Coloured Petri Net for Modelling and Validation of Dynamic Transmission Range Adjustment Protocol in an Ad-Hoc Network

Lopamudra Mohapatra and Debansu Panda

Computer Science and Engineering Department,
National Institute of Technology Rourkela
lopamudra.cse@gmail.com, debansu89@hotmail.com

Abstract. The IEEE 802.11 standard defines two operational modes for WLANs: infrastructure-based and infrastructure-less or ad-hoc. With constrained resources and limited computational capability, it may not be able for a node to serve more number of neighbours at the same time. The Dynamic Transmission Range Adjustment Protocol provides a mechanism for adjusting transmission range of the ad-hoc nodes to register or de-register a communicating node as its neighbour by dynamically varying the transmission range. Coloured Petri Nets is the modelling tool which provides a framework for design, specification, validation and verification of systems. In this paper, this tool is used to model and validate Dynamic Transmission Range Adjustment Protocol.

Keywords: Coloured Petri Nets, Ad-hoc network, Dynamic Transmission Range Adjustment Protocol, Transmission Range, Neighbouring nodes, Topology Control, Routing.

1 Introduction

In an infrastructure-based wireless network, the nodes communicate through an access point that serves as a bridge to a wired-network infrastructure. However, we can have peer-to-peer communication in an infrastructure-less network.

1.1 Ad-Hoc Network

The IEEE 802.11 defines similar modes of operation for both infrastructure-based and ad-hoc networks. The use of ad-hoc mode has an impact on the protocols implemented, and there is no effect on the physical layers (802.11a and 802.11b).

In the MAC layer, the handling of frames and the carrier-sensing responsibilities are quite the same in the modes. However, due to the absence of access-points in the ad-hoc mode, more of the MAC layer tasks must be taken up by the ad-hoc wireless LAN. Factors like saving the cost of purchase or installation of access-points, less set-up time, etc favour the use of ad-hoc network.

N. Meghanathan et al. (Eds.): CCSIT 2012, Part II, LNICST 85, pp. 89–99, 2012.

1.2 Coloured Petri Nets

To validate dynamic and adaptive routing protocols, Coloured Petri Nets serves as an excellent modelling and validation tool to depict discrete events. It combines standard MLs and Petri Nets. CPN models of a system can be built to represent the various states the system is in and the transitions that cause a change of state. Using CPN tools, simulations are run to investigate the behaviour and verify its properties. Free license for this tool is available.

2 Mobile Ad-Hoc Networks (MANET)

MANET, which communicate using radio frequency has a decentralized architecture. With the advent of MANETs, a new art in the field of network management was discovered which enabled a network to be designed without being heavily dependent of the infrastructure establishment. The Quality of Service (QoS) of the network is likely to be affected unless specialized schemes are developed to handle such dynamic networks. The nodes are the transmitting stations and are limited by a transmission range, so in most cases a direct communication between the source and receiver is not facilitated. To solve this issue, the nodes communicate in a multi-hop fashion to relay packets from the source to the destination. Therefore, the nodes function as both hosts and routers. This enables communication between nodes even when they are not in the transmission range of one another. Due to the decentralized nature of the network, it is not possible for a node to know the entire state of the network. So, the nodes frequently transmit messages to get updated about the network. Moreover, the motion of the nodes results in disruption of the links as they may frequently go out of the transmission range, thereby hampering routing in MANETs.

The MANET network interfaces differ from conventional wired network interfaces based on various scenario-dependent factors like:-

-Relative motion of the neighbours resulting in environmental and distance effects
-Dynamic local noise and interference caused by the nodes itself
-Links established between the nodes are often asymmetric in nature
-Time varying communication channels
-Lower layer wireless protocol behaviours.

2.1 Routing Approaches

Routing in MANETs is a dynamic optimization task aimed at providing an optimal path in terms of some criterion and also satisfying some constraints like limited power, transmission range, etc. Therefore, based on when the routing tables are built MANET routing algorithms are categorised as:-

1) **Proactive algorithms:** where routes to all other nodes are maintained at all times[1].

2) **Reactive algorithms:** which are on-demand routing techniques.

