Discrete Event Simulator[4] for opportunistic networks. ONE is implemented on java and provides multiple features for opportunistic networks.

ONE is capable of features like:

- Generating node movement using different movement models like Random Way Point and Map-Based Movement, etc.
- Supports several DTN routing protocols like Epidemic , Spray and wait. etc.
- Providing visualization of both mobility and message passing in real time.
- Importing mobility data from real-world traces.

2.4 Model Checking and MultiVeStA

Implementation and verification of system are important phases in software engineering. Sometimes verification may take more efforts than implementation. Formal methods like model checking[5] provide greater help in terms of analysis and verification. Properties of the system which need to be verified, can be specified using an appropriate logic

systems. Model checking algorithms can then be used to check whether the system satisfies the properties or not.

Probabilistic model checking is mainly of two types – Numerical and Statistical model checking.

- Numerical model checking is a state space exploration based technique. Numerical model checking is accurate, costly and is not suitable for larger networks. For larger networks, we use statistical model checking.
- Statistical model checking is based on simulation and statistical techniques like hypothesis testing. Statistical model checking is fast and scales much better than numerical model checking in larger networks. Statistical model checking is simple to implement but is not completely accurate, especially with small sizes networks.

2.4.1 MultiVeStA

MultiVeStA[8] is a descendant of VeStA[6] model checker. Building on VeStA and PVeStA[7], Sebastio and Vandin developed MultiVeStA.

MultiVeStA is a java based statistical analysis tool which can be easily integrated with existing discrete event simulators. It also provides capabilities of distributed statistical analysis and statistical model checking.

MultiVeStA supports verification of models using queries expressed in logic like PCTL, CSL and MultiQuaTEx queries, which are very powerful. We can ask several logical queries using MultiQuaTEx query language. MultiQuaTEx, a simple extension of QuaTEx, allows multiple simultaneous queries to be answered on same simulation set.

Chapter 3

Approach and Implementation

To implement statistical model checking for opportunistic networks, we need a simulator for opportunistic network i.e. ONE simulator and a model checker i.e. MultiVeStA.

