International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 17 (2015) pp 37992-37996

An Addressing Mechanism for Network Partitioning and Merging in Wireless Ad hoc Networks

N. Ramakrishnaiah
Assistant Professor, Department of Computer Science and Engineering, University College of Engg., JNTUK, Kakinada, E.G. District, Andhra Pradesh, INDIA. nrkrishna27@gmail.com

P. Chenna Reddy
Professor, Department of Computer Science and Engineering, JNTUA College of Engineering, Pulivendula, Kadapa District, Andhra Pradesh, INDIA. pcreddy1@rediffmail.com

Abstract
A mobile ad hoc network (MANET) is an infrastructure less and multi hop wireless network. Because of the mobility of the nodes, one or more new nodes join into the network as well as leave the network at any time. Every node should have a unique address to participate in a communication. Address assignment in ad hoc networks is a critical problem because of its dynamic topology. Any addressing protocol should cope with network dynamics of ad hoc networks. In this paper, we propose an approach for identifying network merge and assigning addresses without conflicts for newly joining network nodes and also reclaim the addresses of leaving nodes for future use. The simulation results shows that our protocol efficiently handles network partitioning and merging with less delay and communication overhead when compare to other protocols.

Keywords: Mobile ad hoc network, Address autoconfiguration, Network partitioning, Network merging.

Introduction

A MANET is a self-configurable and self-organizing network without infrastructure and communication is achieved through multi hop relays. A mobile ad hoc network requires an address autoconfiguration protocol to assign addresses to its mobile nodes [1]. Conventional networks use Dynamic Host Configuration Protocol (DHCP) [2] or IPv6 stateless address autoconfiguration [3] for address assignment. These protocols are not suitable for ad hoc networks because of its infrastructure less and node mobility behavior.

Due to mobility of nodes, different networks can overlap and address conflicts may occur and also one or a group of nodes leave the network. An address autoconfiguration protocol for ad hoc networks can assign unique addresses to mobile nodes with less overhead & delay and also provide a mechanism for handling network partitioning and mergers.

We assume variable length addressing scheme and the network is created as a tree and its nodes are categorized as Normal nodes, Coordinator nodes and Header node. Beginning with some initial length, addresses are represented in binary format and assign from 0 to its range. If its address range is exhausted, the address length will be incremented to accommodate new nodes. Normal nodes simply act as relaying nodes and Coordinator nodes are responsible for assigning addresses to new nodes and maintain address assignment record with Addr_Length and High_Address fields and the list of other coordinator nodes. The first Coordinator node in the network is called as header node and it is responsible for assigning network identifier and incrementing address length when address space is exhausted and also maintains a list of leak addresses.

In our scheme, we propose a mechanism for identifying network merging and assign new addresses for merger nodes and also reclaim the addresses of leaving nodes when there is a need for new addresses. We generate new network id whenever we increment the address length, so that the probability of occurrence of same network id is low and it assigns the addresses efficiently with less delay and communication overhead when a network merging takes place irrespective of its size. It also reclaims the addresses of leaving nodes when there is a need for new addresses, so that it avoids periodic flooding.

The rest of this paper is organized as follows: Section 2 discusses the various addressing schemes available in literature and their merits and demerits. Section 3 gives our proposed addressing process for network partitioning and merging. Section 4 presents simulation results and Section 5 concludes the paper.

Related Work

Addressing protocols for MANETs are classified into neighbor based schemes, decentralized schemes and centralized schemes [4].

In neighbor based schemes, a new node is configured by its adjacent node and address conflicts are resolved by local communication. Each node maintains a disjoint address space for assigning addresses and these schemes do not suffer from centralized control or flooding.

Sonia et al. [5] suggested an addressing scheme, in which address assignment is the responsibility of all nodes in the network and nodes are classified as configuration agent and simple node. Configuration agent assigns addresses to new nodes from its mutually exclusive address space. Simple nodes act as intermediate nodes between configuration agents and new nodes. The communication overhead and latency associated with address allocation is less because the control messages are handled within two-hop distance. If the address pool of configuration agent exhausts, it increases configuration overhead and latency to recover the leak addresses or borrow the address space from other agents.

Longjiang et al. [6] suggested a domain-based auto configuration scheme for large scale mobile ad hoc networks. It groups the nodes into domains and the nodes may roam to different locations in the same domain. The address of a node has two parts: interID, denotes domain id and intralID, denotes node id in the domain. The interID is generated by using a random generator function, so that the uniqueness of interID can't be guaranteed. This scheme uses Passive Duplicate Address Detection (PDAD) mechanism to detect address conflicts. The use of DAD scheme increases the communication overhead and delay of the proposed protocol.

Jang-Ping et al. [7] suggested a distributed address assignment scheme, in which the network is constructed as a virtual tree. The nodes of the network are classified as coordinator node and common node. A coordinator node is responsible for address assignment and maintains a disjoint address space to assign for new nodes. The coordinator node changes its role as common node, if its address space is exhausted. If the address pool at all coordinator nodes exhausts then there is no scope to add new nodes, and the protocol fails to accommodate new nodes.

Wang et al. [8] suggested a tree-based addressing scheme for a MANET (T-BAAP). In this scheme, the network is established as an association relationship between adjacent nodes. An S value is assumed based on size of network, i.e. the highest number of child nodes a father node can configure. As we can't guess the exact size of network at the starting and if the number of child nodes at a father node is greater than S value, then there is no scope to add new nodes.

In decentralized schemes, one of the addressing agents assigns an address to new node and it ensures uniqueness by consulting other agents in the network. In query-based Duplicate Address Detection (DAD) [1], weak DAD [9], and passive DAD [10], a new node selects itself an address and its uniqueness is verified by running DAD procedure. All the protocols using DAD procedure fit in this category and suffer from network-wide flooding.