3) **Hybrid algorithms:** which exhibit reactive and proactive behaviour depending on the circumstances. They maintain routes to nearby nodes even if they are not needed and maintain routes to far away nodes only when needed [2].

The typical design goals for routing protocols in MANETs include: minimal control overhead, minimal processing, multihop routing, dynamically changing topology and prevention of occurrence of loops.

2.2 Protocol Stack

The MANET protocol stack is quite similar to the TCP/IP protocol stack except for the network layer. The network layer is divided into two parts- network and ad hoc routing. Since the nodes in a MANET can act as both hosts and routers, the implement an ad-hoc routing protocol to relay packets. The network part uses the IP protocol where the ad-hoc part uses various other protocols specifically designed for ad hoc networks. The lower layers, the datalink and the physical layer implement protocol designed for wireless communication.

3 Coloured Petri Nets (CPN)

Coloured Petri nets (CP-nets or CPNs) is a language used for modelling and validation of systems having concurrency, communication and synchronization as vital aspects [4]. They provide a framework for the design, specification, validation and verification of systems [5]. CP-nets is used to model discrete-event systems.CPN is a combination of Petri nets and Standard ML. Standard ML is a functional programming language that provides primitives for defining data types and for their manipulation, describing how data can be manipulated and for the creation of compact and parameterisable models [4]. A CPN model describes the various states the system is in at different points of time. It also specifies the events (transitions) that cause the system to change its state. By making simulations, the system design can be investigated, the different scenarios it can possibly be explored and the behaviour of the model can be tested therein. Also, it is possible to verify the properties of the system by state-space methods and a simulation-based performance analysis can be generated. The users can interact with the CPN tool by direct manipulation of the graphical representation using user-interaction techniques like the tool palette and marking menus [6].

3.1 Modelling of States

The basic constructs of the CPN model are:

1) **Places:** These are ellipses or circles and indicate the states of the system.

2) **Types:** Each place is associated with a type, also called as a colour-set. These indicate the type of data a place can hold, very similar to that in programming languages.

3) **Transitions:** These are represented by rectangles. The names of transitions are written inside these rectangles. The actions of the system are represented by means of transitions. An action of a CP-net consists of one or more transitions occurring concurrently.

4) **Markings:** They indicate the state of the system. Marking refers to the tokens positioned over the individual places. Tokens refers to the data value(colour) present at the place. The colour-sets are synonymous to the data-types and the colours are synonymous to the data-values.

5) **Initial Marking:** A CP-net has a distinguished marking called the initial marking, which is used to describe the initial state of the system.

6) **Arcs and Arc Expressions:** Arcs are used to connect transitions and places. A transition can be enabled only when the two criteria are satisfied:-

1) All the arc expressions must evaluate to the tokens which are present on the corresponding places connected to the incoming arc.
2) The guard conditions must be satisfied, if any.

Places and transitions are referred to as nodes. Nodes along with the directed arcs constitute the net structure. An arc always connects a place to a transition or a transition to a place. However, two transitions or two places cannot be connected directly by arcs. A number of tokens along with the token colours on the individual places are representative of the state of the system and is called a marking of the CPN model. On the other hand, the tokens on a specific place form the marking of that place [5].

3.2 Construction of Hierarchical Models

The concept of hierarchical CP-nets allows the modeller to construct a large model by an aggregation of a number of small CP-nets called pages. Substitution transitions are used to implement the hierarchy concept. They are replaced by subpages which contain the detailed description of the activity represented by the corresponding substitution transition. The subpages have a number of places marked with an In-tag, Out-tag, or I/O-tag. These places are called port places. They provide an interface through which the subpage communicates with its surroundings. The port places of the subpage must be assigned to the socket places in the main page in-order to specify the relationship between a substitution transition and the subpage. This is referred to as the port-socket assignment [4]. The Hierarchy palette provided by the GUI helps in the construction of hierarchical models. The different steps to construct hierarchical models are described later in the paper. Heirarchy can be implemented as:

Top-Down Approach:- using The Move to Submodule tool

Bottom-Up Approach:- using The Assign Submodule tool.

