Modeling of AODV Routing protocol using Timed Petri nets

Shraddha

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

Modeling of AODV Routing protocol using Timed Petri nets

Dissertation submitted in

May 2013

to the department of

Computer Science and Engineering

of

National Institute of Technology Rourkela

in partial fulfillment of the requirements

for the degree of

Master of Technology

by

Shraddha

(Roll 211CS3289)

under the supervision of

Prof. S. Chinara

Department of Computer Science and Engineering

National Institute of Technology Rourkela

Rourkela – 769 008, India

Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, India. www.nitrkl.ac.in

Dr. S. Chinara
Professor

May 25, 2013

Certificate

This is to certify that the work in the thesis entitled *Modeling of AODV Routing protocol using Timed Petri nets* by *Shraddha*, bearing roll number 211CS3289, is a record of an original research work carried out by her under my supervision and guidance in partial fulfillment of the requirements for the award of the degree of *Master of Technology* in *Computer Science and Engineering*. Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

S. Chinara

Acknowledgment

Foremost, I would like to express my sincere gratitude to my advisor Prof. S. Chinara for the continuous support of my M.Tech study and research, for her patience, motivation, enthusiasm, and immense knowledge. Her guidance helped me in all the time of research and writing of this thesis. I could not have imagined having a better advisor and mentor for my M.Tech study.

Besides my advisor, I extend my thanks to our HOD, Prof. A. K. Turuk for his valuable advices and encouragement. I do acknowledge the academic resources that I have received from NIT Rourkela. I also thank the administrative and technical staff members of the Computer Science Department for their intime support.

My sincere thanks also goes to Chota Prateek, Debaditya, Mayank Agrawal, Suman kumar choudhary, Bada Pratik for helping me in my work.

I would also like to thank Suresh sir, Balu, Sugandha, Anita and tulika ma'am for motivating me time to time.

Last but not the least I would like to thank my family: my parents Mr. P. L. Piparia and Mrs. Seema Piparia and Prachi, Nikhilesh for constantly supporting me throughout my life.

Whilst it would be simple to name them all, it would not be easy to thank them enough.

I would like to dedicate this thesis to my parents.

Shraddha

Abstract

The growth of interest and research on mobile ad-hoc networks is exponentially in recent years. In a Mobile Ad hoc NETwork (MANET), wireless transmission takes place where one mobile node can send messages directly to other mobile node. One of the reactive protocol (the protocol which creates route in an on-demand basis) defined for MANETs is AODV (Ad hoc On-demand Distance Vector) routing protocol. The node movement in the dynamic environment causes frequent topology changes in the network. 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, the delay associated with a packet is calculated using timed petri net by providing the inputs manually. The same routing protocol is again validated usnig well known NS2 tool. Implementation in CPN tools requires time values to be incorporated amongst the states (i.e. places and transitions) which indicates the delay taken by a router or delay associated over a link or it may be delay due to queuing of packet. This value can be extracted for a particular route and delay value associated with it can be obtained. 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,TPN, routing protocol.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Mobile ad hoc network (MANET) is an autonomous system with no pre existing infrastructure or centralized administration. In MANET, a node can send packet directly to another node or packet transmission can be mutlihop. Network topology changes rapidly due to arbitrary movement of nodes. MANETs are also characterized by a random, dynamic and rapidly changing topology. This makes the routing algorithms fail to perform correctly, since they are not robust enough to accommodate such a changing environment. 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 [1]. AODV routing protocol [4] is a reactive routing protocol, which means that route from source node is created in an on demand basis.

1.1 Mobile Ad Hoc Network (MANET) Characteristics

A Mobile adhoc network is an autonomous collection of mobile nodes that communicate over relatively bandwidth constrained wireless links, limited battery power and etc... MANETs have several salient characteristics [7]:

- Dynamic topologies: Nodes are free to move arbitrarily; thus, the network topology which is typically multihop may change randomly and rapidly at unpredictable times, and may consist of both bidirectional and unidirectional links.

- Bandwidth-constrained,variable capacity links : Wireless links will continue to have significantly lower capacity than theirhardwired counterparts.In addition, the realized throughput of wireless communications after accounting for the effects of multiple access, fading, noise, and interference conditions,etc. is often much less than a radio's maximum transmission range.

- Energy-constrained operation: Some or all of the nodes in a MANET may rely on batteries or other exhaustible means for their energy. For these nodes, the most important system design criteria for optimization may be energy conservation.

- Limited physical security: Mobile wireless networks are generally more prone to physical security threats than are cable nets. The increased possibility of eavesdropping, spoofing, and denial-of-service attacks should be carefullyconsidered. These characteristics create a set of underlying assumptions and performance concerns for protocol design which extend beyond those guiding the design of routing within the higher-speed, semi-static topology of the fixed internet.

The deployment of a mobile ad hoc network is easy due to the absence of setting up any infrastructure for communication. Mostly such kind of network is required in militaly application and emergency rescue operations. But slowly it has been emerged into the areas of gaming, sensing, conferencing and collaborative and distributed computing [6]. This dynamic network is yet to capture most of the commercial applications. Research is still going on in this direction so that the MANET can be deployed in any area where a faster and cheaper network can be setup instantly for data communication. Mostly, the distributed applications where

the entities are strongly coupled, are not adaptive to the connection disruptions. In this regard the mobile ad hoc networks, where the nodes are highly dynamic in nature, can be most suitable for the loosly distributed applications.Here are some of the applications of mobile ad hoc networks:

- Military Services:Military Services are one of the most discussed and common application area of mobile ad hoc networks where installation of any fixed infrastructure is not possible in the enemy territories or inhospitable terrains. In this environment MANET provide the required communication mechanism in no time. Here, the soldiers are considered to be the mobile nodes.So the network is required to remain connected even though the soldiers move freely. This support is provided by the MANET. Another application in this area can be the coordination of the military objects and the personnel in the battle field.

- Emergency Services:In certain situations that are unexpected and unavoidable like search and rescue, crowd control, disaster recovery and commando operations, the use of mobile ad hoc networks is very much suitable. The major factors that favour the deployment of MANET in such situations are its self configuration with minimum overhead, unavailability of fixed or centralised infrastructure as well as freedom and exibility of mobility of the nodes. Since the ad hoc networks require minimum initial network configuration, it can also be deployed in situations where conventional infrastructure based communication is disturbed due to natural calamity or any other reason.

- Collaborative and Distributive Computing: The use of mobile ad hoc network is very much necessary in such situation where a group of researchers want to share their research findings or share their research materials during a conference or on_the_fly.

- Sensing and Gaming: Sensor network is a special case of ad hoc networks where mobility is generally not considered. However the battery power is a

key factor in sensors. Each sensor is equipped with a transceiver, a small micro-controller and an energy source. The sensors relay information from other devices to transport data to a central monitor. The sensors are used to sense the environmental condition such as temperature, pressure, humidity etc.

- Personal Area Networking: Personal communicating devices like laptops, PDAs, mobile phones create a network to share data among each other called the personal area network (PAN). The PAN cover a very short range for communication and can be used for ad hoc communication among the devices or for connecting to a backbone network.

In particular, network performance early in a simulation may differ substantially from the performance later in the simulation.

1.2 Petri nets

A Petri net (also known as a place/transition net or P/T net) is one of mathematical modeling languages for the description of distributed systems. A Petri net consists of places, transitions and arcs. Arcs run from a place to a transition or vice versa, never between places or between transitions. The places from which an arc runs to a transition are called the input places of the transition; the places to which arcs run from a transition are called the output places of the transition.Graphically, places in a Petri net may contain a discrete number of marks called tokens. Any distribution of tokens over the places will represent a configuration of the net called a marking. In an abstract sense relating to a Petri net diagram, a transition of a Petri net may fire if it is enabled, i.e. there are sufficient tokens in all of its input places; when the transition fires, it consumes the required input tokens, and creates tokens in its output places. A firing is atomic, i.e., a single non-interruptible step.

A place in a petri net contains a single token as shown in figure 1.1.

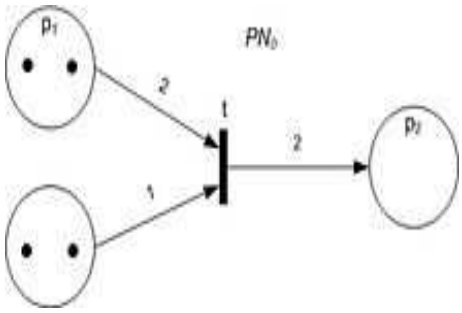


Figure 1.1: A petri net with enabled transition

1.3 Colored Petri nets

Colored Petri Net [8] provides a framework for construction and analysis of concurrent and distributed systems [9]. A Coloured Petri Net model describes the states(the places) that a system may obtain and the possible transitions in between them. The strength of CPNs over traditional Petri Nets is that, it supports hierarchy, colour, and time in the model. Hierarchy in the CPNs indicates that the models can be structured in a number of related modules. This concept is based on the concept of hierarchical structuring of the programming language [14], that supports the bottom-up or top-down style.

