

# Modeling and Performance analysis of AODV Routing Protocol using Coloured Petri Nets

Sudheer Singampalli



Department of Computer Science and Engineering  
National Institute of Technology Rourkela  
Rourkela-769 008, Odisha, India

# Modeling and Performance analysis of AODV Routing Protocol using Coloured Petri Nets

*Thesis submitted in partial fulfillment of the requirements for the degree of*

## Master of Technology

*in*

## Computer Science and Engineering

(Specialization: Software Engineering)

*by*

## Sudheer Singampalli

*(Roll No. 212CS3127 )*

*under the supervision of*

## Prof. S.Chinara



Department of Computer Science and Engineering  
National Institute of Technology Rourkela  
Rourkela, Odisha, 769 008, India

June 2014



Department of Computer Science and Engineering  
**National Institute of Technology Rourkela**  
Rourkela-769 008, Odisha, India.

## Certificate

This is to certify that the work in the thesis entitled *Modeling and Performance analysis of AODV Routing Protocol using Coloured Petri Nets* by *Sudheer Singampalli* is a record of an original research work carried out by him under my supervision and guidance in partial fulfillment of the requirements for the award of the degree of Master of Technology with the specialization of Software Engineering in the department of Computer Science and Engineering, National Institute of Technology Rourkela. Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

Place: NIT Rourkela  
Date: June 2, 2014

**Prof. S.Chinara**  
Professor, CSE Department  
NIT Rourkela, Odisha

## Acknowledgment

I am grateful to all my local and global peers who have contributed towards shaping this thesis. At the outset, I would like to express my sincere thanks to Prof.S.Chinara for her advices during my thesis work. As my supervisor, she has constantly encouraged me to remain focused on achieving my goal. Her observations and comments helped me to establish the overall direction of the research and to move forward with investigation in depth. She has helped me greatly and been a source of knowledge.

I am very much indebted to the Doctoral Scrutiny committee members Prof. Durga Prasad Mahapatra, Prof. Ashok Kumar Turuk, Prof. Bansidhar Majhi and Prof. Sanjay Kumar Jena for their time to provide more insightful opinions into my research.

I extend my thanks to our HOD, Prof. S. K. Rath for his valuable advices and encouragement and time he spent to help in my thesis.

I must acknowledge the academic resources that I have got from NIT Rourkela. I would like to thank administrative and technical staff members of the Department who have been kind enough to advise and help in their respective roles.

I am really thankful from the bottom of my heart to my friend Prerna Kanojia for being with me always and helping me during my thesis work. Thanks a lot to my senior Shraddha, this thesis is not possible without her guidance. Also thankful to Sumit, Pankaj, Vijay, Rajesh, Priyesh, Anshuman and all my friends for helping me during my thesis documentation. My sincere thanks to everyone who have provided me with kind words, a welcome ear, new ideas, useful criticism, or their invaluable time, I am truly indebted.

***Sudheer Singampalli***  
***Roll: 212CS3127***

# Abstract

The growth of interest in mobile ad-hoc networks is increasing exponentially. In a Mobile Ad hoc Network(MANET), wireless transmissions can happen in such a way that mobile nodes can send messages directly to one another through wireless links. The protocols which establish their routes dynamically on demand are called reactive protocols. One of the such reactive protocols defined for MANETs is AODV (Ad hoc On-demand Distance Vector) routing protocol. Since the nodes are mobile in nature, the topology of the network does not remain constant, it keeps on changing frequently. Thus it is very much necessary for every node in the network to keep track of change so that an efficient packet transmission can be done. In this thesis, AODV is modeled using Coloured Petri nets, various performance measures like workload, number of packets sent and received, efficiency of the protocol are evaluated using monitors. The same routing protocol is again simulated using well known NS2 tool. The results of the modeled CPN are compared with NS2 simulator output. We have assumed that all the nodes have sufficient energy while participating in the routing process.

**Keywords:** *AODV, Mobile Ad-hoc Network, Coloured petri net tool, Timed Petri Net, Routing protocol, Monitors.*

# Contents

<b>Certificate</b>	<b>i</b>
<b>Acknowledgment</b>	<b>ii</b>
<b>Abstract</b>	<b>iii</b>
<b>List of Figures</b>	<b>vi</b>
<b>List of Tables</b>	<b>vii</b>
<b>1 Introduction</b>	<b>2</b>
1.1 Introduction . . . . .	2
1.2 Mobile Ad Hoc Network (MANET) . . . . .	3
1.2.1 Characteristics . . . . .	3
1.3 Petri nets . . . . .	4
1.4 Coloured Petri Nets . . . . .	4
1.5 Timed petri nets . . . . .	8
1.6 Thesis Layout . . . . .	10
<b>2 Literature Review</b>	<b>12</b>
<b>3 Modeling Of AODV Protocol Using Coloured Petri Nets</b>	<b>16</b>
3.1 Ad hoc On demand Distance Vector Protocol . . . . .	16
3.2 AODV Terminology . . . . .	16
3.3 AODV Process . . . . .	17
3.4 Simulating AODV in CPN . . . . .	20
<b>4 Simulation based Performance Analysis of Modeled AODV</b>	<b>29</b>
4.1 Timed Protocol for Performance Analysis with Monitoring Simulations: . . . . .	29

4.2	Kinds Of Monitors . . . . .	30
4.3	Performance Analysis . . . . .	32
4.4	Efficiency Calculation . . . . .	34
4.4.1	Simulation in NS2 tool . . . . .	34
<b>5</b>	<b>Conclusion</b>	<b>39</b>
	<b>Bibliography</b>	<b>40</b>

# List of Figures

1.1	Place inscription: place type . . . . .	6
1.2	Place inscription: initial marking . . . . .	7
1.3	Arc Inscription . . . . .	7
1.4	Transition inscriptions . . . . .	8
1.5	Enabled Transition . . . . .	9
1.6	Adding timing values to a transition . . . . .	9
3.1	Main Module . . . . .	20
3.2	Protocol Module . . . . .	21
3.3	Check Route Sub Page . . . . .	21
3.4	RREQInit Subpage . . . . .	22
3.5	RREQProcess Subpage . . . . .	23
3.6	RREPProcess Subpage . . . . .	24
3.7	Firing of generate transition . . . . .	25
3.8	Enabling of NODE_I . . . . .	25
3.9	Enabling of RoutCheck . . . . .	26
3.10	RREQProcess Subpage . . . . .	27
3.11	RREPProcess Subpage . . . . .	27
4.1	Workload for 500 steps of Simulation . . . . .	33
4.2	Log File of Workload for 500 steps of Simulation . . . . .	33
4.3	Number Of packets sent 500 steps of Simulation . . . . .	34
4.4	Log File of Number Of Packets sent for 500 steps of Simulation . . . . .	34
4.5	Trace file . . . . .	36
4.6	NS2 AODV efficiency VS Modeled AODV efficiency . . . . .	37



# List of Tables

4.1	NS2 AODV Efficiency VS CPN AODV Efficiency . . . . .	37
-----	--	----

# **Chapter 1**

**Introduction**

# Chapter 1

## Introduction

### 1.1 Introduction

Mobile ad hoc network (MANET) is a network which is infrastructure less. This network consists of mobile nodes and wireless machine nodes connected to each other [1]. There is no centralized administration mechanism in MANETs. In MANET every node acts as a router. As the name indicates, Mobile means the nodes are not stationary, Ad hoc means no fixed infrastructure. Therefore a network with no fixed topology or infrastructure is called adhoc network. In MANET, a node can send packets directly to another node if both the nodes are in their respective transmission ranges or else packet transmission can be multihop. Due to arbitrary movement of nodes, network topology changes rapidly. Since the nodes are changing much rapidly it is very difficult for the routing protocols to perform their task correctly. Hence a topology approximation mechanism, known as Ad hoc On-demand Distance Vector (AODV) routing protocol, is used to perform simulation of a typical routing protocol which addresses the problem of mobility [2]. AODV routing protocol [3] is a reactive routing protocol, which means that route from source node is created in an on demand basis.

There are different types of MANETs. They are listed as follows

- InVANETs : Short form for Intelligent Vehicular Ad hoc Networks. They make use of concepts like artificial intelligence to avoid vehicle collision.
- iMANET : called as Internet Based Mobile Ad hoc Networks. They help in linking fixed as well as mobile nodes.

- VANETs : Vehicular ad hoc networks enables effective communication with roadside equipment and other vehicles.