3.3 Syntax Check and Code Generation

CPN Tools performs syntax and type checking, and simulation code generation. The processes of syntax checking and code generation are incremental in nature. They are parallel performed with editing.When a simulation is run, the following is the simulation feedback shown:

1) Current markings of places are indicated near the individual places.
2) The number of tokens currently present at a place is given by a green circle.
3) The corresponding token values at the places are shown in green boxes.
4) Green auras are present around enabled transitions.
5) Green underlines are shown on the pages with enabled transitions.
6) The step count and time taken is shown in the index for the net under simulation. [6] The main outcome of the code generation phase is the generation of the simulation code. The simulation code contains the functions for inferring the set of enabled binding elements in a given marking of the CPN model, and for computing the marking reached after an enabled binding element occurs [4].

4 Topology Control in MANETS

The network topology is determined by the links between the nodes that is used by the routing mechanism. MANETs are indeterministic in nature and the topology of the network is dependent on a number of factors like mobility of the nodes, their battery power, the traffic patterns in the network, noise, interference and transmission power of nodes. However, these factors are subject to changes depending on the current state of the network and its present demands.

The network size, referring to the number of nodes in the network and their distribution has an effect on the performance of the network. A sparse network may have numerous network partitions and the entire network may be divided into disconnected portions. Thereby, the connectivity is hampered and it may result in lack of routes for packet transmission. On the other hand, dense networks may have problems like congestion and contention for bandwidth leading to a low packet delivery ratio. More number of collisions are also caused and thereby more energy is expended in overcoming these.

4.1 Dynamic Transmission Range Adjustment Protocol

In a multi-hop ad-hoc network, communication between two nodes may be disrupted if an intermediate node moves out of the fixed transmission range. So the nodes must be capable of dynamic reconfiguration by self-adjusting the variable transmission range. A low transmission range may result in a sparse network and the connectivity among nodes may not be effective. On the contrary, a high transmission range will ensure connectivity but collision and congestion of control packets will increase, which may significantly increase the end-to-end delay. In an operating area, when there are fixed number of nodes distributed uniformly, the optimality in transmission range is essential. Thus, a variable transmission range protocol like the Dynamic Transmission Range Adjustment Protocol (DTRAP) would be highly effective in such a dynamic environment [5].

The Dynamic Transmission Range Adjustment Protocol provides a mechanism for adjusting the transmission range of the ad hoc nodes. The nodes are configured to maintain a threshold number of neighbours based on their available resources. The nodes protect their neighbourhood relationship during data communication by controlling their transmission range. It can register or de-register a node as its neighbour by dynamically varying its transmission range in steps. However, there is a maximum limit of the transmission range. It can register or de-register a node as its neighbour by dynamically varying its transmission range in steps. However, there is a maximum limit of the transmission range beyond which it cannot be increased. If the separating distance between the nodes is less than the maximum transmission range and;

1) if the number of neighbours of a node is less than the threshold value, the node dynamically increases its transmission range in steps until it is ensured of an optimal number of neighbours.
2) if the number of neighbours of a node exceeds the threshold value, the node dynamically decreases its transmission range in steps until it is ensured of an optimal number of neighbour [5].

5 Design and Validation

In the field of multi-hop wireless network, it is almost inevitable for any design to overcome the disturbance in communication process and also a design which allows us to control the packet congestion in a dynamic field, which in turn reduces delay in end-to-end communication. The self adjustable transmission protocol, which forms a part of DTRAP is an efficient protocol with regards to the ad hoc network.

The work is based on two significant phases:

1) Design and modelling the ad hoc network using CPN.
2) Validate the above model for the Dynamic Transmission Range Adjustment Protocol using CPN.

5.1 Installation

The CPN tool software was downloaded from the Aarhus University site which provides license for the use of this tool. On downloading the set up file, it was successfully installed alongwith its component files.

As a start up the help page is referred to get used to the different options and components of of this tool.

5.2 Establishing Link by Finding Neighbours

This forms the core part of our project where in we have taken five static nodes in an ad hoc network. These five nodes share a common message store *MsgStore.* The page *MsgStore* shows the five nodes and their messages being sent to the *MsgStore.*

which portrays the establishing of link of each node with its neighbours. The subpage *AH(3)* demonstrates the neighbour node determination. Here the parameters of the node such as its co-ordinates, battery power, mobility and counter (which keeps a note of its number neighbour, initialized as zero) is broadcast to the *MsgStore*. The appropriate or intended node retrieves the *MSG* from the *MsgStore*.