Modules created can be reused in several parts of the CPN model and further sub-modules can be created from it [10]. The modules of the CPNs are called pages. A complex model can have as many as hundreds of pages similar to a lengthy and complex program, that is divided into several modules. In hierarchical CPNs, a transition and its associated components make a link to another CPNs providing a more precise and detailed description of the activity represented by the transition. Such transitions are called the substitution transitions. The hierarchy inscription in the substitution transition defines the details of the substitution in separate modules called the subpages [17]. The places in a sub-page are marked with an input tag In tag,output tag Out tag or input/output tag I/Otag. These places are called the port places [17]. They constitute the interface through which the sub page communicates with its surroundings.

The sub page receives tokens from its surroundings through the input port. It delivers tokens 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 port places of the sub pages are related to the socket places of the substitution transition by providing the port assignments [practitioner's guide]. When a port place is assigned to a socket place, the two places become identical. The port place and the socket place are two different representations of a single conceptual place, i.e. the port and the socket places have always identical markings. When an input socket receives a token from the surroundings of the substitution transition, that token also becomes available at the input port of the sub-page [17], and hence the token can be used by the transitions on the sub-page. Similarly, the sub-page may produce tokens on an output port. Such tokens are also available at the corresponding output socket and hence they can be used by the surroundings of the substitution transition.

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 [1]. As has been done in (Gordon 2001), one of the approach to ensure the correctness and validity of existing routing protocol is creating a formal model for the protocol [18] 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 [19] 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 [1].

Another concept of hierarchical CPNs [2] is the fusion places. This indicates that a number of individual drawn places can be considered i.e. they all represent a single conceptual place. When a token is added or removed at one of the places, an identical token will be added or removed at all other places in the fusion set. So it is clear that the relationship between the members of a fusion set is similar

to the relationship between two places which are assigned to each other by a port assignment. When all members of a fusion set belong to a single page and that page has only one page instance, place fusion is nothing more than a drawing convenience to avoid too many crossing arcs in the model. But the situation is complex and interesting when the members of a fusion set belong to several different subpages or to a page that has several page instances. The various kinds of fusion sets are the global fusion sets, page fusion sets and instance fusion sets. The global fusion set can have members from different pages whereas the page fusion set and instance fusion sets only have members from a single page. Colors associated with each place in the CPNs determine the type of data it may handle [16]. The types of the places shown in figure **??** are similar to the types in programming languages. It can be a

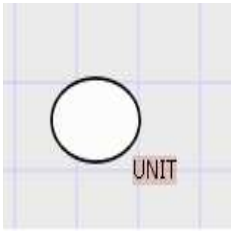


Figure 1.2: Place inscription: place type

complex type as the record which may contain heterogeneous data types. 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 state of a CPNs is called as its state that shows the number of tokens distributed on the individual places. Each token carries a value that belongs to the type of the place on which the token resides. The tokens present on a particular place denotes the marking of that place. The initial state of a place is denoted as its initial marking. It is usually written in the upper left or right of the place as shown in figure 1.3. Every place in coloured petrinet is connected to a transition which fires the tokens from one place to another. No two places and transitions can be connected.The

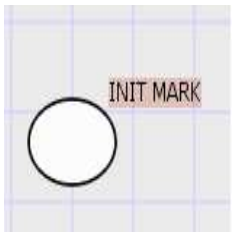


Figure 1.3: Place inscription: initial marking

place and transition are connected through arcs and an arc has an inscription which may contain a condition or an expression as shown in Figure 1.4. The inscription of

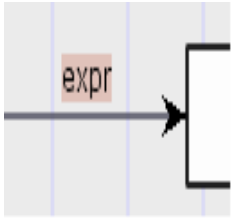
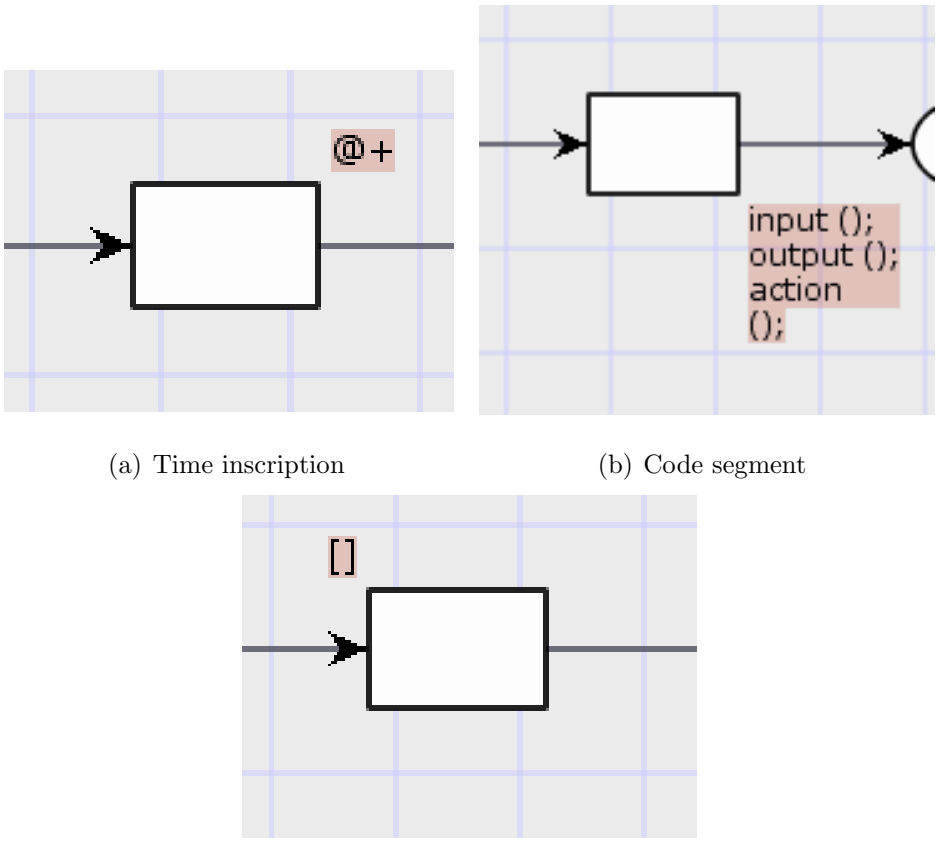


Figure 1.4: Arc inscription

transition are shown in Figure 1.5. Figure 1.5(a) shows how time values are added to the transition. Guard functions are written in top left corner of the transition as shown in Figure 1.5(c). Figure 1.5(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 1.6.

As a mathematical tool, a petri net model [3] can be described by a set of linear equations or other mathematical models reflecting the behavior of the system. This opens a possibility for formal analysis of the model. A formal check of properties can be done related to the behavior of underlying systems e.g. precedence relations amongst events, concurrent operations, appropriate synchronization, free from deadlock, repetitive activities and mutual exclusion of shared resources to some extent. Timed petrinets provides a uniform environment for modeling, design and



(a) Time inscription (b) Code segment

(c) Guard

Figure 1.5: Transition inscriptions

performance analysis of discrete event systems.

1.4 Timed petri nets

A timed CPN [15] can model how much time certain activities require and how much time passes between other activities. In most cases,its insufficient to model the average amount of time a certain activity takes- it is necessary to include a more precise representation of timing of the system. Introducing time conception into the CPNs can be redefined as timed CPNs. This introduces the concept of global clock. The clock value which is either discrete or continuous represent the model time. In the timed CPNs, each token carries a time value called the time

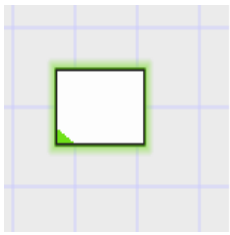


Figure 1.6: Enabling of a transition

stamp. The time stamp describes the earliest model time at which the token can be used, that is it 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 CPNs. However, to be enabled, the time stamps of the tokens to be removed must be less than or equal to the current model time. Figure 1.7 shows how timing values are incorporated in a transition. The marking of a place where

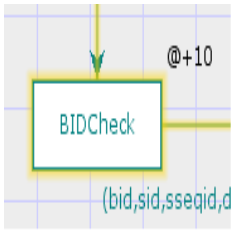


Figure 1.7: Adding timing values to a transition

the tokens carry a time stamp becomes a timed multi-set specifying the elements in the multi set together with their number of appearances and their time stamps. 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.

1.5 Network delay

Network delay is a significant design and performance feature of a computer network. The delay associated with a network tells how long it takes for data packet to travel across the network from one node or end point to another. It is typically measured in multiples or fractions of seconds. Delay may differ slightly, depending on the location of the specific pair of communicating nodes. Thus maximum and average delay needs to be calculated and the delay can be divided into several parts:

- Processing delay : time routers take to process the packet header.
- Queuing delay : time the packet spends in routing queues.
- Transmission delay : time it takes to push the packet's bits onto the link.
- Propagation delay : time for a signal to reach its destination.

1.5.1 Processing delay

In a network based on packet switching, processing delay is the time it takes routers to process the packet header. Processing delay is a key component in network delay. During processing of a packet, routers may check for bit-level errors in the packet that occurred during transmission as well as determining where the packet's next destination is. Processing delays in high-speed routers are typically on the order of microseconds or less.

After this nodal processing, the router directs the packet to the queue where further delay can happen (queuing delay).Routers performing network address translation also have higher delay than normal processing delay because those routers need to examine and modify both incoming and outgoing packets.