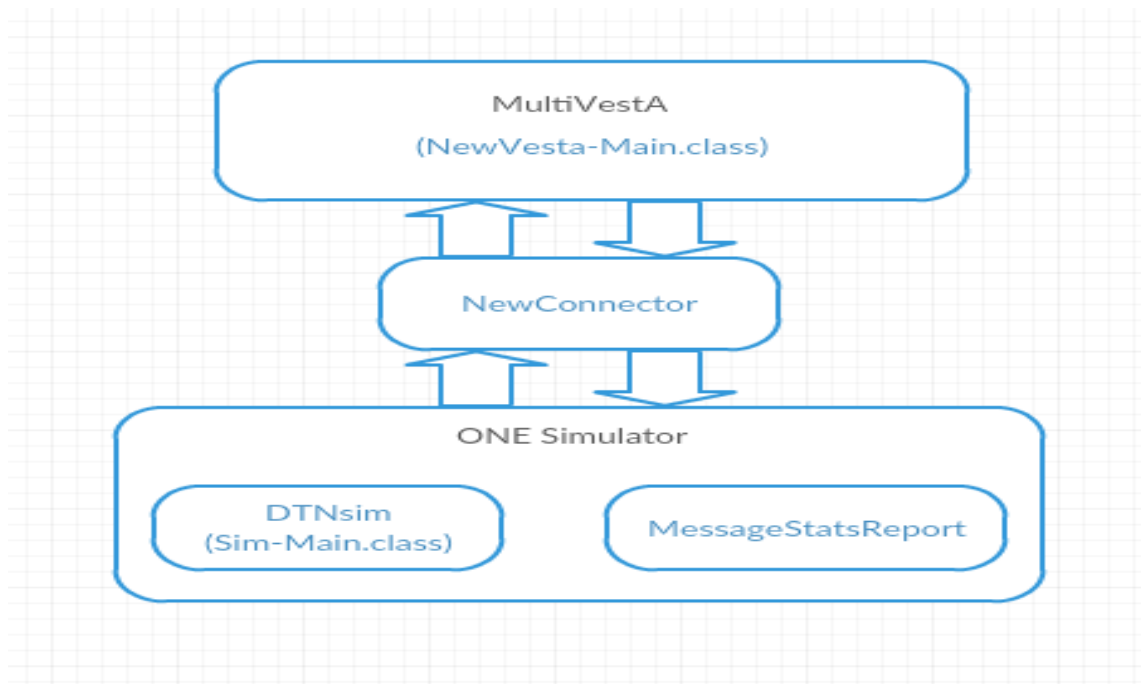


FIGURE 3.1: Linking between ONE and MultiVeStA

The linking between ONE simulator and MultiVeStA is show in figure 3.1 above. The **NewConnector** class extends the **NewState** class of MultiVeStA. NewConnector is used to link ONE and MultiVeStA, and works as an intermediary class. This class keeps track of simulation variables which are relevant for MultiQuaTEx queries.

ONE simulator has **MessageStatsReport** class, which provides the values needed for MultiQuaTEx queries. The **MessageStatsReport** class was made singleton to synchronize updated simulation variables among different classes.

For accessing the variables and their values, we use **rval(int arg)** function. For every value of *arg*, we have a corresponding output. We have used the following map of arguments to the simulation variables:

- **rval(0)** returns true if the simulation is completed and false otherwise.
- **rval(1)** returns the current simulation time.
- **rval(2)** returns the number of messages created during a simulation.
- **rval(4)** returns the number of **active messages** – messages which have not yet been dropped or delivered.
- **rval(5)** returns the ratio of the number of delivered messages to the number of created messages (also called the delivery ratio).
- **rval(9)** returns the ratio of the number of aborted to created messages.

MultiVeStA provides two variations for model checking: **Stepwise Simulation** and **Whole simulation**. Stepwise simulation checks for the properties of system after each discrete step of simulation. In case of Whole simulation, the properties are checked only after the simulation is completed.

In our experiment we perform each simulation in a map area of dimension 4000×3000 meters. Each map contains 125 nodes which are randomly placed on the map by ONE Simulator. We use the Random Waypoint mobility model, with a speed in the range of 0.5 to 1.5 m/s for our experiments. The communication range of each node in the simulation is 10 meters, which falls in the category of Bluetooth interfaces. The **MessageEventGenerator** class is used to create messages uniformly at random in a 10 second interval after every 25 seconds. Each message has a TTL of 300 minutes.

Chapter 4

Results and Analysis

For running MultiQuaTEx queries and getting results, we need MultiVeStA *Client and Server* model. We provide a server list and port numbers available for model checker. We start *Server* in one terminal and *Client* in another terminal.

- Server Command

```
java -cp ./lib/* vesta.mc.NewVestaServer 49141(PortNumber)
```

In this command, we provide classpath of **MultiVeStA.jar** to start the *VestaServer*.

- Client Command

```
java -cp ./lib/* vesta.NewVesta -sd core.NewConnector -m ModelName  
-f quateX/queryname.quateX -l ServerList/Server -bs 10 -ms 100 -a 0.005  
-d1 0.0004 -se core.NewEvaluator -osws ONESTEP
```

In this command, we provide classpath of *NewVesta* class which contains *main* function.

- **sd** is for state descriptor.
- **m** is for Model name (usually initial setting file).
- **f** is for the file name of QuaTEx query.
- **l** is for the file containing list of available servers.
- **bs** is for batch size. We use batch size 10 in over experiment.
- **ms** is for maximum number of simulations allowed. We use 100 simulations i.e 10 batches of 10 simulations each.

Using MultiQuaTEx queries, we studied some properties of opportunistic network routing protocols.

- **Example 1**

Expected value of delivery ratio at the end of the simulation for different routing protocols.