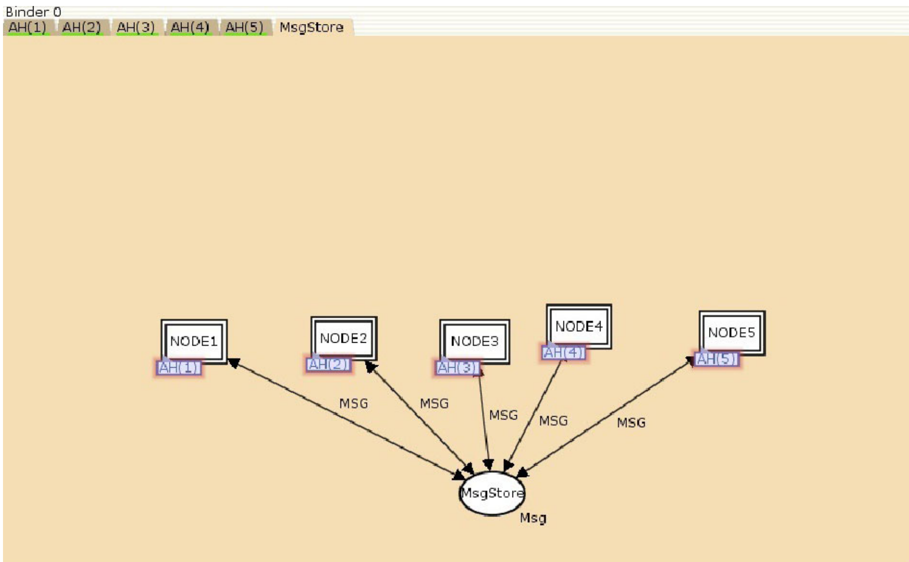

Fig. 1. Model of five nodes communicating

Distance Calculation: *CalculateDistance* transition is fed with the senders and the receivers co-ordinates. On _ring this transition it calculates the distance between them. If the distance is within a specified transmission range then it returns a Boolean yes else no.

```
fun GetDistance(k:INT,l:INT,p:INT,q:INT)=if (((k-p)*(k-
      p)+(l-q)*(l-q))<961) then yes else no;
```

Counter Increment: *IncrememntCounter* is fed with the previous step counter value and the boolean returned by *CalculateDistance*. So initially it is fed with zero. Whenever is the boolean is yes, the counter gets incremented by1.

```
fun Update Cnt(m:BOOL)=if(m=yes) then 1 else 0;
              c+Update Cnt(b);
```

The place CntNei keeps a track of the counter value for each and every node.

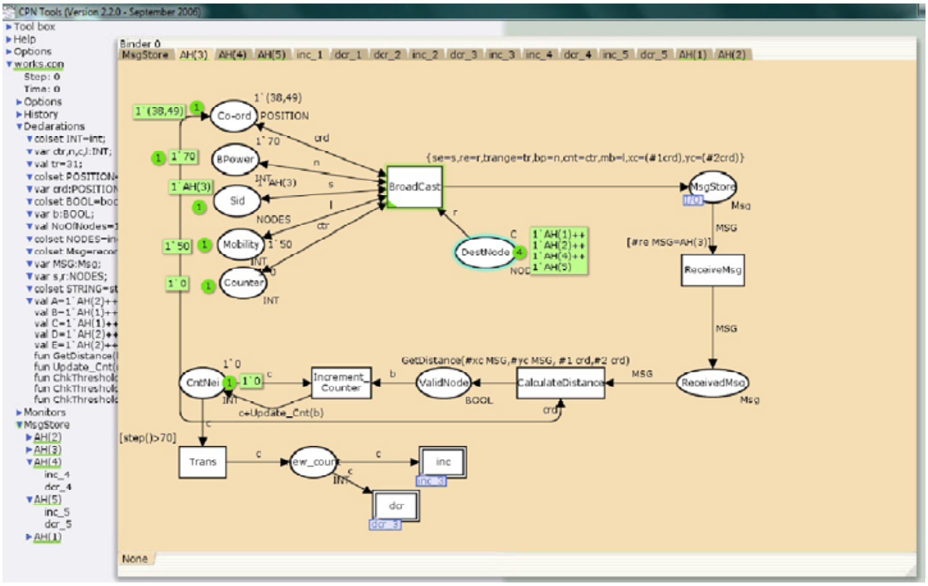

Fig. 2. Extensive view of AH(3)