1.5.2 Queuing delay

The queuing delay (or queueing delay) is the time a job waits in a queue until it can be executed. It is a key component of network delay. This term is most often used in reference to routers. When packets arrive at a router, they have to be processed and transmitted. A router can only process one packet at a time. If packets arrive faster than the router can process them (such as in a burst transmission) the router puts them into the queue (also called the buffer) until it can get around to transmitting them. Delay can also vary from packet to packet so averages and statistics are usually generated when measuring and evaluating queuing delay.

As a queue begins to fill up due to traffic arriving faster than it can be processed, the amount of delay a packet experiences going through the queue increases. The speed at which the contents of a queue can be processed is a function of the transmission rate of the facility.

The maximum queuing delay is proportional to buffer size. The longer the line of packets waiting to be transmitted, the longer the average waiting time is. The router queue of packets waiting to be sent also introduces a potential cause of packet loss. When the buffer fills the router must drop packets to remain functional, resulting in data loss. Since the router has a finite amount of buffer memory to hold the queue, a router which receives packets at too high rate may experience a full queue. In this case, the router has no other option than to simply discard excess packets. If required, these may later be retransmitted by a transport protocol.

1.5.3 Transmission delay

In a network based on packet switching, transmission delay (or store-and-forward delay, also known as packetization delay) is the amount of time required to push all of the packet's bits into the wire. In other words, this is the delay caused by the data-rate of the link. Transmission delay is a function of the packet's length and has nothing to do with the distance between the two nodes. This delay is proportional

to the packet's length in bits.

Most packet switched networks use store-and-forward transmission at the input of the link. A switch using store-and-forward transmission will receive (save) the entire packet to the buffer and check it for CRC errors or other problems before sending the first bit of the packet into the outbound link. Thus store-and-forward packet switches introduce a store-and-forward delay at the input to each link along the packet's route.

1.5.4 Propagation delay

Propagation delay is the amount of time it takes for the head of the signal to travel from the sender to the receiver. It can be computed as the ratio between the link length and the propagation speed over the specific medium. Propagation delay is equal to d / s where d is the distance and s is the wave propagation speed. In wireless communication, s=c, i.e. the speed of light. This delay is the major obstacle in the development of high-speed computers.

1.6 Calculation of delay in AODV routing protocol

1.6.1 Calculation of processing delay

The processing delay, D_{proc}, is calculated as:

D_{proc}= time it takes routers to process the packet header.

1.6.2 Calculation of Queuing delay

The queuing delay, D_{que}, is calculated as:

D_{que}= time a job waits in a queue until it can be executed.

1.6.3 Calculation of transmission delay

The transmission delay,D_{trans}, is calculated using the formula:

D_{trans}=N/R

where

D_{trans} is the transmission delay

N is the number of bits, and

R is the rate of transmission (say in bits per second)

1.6.4 Calculation of propagation delay

The propagation delay, D_{prop}, is calculated using the formula:

D_{prop}=d/s

where

d is the distance and

s is the wave propagation speed.

1.6.5 Calculation of end to end delay

End-to-end delay refers to the time taken for a packet to be transmitted across a network from source to destination. Hence it is calculated by adding all the delays.

$d_{end-end}$= N[D_{trans}+D_{prop}+D_{proc}+D_{que}]

where:

$d_{end-end}$= end-to-end delay

D_{trans}= transmission delay

D_{prop}= propagation delay

D_{proc}= processing delay

D_{que}= queuing delay

N= number of links (Number of routers + 1)

1.7 Delay calculation in NS2

In NS2, delay is calculated by obtaining the difference between the start time and the end time.

1.8 Delay calculation in Timed Petri Net

In CPN tool [3], several delays are added at some transitions like adding propagation delay at transition "Broadcast",processing delay at transition "AddRout". Hence end to end delay is calculated by adding all these values.

1.9 Thesis Layout

Rest of the thesis is organized as follows —

Chapter 2: Related Work In this chapter,literarture survey is done in AODV,Coloured Petri Net tool and Timed petri net.

Chapter 3: Timed petri net modeling of AODv protocol to calculate the routing delay 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 end to end delay is calculated.

Chapter 4: Simulation in Timed Petri Net with help of NS2 In this work, the existing AODV routing protocol is simulated in NS2 tool and using generated trace file, necessary information for calculating delay value is extracted. These values are feeded into the CPN tool after the calculation of end to end delay.

Chapter 2

Literature Survey

Ad hoc On-Demand Distance Vector (AODV) Routing is a routing protocol for mobile ad hoc networks (MANETs) and other wireless ad-hoc networks. 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.

The nodes in AODV keep changing their position dynamically thus making it difficult to model the system.

Colored Petri Nets has been proved to be a powerful tool for simulating and analysing the non-determinism, concurrency and different level of abstraction of any communication protocol. Coloured Petri Nets have been used by some of the researchers for validating and modeling some of the features of the mobile ad hoc networks. Chinara et al. [8] have proposed the validation of neighbor detection protocol for ad-hoc network by using the CPN tools.

Erbas et al. [9] proposed a two designed position based routing approach based model on Colored Petri Nets for mobile ad-hoc network. Here the author shows that the multicast routing protocol delivers better result than the basic ODMRP (On Demand Multicast Routing Protocol). This model (CPN model) developed reliable unicast and multicast routing method based on geographical position of a node.

Kodikara et al. [10] 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. [11] presented new route selection criteria instead of hop count. This scheme adds two parameter to route request packet: number qmean and number maxq, shows the average queue length and maximum queue length of the nodes in the path respectively. Nakhaee et al. [12], presented an approach to improve AODV protocol routing reliability. The approaches of both the papers ([11] and [12]) are almost same. mohamed et al. Mobile ad-hoc network has several charecteristics which secernate it from conventional one.Nodes in the network suppots the network infrastruture and security required when the packets are passing.Author proposed security approach based on immunty and multiagent paradigm.It depict the how the mobile ad-hoc network is secured by different agent coordination.It has been studied that much work has not been done on the mobility pattern of the ad hoc networks by using CPN tools. This provides a motivation for the current work where CPN tools have been used for the modeling of the Random Way Point mobility for MANET [13].

The paper [19] addresses the mobility peoblem.Colored petri nets used as a simulation tool , which simulate the network without knowing the network topology.Colored petri net is a tool which provide great perceptivity to 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 comparitively 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:

1. All the nodes in the MANET have equal transmission range.
2. Every node in the MANET has the same number of neighbors which is equal to the average degree of the MANET graph.

Here, the second assumption is not a realistic approach. Because 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 [12]. The CPN tools are used by the authors for the elegant and simple modelling 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 [10].

An extension to AODV (known as MAODV) was made in [3] to support multicast routing in MANETs. In conventional AODV routing protocol, every destination node maintains a unique sequence number. In multicast routing algorithm periodic broadcast of HELLO messages in a network is done by a group leader amongst the nodes (which is by default the first node in the network until it moves out of network otherwise any random node can be chosen). The proposed algorithm creates a multi cast tree in an on demand basis and is constructed in the same way as that of route discovery process. Due to this tree formation loop is prevented.

Chapter 3

Timed petri net modeling of AODV protocol to calculate the routing delay

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

3.1 Regarding Adhoc On Demand Distance Vector Routing Protocol

AODV is a routing protocol defined for MANETs and other wireless ad-hoc networks [4]. Mobile nodes in MANET quickly discovers route for new destinations and responds to link breakages and changes in network topology [5]. AODV makes use of a destination sequence number in order to identify the most recent path [20]. Route request and route reply query cycle [11] is used in order to build the routes in AODV routing protocol. Whenever there is a link breakage, affected nodes are notified so that they invalidate the routes using that link. As has been done in (Gordon 2001), one approach to ensure the correctness of an existing routing protocol is to create a formal model for the routing protocol and analyze the model to determine if the

protocol provides the defined services correctly.

3.2 AODV terminology

- Active route/valid route: A route towards the destination node which has a routing table entry marked as valid. Only active routes in the network can be used for transmitting data packets.
- Destination: An IP address to which the data packets are transmitted. A node identifies itself as the destination node by looking upon the appropriate field in the packet.
- Forwarding node: A node that agrees to forward packets destined for another node, by retransmitting them to a next hop that is closer to the unicast destination along a path that has been set up using routing control messages.
- Forward route: A route set up to send data packets from a node originating a route discovery operation towards its desired destination.
- Invalid route: A route that is expired, denoted by the state invalid in routing table entry. An invalid route cannot be used to forward data packets, but it can provide information useful for route repairs, and also for future RREQ messages.
- Originating node: A node that starts an AODV route discovery process and possibly retransmit the message by other nodes in the network.
- Reverse route: A route set up back to the originator to forward a reply (RREP) packet from the destination node or from an intermediate node having a valid route to the destination node.
- Sequence number: A monotonically increasing number maintained by each originating node. In AODV messages, it is used by nodes to determine freshness of the information 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 [22] i.e. A node can send packet to another node beyond its transmission range using other nodes as relay point and thus a node can function as a router. Each node in the network maintains its own neighbor table.

If a node N wants to send a data packet, it first checks its routing table to find whether there is an existing path to the destination node 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: If the route between source and destination is not available, a RREQ (Route Request) packet is broadcasted throughout the network. 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 [2].

