IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 14, Issue 1 (Sep. - Oct. 2013), PP 33-37
www.iosrjournals.org

Mobile Networking and Mobile Ad Hoc Routing Protocol Modeling

[1]Simanta Sarma, [2]Binita Devi,
([1]HOD & Asstt. Professor, S.B.M.S. College, Sualkuchi, Assam, India)
([2]Principal, Surojit Academy, Guwahati, Assam, India)

Abstract: *In this Research paper we describe mobile ad hoc networking and routing protocols modeling. Moreover we discuss system model and ad hoc wireless networks about the routing protocols. We decisions of report Solutions for Routing Protocols in Ad Hoc Networks. This paper we discuss verification of the Ad Hoc Routing. We discuss for another process of copying with the state explosion in Ad Hoc Networks and checking of the mobile ad hoc network. Finally we survey the current checking of the modeling solutions for the mobile ad hoc network.*

Keyword: *Networking, Modeling, Protocol, node, Verification, Routing, UPAAL, BDD, SPIN.*

I. Introduction

The representation of mobility is a key issue in mobile networks simulation. The literature mentions loads of mobility models which cover various aspect of human mobility. If it were possible to combine them, the community would be provided with a fairly complete representation of how people move. However in practice, only one of them is of common use: the random waypoint mobility model. Researchers propose more and more sophisticated mobility models, using for example real traces or realistic radio waves propagation models. A mobility model is a set of rules used to generate trajectories for mobile entities. Mobility models are used in network simulations to generate network topology changes due to node movement.

II. Modeling The Protocol And Network:

In order to perform verification of an ad hoc routing protocol, we first need to construct a formal model. In addition to modeling the protocol instance running on each node, we must specify the way in which nodes are connected. There is a question of how abstract the model should be made, the tradeoff being between verification complexity and relevance of the results. In this work we use a rather rough model of layers above and below the network layer, where routing functionality resides. Upper layers are seen as generators and destinations of Internet Protocol (IP) packets destined for a certain network address. For the lower layers (link and physical) we specify if nodes are connected or not. No intermediate state is possible. Connected nodes can send packets to each other without loss or corruption during the time of connectivity. In the UPPAAL verifications, where timed automata are used for the modeling, we additionally include a nondeterministic packet delivery delay. Regarding the modeling of network configuration, we consider the following variants.

- Explicit modeling of topology and transitions. This means that the connectivity between nodes and changes in this connectivity are explicitly expressed in the model. This is the strategy we use initially.
- Parameterization, for example on the number of network nodes. In our case we parameterize on network diameter and number of link breaks.
- Universal quantification of the parameters. This provides a general network model which encompasses all numbers of nodes and all possible topology transitions. This is the topic of our future work.

III. System Model:

For designing cellular system, the most important parameter is the signal to interference ratio. In our system for standard voice quality, we are assuming signal to interference ratio to be 18 db
From equation

$$\frac{S}{I} = \frac{1}{\sum_{k=1}^{6} (q)^{-\gamma}}$$

$$q = \left\{ 6 \left[\frac{S}{I} \right] \right\}^{\frac{1}{\gamma}}$$

Where q= reuse ratio

γ = propagation loss constant from above equation for our assumed signal to interference ratio we get reuse ratio q= 4.41
Now by using formula

$$q = \sqrt{3K}$$

Where k= cluster size i.e. number of cells in a cluster
From above equation we get k=7
In our system coverage area considered is 25 sq.km
So area per cell=3.57 sq.km
So by using formula for area of cell we got radius of each cell to be R=1.17 km

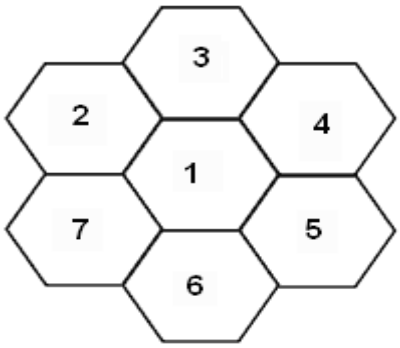

Figure 1: Cellular structure for assumed system

Above figure shows the cell structure considered. It is a seven cell cluster. In fixed channel allocation scheme every cell can access the channels given to it only. But in borrowing channel allocation scheme cell uses its own channels and then looks for channels from neighboring cells. In our system we are using borrowing from richest neighbor scheme. In this scheme, every cell will borrow a channel from its neighboring cell having most number of free channels. In our system we have considered 200 users in each cell and every user makes a call attempt thrice in one hour. Hence we get number of calls per hour per cell i.e. Q to be 600. Assuming call holding time T to be 1.76 minutes. So the traffic is A = 17.6 Erlang. Assuming blocking probability to be 0.02 , using the Erlang-B table , for given load we got number of channels per cell N=25.

