2011 International Conference on Computer and Software Modeling
IPCSIT vol.14 (2011) © (2011) IACSIT Press, Singapore

Validation of Ad hoc On-demand Multipath Distance Vector Using Colored Petri Nets

Mohammad Ali Jabraeil Jamali[a] , Tahere Khosravi[b]

Department of Computer Science, Shabestar Branch, Islamic Azad University, shabestar, Iran

[a]Khosravi.tahere@yahoo.com

[b]M_jamali@itrc.ac.ir

Abstract. For a successful application of MANETS, it is very important to ensure that a routing protocol is unambiguous, complete and functionally correct. One approach to ensuring correctness of an existing routing protocol is to create a formal model for the protocol, and analyze the model to determine if needed the protocol provides the defined service correctly. Colored Petri Nets (CPNs) are a suitable modeling language for this purpose. However it is not easy to build a CPN model of a MANET because a node can move in and out of its transmission range and thus the MANET' topology dynamically changes. So a topology mechanism has been proposed to address this problem of mobility and perform simulations of routing protocol called Ad Hoc On demand Multipath Distance Vector routing (AOMDV). We compare Ad Hoc On demand Multipath Distance Vector routing (AOMDV) and Dynamic Source Routing Protocol (DSR) based on the simulation results of our CPN model.

Keywords: MANET, AOMDV Routing Protocol, DSR Routing Protocol, Colored Petri Net**.**

1. Introduction

Mobile ad hoc networking is a technology which works without requiring an already established infrastructure and centralized administration and provides for the cooperative engagement of a group of mobile nodes. In a mobile ad hoc network (MANET), each node can function as router and thus mobile nodes directly send messages to each other via wireless to a destination node beyond its transmission range by using other nodes as relay points. There are additional constraints compared with its hardwired counterpart, constraints such as bandwidth-constraint, energy-constrained and limited physical security [1]. With the explosive growth of the internet and mobile communication networks, challenging requirements have been added into MANETs and designing routing protocols have been added into MANET and the designing routing protocols has become more and more complex [2]. One approach to ensuring correctness of an existing routing protocol is to create a formal model for the routing protocol and analyze the model to determine if needed the protocol provides the defined service correctly. Colored Petri Nets (CPNs) is a language for the modeling and validation of systems in which concurrency, communication, and synchronization play a major role. Colored Petri Nets is a discrete-event modeling language combining Petri nets with the functional programming language Standard ML [3]. Petri nets provide the foundation of the graphical notation and the basic primitives for modeling concurrency, communication, and synchronization. Standard ML provides the primitives for the definition of data types, describing data manipulation, and for creating compact and parameterisable models. The CPN language makes it possible to organize a model as a set of modules, and it includes a time concept for representing the time taken to execute events in the modeled system [4]. Using CPN Tools, it is possible to investigate the behavior of the modeled system using simulation, to verify properties by means of state space methods and model checking, and to conduct simulation- based performance analysis [7]. The rest of this paper is organized as follows. Section 2 discusses the mobility problem of MANETs and the

topology mechanism. Section 3 presents our CPN models of AOMDV protocol. Simulation results are given in Section 4, and conclusions in Section5.

2. Mobility Problem and Previous Works

A MANET can be represented by a graph G (V, E), where V is the set of nodes representing mobile hosts and E is the set of edges representing links interconnecting mobile hosts. It is reasonable to disregard the exact locations or movements of nods when we build a CPN model for a MANET. Nevertheless, to verify the performance of a routing protocol, a CPN model still should simulate the mobility of a MANET, while it is difficult to build a dynamic structure in CPN modeling. In addition, to find a route from a source node to a destination, the source node in the CPN model should have its neighbor's identifications to send messages. It is difficult to capture structure changes in a MANET because the structure of a MANET changes in an unpredictable way. Nodes in a MANET move at will and their movements change neighboring relationships unpredictably. It is not easy to build a CPN model of a MANET as nodes can move in and out of their transmission ranges and thus MANET's topology graph dynamically changes.