## 1.2 Mobile Ad Hoc Network (MANET)

### 1.2.1 Characteristics

- In MANET, every node can act as both host as well as a router. So, it has an autonomous behavior.
- Whenever the source and destination are not reachable, MANETs are capable of multi-hop routing.
- It has a circulated nature of operation for security, directing and host setup. An unified firewall is missing here.
- The hubs can join or leave the system at whatever time, making the system topology changing in nature.
- Versatile hubs are described with characteristics such as light weight, less memory and power.
- The unwavering quality, productivity, security and limit of remote connections are frequently subpar when contrasted and wired connections. This shows the fluctuating connection transfer speed of remote connections.
- Mobile and spontaneous conduct which requests least human intercession to arrange the system.
- All hubs have indistinguishable characteristics with comparative obligations and proficiencies and thus it structures a totally nature.
- High client thickness and extensive level of client portability.
- Nodal integration is discontinued.

## 1.3 Petri nets

Petri nets were introduced by Dr. Carl Adam Petri in 1962 [4]. These are a powerful modeling mechanism in computer science, system engineering and many other disciplines. Petri nets combine a graphical representation of the dynamic behavior of systems with well-defined mathematical theory. Modeling and analysis of the system is done by the theoretical aspect of Petri nets, while the visualization of the modeled system is done by the graphical representation. This combination of theory and graphics is the main reason for the great success of Petri nets [5]. Consequently, Petri nets have been used to model various kinds of dynamic event-driven systems like computers networks, communication system, real time computing system etc.

Petri nets have been mostly and successfully used in studying concurrent systems. Time features were also extended to Petri nets to specify real-time and safety-critical systems.

## 1.4 Coloured Petri Nets

Colored Petri Net [6] provides a framework for modeling and analysis of concurrent and distributed systems. A Coloured Petri Net model describes the states(the places) that a system may obtain and the possible transitions in between them.

A Coloured Petri Net is mathematical model used to describe any system. It is a directed bipartite graph defined as a 9 tuple [7]  $(\Sigma, P, T, A, N, C, G, E, I)$

- $\Sigma$  is a finite set of non-empty types, called color sets.
- $P$  is a finite set of places.
- $T$  is a finite set of transitions.
- $A$  is a finite set of arcs.
- $N$  is a node function.
- $C$  is a color function. It is defined from  $P$  into  $\Sigma$ .

- G is a guard function.
- E is an arc expression function.
- I is an initialization function.

The main advantage of CPNs is that it supports colour, hierarchy and time. Hierarchy allows the user to construct a large flat net into subnets that can be easily understood and maintained. This is done by the hierarchical structuring of the programming language [8], that supports the bottom-up or top-down style.

In hierarchy, modules created can be reused in several parts of the CPN model and further sub-modules can be created from them [9]. The modules of the CPNs are called pages. A complex model can have many pages. This is similar to a lengthy and complex program, that is divided into several modules. The substitution transitions in hierarchical CPNs, provide a more detailed description of the activity of a transition and its associated components. The details of the substitution transition are defined in separate modules called the subpages [10]. The places in a sub-page are marked with an input tag In tag, output tag Out tag or input/output tag I/O tag. These places are called the port places [10]. Through these ports the subpage communicates with its surroundings.

The subpage receives tokens from its surroundings through the input port and tokens are delivered to its surroundings through the output ports and the I/O port communicates to its surroundings in both ways. The places associated with a substitution transition are called the socket places. The input and output places of the substitution transition are called sockets. By providing the port assignments [practitioners guide], the port places of the sub pages are related to the socket places of the substitution transition. When a socket place is assigned to a port place, the two places become identical. They are the two different views of a single conceptual place, therefore the port and the socket places always have identical markings. Whenever a token is produced by the subpage on its output port it is available at the output socket of the transition, similarly if a token arrives at the input socket of the substitution transition it will be available at the input port [10].

Communication is important for modeling of AODV routing protocols. Systems where concurrency, communication and synchronization play an important role can be modeled using Colored Petri Net tool [2]. As has been done in (Gordon 2001), one of the approaches to ensure the correctness and validity of existing routing protocol is creating a formal model for the protocol [11] and analysis of model to determine if the existing protocol provides the defined services correctly or not. By verifying the routing protocol using formal modeling, one can gain CPN is a graphically oriented language [12] for design, specification, simulation and verification of systems. An advantage to use CPNets is that it is possible to use the same model for performance analysis as well as for checking the logical and functional correctness of a system [2].

Colors associated with each place in the CPNs determine the type of data it may handle. The types of the places shown in figure 1.1 are similar to the types in programming languages. It can be a complex type as the record which may contain heterogeneous data types.

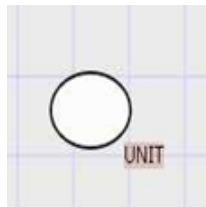


Figure 1.1: Place inscription: place type

The color set is usually defined as:

`colset No = int;` where `colset` is a keyword to declare the color set, `No` is the name of the colour set and `int` indicates that this colset can have integral values as tokens. The number of tokens distributed on the individual places of a CPNs is called as its state. Each token carries a value. The type of the token is equal to the type of place on which it resides. The marking of a place is denoted by the tokens present on a particular place. The initial state of a place is denoted as its initial marking. It is written in the upper left or right of the place as shown in figure 1.2. Every place in Coloured petri net is connected to a transition. The transition fires the tokens from one place to another. The Petri net is a bi-partite

graph where the two types of nodes are transitions and places. No two places and transitions can be connected.

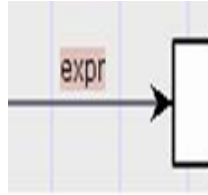


Figure 1.2: Place inscription: initial marking

The place and transition are connected through arcs. The arc has a condition or an expression as shown in figure 1.3.

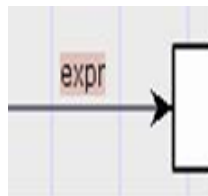


Figure 1.3: Arc Inscription

Similarly a transition can also have a condition. The condition on a transition applies on the bindings of the elements for the transition to fire. In the figure 1.4 (a) shows how time values are added to the transition. (b) shows the code segment where the inputs and outputs are written and the code written action part is performed whenever a transition is enabled. A transition is enabled at that time only when all the inputs to a transition are available. A green aura is created around the rectangular box whenever a transition is enabled as Figure 1.5. Guard functions are written in top left corner of the transition as shown in figure (c).



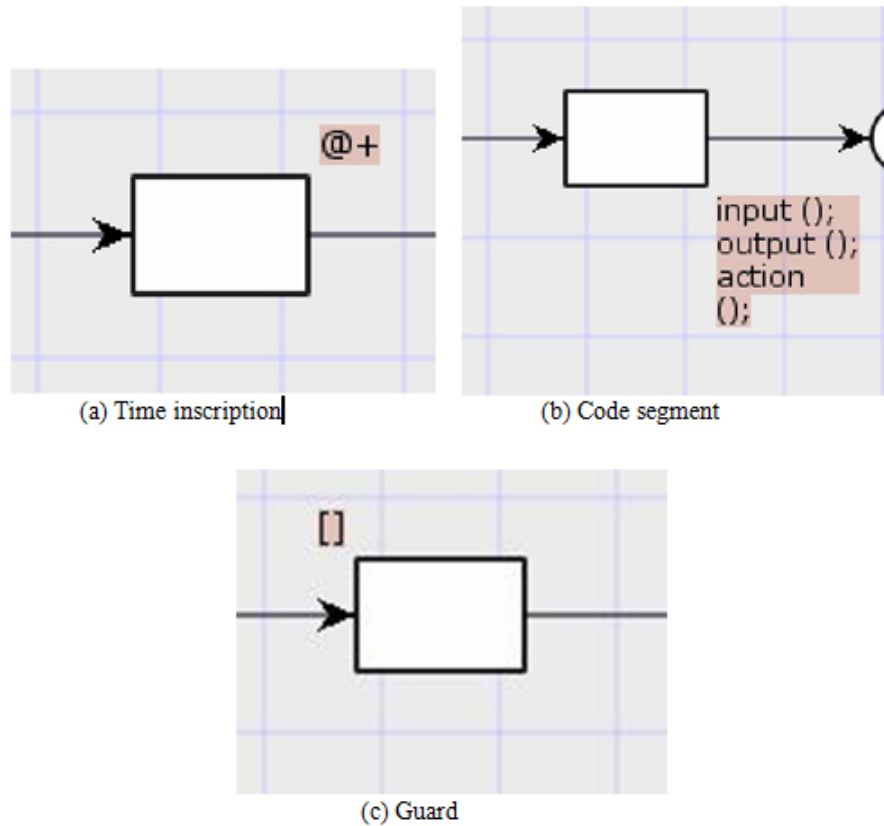


Figure 1.4: Transition inscriptions

## 1.5 Timed petri nets

A timed CPN [13] can model the amount of time certain activities require and the amount of time passes between other activities. In most of the cases, it is not sufficient to model the amount of time a certain activity takes, it is necessary to include a more precise representation of timing of the system. Introducing time concept in the CPNs is redefined as timed CPNs. This introduces the concept of global clock. The clock value which is either discrete or continuous and represent the model time. In the timed CPNs, each token carries a time stamp.