IV. Analytical Model :

Resource planning:

1. Traffic model used: Priority scheme. In this scheme priority is given to handoff requests by assigning some channels exclusively for handoff calls among the total number of channels in the cell.
2. Every cell is given N channels. Those channels act as primary channels for that cell and secondary channels for neighboring cells.
3. In fixed channel assignment, cell will use its free channels only.
4. In borrowing channel assignment, cell will first use all its primary channels and if it requires more channels, it will access the secondary channels.
5. Primary channels of each cell are divided in two parts:

a) Handoff call channel (Nh)
b) New call channel (Nc)

6. Nh channels are reserved exclusively for handoff calls only. So if N are the total channels available for the cell, newly arrived call can access only (N-Nh) channels.

V. Initial Verification Approach :

Our first approach is to model each protocol instance as well as the connectivity between nodes explicitly. Topology changes are also included in their exact form in the model, for example as “nodes 0 and 2 are initially connected, but at any time this connection can break.” We make sure that the dedicated sender and receiver nodes are at all times connected by some path. Lower network layers are abstracted away from by providing a channel on which packets are delivered provided there is connectivity. For simplicity, intermittent transmission errors at the link/physical layer are treated as link breakages in our model. The maximum topology that we are able to verify consists of six nodes where one can go down at any time. We are, however, able to check livens properties on the form “route setup will eventually succeed,” etc. at the push of a button. Furthermore, we extract bounds on route setup and initial IP packet delivery times.

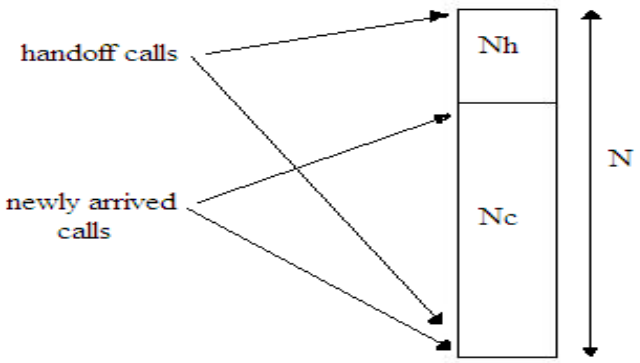

Figure 2: System model

Even though the ad hoc horizon limits the usable size of an ad hoc network, we would like to extend our verification to handle networks larger than a few nodes and, more importantly, for general types of node transitions.

VI. Coping With The State Explosion :

6.1 Symbolic Representation

To evaluate the usefulness of symbolic state space representation methods we started to model LUNAR in the NuSMV tool. The model was explicit in the sense that we adopted a network centered view with all the messages exchanged between nodes modeled using equivalent data types. However, explicit modeling turned out to be highly suboptimal due to the conversion of scalars and arrays to boolean encoding. This led us into thinking of further abstractions that could be implemented in a suitable NuSMV model. Once we had developed the new modeling strategy, accounted for below, we were no longer as constrained by state space storage. We then chose to use the UPPAAL tool for our experiments instead because it enabled us to include timing in our model in a straightforward way. The purpose of extracting time ranges was then as an extra check to validate that we had not constructed an erroneous model. As mentioned above, the UPPAAL tool also uses symbolic representation for the time ranges. In our SPIN verification we exploited a form of symbolic representation as well by using COLLAPSE compression [22]. It did not, however, in this case enable us to verify larger scenarios.