We use a receiving function to decide which node receives which message. Assume every node in a MANET has the same number of neighbors which is equal to average degree of MANET graph. Since every node has the same capacity and responsibility, every node in a MANET has the same chance of receiving or forwarding broadcast message. In a real situation, neighboring nodes will directly receive a broadcast message sent by this node. In CPN modeling, it creates a place called Store to hold the entire message in transmission. After the place Store receives messages, a receiving function directs messages to the corresponding neighboring node. Clearly a node's maximum number of neighbors is equal to (n-1), where n is the number of nodes in a MANET, and all the node's neighbor will receive broadcast message sent by the node. Thus a node should maximally send (n-1) copies of a broadcast message to the place Store. For convenience in CPN modeling, we always choose (n-1) as the number of copies of the broadcast message sent or forwarded by a node. But only x out of these (n-1) copies of the broadcast message will actually received by other nodes where x is the number of neighboring nodes of this node and (n-1-x) copies will be thrown away by the CPN model. The function of how many copies of broadcast message will be received by other nodes is achieved by the receiving function in using probability bar (PB). PB is the probability of a node that will receive a broadcast message. Let d be the average degree of the MANET graph. Then on average, nodes will receive a broadcast message among all (n-1)(n-1) broadcast messages. Thus it results [5]:

$$\mathrm{PB} = d / \left((n-1) \times (n-1) \right) \tag{1}$$

Nodes in a MANET also receive unicast message such as route reply (RREP) message, which are different from broadcast message such as route request (RREQ) message. Thus the receiving function needs to check if a message is a unicast or broadcast message[6].

3. CPN Model of a MANET with AOMDV Routing Protocol

Ad hoc on-demand multipath distance vector routing (AOMDV) is based on the distance vector concept and uses hop-by-hop routing approach. It finds routes on demand using a rout discovery procedure. In this model we have 5 pages which are: (1) Node Instance Page (2) Node Mechanism Page(3)Route Request Initialization Page (4) Route request Processing Page (5) Route Reply Processing Page. Other Than These 5 pages we have one top level hierarchy Page.

We have modeled the AOMDV routing protocol with the following pages:

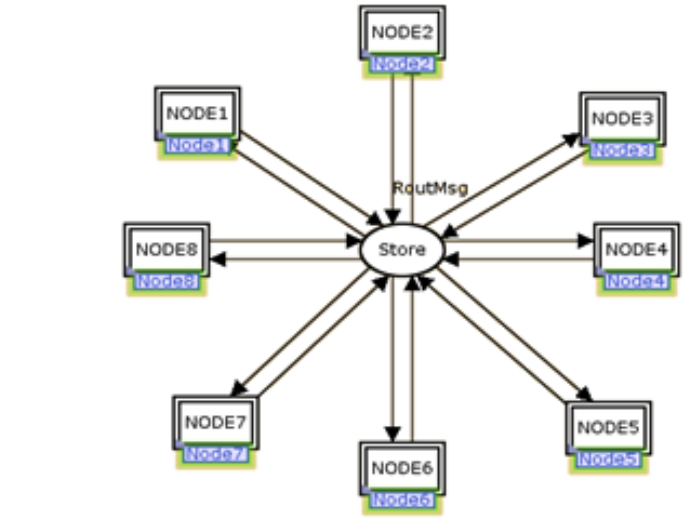
Top Level Page: In this page we can declare the instances of various nodes in the network and connect them to a common shared place "STORE" from where they communicate with each other Fig.1(a). Each of the messages that have to be sent or received is stored in a common place "STORE". Now if a node wants to send a message then it sends its packet or broadcast a definite number of packets and store it in the common place from where based on the CPN Tool simulation any node can receive the packet randomly by taking a packet from the store if it satisfies the criteria for the receiver. Similarly if a

node wants to send a reply message back a node then it provides a single copy of the packet to the store that can be sent via the path that was determined earlier either by the route cache or by the request packet. We have considered bi directional communication within the network.