- **MultiQuaTEx Syntax**

Delivery Ratio : the number of delivered messages to the created messages

s.rval(0) : returns whether the simulation has ended.

s.rval(5) : returns the delivery ratio

```
delivery()= if { s.rval( 0 )== 1.0 } then {s.rval(5) } else {0} fi;
eval E[ delivery( ) ] ;
```

- **Results**

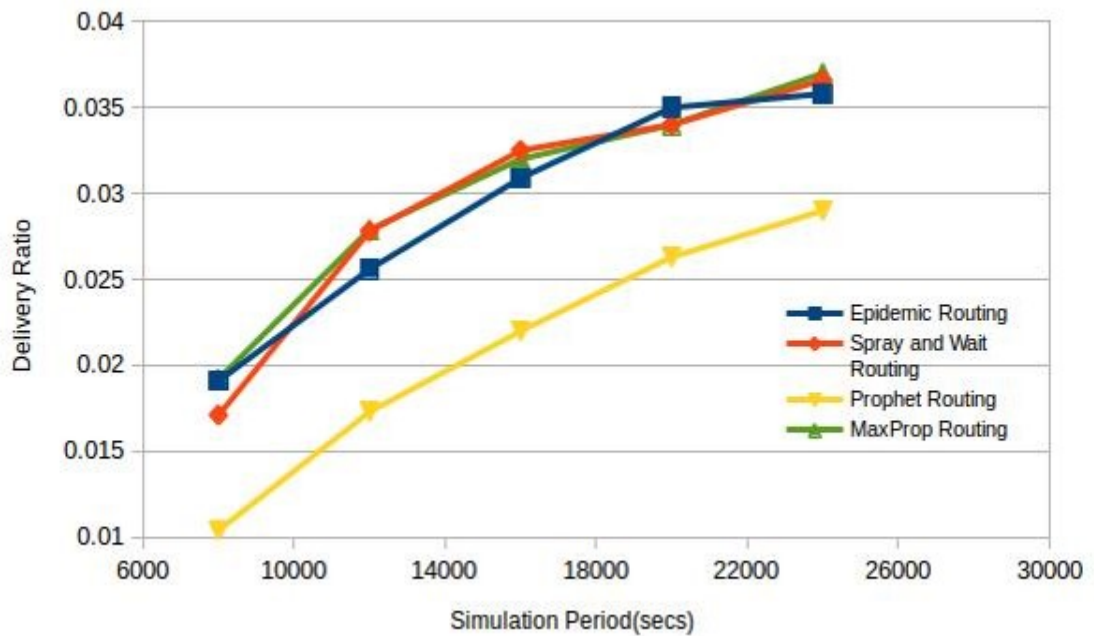


FIGURE 4.1: The Delivery Ratio

- **Analysis**

Epidemic, Spray and Wait and MaxProp are Flooding based Routing and results show the similar behavior but Prophet is prediction based which is different from other three.

- **Example 2**

Probability that the delivery ratio remains low(in range 0.015 to 0.03).

- **MultiQuaTEx Syntax**

s.rval(2) : returns the number of created messages.

s.rval(5) : returns the delivery ratio

```
del_ratio() = if { s.rval( 0 ) == 1.0 && s.rval( 2 ) == 0.0 } then {0} else
if{ s.rval( 5 ) =0.015 && s.rval( 5 ) = 0.03 } then {1} else {0} fi fi ;
eval E[ del_ratio()];
```