- Route maintenance: Every node in the network periodically broadcasts HELLO messages to its neighbors to indicate its presence. If the node does not receive a HELLO message from its neighbor then it marks that particular route as invalid and the node is considered to be exhausted or moved away from the network. Hence route table is updated and a RERR (Route Error) packet is sent to all the affected nodes linked with that particular node [2]. Every node in the network maintains a sequence number to ensure that no loop exists among the nodes. 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). A node with larger sequence number is always preferred since it indicates the most recent path to the destination [2].

Whenever a source node was to send data packet, routing table is checked for an unexpired entry, which if exists, packet is transmitted directly by using that route otherwise RREQ packet is broadcasted. This packet contains source ID, broadcast ID, sequence number of source, 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 increment by one every time there is a link breakage [1]. 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 uni casted to the source along the reverse path to that of RREQ packet [21]. When an intermediate node receives this RREP packet, it sets a forward path entry to the destination. AODV protocol has following 4 states:

- Routcheck: check the routing table to find if the source node has an unexpired path to the destination node. This is done using the guard has_validRoute().
- RREQInit: Initiate RREQ message when necessary. Its main function is to direct the RREQ message and rebroadcast it using the function arc_rebroadcast() if necessary.
- RREQProcess: Process the RREQ message received and output proper results. Its function is to initiate RREP message if possible or forward RREQ message if necessary. This functionality is achieved by arc_newBID() and arc_initiateRREP().
- RREPProcess: Process the incoming RREP message and output proper result. The main function of this subpage is to update the route table and 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 wants to send data packet, it first 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 [1].

Because protocol design is not yet an exact science, designers should take advantage of those tools which may aid them in validating the operation of their protocols. The use of design verification tools can aid in the examination of each possible usage case, and can validate the operation of a protocol in each of these situations. Because of the difficulty in enumerating all possible usage cases and node failure scenarios, these tools should be considered an important part of the protocol design process.

3.4 Simulating AODV in Timed Petri Net

The simulation of AODV routing protocol in Timed petri net is done with the help of hierarchical colored petri nets. 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 3.5 shows the first module of AODV. This page contains a set of information which send to the next level. It contains: (Destination ID, Source ID, Lifetime, Source Sequence No, Broadcast id, Destination Sequence No)

The transition NODE is associated with a timing delay of 10 time units which indicates the delay. The packet containing all this information moves to the next page where the path is selected.

As shown in Figure 3.5, firstly routing table is checked for an existing path between pair of source and destination node. If path exists, the packet is sent using that route only. 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

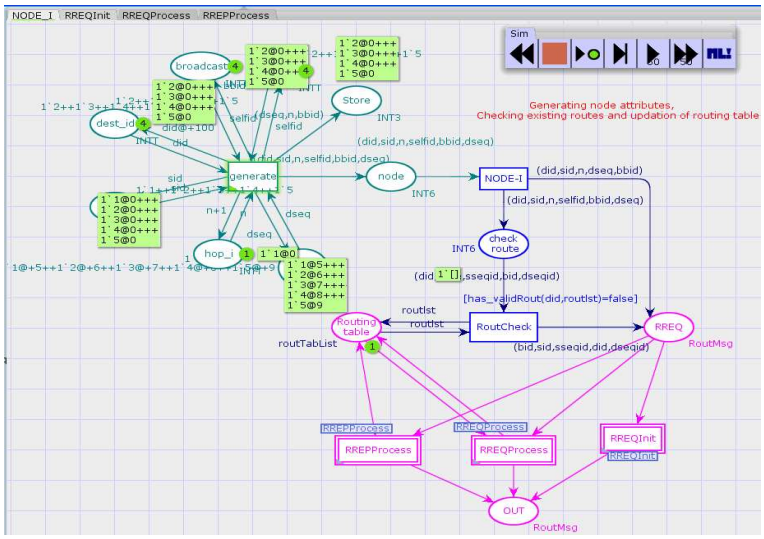


Figure 3.1: The main page of the simulation

nodes. Some of the transitions are associated with timing values like RoutCheck, BIDCheck, AddRout, etc.

A node, upon receiving the RREQ packet enters into a RREQProcess state and checks whether it is destination node or not. If it's not the destination node then guard_broadcast is enabled and RREQ packet is forwarded until it reaches the destination node. If it is the destination node then guard_sendRREP is enabled and RREP packet is initiated from the destination node using the function arc_initiateRREP().

This page has 3 substitution transitions RREQInit, RREQProcess and REPProcess.

- RREQInit sub page: It simply broadcasts the RREQ packets received. This forwarding of RREQ packet to its neighboring nodes is done until it reaches the destination node as shown in Figure 3.2. The place RREQ transmits the packet to the transition Broadcast which takes 10units of time delay to send the packet to the place OUT.

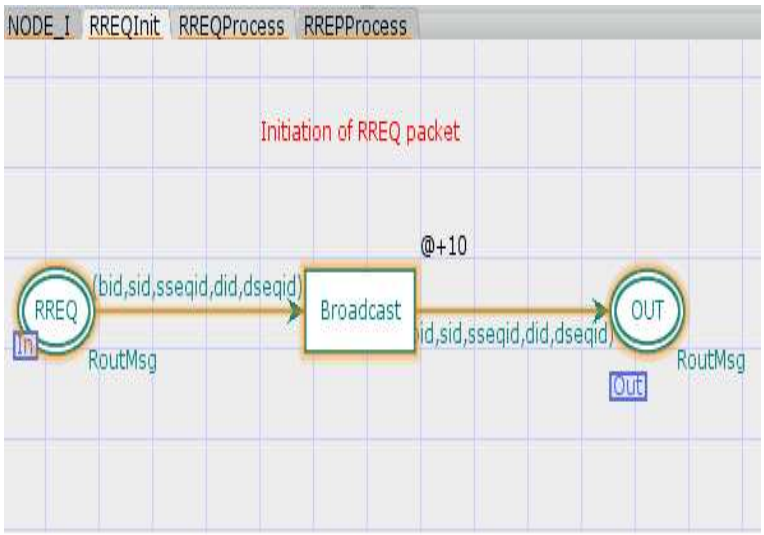


Figure 3.2: RREQInit subpage

- RREQProcess sub page: 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 3.3. This page contains the following transitions:

  - BIDcheck: this transition checks whether the broadcast_id received is fresh one or not. It takes 10units of time delay.
  - Broadcast: If the current node is not the destination node then it simply forwards the RREQ packet recieved to its neighbors. This condition is checked using the guard function guard_broadcast(). It takes 2 units of time delay.
  - Send RREP: This transition initiates the RREP packet when the node itself is the destintion node. This functionality is checked using the guard function guard_sendRREP() and the RREP packet is initiated using the function arc_initiateRREP() as shown in figure 3.3.

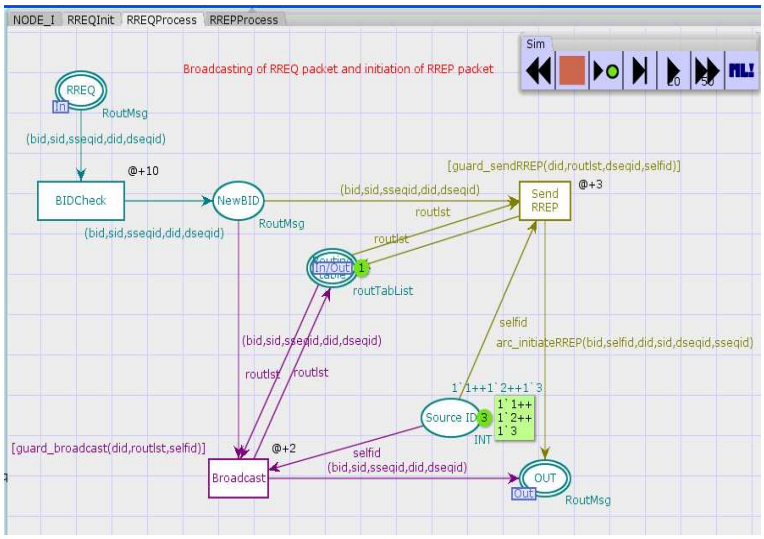


Figure 3.3: RREQProcess sub page

This page also contains the following places:

- RREQ: It transmits the packet received from RREQInit page to this page.
- NewBID: Contains all the tokens which havent 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.

- RREPProcess sub page: RREP packet is unicasted along the route reverse to that of RREQ packet. The page shown in Figure 3.4 page contains the following main transitions :
  - AddRout: Since the reply packet contains the route from source node to destination node hence this route is added to the routing table using the function arc_updateRout().
  - Calculate delay: This transition computes the time taken by a packet to travel from source node to destination node. A function extract_time() is used at the coding part of the transition to extract the delay values.
  - Combine: This transition combines the tokens and the delay values.