Table 1: SPIN partial order reduction gain

Without partial order reduction			Using partial order reduction		Reduction gain	
Scene.	States gen.	Trans.	States gen.	Trans.	States	Trans.
(a)	9653	25049	5715	12105	41%	52%
(b)	539601	1.70e+06	269886	731118	50%	57%
(c)	106621	302862	53614	128831	50%	57%
(e)	2.78e+06	1.03e+07	1.41e+06	4.59e+06	49%	55%

6.2 Partial Order Reduction

We exploited Partial Order Reduction in our SPIN verification. Table 1 shows a comparison of the SPIN verifications of performed with partial order reduction off and on respectively. We have only included the scenarios where exhaustive verification could be performed with partial order reduction turned on. The table shows that the state space needed to search in exhaustive verification decreases with 41-50% for these scenarios. The gain from using partial order reduction seems to increase with scenario complexity although we cannot be convinced by these few measurements. Partial order reduction has not been implemented in the currently available UPPAAL versions [1].

6.3 Abstraction Techniques

Manual abstraction can be applied to a general class of protocols. The flooding operation was in that paper explicitly included in the routing protocol model. The reason is due to the duplicate packet suppression which makes sure that the destination and all other nodes only take action on the first request they receive. By traversing the broadcast delivery path backwards from the destination node to the source we identify what we refer to as the unique PLBD path which is the same as the fastest path. The effect is that we can move the PLBD

operation out of the model and assume it to be a primitive. In other words, it may be simpler to think of the PLBD primitive as a network layer that provides unique copies of each received PLBD and only performs one resend. This does not mean that we are not handling packet collisions. They are treated as before, namely as link breakages. Still, however, there is a tradeoff between network size and number of link breaks. The protocol cannot be expected to guarantee successful route discovery if more link breaks occur, since all retries may then be lost. Using a modified protocol model, we would like to extend the analysis to an arbitrary number of nodes and link breaks. One option is to construct a fair transition system model of each protocol instance and use a very general network model in which packets are collected in a pool, an unbounded set of messages. All nodes can take action based on which messages are in the pool. This is a very general topology model in which packets can be repeated infinitely many times, arrive out of order, and in which packets may arrive at an arbitrary time to their destination. It has been used by Das and Dill [16] in their verification of the AODV routing protocol, later followed by Lahiri and Bryant [31, 30]. With this network model it is not possible to prove eventuality properties on the form "a route will eventually be set up." Invariant properties are instead used, in this case a formulation of loop freedom. Xiong et al. [53, 52] have modeled AODV using Petri nets. A topology approximation mechanism describes dynamic topology changes. They report on a looping situation found during a state space search of a general ten node topology. Their broadcast model works by sending out uncast messages to an average number of nodes. This number is based on the average degree and the total number of nodes in the graph. The resulting PLBD implementation is less abstract than ours and models redundant packet transfers between nodes not on the fastest path between the sender and receiver. In contrast to our approach link failure effects are not included in their model as they assume regular uncast transmissions to be globally receivable regardless of topology.

VII. Applying The Method:

7.1. Model Checking

There are two main advantages of model checking in comparison to deductive methods. The first one is that once the system model has been constructed and the verification properties devised, the process is completely automatic and outputs a "yes" or "no" answer. The other advantage is the possibility to generate error traces in case a property is not fulfilled by the system. This makes it possible for the user to modify the model accordingly. The main disadvantage of model checking is its limitation to finite state systems. It can, however, be used in hybrid infinite state verification approaches where model checking is, for example, a component in a CEGAR (Counter-Example Guided Abstraction Refinement) loop [12]. Furthermore, model checking of symbolically represented systems can be regarded as infinite state since the original system may contain an unlimited element (such as continuous time). Using model checking, one can check safety as well as livens properties. Model checking algorithms work by exploring the state space whereby the search stops at the first violation or when the complete execution tree has been examined. Methods can be divided into explicit state and symbolic model checking depending on if the individual states or groups (sets) of states are used to represent the state space.

7.2. Deductive Verification

In deductive verification the goal is to prove that a conclusion, the property to be verified, can be drawn from a given set of premises, the system description. This was previously a tedious manual process which has been speeded up with the emergence of semi-automatic tools, so called theorem proves. One advantage of this method is that it can be used to prove properties of infinite state systems, for example a protocol running in a network with an unbounded number of nodes. An invariant is an assertion that is true in all states of a system. A safety property, expressed as an invariant, can be proven using mathematical induction. First it needs to be proven that the initial system configuration implies the assertion. In the inductive step it is then checked whether all state transitions preserve the property, that is, if the assertion holds before the transition it will also hold after it. Hence, the verification does not require an explicit state space search. This avoids the state explosion problem at the cost of a more cumbersome proof process. The manual method was used by Ogier [40] to make a proof of correctness for the TBRPF [39] protocol. For the discovery module he further presents a proof that the neighbor information exchanged is sufficient for the functionality of the protocol.