The time stamp indicates the earliest model time at which the token can be used. That is only after the global clock exceeds the time stamp the can be removed by the occurrence of a binding element. In a timed CPNs a binding element is said to be color enabled when it satisfies the enabling rule for un-timed

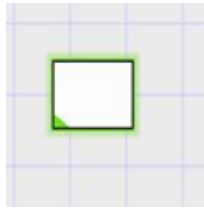


Figure 1.5: Enabled Transition

CPNs. Figure 1.6 shows how timing values are incorporated in a transition. The marking of a place where the tokens carry a time stamp becomes a timed multi-set.



Figure 1.6: Adding timing values to a transition

It specifying the tokens in the multi set together with their time stamps and the number of appearances. The timed color sets are declared as `: colset INTDATA =product INT*STRING timed;` and the possible marking of a place with timed token is as: `2(1,colour)@[19,45]` This indicates the marking contains two tokens with value `(1, colour)` and time stamps 19 and 45 respectively. The `@` symbol can be read as at and the symbol `[ ]` is used to specify the time stamps.

## 1.6 Thesis Layout

Rest of the thesis is organized as follows:

**Chapter 2: Literature Review:** In this chapter, literature survey is done in AODV, Coloured Petri Net tool and Timed petri net.

**Chapter 3: Modeling of AODV protocol using CPN:** This chapter presents a model of AODV routing protocol implemented in Coloured petri net tool. The protocol is simulated and hence validated using timed petri net concept. Here various performance metrics are calculated.

**Chapter 4: Simulation based Performance Analysis of Modeled AODV:** In this, the existing AODV routing protocol is simulated in NS2 tool and using generated trace file, necessary information for calculating the number of packets sent and received is extracted. These values are compared with the once obtained through CPN tool and a relative comparison is made.

# **Chapter 2**

## **Literature Review**

## Chapter 2

# Literature Review

Since the mobile Ad hoc networks (MANETs) do not have a fixed topology, selection of a routing protocol to serve its needs was very difficult. They needed a reactive protocol which can behave dynamically to the network changes. Therefore the ad hoc on demand distance vector (AODV) is developed. It is jointly developed in Nokia Research Center, University of California, Santa Barbara and University of Cincinnati by C. Perkins, E. Belding-Royer and S. Das [14]. The nodes in AODV keep changing their position dynamically thus were making it difficult to model the system.

Colored Petri Nets has been proved to be a powerful tool for simulating and analyzing the non-determinism, concurrency and different level of abstraction of any communication protocol. Some of the researchers used Coloured Petri nets for validating the MANETs whereas some used it for modeling the features of the mobile ad hoc networks. Chinara *et al.* proposed a method for validating the neighbor detection protocol for ad-hoc network using CPN [15].

Erbas *et al.* proposed a two designed position based routing approach model for ad-hoc network using CPN [16]. In this paper it is shown that better result is delivered by multi-cast routing protocol than the basic On Demand Multi-cast Routing Protocol. Based on geographical position of a node, this CPN model developed reliable unicast and multicast routing method.

Kodikara [17] proposed the simulation model of context exchange based on hierarchical coloured petri nets. Simulation of vertical communication model has done which is cross layer info exchange module of context exchange (conEX). State

---

space used for verification of Petri nets dynamic behavioral properties like liveness, home property, boundedness and fairness.

To improve quality of route selection for MANET routing protocol, Nakhaee *et al.* [18] presented new route selection criteria instead of hop count. This scheme adds two parameters to route request packet: number  $q_{mean}$  and number  $q_{max}$ , shows the average queue length and maximum queue length of the nodes in the path respectively. Nakhaee presented an approach to improve AODV protocol routing reliability. Mohamed *et al.* Mobile ad-hoc network has several characteristics which separate it from conventional one. Nodes in the network support the network infrastructure and security required when the packets are passing. Author proposed security approach based on immunity and multi-agent paradigm. It depicts how the mobile ad-hoc network is secured by different agent coordination. It has been observed that, much work has not been done on the mobility pattern of the ad hoc networks by using CPN tools. The motivation for the current work is to construct the CPN model for the modeling and validation of Mobile Ad-hoc network [19].

The paper [12] addresses the mobility problem. Colored petri nets used as a simulation tool, simulate the network without knowing the network topology. Colored petri net is a tool which provide great perceptive in modeling a mobile ad-hoc network protocol. The author finds the route in the simulation part. This paper compares the DSR and AOMDV. For small network AOMDV has lesser delay to find out routes and in the other hand DSR has better efficiency than AOMDV but comparatively high delay in finding the routes.

Further to improve the TCP performance over MANET, Xiong *et al.* [Xiong, Yim, Leigh and Murata] have proposed a reactive approach TCP-MEDX to detect the causes of packet loss. As mobility of nodes is the biggest challenge in MANET, the round trip time is replaced with an average propagation delay for indicating congestion. The authors claim that the TCP-MEDX mechanism is able to detect the packet loss much accurately. Xiong *et al.* have created a formal CPN model of the very well known routing protocol for MANET, the Ad hoc On-Demand Distance

---

Vector (AODV) to analyse its correctness in service [11]. To meet the challenge of dynamism of the nodes, the authors have proposed a topology approximation (TA) mechanism. The TA mechanism works with certain assumptions. They are:

- All the nodes in the MANET have equal transmission range.
- It is also assumed that in a MANET every node has the same number of neighbors. This is equal to the average degree of the MANET graph.

Here, the second assumption is not a realistic approach. In MANET, the node movement frequently changes the degree of connectivity among the nodes and also the network topology. In such a non-deterministic environment, the mechanism of topology approximation may not be possible. Further, the dynamic operation of MANET using CPN is illustrated by Yuan *et al.* in [18]. The CPN tools are used by the authors for the elegant and simple modeling of the protocol without using any assumption or approximation. By using the formal specification and verification method of the modeling tool, the authors could be able to find the errors existing in the protocol and suggested the modification to eliminate those errors. There is very few work done in simulation of AODV protocol using Timed Petri nets [17].

# Chapter 3

Modeling Of AODV Protocol Using Coloured Petri Nets



## Chapter 3

# Modeling Of AODV Protocol Using Coloured Petri Nets

This chapter deals with modeling of AODV routing protocol using Coloured petri net.

### 3.1 Ad hoc On demand Distance Vector Protocol

AODV is a routing protocol defined for MANETs and other wireless ad-hoc networks [20]. Routes for new destinations are quickly discovered by the mobile nodes in MANET and they also respond to link breakages and changes in network topology [5]. Whenever there is a link breakage, effected nodes are notified so that they invalidate the routes using that link. In order to identify the most recent path, AODV makes use of a destination sequence number [20]. In order to build the routes in AODV routing protocol, Route request and Route reply query cycles [11] are used. As has been done in (Gordon 2001), creating a formal model for the routing protocol and analyzing the model to determine if it provides the defined services correctly is one of the approach to ensure the correctness of an existing routing protocol.

### 3.2 AODV Terminology

- **Active route/valid route:** A route is said to be active route if it has a corresponding entry in the routing table entry marked as valid. For trans-

mitting data packets only active routes in the network can be used.

- **Destination:** It is an IP address to which the data packets should be transmitted. A node identifies itself as the destination node by looking upon the appropriate field in the packet.
- **Forwarding node:** It forwards the packets which are destined for another node. This node retransmits packets to a next hop along a path that has been set up using routing control messages.
- **Forward route:** A route set up which is used to send data packets from a node towards its desired destination by following the route discovery process.
- **Invalid route:** It is denoted as invalid in the routing table entry. This route is expired. It cannot be used to forward data packets, but it helps in providing information useful for route repairs, and also for future RREQ messages.
- **Originating node:** This node is responsible for starting an AODV route discovery process and possibly retransmit the message by other nodes in the network.
- **Reverse route:** This route is used to send the RREP message to the source once the destination is found. Any intermediate node which has the destination's address can invoke this.
- **Sequence number:** It is an automatically increasing number maintained by each node. It is used to determine the freshness of the information that is originating from the originating node.

### 3.3 AODV Process

As AODV is reactive protocol i.e. the route is created only when it is required. Packet transmission in AODV routing protocol can be multi hop i.e. using other

nodes as relay points, a node can send packet to another node beyond its transmission range and thus a node can function as a router. Every node in the network maintains its own neighbor table.