- **Results**

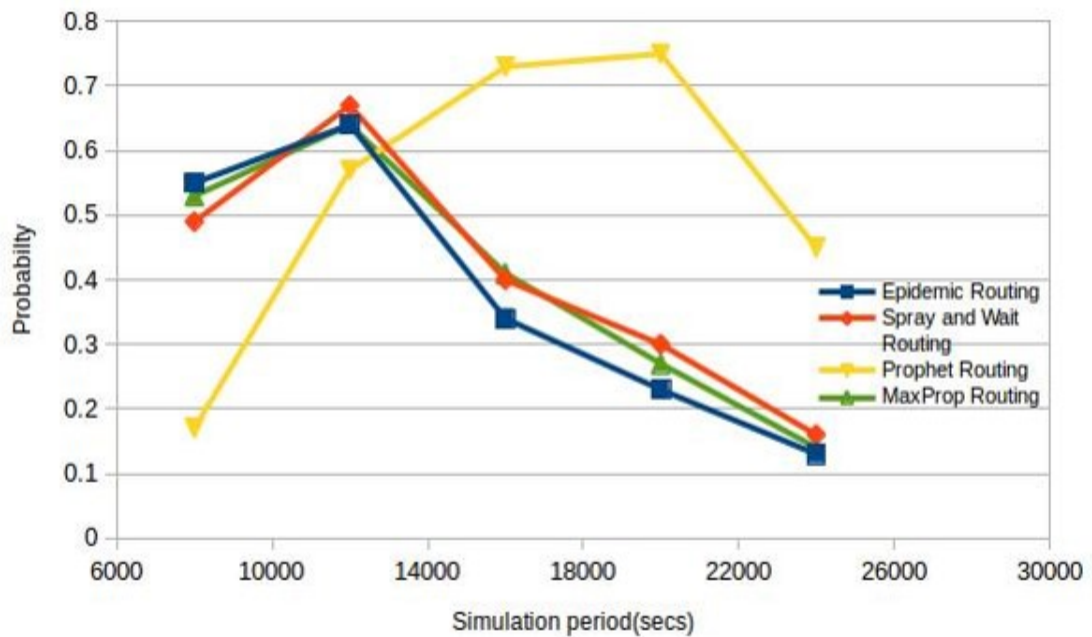


FIGURE 4.2: The Delivery Ratio

- **Analysis**

At the initial stage, the delivery ratio remains low so the corresponding probability is more but when time increases delivery ratio also increases and thus the probability decreases.

- **Example 3**

Expected value of active messages at different timestamps for different routing protocols.

- **MultiQuaTEx Syntax**

```
active_msgs( x )= if { s.rval( 1 ) ≥ x } then { s.rval( 4 ) } else #active_msgs({ x })fi;
eval parametric( E[ active_msgs( x ) ] , x , 0 , 2 , 20 ) ;
```

- **Results**

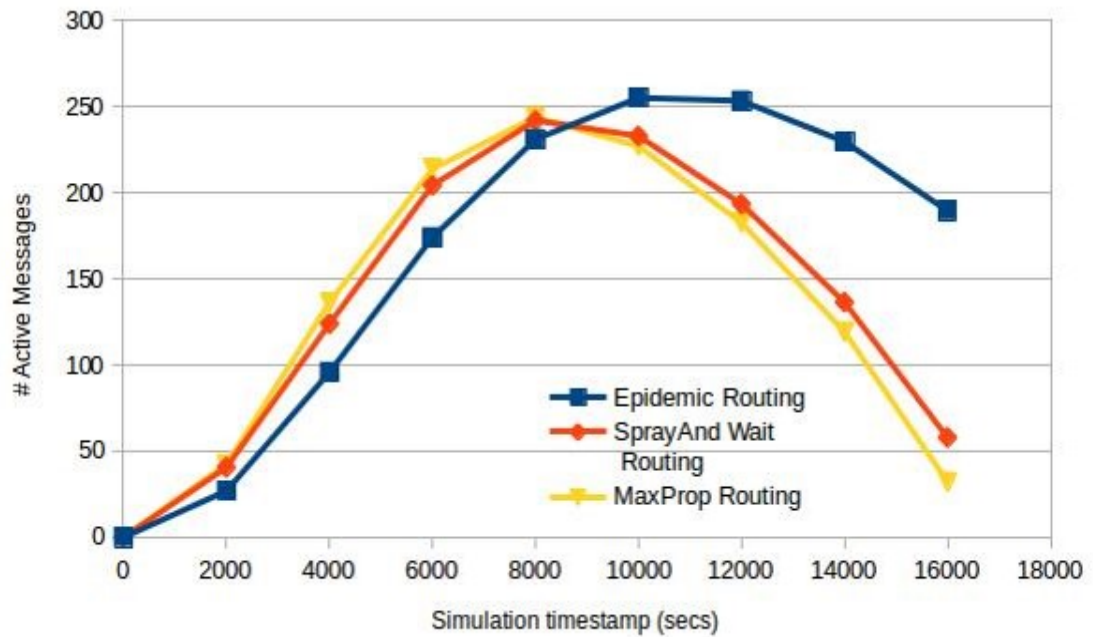


FIGURE 4.3: The Delivery Ratio

- **Analysis** Initially, we have less active nodes. As time increases, active nodes also increase but after a particular time period, either message is delivered or aborted, so the number of active messages decrease.