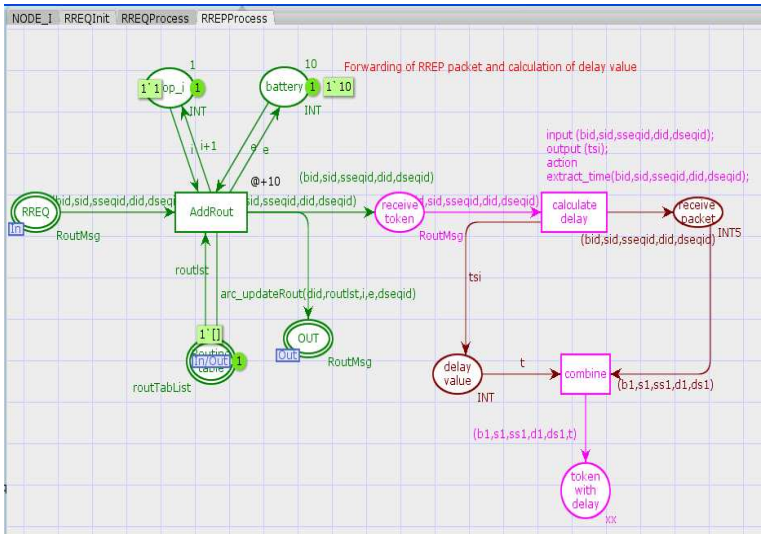


Figure 3.4: RREPProcess sub page

This page contains the following places:

- RREQ: It transmits the packet from RREQProcess subpage to add rout transition in this page.
- Hop_i: It gives the hop count.
- Battery: It assigns a very large value as battery.

- Routing table: It stores the routes using which data packet is sent. It updates the routing table.
- OUT: It contains all the information about the routes.
- Receive token: It transmits the tokens to the transition calculate delay.
- Receive packet: It contains the packet without timing values.
- Delay: It contains the delay associated with each of the packet.
- Token with delay: It combines the tokens from the place receive packet and delay i.e. contains the routes and their associated delay values.

The simulation process can be explained as follows:

Initially the transition generate contains all the attributes of a packet i.e. source id, destination id, braodcast id, destination sequence id and hop count as shown in Figure 3.5.

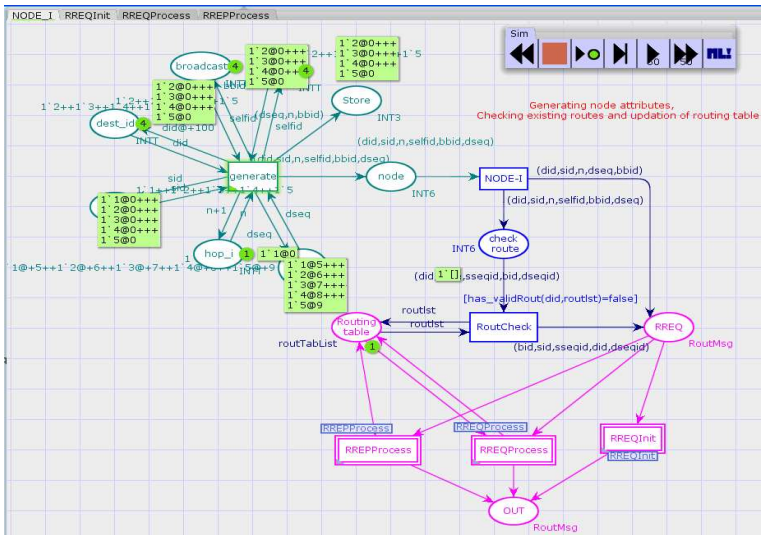


Figure 3.5: Start of simulation process

The information generated about a packet is transmitted to the transition NODE_I. Thereafter routing table is checked for an existing entry.

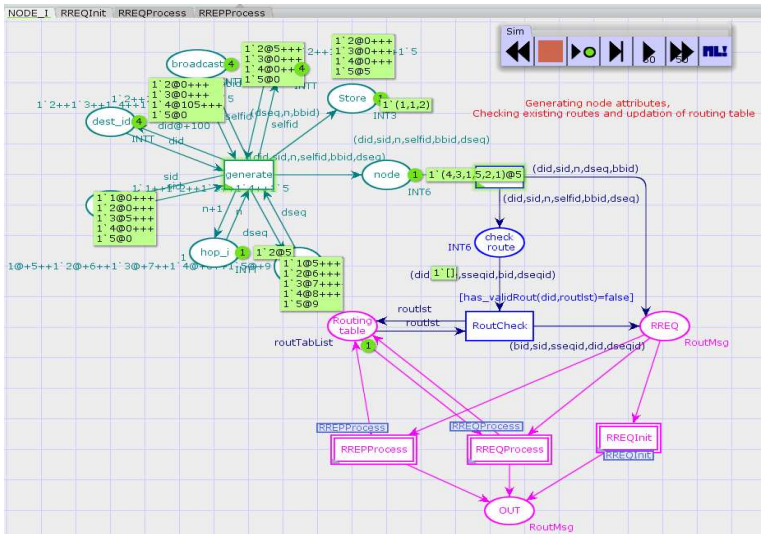


Figure 3.6: Simulation after the occurence of transition “generate”

Whenever transition “RoutCheck” is enabled, the guard function has_valid route() is evaluated. A guard is a CPN ML Boolean expression that evaluates to TRUE or FALSE. Each expression used in a guard must be a Boolean expression. Here if the expression evaluates to false then transition is enabled and the packet moves to the next subpage i.e. RREQInit subpage.

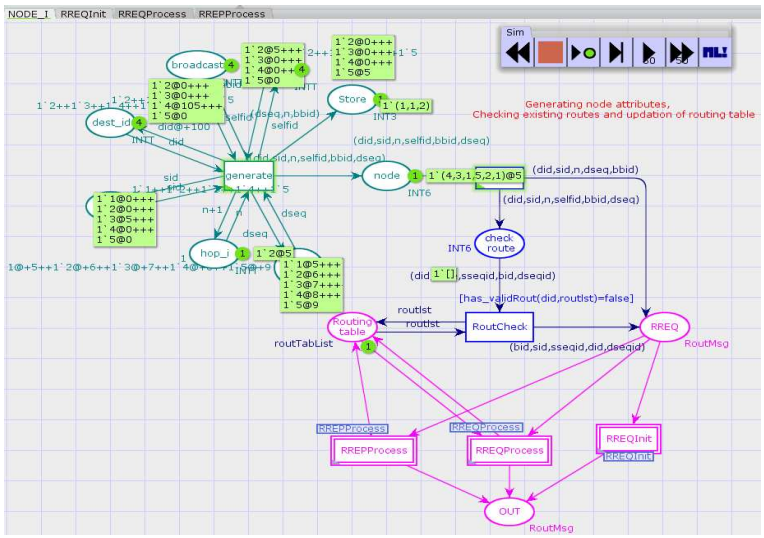


Figure 3.7: Simulation after the occurence of transition "generate"

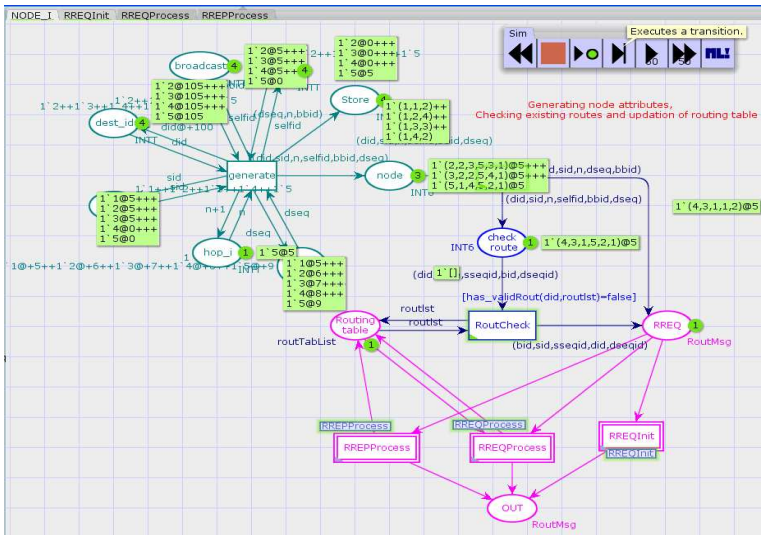


Figure 3.8: Simulation after the occurence of transition "NODE-I"

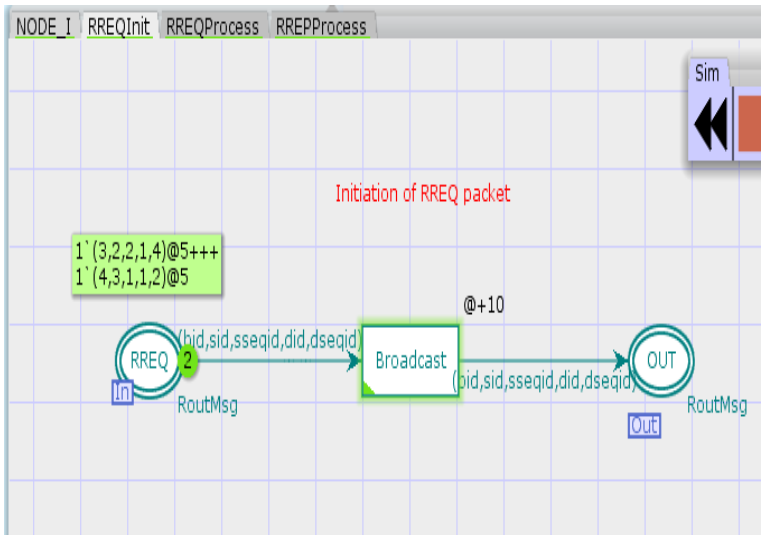


Figure 3.9: Simulation after the occurence of transition "RoutCheck"

In this subpage the transition "Broadcast" is enabled and the tokens are fired with a time delay of 10units. A transition "delay" must be a positive integer expression. The expression is preceded by @+, and this means that the time inscription has the form @+ delay-expr.Time delay is always added relative to the current time. For example here current time is 5time units and the time delay is @+10 then the time stamp of tokens sent to the output places will be 15time units.