Before sending a data packet every node N first checks its routing table to see if there is a valid route to the destination or not. If it has, it sends the data packet along this route immediately. Otherwise, the route discovery procedure starts which is as follows:

- **Route discovery:** RREQ (Route Request) packet is broadcasted throughout the network if the route between source and destination is not available. As soon as a node receives a RREQ packet, it first checks whether it has received this packet earlier. If yes, then the node simply discards the packet and if not, a reverse routing entry towards the originator of RREQ packet is created. This route can be used to forward route reply later on. If any intermediate node has a valid route towards the destination node, it unicasts a RREP (Route REPLY) packet towards the source node. A node on receiving RREP packet creates a reverse route entry towards the originator of RREP packet [6].
- **Route maintenance:** In order to indicate its presence, every node in the network periodically broadcasts HELLO messages to its neighbors. A particular route is marked as invalid if a node does not receive a HELLO message from its neighbor and the node is considered to be exhausted or moved away from the network. Hence a RRER packet is sent to all the nodes and the routing table is updated [6]. In order to get rid of loops, a sequence number is maintained by every node in the network. This sequence number is incremented by the node every time a packet is sent and is stored along with the route information in the route table. It is sent along with RREQ (for source) and RREP (for destination). The most recent path to the destination is indicated by the node with larger sequence number. Therefore such nodes are always preferred [6].

If a source node wants to send a data packet, it first checks the routing table

for a valid path to the destination. If it is available, then the packet is forwarded through that route otherwise RREQ is broadcasted. This packet contains source ID, sequence number of source broadcast ID, sequence number of destination, previous ID, hop count and destination ID. The purpose of destination sequence number is to prevent the loop in the route discovery process. Every node has its own sequence number which is incremented by one every time there is a link breakage [2]. A node upon receiving this packet checks whether it has received this packet earlier, if it has it simply discards the packet otherwise a RREP packet is unicasted to the source along the reverse path to that of RREQ packet [21]. Setting of a forward path entry to the destination is done when an intermediate node receives this RREP packet. AODV protocol has following 4 states:

- **Routcheck:** It checks the routing table to find whether the source node has an unexpired path to the destination node or not. This is done using the guard `has_validRoute()`.
- **RREQInit:** Its main function is to initiate the RREQ message when necessary. The RREQ message is rebroadcasted using the function `arc_rebroadcast()` if necessary.
- **RREQProcess:** This subpage initiates the RREP process if there is a route to the destination, or if there is no route it forwards RREQ message. This functionality is achieved by `arc_newBID()` and `arc_initiateRREP()`.
- **RREPProcess:** This subpage plays an important role in the updation of the routing table and also to forward the RREP message if necessary. To achieve these functionalities, two functions `arc_updateRoute()` and `arc_forwardRREP()` are used. RREQInit, RREQProcess and RREPProcess are three substitution transitions. If a node desires to send data packet, primitively it enters Routcheck state to check for an existing path. If there is no existing route in the routing table, the node enters RREQInit state and initiates route discovery process i.e broadcasting of RREQ packet [2].

Since there is no clear cut method to design a protocol, the designers should utilize those tools which help them in validating the operation of the protocol. The examination of each possible usecases can be done with the help of design verification tools. They are also useful to validate the operation of the protocol in every situation. Since enumerating all possible usecases manually is very difficult, the design verification tools play an important part.

### 3.4 Simulating AODV in CPN

Hierarchical colored petri nets are used for the simulation of AODV routing protocol. Codes are written in arc, place and transition, which are known as arc inscription, place inscription and transition inscription. These all inscription are written in CPN ML programming language. The figure 3.1 shows the main module of AODV. It contains the set of information which a node carries. It contains: (Source ID, Destination ID, Broadcast id, self id, Store, hop count, Destination Sequence No).

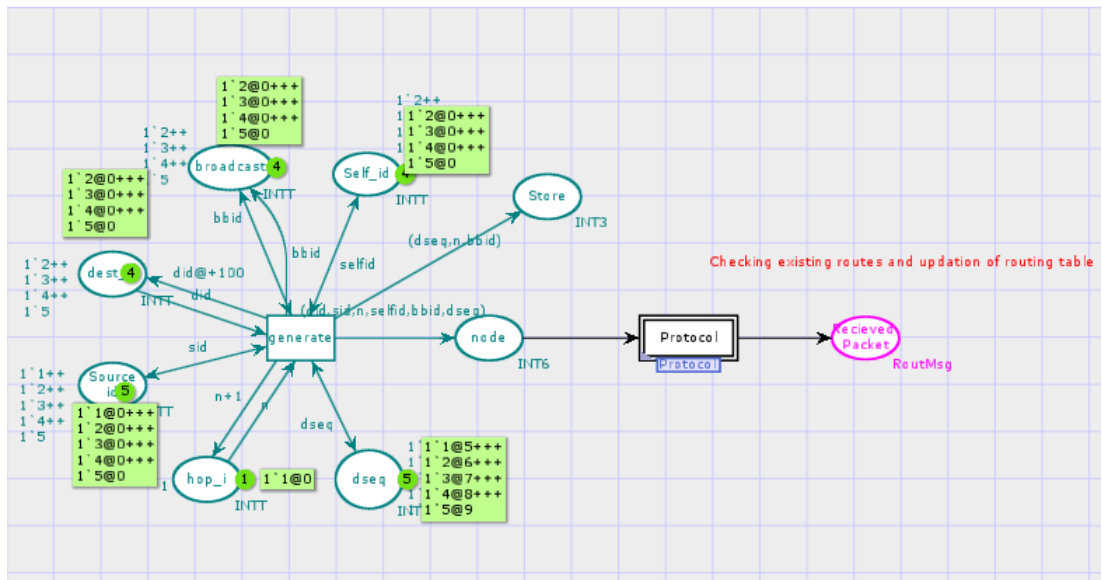


Figure 3.1: Main Module

The whole data is transmitted to the place node when the transition generate fires. Then this data is transmitted to the substitution transition Protocol.

The figure 3.2 shows the subpage of the Protocol Module. This subpage con-

tains, four places named node, RREQ, Routing table, OUT and four substitution transitions namely CheckRoute, RREQInit, RREQProcess and RREPProcess.

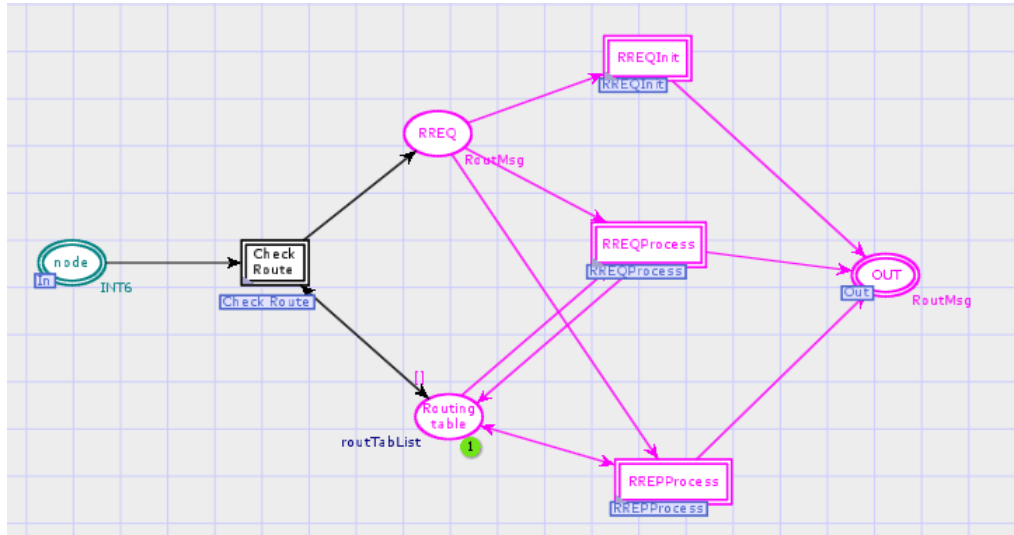


Figure 3.2: Protocol Module

The subpage of the CheckRoute substitution transition is shown in figure 3.3. Here, firstly in order to check if there is a valid route between the source and destination the routing table is checked.