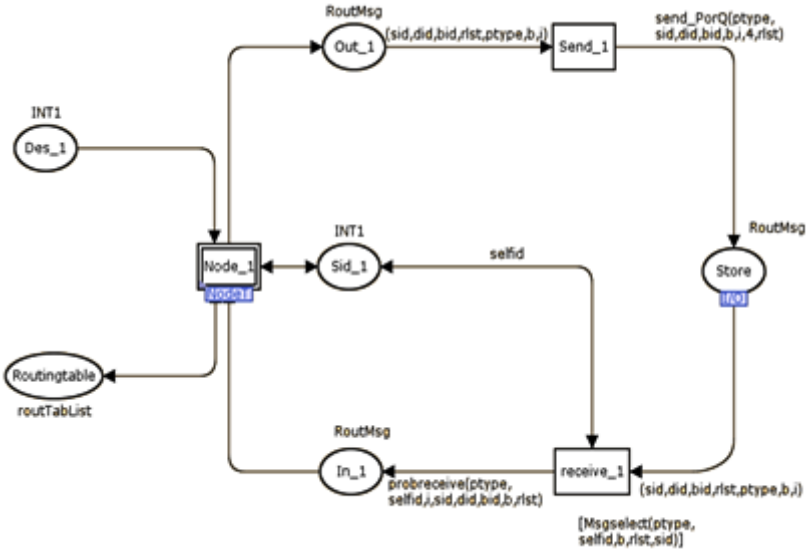
Node Instance Page: The function of the node template is just similar to that of class in an object-oriented programming language. If we want to create a node instance in a MANET, we can simply do it by instantiating the node template. Like an object in an object–oriented programming language, every instantiated node has its own local variables and provides interface to the outside world. In our CPN modeling, an instantiated node has its own local variables and provide interfaces to the outside world. In our CPN modeling, an instantiated node is represented by a substitution transition, and sockets provided by CPN Tools are its interfaces to the outside world. If more nodes are to be added to this MANET model, we can simply add substitution transition to the model. Fig.1(b). Shows a node instance in a MANET, which is a part of the CPN model in the top level. For each and every node there is one node instance page where there are places that stores the values associated with each node which are namely routingtable, selfid, destination id etc. At this level of abstraction we can actually see the packets that are sent and received by a node at any point of the time. Here we can set parameters for a node to communicate with other nodes. In this page there are two places which are "IN" and "OUT" that acts as a buffer for the packets that are being received and sent. Here also we can see the packets that are present currently in the global "STORE" that is shared by all the nodes present in the network. "Routingtable" is a place to store the routing table information about the network present with the node. Every node has a unique identification number which is used to distinguish a node from others which is stored at a place called "Self id". For every node the destination node's identification number is stored at "Destination Id". Here we have 2 transitions which are "SEND" and "RECEIVE" respectively. The "RECEIVE" transition has a guard associated with it which ensures that the node is receiving the right packet that is meant for it. The "SEND" transition simply provides the store with multiple copies of the packet if the message has to be broadcasted otherwise in case of direct reply it provides a single copy of the packet meant just for the valid node.

Node Mechanism Page: This page performs all the functionalities associated with a node that are defined in AOMDV. Every node that wants to send packet first checks its own routing table to see if there is already a path available in it which is done by the transition "ROUTE CHECK". If there is no definite route available in the table then it initialize the route discovery mechanism by the transition "RREQ INIT" which initialize the route request mechanism and broadcasts a RREQ packet in the network asking for a route to the destination node. Once a node receives a RREQ packet it can act on it by the functionality provided by the "RREQ Process" transition. If the destination node gets the broadcasted message then it sends back a RREP packet to the source node by following the path that was followed by the RREQ packet. Once a node receives a RREP packet then it can act on it by using the transition "RREP process". Also there is a unique id associated with each packet that is sent over at the network so that the same packet is not received again and again by the same node thus avoiding the loops. Fig.1(c) shows the template of a CPN node for implementing AOMDV routing protocol.