Chapter 5

Propagation of Epidemics and Malware in Human and Computer Networks

In this chapter, we discuss another application of statistical model checking using MultiVeStA – propagation of epidemics and malware in human and computer networks respectively.

We used a java based DES which works on concept of SIR model. SIR stands for *Susceptible, Infected and Recovered*. SIR model is an epidemic model which computes the number of infected nodes with a contagious illness in a closed system. DES also supports early vaccination scheme. Vaccination is based on four methods: 1. Random vaccination 2. Degree distribution 3. Betweenness centrality 4. Closeness centrality.

In next section, we discuss linking between DES and MultiVeStA.

5.0.1 Linking between DES and MultiVeStA

Linking between DES and MultiVeStA is shown in figure 5.1 below. We implemented **NewConnector** class to link DES and MultiVeStA.

As discussed about **Server and Client** method of MultiVeStA in previous chapter, we use the same method here for asking MultiQuaTEx queries. First, MultiVeStA calls the NewConnector class to start simulations. We use same graph for all the queries.

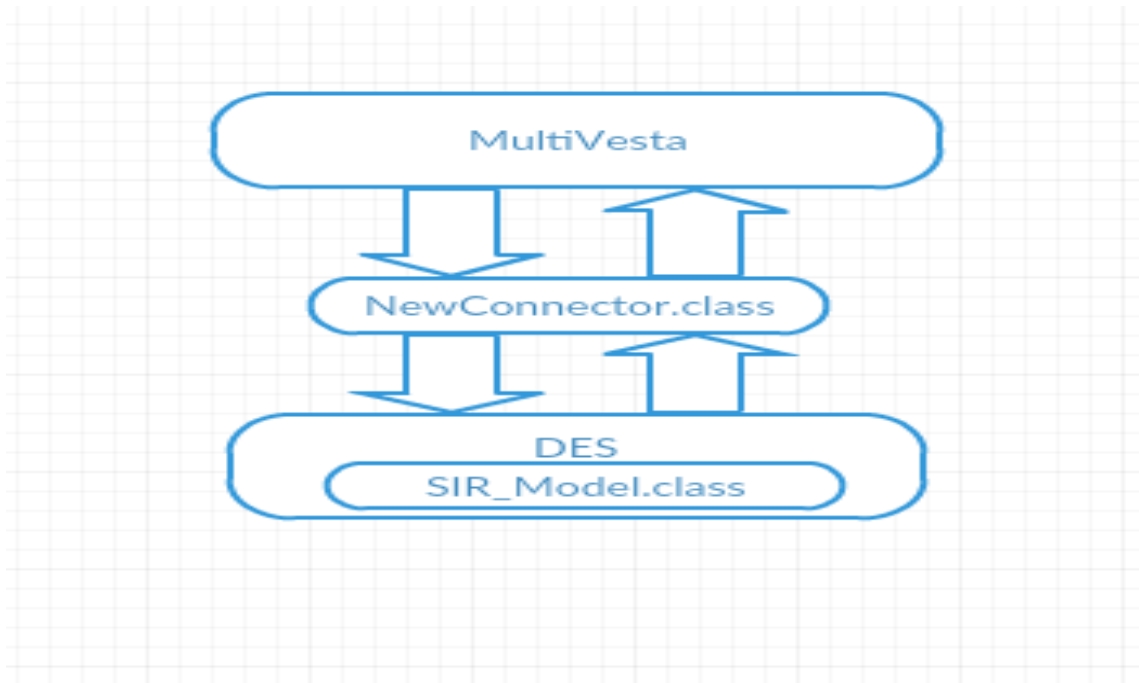


FIGURE 5.1: Linking Between DES and MultiVeStA

SIR_Model class then computes the result, which we will get in our *NewConnector* class. MultiVeStA gets the required values using *rval* function. We use the following map of the arguments to the simulation variables.