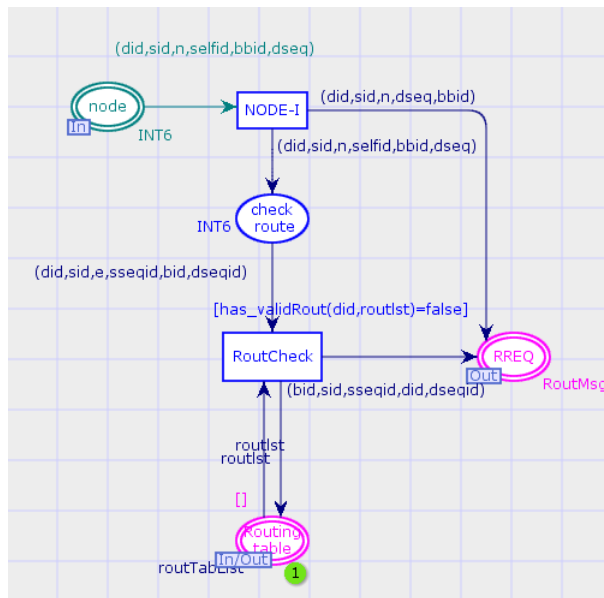


Figure 3.3: Check Route Sub Page

As shown in figure 3.4, the packet is sent using that route if path exists. If the route doesn't exist then it enters RREQInit state and route discovery process starts. The source node sends RREQ packet to its neighboring nodes.

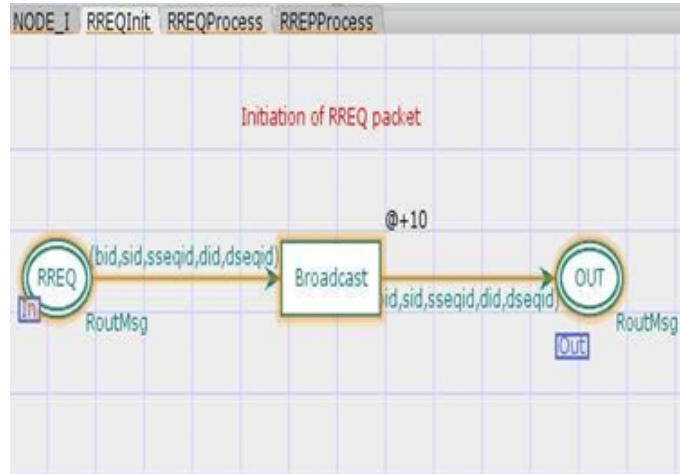


Figure 3.4: RREQInit Subpage

The figure 3.5 shows the RREQProcess subpage. Whenever a node receives RREQ packet, it checks whether it is the destination node or not (looking upon the destination field in the packet header). Hence two paths are shown in Figure 3.5 This page contains the following transitions:

- **BIDcheck**: this transition checks whether the broadcastid received is fresh one or not. It takes 10 units of time delay.
- **Broadcast**: If the current node is not the destination node then it simply forwards the RREQ packet received to its neighbors. This condition is checked using the guard function `guard_broadcast()`. It takes 2 units of time delay.

**fun** `guard_broadcast(did,routlst,selfid) = (not(has_validRout(did,routlst))) andalso (di  $\leq$  selfid);`

- **Send RREP**: This transition initiates the RREP packet when the node itself is the destination node. This functionality is checked using the guard function `guard_sendRREP()` and the RREP packet is initiated using the function `arc_initiateRREP()`.

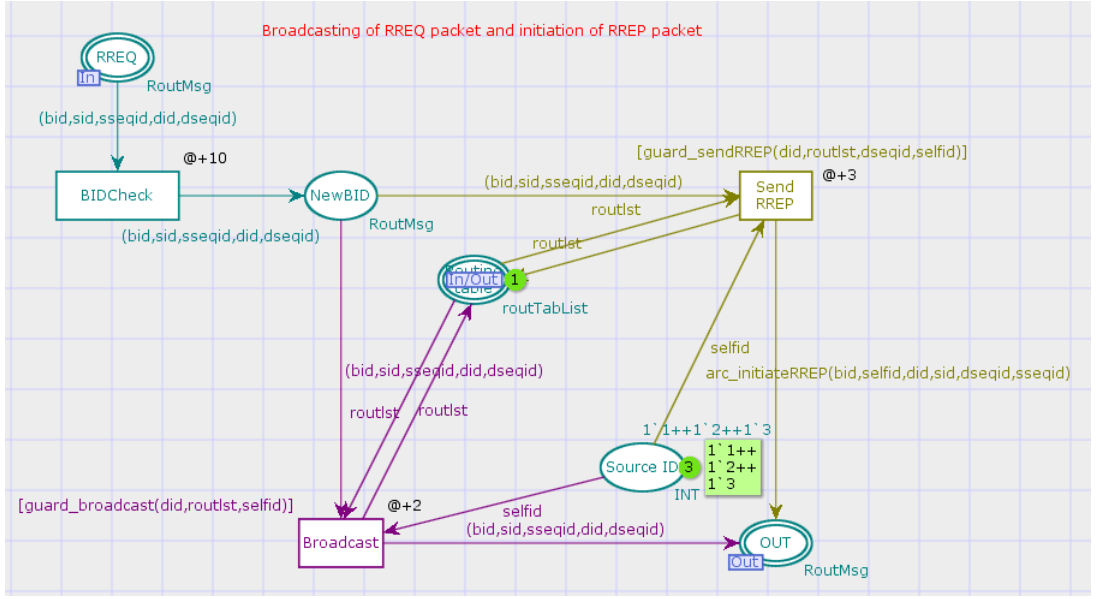


Figure 3.5: RREQProcess Subpage

```
fun guard_sendRREP(did,routlst,dseqid,selfid) = if((dseqid ≤ get_dseq(did, routlst)
andalsohas_validRout(did, routlst))or else(did = selfid))then true else false;
```

```
fun arc_initiateRREP(bid,selfid,did,sid, dseqid,sseqid)= (bid,sid,sseqid,did,dseqid);
```

The RREQProcess subpage has the following places:

- **RREQ**: It transmits the packet received from RREQInit page to this page.
- **NewBID**: Contains all the tokens which haven't received earlier.
- **Routing table**: Provides the routes using the list *routlst* to transitions Broadcast and Send RREP.
- **Source ID**: Provides the source id to both the transitions( Broadcast and Send RREP)
- **OUT**: Contains all the tokens.

The figure 3.6 shows the RREPProcess subpage. RREP packet is unicasted along the route reverse to that of RREQ packet. The page shown contains the following main transition :





When the transition generate fires, a token is placed on the place **node** as shown in the figure 3.7. As soon as the token reaches the place "node" the transition **NODE\_I** gets enabled as shown in figure 3.8

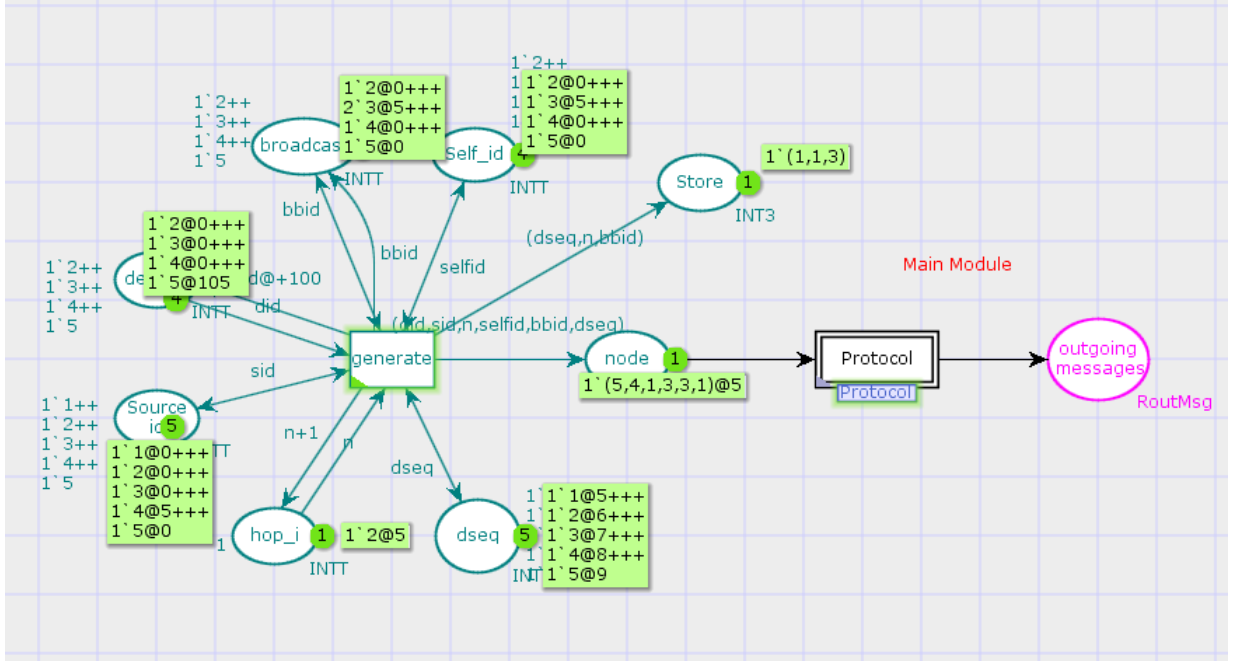


Figure 3.7: Firing of generate transition

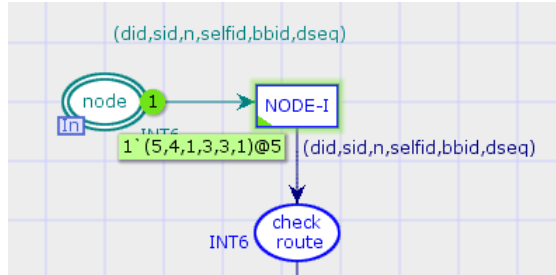


Figure 3.8: Enabling of NODE\_I

Consequently when the **NODE\_I** fires tokens are available at the place **check-route** and the transition **RoutCheck** gets enabled as shown in figure 3.9. The rectangular box next to **RoutCheck** gives the set of binding elements.