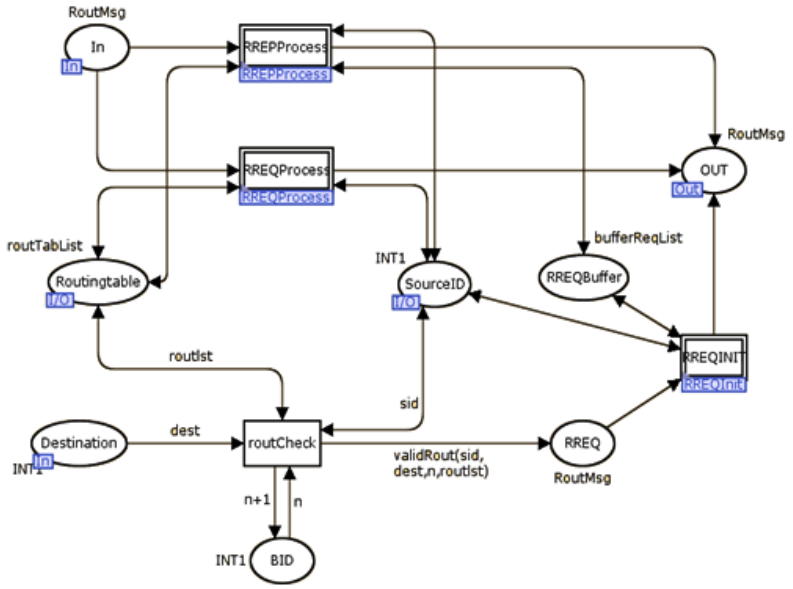
Route Request Initialization Page: As mentioned earlier once the route check returns no valid route in the table it sends a copy of packet to be broadcasted in the "RREQBuffer" in Fig.2(a). In this page that packet is taken from the RREQ Buffer and is broadcasted in the network and the required number of copies gets stored in the "STORE". While broadcasting nodes also add its own Identification Number in the route list present with the packet. Also one separate copy of the packet is stored in the "SEND BUFFER" to be rebroadcasted if the timeout occurs which is defined at the place "TIMEOUT". Also the copy can be rebroadcasted only a limited number of times that is predefined and is stored at"COUNTER". So if timer expires and the counter value is valid then the same packet is again rebroadcasted into the network.



(a)

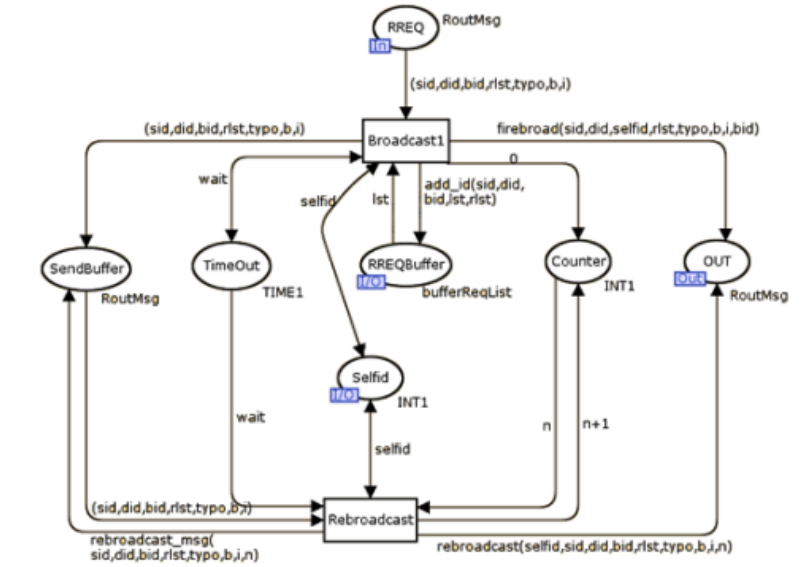


(b)