The token after moving from this subpage goes to the main page after which it is fired to RREQProcess subpage.

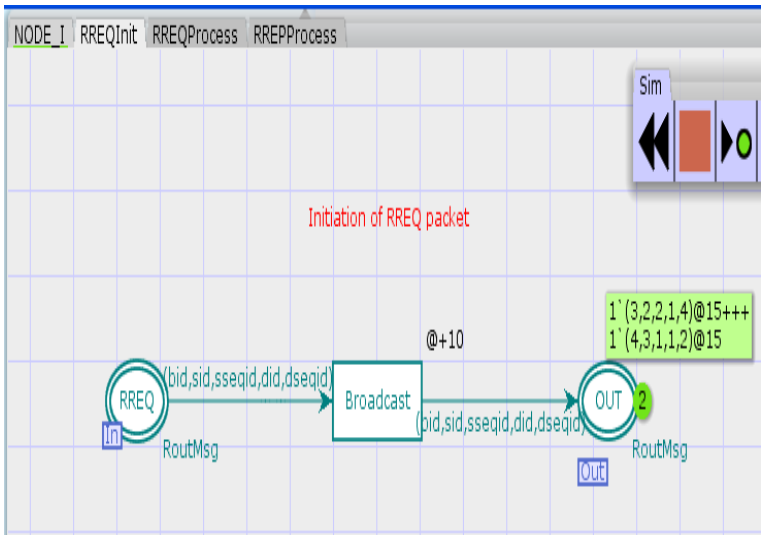


Figure 3.10: Tokens after the occurence of transition "Broadcast"

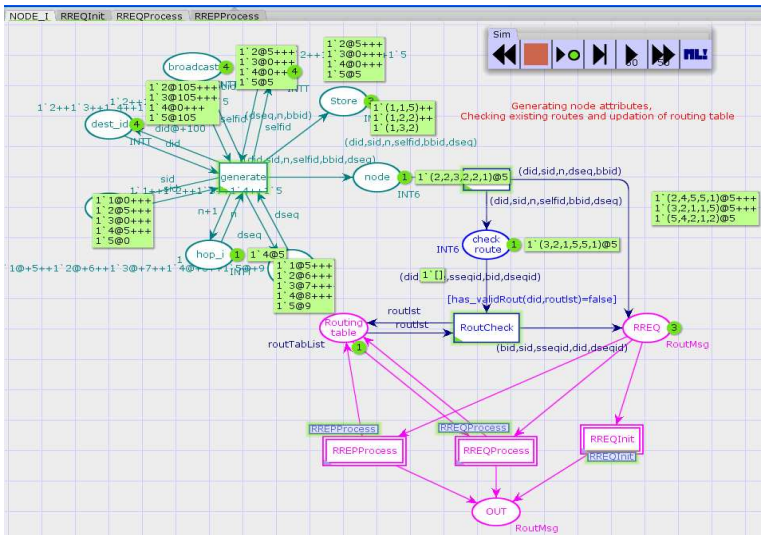


Figure 3.11: Simulation after the occurence of transition "Broadcast"

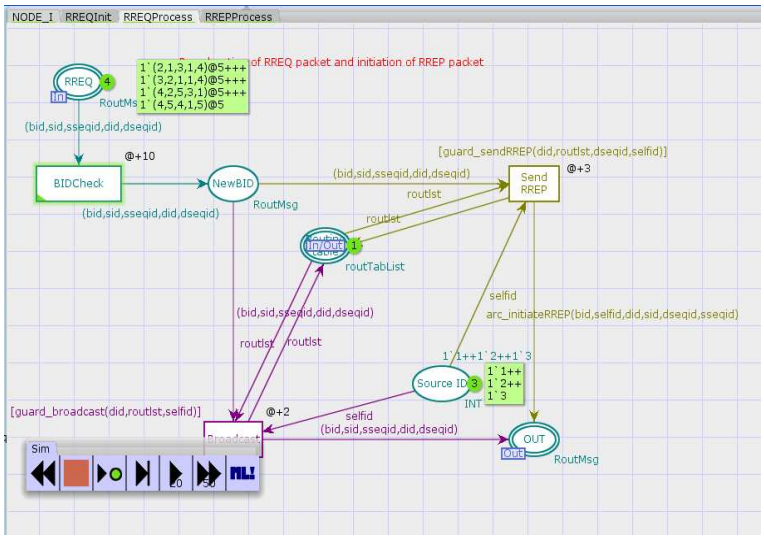


Figure 3.12: Simulation after the occurence of transition "RoutCheck"

The 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 3.14 this transition is enabled when did and self id are same like in this case its 3. Hence the transition is fired with addition of time delay of 3units as shown in figure 3.14.

Similarly the transition "Broadcast" has a guard guard_broadcast() which is enabled when the destination id and self id are not same. For example here the destination id is 2 and self id is 1, so the transition fires this token by adding time delay of 2units.

The packet then moves to RREPProcess subpage where forwarding of RREP packet and calculation of delay takes place. The packet first moves to the transition "AddRout". As shown in figure 3.16, the tokens from transition "AddRout" is transferred to two places "OUT" and "receive token". When the token is transferred to the place "OUT", the Routing table is updated using the arc inscription arc_updateRout. When the tokens are available at place "receive token", the

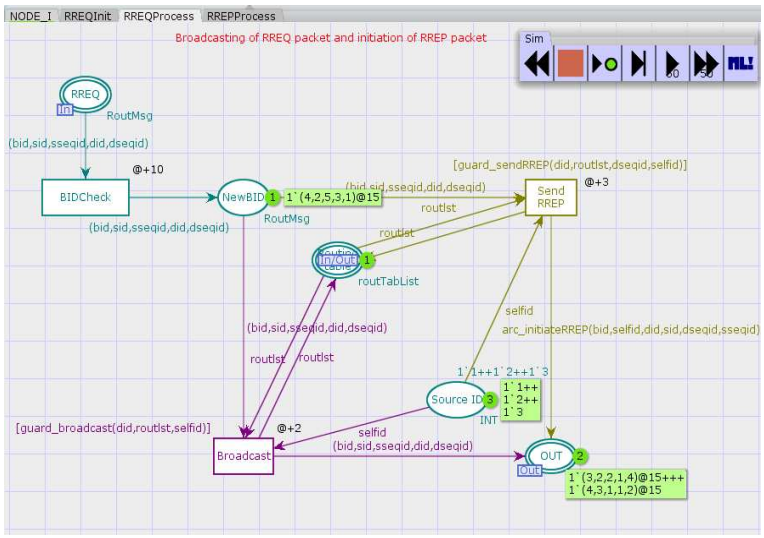


Figure 3.13: Simulation after the occurence of transition "BIDCheck"

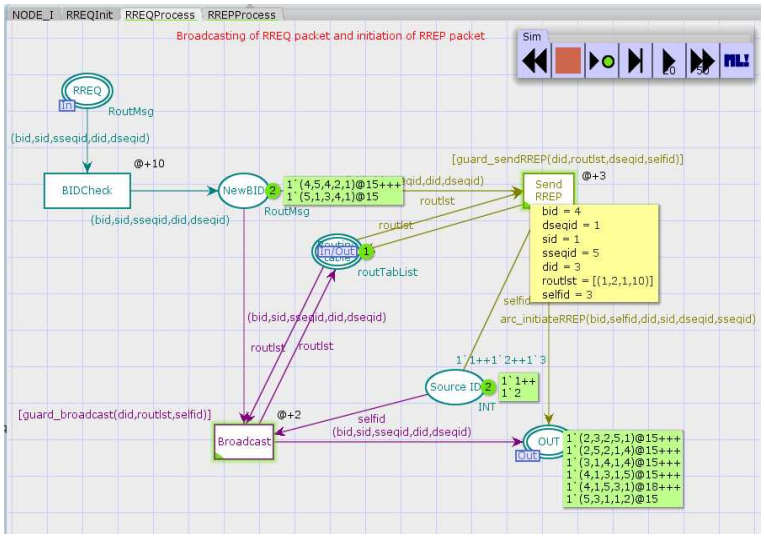


Figure 3.14: Tokens at transition "SendRREP"