Here the guard **has\_validroute()==false** is used to determine if there is a valid route to the destination in the routing table. If the function returns true then it means that there is a valid route to the destination and hence the protocol does not do anything. But, if the result of the function is false then **RREQInit**

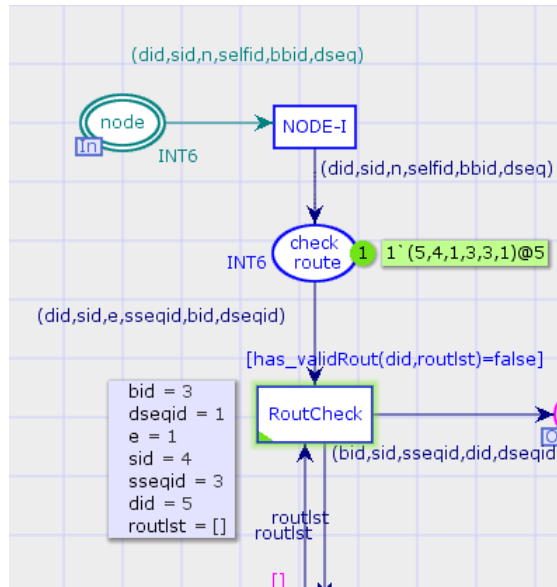


Figure 3.9: Enabling of RoutCheck

is initiated. Subsequently the RREQInit subpage enables. When tokens from the place RREQ moves to the place BIDCheck, two paths exists from this transition. If the destination node mentioned is same as that of self id, then the packet moves towards the transition RREP otherwise, same RREQ packet is broadcasted using the transition Broadcast. The transition SendRREP has a guard function `guard_sendRREP()`. As shown in figure 3.10 this transition is enabled when `did` and `self id` are same like in this case its 3.

If an intermediate node has a route to the destination then it unicasts a RREP message to the source. This is done by RREPPProcess subpage shown in figure 3.11. **AddRoute** transition is used to add the route to the routing table.

Similarly, rest of the tokens are also fired and all tokens moves to the place **OUT**. This place contains all the tokens along with their delay values.

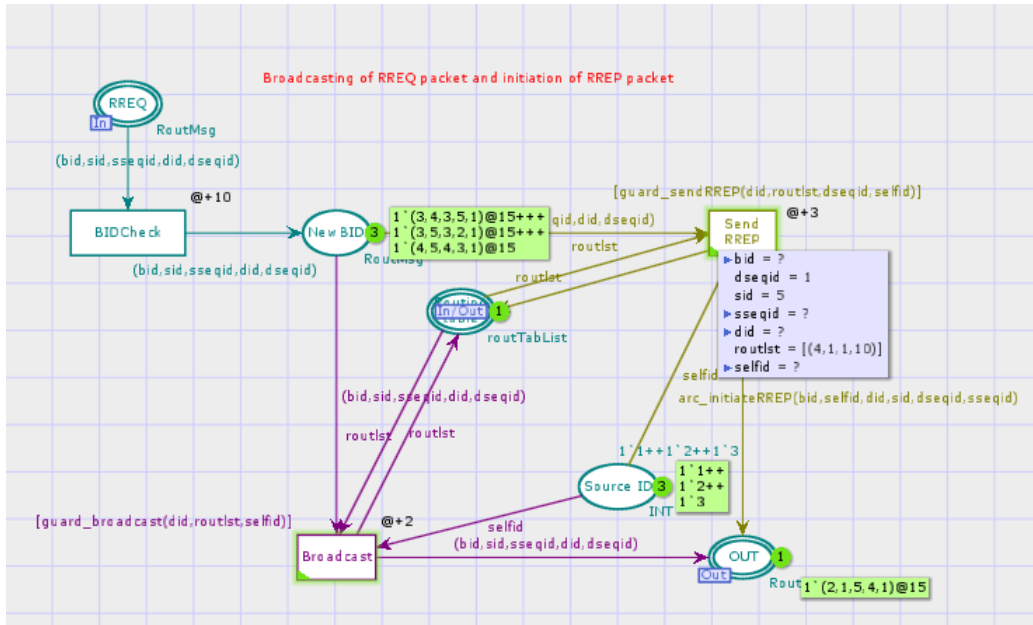


Figure 3.10: RREQProcess Subpage

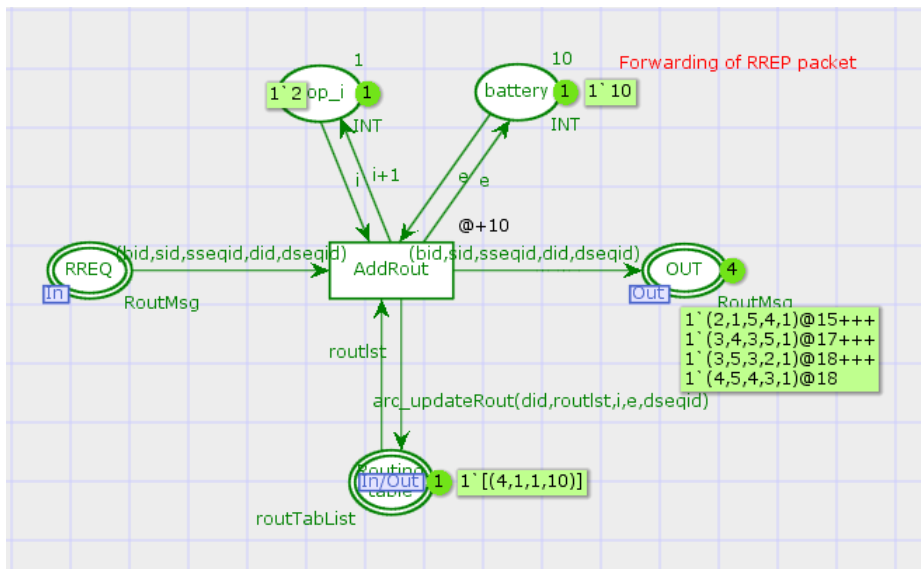


Figure 3.11: RREPPProcess Subpage

# Chapter 4

Simulation based Performance Analysis of Modeled AODV

## Chapter 4

# Simulation based Performance Analysis of Modeled AODV

In this chapter, we describe the way in which the simulation of CPN models can be used to analyze the performance of systems and thereby their efficiency. Performance plays an important role in designing a concurrent system. We can build a number of models and compare the performance of each in order to find the optimal configuration. Typically, performance measures include throughput, average delays and queue lengths. Statistical investigation of output data is involved in simulation based performance analysis.

### 4.1 Timed Protocol for Performance Analysis with Monitoring Simulations:

We consider a timed model of the protocol for performance analysis because there are a lot of interesting facts regarding performance measures that can be done including the throughput, the average packet queue lengths at the sender and in the network.

It is very useful and good way to extract useful information from the model during simulation. That information may include data which is numerical in nature and which is used for measuring the performance analysis, can be strings which can be saved to a file, otherwise they can be data values that are used for many distinct purposes.

A simulation tool should be in such a way that there is a clear distinction

between modeling and monitoring the behavior of the model. In CPN is possible with **monitors**. Therefore without modifying the structure of the system, monitors are used to inspect and control a simulation.

## 4.2 Kinds Of Monitors

As discussed above, a monitor can be used to inspect and control a simulation. Every monitor can examine both the events that occur and states of the model during simulation. The different kinds of monitors:

1. **A Simulation breakpoint monitor:** Whenever a simulation should be stopped in accordance with the occurrence of a condition, these monitors are used. There are two types of break point monitors:
  - **Place contents monitor:** It can halt a simulation depending on the numerical value of tokens present on a single place.
  - **Transition enabled monitor:** It is can halt a simulation when a particular transition is enabled.
  - **Generic break point monitor:** It is a user defined monitor, where the user can define his own breakpoint.
2. **Data collection Monitors:** These monitors can be used to collect data from the designed model. Statistics can be calculated from the data collected. Performance analysis is done by these monitors. The different types of data collection monitors are:
  - **Marking size Monitor:** On a particular place, it counts the numerical value of tokens. During a simulation, these monitors can be used to calculate the maximum and minimum number of tokens at a place.
  - **Count Transition occurrence monitor:** It counts the number of times a transition is enable i.e, it counts the number of occurrences of an event.