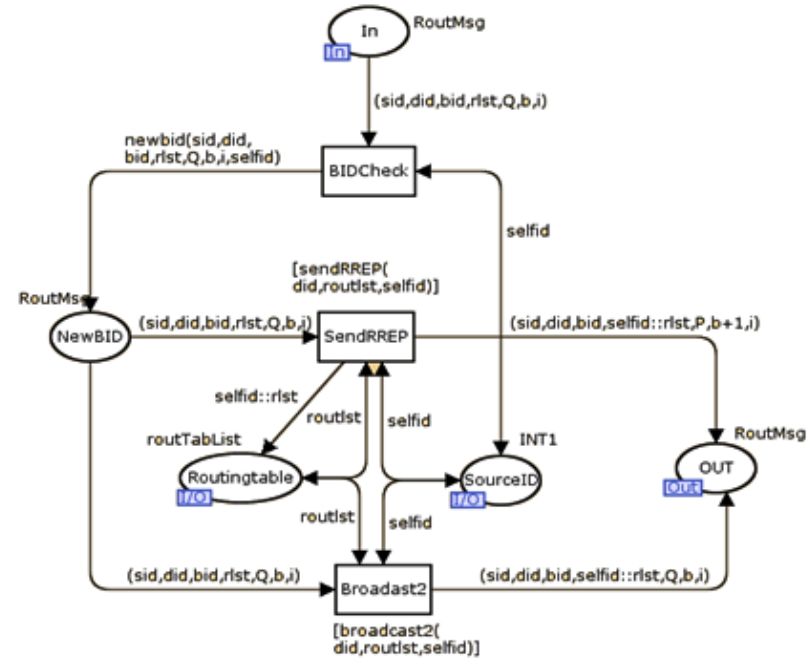


(c)

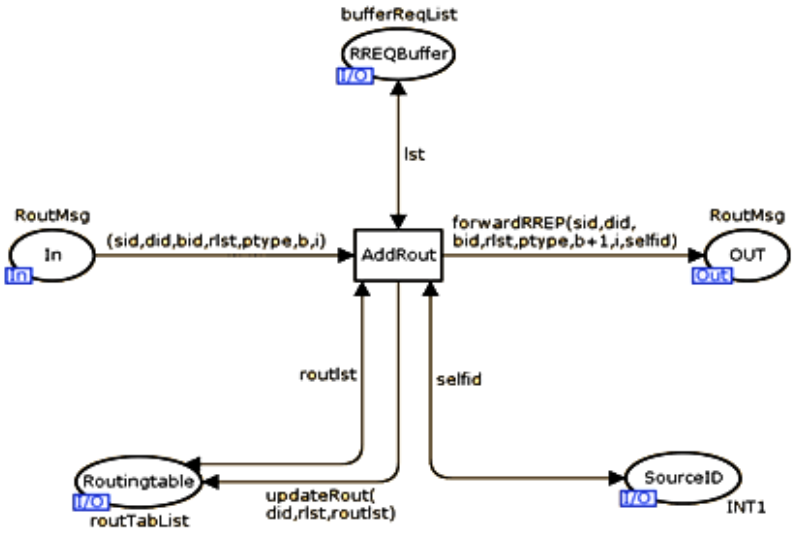
Fig.1: (a) Nodes connect to the store, (b) Node instance of a MANET at the top level showing nodes'local variables, and connections with the place Store, (c) CPN model for a node template



(a)



(b)



(c)

Fig. 2: (a) RREQInit subpage, (b) RREQProcess subpage, (c) RREPProcess subpage

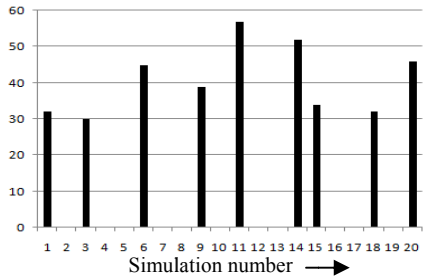
Route Request Processing Page: Now if a node receives a RREQ packet then it checks, if its own identification number is already in the route list present with the packet, check is performed by the transition "BIDCheck" in Fig.2(b). If it is not listed then it means that it is coming to this node for the first time so the route table present with the node is checked whether this node is destination or if it is having a route to destination. If this is the destination then a Route Reply packet is sent to the source or if it is having the route to the destination then it appends the route in the route contained with the packet and sends back Route Reply packet to the source. If it is none of the above two then it simple updates the packet's route list and broadcast it on the network.