7.3. The State Explosion Problem and Remedies

The state explosion problem in model checking refers to the situation in which the state space storage overhead grows exponentially with the size of the model. This problem occurs because of the large number of possible interleaving between processes in a reactive concurrent system. Verification may thereby fail simply because the available amount of computer memory is limited. There have been a number of suggestions for coping with the state explosion, that is, to make verification feasible for realistically sized systems. We list the major remedies below following the description by Clarke et al. [9].

7.4. Symbolic representation.

Symbolic representation refers to the use of compact data structures for representing state space. For example, by encoding the transition relations of a Kripke structure as a Binary Decision Diagram (BDD) it is possible to save storage by exploiting the often inherent regularity of a hardware or software system. Constraint system representation of continuous parameters such as clock ranges, which is done in UPPAAL, is another example of a symbolic representation. In that case it would not even be possible to store all time points explicitly regardless of the amount of available memory.

7.5. Partial order reduction.

Partial order reduction [24] is an optimization, for example implemented in the SPIN tool. If a group of concurrently executing processes do not exchange any data throughout their lifetime, then it does not make a difference for the end result if they are run one after the other or in parallel. This makes verification simpler since the processes can be verified in isolation. However, once processes cooperate, for example by message passing, which is certainly the case for protocol implementations, then the possible interleaving of operation have to be taken into account when verifying the system. Partial order reduction is a way of disregarding process interleaving that produce the same global state as some other interleaving. Note that the verification property also needs to be taken into account since it might introduce additional data dependencies between processes. Keeping as much as possible local to each modeled process can thus promote partial order reduction.

VIII. Conclusion & Future Work:

It is considered that the Routing Protocol is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on. The main aim of different routing architectures is to select a path (route) to destination with minimum cost and error. Also, the route should be reliable and short so that the messages can send with minimum cost.

We have examined how to prove general properties of ad hoc routing protocols automatically. A requirement has been to not demand anything from the protocol verifier but the most basic knowledge about formal methods. This work is to our knowledge the first to study a range of topologies in order to determine where the limit actually is when performing model checking on an ad hoc routing protocol.

In our future work we intend to apply this method to be able to provide a general proof of correctness in an automatic way. We would like to improve the method by attempting different optimization strategies to construct a faster tool. The tool will be developed with the additional goal of being user friendly. We completely introduced computer network and mobile ad hoc with routing protocols validation process in my research work.

References

[1]. Arup Acharya, Archan Misra and Sorav Bansal. .A Label-switching Packet Forwarding architecture for Multi-hop Wireless LANs.
[2]. Proc WoWMoM'02, Sep 2002, Atlanta, USA Lundgren, H.: Implementation and Real-world Evaluation of Routing Protocols for Wireless Ad hoc Networks. Licentiate thesis, Uppsala University (2002).
[3]. IETF MANET Working Group: MANET charter. http://www.ietf.org/html. charters/manet-charter.html (2004).
[4]. Perkins, C., Belding-Royer, E., Das, S.: Request for Comments: Ad hoc on-demand distance vector (AODV) routing. http://www.ietf.org/rfc/rfc3561.txt (2003).
[5]. Johnson, D.B., Maltz, D.A., Hu, Y.C.: Internet draft: The dynamic source routing protocol for mobile ad hoc networks (DSR). http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-09.txt (2003).
[6]. Xiong, C., Murata, T., Tsai, J.: Modelling and simulation of routing protocol for mobile ad hoc networks using coloured Petri nets. In: Proc. Workshop on Formal Methods Applied to Defence Systems in Formal Methods in Software Engineering and Defence Systems. (2002)
[7]. Tian-Tsair SutS, Po-Chiun Huangt, and Chung-Ju Changt,—A channel borrowing protection scheme for handoffs in a cellular based pcs systeml , 0-7803-2955-4/95/ IEEE
[8]. Holzmann, G.: The Spin Model Checker, Primer and Reference Manual. Addison-Wesley, Reading, Massachusetts (2003)