- **List length monitor:** The count of the position of a particular token on a place can be calculated using this monitor. These are generally used to model the queues or stacks of the system.
  - In order to monitor model specific data collection a generic **data collection monitor** is used.
3. **Write in file monitor:** This monitor is used to update a file. Updation can be appending text before or after the simulation.
  4. **User defined monitor:** The user defined monitors can be used to serve other tasks that cannot be done by the above monitors.

A monitor starts when the simulation starts, it is activated when that particular place or transition is enabled and is stopped when the simulation is stopped. There are a number of functions for each monitor called as monitoring functions.

1. **Initializing functions:** Data is extracted from the the initial state of a model with the help of initialization functions.
2. **Stop functions:** During the ending of a simulation, these functions are used to get the data from the model.
3. **Predicate functions:** These functions tell the monitor "when" to extract information from the model. This function either returns true or false.
4. **Observations Function:** This is used to determine what data should be extracted from the model. This function applies only if the Predicated function returns true.
5. **The action function:** The action function uses the values that the observation function returns. In a Data Collection monitor, the action function is used to update statistics and save the data value in a log file, whereas in a write-in-file monitor, the action function is used to add a string to a file.



## 4.3 Performance Analysis

Performance Analysis on Coloured Petri nets can be done with the help of a new feature known as monitor. The Performance analysis is done upon the numerical data that is extracted from the places and transitions. A number of performance measures can be analyzed such as

- **Workload:** In any system workload plays an important role. For eg: In a bank the number of customers act as workload. Similarly in a routing protocol the packets that are to be transmitted act as workload.
- **Queue Size:** This performance measure helps in knowing the number of packets in queue at particular place. Like the above mentioned there can be a lot of performance measures.

There are two types in which performance can be calculated. They are:

- **Log Files:** These are the files which are generated by applying a data collection monitor in any of the place in the model. The log files generally contain numerical data.
- **Performance Report:** The performance report is used to collect the continuous data. With the help of performance report one can find the maximum, minimum and average number of tokens at any place.

In this section performance analysis is done for the above modeled AODV protocol.

Following are the different types of Monitors applied on the modeled AODV protocol.

1. **Workload:** Workload for the modeled AODV is the number of packets generated. Therefore this is done by applying a **Count Transition** monitor on the transition **Generate**.

For 500 steps of simulation, totally 240 packets are generated. This is given by the performance report output as shown in the figure 4.1 In the above figure it is observed that there is minimum one token and maximum one

Untimed statistics					
Name	Count	Sum	Avrg	Min	Max
Count_trans_occur_Check_Route'RoutCheck_1	19	19	1.000000	1	1
Count_trans_occur_RREQInitBroadcast_1	6	6	1.000000	1	1
Number_of_packets_present_in_the_network	238	238	1.000000	1	1
RREP_request_packets	2	2	1.000000	1	1
RREQ_request_packets	9	9	1.000000	1	1
Workload	240	240	1.000000	1	1

Figure 4.1: Workload for 500 steps of Simulation

#data	counter	step	time
1	1	1	5
1	2	4	5
1	3	5	5
1	4	11	5
1	5	20	105
1	6	22	105
1	7	24	105
1	8	25	105
1	9	34	205
1	10	36	205

Figure 4.2: Log File of Workload for 500 steps of Simulation

token on the transition Generate. Figure 4.2 shows the log file. The log file gives a lot of information. For eg the last line of the log file says that there was one token at the transition Generate after step 36 at a model time 205 and this is the 10<sup>th</sup> measurement of the monitor.

2. **Number of packets ready to be sent:** The number of packets that are ready to be sent are calculated by applying a **Count Transition** monitor on the transition NODE.I. Figure 4.3 shows the total number of packets that are ready to be sent. The figure 4.4 shows the data collection log file of the packets to send monitor. The log file gives a lot of information. For eg the last line of the log file says that there was one token at the transition NODE.I after step 40 at a model time 205 and this is the 10<sup>th</sup> measurement of the monitor.

Untimed statistics					
Name	Count	Sum	Avrg	Min	Max
Count_trans_occur_Check_Route'RoutCheck_1	14	14	1.000000	1	1
Count_trans_occur_RREQInitBroadcast_1	7	7	1.000000	1	1
Number_of_packets_ready_to_be_sent_	234	234	1.000000	1	1
RREP_request_packets	0	0	0.000000	0	0
RREQ_request_packets	3	3	1.000000	1	1
Workload	235	235	1.000000	1	1

Figure 4.3: Number Of packets sent 500 steps of Simulation

#data	counter	step	time
1	1	4	5
1	2	8	5
1	3	9	5
1	4	10	5
1	5	20	105
1	6	24	105
1	7	29	105
1	8	30	105
1	9	37	205
1	10	40	205

Figure 4.4: Log File of Number Of Packets sent for 500 steps of Simulation

Similarly as shown in the figures, monitors are applied upon different places and transitions to calculate different performance measures.

## 4.4 Efficiency Calculation

The efficiency of any routing protocol is defined as the ratio between the number of packets sent and the number of packets received. In this section the AODV protocol is simulated in NS2 simulator and its efficiency is compared with the CPN modeled AODV.

### 4.4.1 Simulation in NS2 tool

The `aodv.tcl` file is inputted to the ns-2 simulation tool. The file contains the code for existing AODV routing protocol. It is simulated in NS2 and it gives a trace file as its output. This trace file contains the following information:

1. **Occurred Event:** 's' for SENT, 'r' for 'RECEIVED, 'd' for DROPPED.
2. **Time at which event occurred:** example 10.000.
3. **Node at which the event occurred:** Node ID like 0.
4. **Layer at which event occurred:** AGT Application layer, 'RTR Routing layer, LL Link layer, IFQ Interface Queue, MAC Mac layer,ARP link layer ARP packet.
5. **Flags**
6. **Destination Sequence number of packets:** example 0.
7. **Packet Type:** cbr CBR packet, DSR DSR packet,RTS RTS packet generated by MAC layer.
8. **Packet Size:** Packet size increases when a packet moves from an upper layer to lower layer and decreases when it moves from lower layer to an upper layer.
9. [...]: It shows the information about packet duration, mac address of destination,mac address of source and the mac type of the packet body.
10. **Show flags**
11. [...]: It shows information about source node ip: port number, destination node ip(-1 means broadcast):port number,ip header ttl and ip of next hop(0 means node 0 or broadcast).

A sample AODV.tr file is shown in the figure 4.5

The trace file generated by simulating the code in ns2 tool contains the information as shown in the figure 4.5. On the basis of these values we can extract the number of packets sent and received and also the efficiency of the protocol. This information is outputted to a text file.

```

M 1.00000 6 (400.00, 500.00, 0.00), (390.00, 390.00), 100.00
M 1.00000 7 (600.00, 200.00, 0.00), (400.00, 350.00), 100.00
M 1.00000 8 (600.00, 400.00, 0.00), (490.00, 450.00), 100.00
s 3.0000000000 _0_ AGT --- 0 tcp 40 [0 0 0 0] ----- [0:0 7:0 32 0] [0
0] 0 0
r 3.0000000000 _0_ RTR --- 0 tcp 40 [0 0 0 0] ----- [0:0 7:0 32 0] [0
0] 0 0
s 3.0000000000 _0_ RTR --- 0 AODV 48 [0 0 0 0] ----- [0:255 -1:255 30
0] [0x2 1 1 [7 0] [0 4]] (REQUEST)
s 3.0006350000 _0_ MAC --- 0 AODV 106 [0 ffffffff 0 800] ----- [0:255 -
1:255 30 0] [0x2 1 1 [7 0] [0 4]] (REQUEST)
r 3.001483266 _4_ MAC --- 0 AODV 48 [0 ffffffff 0 800] ----- [0:255 -
1:255 30 0] [0x2 1 1 [7 0] [0 4]] (REQUEST)
r 3.001483277 _2_ MAC --- 0 AODV 48 [0 ffffffff 0 800] ----- [0:255 -

```

Figure 4.5: Trace file

The efficiency of the Modeled AODV is calculated by taking into account the number of packets sent and the number of packets successfully received. The below table shows the comparison between Modeled AODV and NS2 AODV.