- **rval(1)** returns true if the simulation is completed and false otherwise.
- **rval(2)** returns number of infected nodes at the current time.
- **rval(3)** returns the number of recovered nodes at the current time.
- **rval(4)** returns the number of susceptible nodes at the current time.
- **rval(5)** returns the current simulation time.
- **rval(6) and rval(7)** return status of node A and B respectively.
- **rval(8) and rval(9)** return simulation time when node A and B get infected respectively.
- **rval(10)** returns difference of infection time of node A and B.
- **rval(11)** returns the percentage of the nodes which are not infected at the current time.

In our experiment, we perform each simulation for 100000 nodes on a small world graph of a fixed topology. Each experiment contains fixed 10 initially infected nodes and 30000 vaccinations.

5.0.2 Results and analysis

Example 1

Probability that node A is infected before node B at the end of the simulation for different vaccination schemes.

MultiQuaTEx Syntax

`s.rval(1)`: returns true if the simulation is completed and false otherwise.

`s.rval(8)` and `rval(9)` : return simulation time when node A and B get infected respectively.

```
AbeforeB() = if { s.rval (1)==1.0 } then if { s.rval(8) > s.rval(9)
} then {1} else {0} fi else # AbeforeB() fi ;
eval E[ AbeforeB()];
```

Results

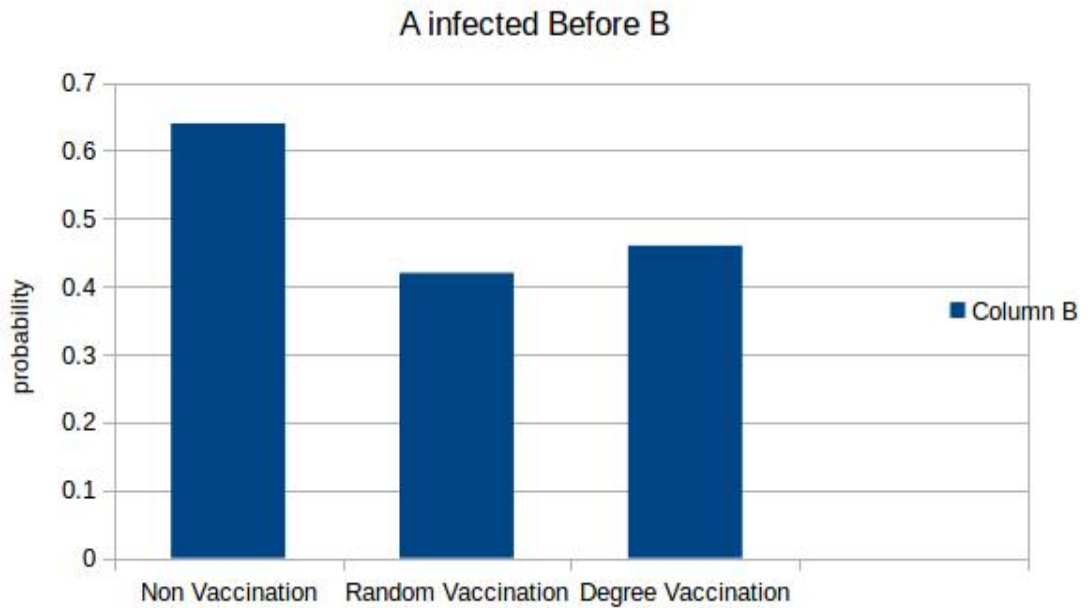


FIGURE 5.2: Probability of node A infect before node B

Analysis

In case of non vaccination the probability of A is infected before B is high. In case of random vaccination and degree vaccination probability relatively low. We use node number 2670 as A and 56700 as B.

Example 2

Expected value of infected nodes at different timestamps for different vaccination scheme.