5.3 Validation of DTRAP Using CPN

TRANS: We use a transition Trans which is fed with updated final counter value. This transition is fired only when the no. of steps is greater than the no. of steps taken so that no token value is left behind at any place. Here we use the control flow concept. We only take the counter value when it has reached to the end of execution leading to no leaving behind of tokens in any of the places in the model. The advantage of using such control flow mechanism is: The place *CountNei* acts as a store and the value stored in this place can be used subsequently in the DTRAP validation.

5.4 Transmission Range Adjustment

The output of the Trans is fed to a place, new count which is the node parameter as specified earlier. This dynamic counter keeps the updated value of the count of number of nodes. This updated c is passed to two substitution transitions:

1)inc
2)dcr

Here we use the hierarchy concept of CPN tools. *inc* and *dcr* are substitution transitions and when fired, they invoke two subpages , *inc #* and *dcr #* respectively.

Subpages are used to replace the substitution transition *inc* in order to increment the transmission range whenever the number of neighbours falls short of the threshold value. The increment is done in steps of 5. The updated transmission range is stored in a place, *newrng*. Subpages are used to replace the substitution transition dcr in order to decrement the transmission range whenever the number of neighbours falls short the threshold value. The increment is done in steps of 5. The updated transmission range is stored in a place, *newrng*.

```
fun ChkThreshold(c:INT)=if(c>=2)then tr-5 else tr+5;
```

The steps to construct a hierarchical model are:-

1) Move a transition or group to a new submodule.
2) Replace a substitution transition with its submodule.
3) Assign a submodule to a substitution transition.
4) Assign a port to a socket.
5) Set port type to input.
6) Set port type to output.
7) Set port type to input/output.
8) Assign a place to a fusion set. [4]

For example; if we take node 3: the no. of neighbours=3. Initial transmission range was 31. So now since 3>2 (as used in the simulation, the threshold no. of nodes are 2), hence new range is 31-5= 26.