Route Reply Processing Page: Once a packet reaches its destination or a node that has a valid route to the destination, a route reply packet is sent back to the source retracing the path followed by the packet that is stored in the form of route along with the packet. So if a node receives a route reply then it can be the source or it can be the intermediate node. If it is the intermediate node then it updates its route table with the information contained in the packet and sends the packet to the next node in the path to the source. If it is source then it simple updates its route table and now when it has a valid route to the destination it can send its message to the destination. Fig.2(c) shows RREQProcess subpage.

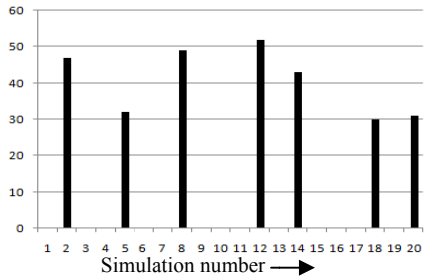
4. Simulation Experiment

For our illustrative purpose, we use a MANET example consisting of 6 nodes. Thus when a node send a broadcast message, it will send 5 copies of the message to the place called Store.

The main function of a routing protocol is to discover routes. Thus we are interested in routes found at the end of each simulation by the CPN model and validate it. The results of the simulation show that this CPN model of AOMDV can find routes. The results of simulation are shown in Fig.3. We compare AOMDV and DSR that have modeled with this method. The results show that DSR has better efficiency with lesser number of nodes present in the network but has a higher delay in network discovery which can be accounted due to the source rout present in every packet. While AOMDV has a low delay in network discovery for a small network so based on these factors there is a trade-off between delay due to network discovery and efficiency of the network or throughput.



(a) DSR protocol:
1. Throughput= %35
2. Average Delay in Network Discovery= 36



(b) AOMDV protocol:
1. Throughput= %45
2. Average Delay in Network Discovery= 42

Fig. 3: Comparison of the AOMDV and AODV protocol based on the simulation

5. Conclusions

This paper presents a CPN model and simulation of a MANET. Routing protocols used for MANETs are not novel in themselves and there have been few formal methods presently available for designing and testing protocols for MANET's due to their dynamically changing graph structures. This paper used a method to address the problem of mobility, and used this mechanism to build a CPN model of AOMDV routing protocol. The simulation result show that we can model protocols of MANET using this method since it can simulate the mobility of a MANET without knowing its actual topology. The simulation results also show that we can gain great insight into MANET routing protocol design by CPN models and CPN Tools as a simulation tool. Latency or delay is a very important consideration in

deciding routes in MANETs. As a future work, we can model the other routing protocols of MANET, validate, verify and simulate them for our purpose.

6. References

[1] M. K. Marina and S. R. Das. On-demand Multipath Distance Vector Routing in Ad Hoc Networks, Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/wcm.432.2006.

[2] M. Basagni, M. Conti, S. Giordno and I. Stojmenovic. Mobile Ad-Hoc Networking, by Wiley Interscience. 2004.

[3] K. Jensen, M. Kristensen and L.Wells. Colored Petri Nets and CPN Tools for Modeling and Validation of Concurrent Systems, Springer-Verlag. Textbook, in preparation, 2007.

[4] K. Jensen. Colored Petri Nets. Basic Concepts, Analysis Method and Practical Use, Volume 1&2: *Basic Concepts*. Springer-Verlag, 1992.

[5] C. Xiong, T. Murata and J. Tsai. Modeling and Simulation of Routing Protocol for Mobile Ad Hoc Networks Using Colored Petri Nets, 2005.

[6] P. Prasad, B. Singh and A. Sahoo. *Validation of Routing Protocol for Mobile Ad- Hoc Networks using Colored Petri Nets*. A thesis subiItted in partial fulfillment of the requirements, 2009.

[7] A. Salah and M. Khaled. Protocol Verification and Analysis Using Colored Petri Nets, July 2003.

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