MultiQuaTEx Syntax

s.rval(5): returns the current simulation time.

s.rval(2) : returns number of infected nodes at the current time.

```
infect_no(x)=if { s.rval(5)==x} then { s.rval(2) } else
#infect_no({x}) fi;
eval parametric(E[infect_no(x)],x,2,3,100);
```

Results

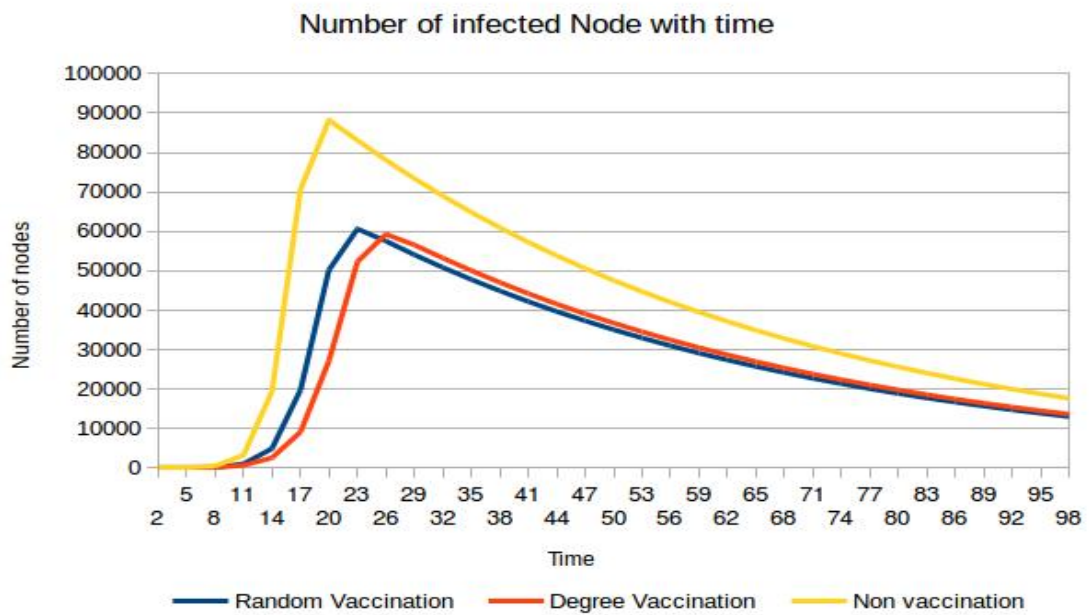


FIGURE 5.3: Probability of node A infect before node B

Analysis

In case of non vaccination number of infected node is much high. In case of vaccination scheme number of infected node remains low.

Example 3

Probability that 75% of the total node are not infected at different timestamps for different vaccination scheme.

MultiQuaTEx Syntax

s.rval(5): returns the current simulation time.

s.rval(2) : returns number of infected nodes at the current time.

```

noninfect(x)=if { s.rval(5) ≥ x } then if { s.rval(11) ≥ 0.75 } then
{1} else {0} fi else # noninfect({x}) fi;
eval parametric(E[noninfect(x)],x,15,3,100);
    
```