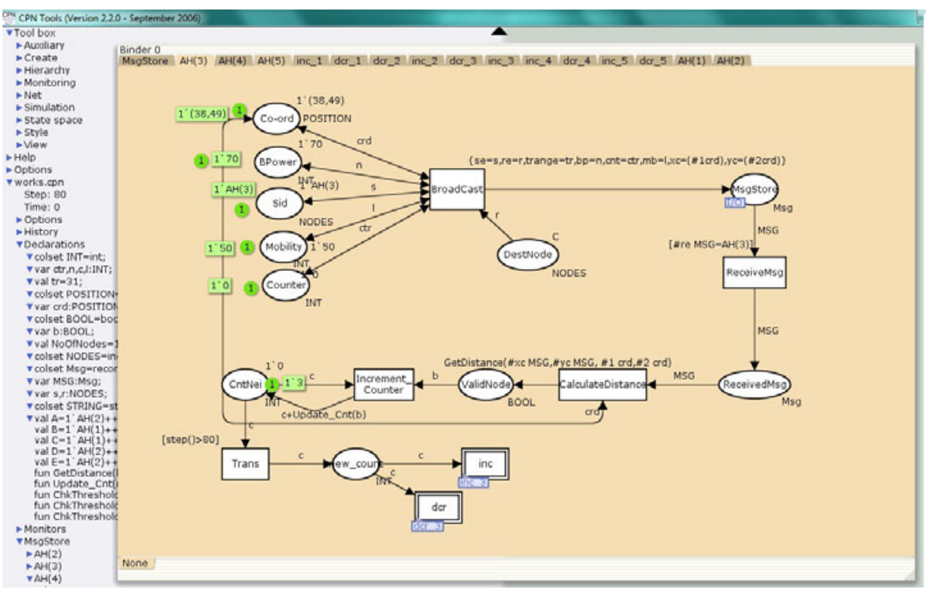

Fig. 3. Counter value of AH(3) after simulation is run

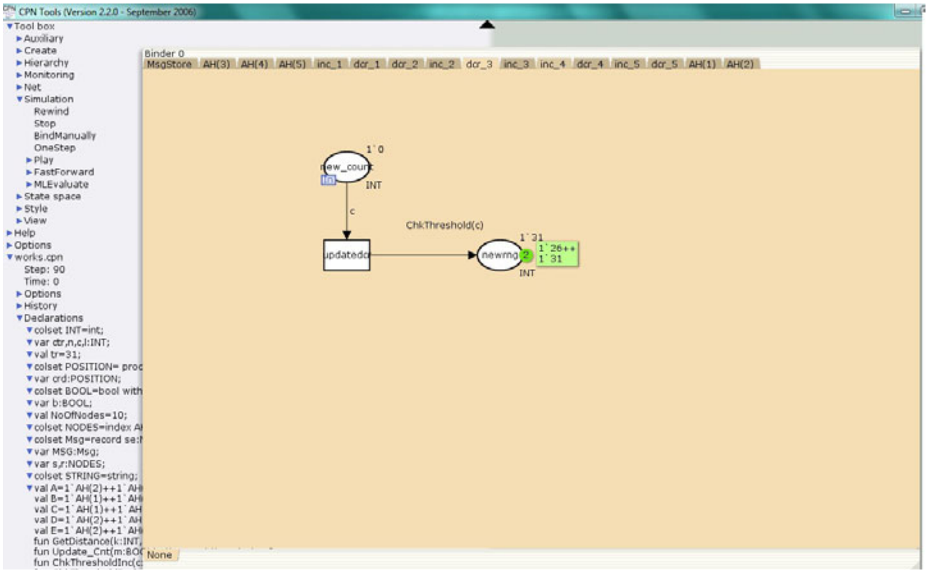

Fig. 4. Transmission range of AH(3) after simulation is run

The Dynamic Transmission Range Adjustment protocol is implemented here using five nodes. The situation of incrementing the transmission range has been successfully achieved. The next task towards the validation of the Dynamic Transmission Range Adjustment protocol is by validation of the decrementing the transmission range for the same, using Coloured Petri Net Tool.

6 Conclusion

Our work is based on ad hoc network which works on static nodes. It can be further extended to mobile nodes. Validation of this protocol ensures that the self adjusting transmission range protocol can be implemented in ad hoc network for bringing optimality in the resource utilization of nodes. It has got variety of application in the field of sensor network as well and also in emergency situations like natural disasters or military conflicts.

References

[1] Basagni, S.: Mobile Ad Hoc Networking. John Willey and Sons Publication (2004)
[2] Hu, Y.-C., Johnson, D.B.: Securing Quality-of-Service Route Discovery in On- Demand Routing for Ad Hoc Networks (1999)
[3] Paul, K., Bandopadhyay, S.: Self Adjusting Transmission Range Control of Mobile Hosts in Ad Hoc Wireless Networks for Stable Communication. In: Banerjee, P., Prasanna, V.K., Sinha, B.P. (eds.) HiPC 1999. LNCS, vol. 1745. Springer, Heidelberg (1999)

[4] Jensen, K., Kristensen, L.M., Wells, L.: Coloured Petrinets and CPN Tools for Modelling and Validation of Concurrent Systems. Int. J. Softw. Tools Technol. Transfer 9, 213–254 (2007); University of Aarhus (1996)
[5] Kristensen, L.M., Christensen, S., Jensen, K.: The Practitioners Guide to Coloured Petri nets. Int. J. STTT 2, 98–132 (1998); University of Aarhus (1998)
[6] Tan, H.X., Seah, W.K.G.: Dynamic Topology Control to Reduce Interference in MANETs, Department of Computer Science, School of Computing, National University of Singapore. In: Proc. of 2nd International Conference on Mobile Computing and Ubiquitous, 2005 Networking (2005)