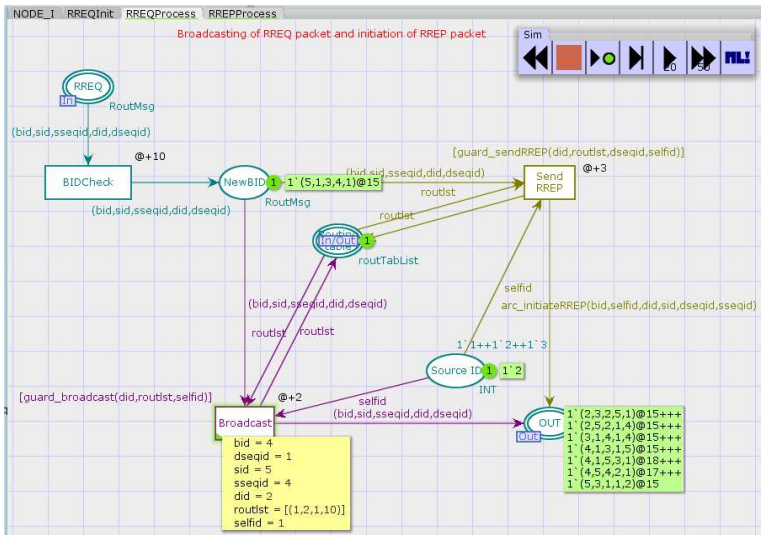


Figure 3.15: Tokens at transition “Broadcast”

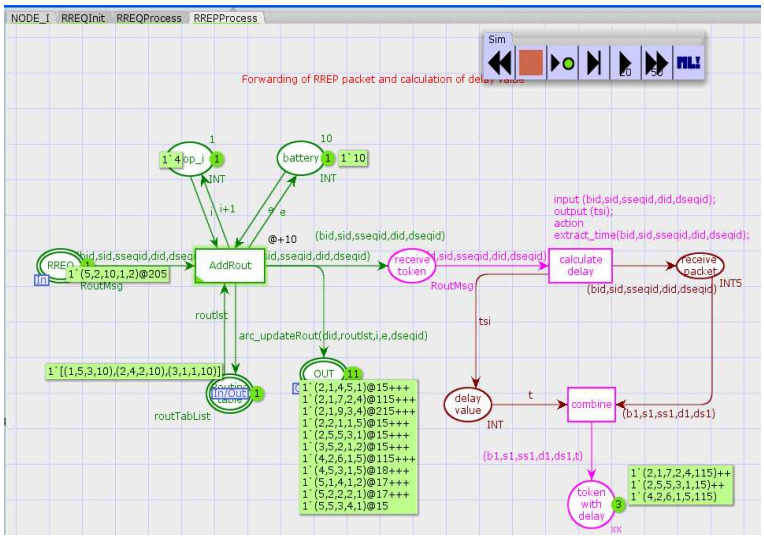


Figure 3.16: Tokens available in subpage “RREPProcess”

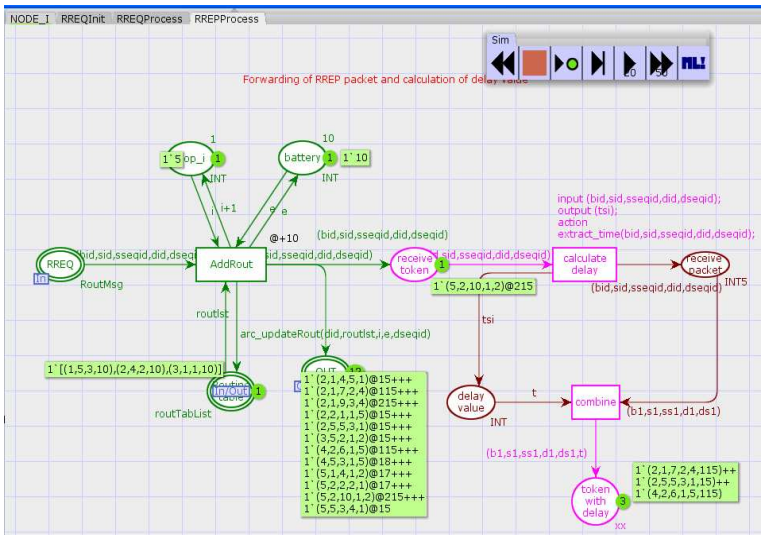


Figure 3.17: Simulation after the occurence of transition "RoutCheck"

transition "calculate delay" is enabled. This transition is associated with a coding part where the input part accepts the token and output part obtains the delay value associated with that token. The delay value is obtained using the function extract_time(). Figure 3.19 shows the place receive packet contains only the attributes of the node i.e. (5,2,10,1,2) and the place delay value contains the delay associated with that token i.e.215.

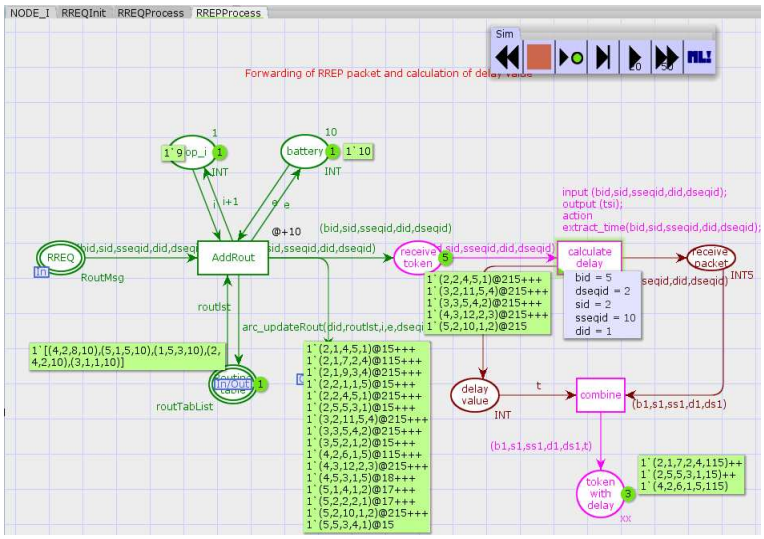


Figure 3.18: Tokens available at transition “calculate delay”

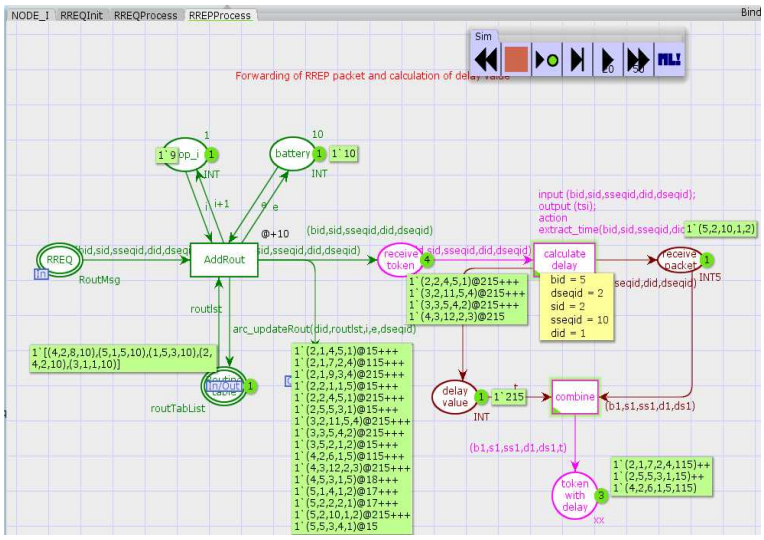


Figure 3.19: Simulation after the occurence of transition “calculate delay”

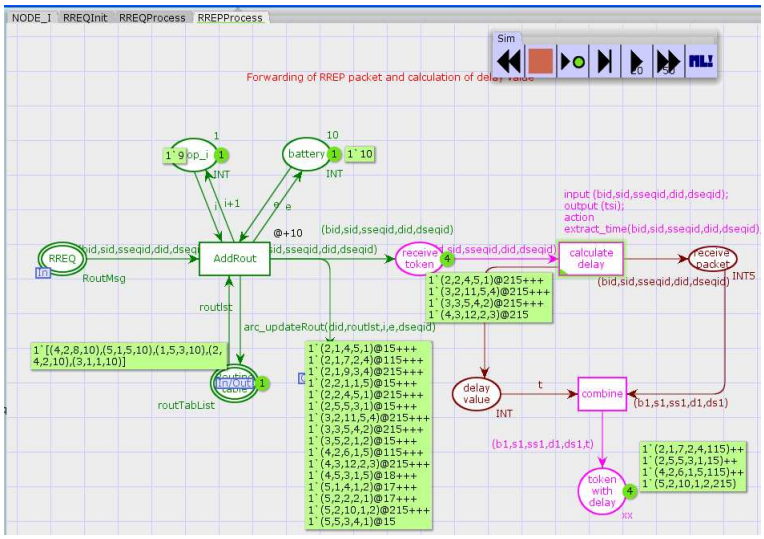


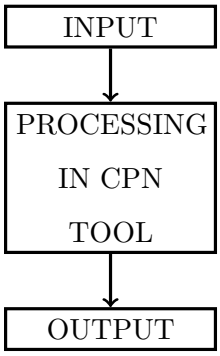
Figure 3.20: Simulation after the occurence of transition "combine"

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

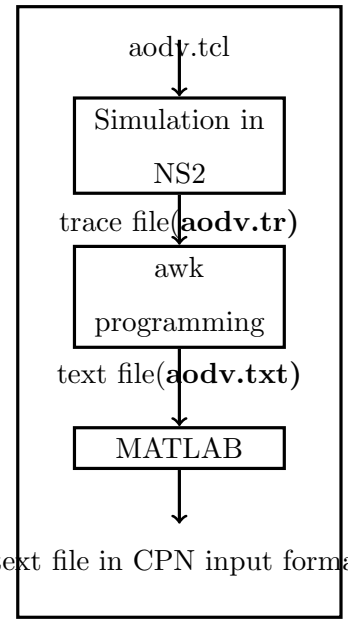
Chapter 4

Simulation in Timed Petri Net with help of NS2

In previous section, the AODV routing protocol is validated by providing the inputs manually. To improve the efficiency of simulation, the inputs are fed from a well known Network Simulation tool, NS2. The routing protocol is simulated by providing the scenario file as input. The delay is calculated using awk programming by obtaining the start time and end time from trace file. These values are modeled in CPN tool. The aimulator output remains same when the values are feeded through NS2.