Results

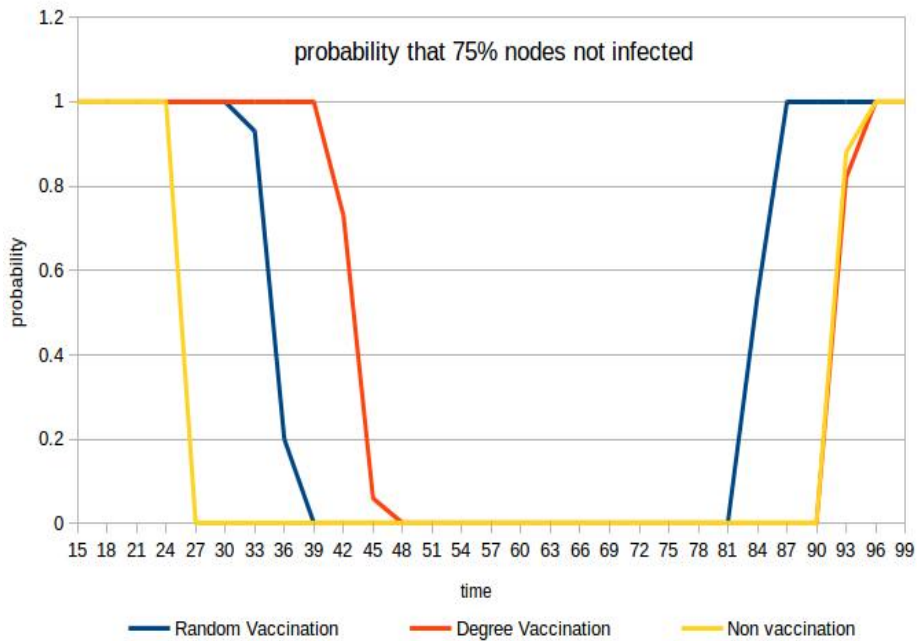


FIGURE 5.4: Probability of node A infect before node B

Analysis

In this result we clearly seen that in case of non vaccination nodes start getting infected early and in vaccination scheme it will take longer time. In vaccination scheme less people are infected so they also recovered early.

Chapter 6

Conclusion

We demonstrated a linking of the MultiVeStA statistical model checker with the ONE simulator for opportunistic networks. We believe that such a composition would provide a powerful tool for the analysis of routing algorithms in such networks. We also demonstrated a linking between SIR model simulator and MultiVeStA for a small world graph. We show how powerful statistical model checking is and how can it be applied to real life problems.

Chapter 7

Reference

1. Xue, Jingfeng, et al. "Advanced PROPHET routing in delay tolerant network." *Communication Software and Networks*, 2009. ICCSN'09. International Conference on. IEEE, 2009.
2. Distl, Bernhard, and Stephan Neuhaus. "Social power for privacy protected opportunistic networks." *Communication Systems and Networks (COMSNETS)*, 2015 7th International Conference on. IEEE, 2015.
3. Poonguzharselvi, B., and V. Vetriselvi. "Survey on routing algorithms in opportunistic networks." *Computer Communication and Informatics (ICCCI)*, 2013 International Conference on. IEEE, 2013.
4. Ari Ker anen, J org Ott, and Teemu K arkk ainen. The ONE simulator for DTN protocol evaluation. In *Proc. of the 2nd Intl. Conf. on Simulation Tools and Techniques for Communications, Networks and Systems, SimuTools 2009*, Rome, Italy, March 2-6, 2009, page 55, 2009.
5. Baier, Christel, Joost-Pieter Katoen, and Kim Guldstrand Larsen. *Principles of model checking*. MIT press, 2008.
6. Koushik Sen, Mahesh Viswanathan, and Gul A. Agha. VESTA: A statistical model-checker and analyzer for probabilistic systems. In *Second Intl. Conf. on the Quantitative Evaluation of Systems (QEST 2005)*, 19-22 September 2005, Torino, Italy, pages 251-252, 2005.
7. AlTurki, Musab, and Jos Meseguer. "PVeStA: A parallel statistical model checking and quantitative analysis tool." *International Conference on Algebra and Coalgebra in Computer Science*. Springer Berlin Heidelberg, 2011.

-
8. Stefano Sebastio and Andrea Vandin. MultiVeStA: statistical model checking for discrete event simulators. In 7th Intl. Conf. on Performance Evaluation Methodologies and Tools, ValueTools13, Torino, Italy, December 10-12, 2013, pages 310315, 2013.
 9. Pelusi, Luciana, Andrea Passarella, and Marco Conti. "Opportunistic networking: data forwarding in disconnected mobile ad hoc networks." *Communications Magazine*, IEEE 44.11 (2006): 134-141.
 10. Boldrini, Chiara, Marco Conti, and Andrea Passarella. "Impact of social mobility on routing protocols for opportunistic networks." *World of Wireless, Mobile and Multimedia Networks*, 2007. WoWMoM 2007. IEEE International Symposium on a. IEEE, 2007.
 11. Newman, Mark EJ, Duncan J. Watts, and Steven H. Strogatz. "Random graph models of social networks." *Proceedings of the National Academy of Sciences* 99.suppl 1 (2002): 2566-2572.