The reason for the less efficiency of the modeled CPN is that we are halting the simulation before all the packets have reached the destination. Therefore most of the packets are present in the network when we have taken the simulation output. We are halting simulation after a specified number of steps because, the modeled AODV goes on generating packets infinitively.

The figure 4.6 shows the efficiency graph between AODV modeled in NS2 VS AODV modeled in CPN.

		NS2 Output	CPN OutPut
Simulation 1	Packets Sent	56	56
	Packets Received	38	34
	efficiency	67%	62%
Simulation 2	Packets Sent	93	93
	Packets Received	73	67
	efficiency	78%	72%
Simulation 3	Packets Sent	170	170
	Packets Received	150	136
	efficiency	87.7%	80%
Simulation 4	Packets Sent	530	530
	Packets Received	490	450
	efficiency	92.08%	85%
Simulation 5	Packets Sent	814	814
	Packets Received	774	670
	efficiency	95.08%	82%

Table 4.1: NS2 AODV Efficiency VS CPN AODV Efficiency

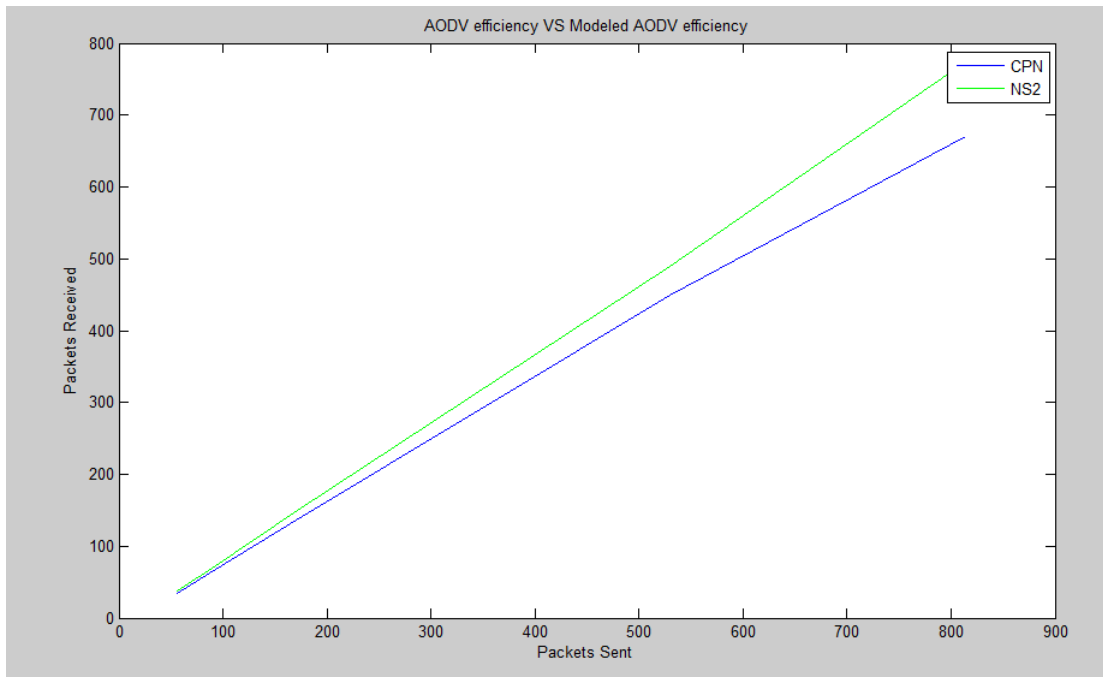


Figure 4.6: NS2 AODV efficiency VS Modeled AODV efficiency

# **Chapter 5**

**Conclusion and Future Work**

# Chapter 5

## Conclusion

In this thesis, AODV routing protocol is modeled using Coloured petri net and the performance is analyzed with the help of monitors. While modeling, various concepts like hierarchy, colour, timing were considered. The obtained performance results of the CPN modeled AODV are compared with well known Network Simulator output.

In **future work**, the modeled AODV algorithm should be constructed in such a way that there should be a finite number of packets generated. By generating a finite number of packets the validation of the protocol by state space analysis can be done. State space analysis is a new feature of Coloured Petri nets which allows to analyze the behavioral aspects of a system.



# Bibliography

- [1] D. Ahirwar, A. Kesharwani, S. K. Tehariya, and A. A. Khan, “A secure routing approach of optimized link state routing protocol,”
- [2] C. Xiong, T. Murata, and J. Tsai, “Modeling and simulation of routing protocol for mobile ad hoc networks using colored petri nets,” in *Proceedings of the conference on Application and theory of petri nets: formal methods in software engineering and defence systems-Volume 12*, pp. 145–153, Australian Computer Society, Inc., 2002.
- [3] G. Kramer, I. Marić, and R. D. Yates, “Cooperative communications,” *Foundations and Trends® in Networking*, vol. 1, no. 3, pp. 271–425, 2006.
- [4] C. A. Petri, “Communication with automata,” 1966.
- [5] L. M. Kristensen, S. Christensen, and K. Jensen, “The practitioners guide to coloured petri nets,” *International Journal on Software Tools for Technology Transfer (STTT)*, vol. 2, no. 2, pp. 98–132, 1998.
- [6] K. Jensen and L. M. Kristensen, *Coloured Petri nets: modelling and validation of concurrent systems*. Springer, 2009.
- [7] K. Jensen, *Coloured petri nets*. Springer, 1987.
- [8] A. Zimmermann, “Modeling and evaluation of stochastic petri nets with timenet 4.1,” in *Performance Evaluation Methodologies and Tools (VALUE-TOOLS), 2012 6th International Conference on*, pp. 54–63, IEEE, 2012.

- [9] R. Kodikara, S. Ling, and A. Zaslavsky, "Evaluating cross-layer context exchange in mobile ad-hoc networks with colored petri nets," in *Pervasive Services, IEEE International Conference on*, pp. 173–176, IEEE, 2007.
- [10] "Cpntools." <http://cpntools.org/>.
- [11] A. Lodhi, G. Kassem, and C. Rautenstrauch, "Modeling and analysis of business processes using business objects," in *Computer, Control and Communication, 2009. IC4 2009. 2nd International Conference on*, pp. 1–6, IEEE, 2009.
- [12] L. Wells, "Performance analysis using coloured petri nets," in *Modeling, Analysis and Simulation of Computer and Telecommunications Systems, 2002. MASCOTS 2002. Proceedings. 10th IEEE International Symposium on*, pp. 217–221, IEEE, 2002.
- [13] W. M. van der Aalst, "Interval timed coloured petri nets and their analysis," in *Application and Theory of Petri Nets 1993*, pp. 453–472, Springer, 1993.
- [14] N. Kulkarni, R. Prasad, H. Cornean, and N. Gupta, "Performance evaluation of aodv, dsdv & dsr for quasi random deployment of sensor nodes in wireless sensor networks," in *Devices and Communications (ICDeCom), 2011 International Conference on*, pp. 1–5, IEEE, 2011.
- [15] S. Chinara and S. K. Rath, "Cpn validation of neighbor detection protocol for ad hoc networks," in *Information, Communications and Signal Processing (ICICS) 2011 8th International Conference on*, pp. 1–5, IEEE, 2011.
- [16] F. Erbas, K. Kyamakya, and K. Jobmann, "Modelling and performance analysis of a novel position-based reliable unicast and multicast routing method using coloured petri nets," in *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, vol. 5, pp. 3099–3104, IEEE, 2003.
- [17] R. Kodikara, S. Ling, and A. Zaslavsky, "Evaluating cross-layer context exchange in mobile ad-hoc networks with colored petri nets," in *Pervasive Services, IEEE International Conference on*, pp. 173–176, IEEE, 2007.

- [18] A. Nakhaee, A. Harounabadi, and J. Mirabedini, “A novel communication model to improve aodv protocol routing reliability,” in *Application of Information and Communication Technologies (AICT), 2011 5th International Conference on*, pp. 1–7, IEEE, 2011.
- [19] S. Sesay, Z. Yang, and J. He, “A survey on mobile ad hoc wireless network,” *Information Technology Journal*, vol. 3, no. 2, pp. 168–175, 2004.
- [20] C. E. Perkins and E. M. Royer, “Ad-hoc on-demand distance vector routing,” in *Mobile Computing Systems and Applications, 1999. Proceedings. WMCSA '99. Second IEEE Workshop on*, pp. 90–100, IEEE, 1999.
- [21] Y. Kim, J. Jung, S. Lee, and C. Kim, “A belt-zone method for decreasing control messages in ad hoc networks,” in *Computational Science and Its Applications-ICCSA 2006*, pp. 64–72, Springer, 2006.