(a) Flow chart showing the process



(b) Flow chart for "INPUT" part of figure (a)

Figure 4.1: Flow chart depicting the steps involved

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:

Table 4.1: Information contained in trace file

Column No	What happened ????	Values for instance....
1	It shows the occured event	's' SEND, 'r' RECEIVED,'D' DROPPED
2	Time at which the event occurred	10.000000000
3	Node at which the event occurred	Node id like 0
4	Layer at which event occured	'AGT' Application layer, 'RTR' Routing layer, 'LL' Link layer,'IFQ' Interface Queue, 'MAC' Mac layer,'ARP' link layer ARP packet
5	Show flags	–
6	Shows the sequence no of packets	0
7	Shows the packet type	'cbr' CBR packet, 'DSR' DSR packet, 'RTS' RTS packet generated by MAC layer
8	Shows the size of packet	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)

4.2 awk Programming

The trace file generated by simulating the code in ns2 tool contains the information as shown in the table. On the basis of these values we can extract the start time and end time for a particular packet by using the sequence no of the packet. The information is outputted to a text file. This file contains the source node, the destination node and the end to end delay.

4.3 MATLAB programming

The text file produced using above awk programming contains the values separated by space. In order to make it readable by cpn tools, a matlab programming is done. It converts comma separated values into one which can be inputted to CPN tools.

4.4 Simulation using Timed Petri Net

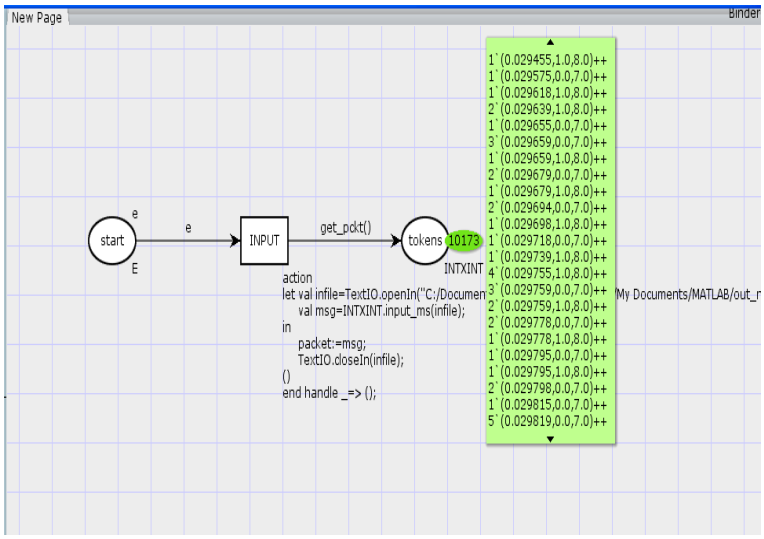


Figure 4.2: Inputting values into CPN tool

The figure 4.2 consists of a transition named “INPUT” where the code is written for inputting the values from a text file named “out_new.txt”. All the values

inputted are accumulated in the place "tokens". The vlaues inputted contains the source node,the destination node and the end_to_end delay.

Chapter 5

Conclusion

In this thesis, AODV routing protocol is modeled using Timed petri net and the delay associated with a packet is calculated. The delay can be processing delay, queuing delay,transmission delay or propagation delay. Implementation in CPN tools require timing values to be incorporated amongst the states which represents any of these delays. The addition of all these values for a particular token gives the delay associated with a packet which is termed as end to end delay. The simulation is done using CPN tools by providing the inputs manually and similar kind of result is obtained. Moreover the simulation values are imported from NS2 tool.

The future work includes the validation of the protocol by taking the inputs from well known simulator NS2.

Chapter 6

Dissemination of work

Published

Shraddha and Suchismita Chinara. Modeling of AODV routing protocol using Timed Petri Net. Proceedings of International Conference on Global Innovations in Technology and Sciences,283-288,4th-6th April 2013.

Bibliography

[1] C. Xiong, T. Murata, J. Tsai. Modeling and Simulation of Routing Protocols for Mobile Ad Hoc Networks Using Colored Petri Nets. *Conferences in Research and Practice in Information Technology*, Vol12, 2002.

[2] K. Jensen and L.M. Kristensen . Coloured Petri Nets - Modelling and Validation of Concurrent Systems. *Springer*, ISBN: 978-3-642-00283-0, 2009.

[3] T. Murata. Petri Nets: Properties, Analysis and Applications. *Proceedings of the IEEE* 77 (4), 1989, pp. 541-580.

[4] C. E. Perkins , E. M. Royer. Ad-hoc On-Demand Distance Vector Routing. *Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications*, p.90, February 25-26, 1999 .

[5] H. Zhou. A Survey on Routing Protocols in MANETs. In *Eas. Lansing, MI 48824-1027 Zhouhon1@msu.edu,Technical Report: MSU-CSE-03-08*, Mar 28, 2003.

[6] C Siva Ram Murthy and B S Manoj. Ad Hoc Wireless Networks. In *Pearson Education*, 2005.

[7] C K Toh. Ad Hoc Mobile Wireless Networks: Protocols and Systems. In *Prentice Hall PTR*, 2002.

[8] Suchismita Chinara, Santanu Kumar Rath. CPN validation of neighbor detection protocol for ad-hoc networks. In *8th International conference on information, communication and signal processing*, 2011.

[9] Fazli Erbas, Kyandoghere Kyamakya, Klaus Jobmann . Modeling 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, Volume: 5.

[10] Ruwini Kodikara, Sea Ling, and Arkady Zaslavsky. Evaluating Crosslayer Context Exchange in Mobile Ad-hoc Networks with Colored Petri Nets. In *IEEE International Conference on 2007*, Page(s): 173-176.

[11] Ali Nakhaee, Ali Harounabadi, Javad Mirabedini. A Novel Communication Model to Improve Mobile Ad hoc Network Routing Reliability. In *Application of Information and Communication Technologies (AICT), 5th International Conference on 2011*, Page(s): 1 - 7.

[12] Ali Nakhaee, Ali Harounabadi, javad Mirabedini. A Novel Communication Model to Improve AODV Protocol Routing Reliability. *Application of Information and Communication Technologies (AICT), 2011 5th International Conference on 2011*, Page(s): 1- 7

[13] E. M. Royer and C. E. Perkins. Multicast Operation of the Ad hoc On-Demand Distance Vector Routing Protocol, Proce. In *Proceedings of IEEE MOBICOM99, Seattle, WA*, August 1999, pp. 207-218.

[14] K. Jensen, L.M. Kristensen and L. Wells. Coloured Petri Nets and CPN Tools for modeling and validation of concurrent systems. In *International Journal on Software Tools for Technology Transfer*, June 2007, Volume 9, Issue 3-4, pp 213-254.

[15] Wil M.P. van der Aalst,prof.dr. j. wessels,prof dr K M vanhee. Timed coloured petrinets and their application to logistics. In *Department of Computer Science Technical University Eindhoven*, 1992.

[16] Agrahari S., and S. Chinara. Simulation of Random Waypoint Mobility Model Using Coloured Petri Nets. In *ICCTS, 18th 19th August, New Delhi*, 2012.

[17] CPN Tools. http://cpntools.org/

[18] Kurt Jensen. A brief introduction to coloured petri nets. *Tools and Algorithms for the Construction and Analysis of Systems*, pages 203-208. Springer,1997.

[19] LisaWells. Performance analysis using coloured petri nets. *Modeling, Analysis and Simulation of Computer and Telecommunications Systems.MASCOTS 2002. Proceedings, 10th IEEE International Symposium on*, pages 217-221. IEEE, 2002.

[20] Bai R., Singhal M. Salvaging Route Reply for On-Demand Routing Protocols in Mobile Ad-Hoc Networks. In *MSWIM 2005, Montreal, Quebec, Canada* October 2005.

[21] Kim Y., Jung J., Lee S., Kim C. A Belt-Zone Method for Decreasing Control Messages in Ad Hoc Networks. In *Gavrilova M.L., Gervasi O., Kumar V., Tan C.J.K., Taniar D., Lagan A., Mun Y., Choo H. (eds.) ICCSA 2006. LNCS, vol. 3982, pp. 64-72. Springer, Heidelberg*, 2006.

[22] Simaremare, Harris. Energy consumption analysis of modified AODV routing protocol under random waypoint and reference point group mobility models. *Advanced Computer Science and Information Systems (ICACSIS), 2012 International Conference on IEEE*, 2012.

[23] Hadjidj, Rachid, and Hanifa Boucheneb. *Efficient reachability analysis for time Petri nets.* Computers, IEEE Transactions on 60.8 (2011): 1085-1099.