Sanket et al. [11] proposed an addressing scheme for mobile ad hoc networks, called as MANETconf. Every node maintains a distributed common table to assign the addresses to new nodes. A new node sends a request for an address and hears the responses from its neighbors and chooses an initiator from the responses. The initiator selects an address by checking its allocation tables and confirms the uniqueness from all the nodes and assigns this address to a new node. The address assignment cost is relatively high to maintain the consistency of address allocation tables.

Jeff Boleng [12] suggested an addressing scheme in which addresses are represented in binary format and address allocation is done by maintaining a distributed common record at each node. The cost of the protocol increases along with its network size to maintain the consistency of the record available at all nodes. In this scheme, address reclamation procedure is not discussed.

In M. R. Thoppian et al. [13], every configured node has a set of unassigned addresses and offers half of its address space to new nodes. It uses periodic flooding mechanism for recovery of leak addresses, so that the cost of the protocol is high.

In centralized schemes, a single entity, called as a server in the network is responsible for assigning addresses. This central server maintains all the information related to address assignment and if it fails then the total system fails. These schemes guarantees address uniqueness, but results in high overhead in maintaining the server.

In Stephen et al. [14], a single entity called as leader is in charge for assigning addresses to new nodes in MANET. A new node sends an address request and a neighbor node within the transmission range of new node forwards the request to leader and gets a free address and allocates it to the new node. The leader node is responsible for entire addressing mechanism, and if it fails then the total system collapses.

M. F. Al-Mistarihi et al. [15] suggested a tree based dynamic address auto-configuration protocol (T-DAAP). The nodes of the network are categorized as a normal node, a leader node, and a root node. The root node maintains information about all leader nodes and is responsible for address recovery. The leader nodes have disjoint address space and are responsible for assigning addresses to new nodes. A normal node acts as an intermediate node between leader node and a new node. If the address pool exhausts at any leader node or the root node fails, then getting of addresses from other leaders or election of new root increases the cost and delay of this scheme.

The above discussion tells that the addressing schemes in MANETs need further research. Our scheme provides a simple and efficient mechanism to reclaim used addresses when network partitioning takes place and also to assign addresses when network merging takes place. The cost and delay of our scheme are very less when compared to other schemes and also scalable.

Proposed System

This section describes how the addresses are recovered when a node leaves from a MANET and also how the addresses are assigned when network merging takes place.

A. Network Partitioning.

i. Node leaving with prior notice:

The nodes in the network can leave either unexpectedly or gracefully. A normal node which wants to leave the network gracefully sends a *Leave* message with its address and type of node before leaving the MANET. The coordinator nodes forward the same message to header node. The header node adds the leaving node address to the list of leak addresses. If the header node receives the multiple copies of same *Leave* message, it discards the subsequent copies except the first one. If a coordinator node wants to leave the network gracefully, it sends a *Leave* message with its address and type of node to all the coordinator nodes in the C-Node list before leaving the MANET. The C-Nodes upon receiving *Leave* message, deletes its address from C-Node list and also adds this address to Leak addresses list.

If the header node (first coordinator node) wants to leave the network gracefully, it sends a *Leave* message with its address, type of node and Leak addresses list to the second coordinator node in the C-Node list before leaving the MANET. The C-Node upon receiving this *Leave* message declares itself as next header, *Leave* message address is deleted from C-Node list and added to leak addresses list and sends header announcement message to all coordinator nodes in the C-

Node list. The coordinator nodes on receiving new header announcement delete the previous header address from its C-Node list.

ii. Node leaving without notice:

The addresses of the nodes that leave the network unexpectedly are recovered when there is a need for an address. A new node sends an address request to the chosen coordinator node. The coordinator node verifies address allocation record and if there is no free address then it sends a request to header node for leak address. The header verifies leak address list and if it is null, it initiates address reclamation procedure. The header starts a timer and floods *Header_Alive* message to all nodes to know the list of active nodes in the network. In response to this message all normal nodes sends an acknowledgement to C-Node and in turn the C-Node forwards list of addresses including itself to header node. After timer expires, header sorts out all addresses in the range and finds the address leaks and adds them to the list of leak addresses.

If the header node leaves the network abruptly, the next node in C-Node list becomes header and leak addresses are recovered in subsequent address reclamation process.

B. Network Merging.

We use the network identifier (NetID) to detect network merging. The header node (first coordinator node) defines the NetID. It consists of four fields: Header IP address, Header's MAC address, Random number and Timestamp. The probability of occurrence of same NetID is negligible with this four tuple value. Whenever the address length is incremented or header changed, the NetID is also changed.

Whenever a node (say, X) receives a packet with different NetID from its neighboring node (say, Y), node X notices network merging. Nodes X and Y acts as mediators between two merging networks. Node X sends a unicast message *MergeDetect* to its header by extracting header address from NetID. In response to *MergeDetect* message, header node sends it's Address Assignment Record and Coordinator Nodes list to X, in turn X forwards these records through node Y to other network header. Node Y of other network also does the same job, i.e. Node Y sends a unicast message *MergeDetect* to its header by extracting header address from NetID and in response to *MergeDetect* message, header node sends its Address Assignment Record and Coordinator Nodes list to Y; in turn Y forwards these records through node X to other network header.

The header, whose High_Addr is greater than High_Addr of other network header or if the High_Addr of both the networks are equal then the header which has less IP address will be the new header for whole network and does the following: It increments the Addr_ Length by one, generates new NetID, updates the C-Node list by adding other network list at the end of the list. It floods an update message to all the nodes in its networks to update new NetID and increments address length by adding 1 to the left of the address and also send the updated C-Node list to all coordinator nodes. It unicasts the new NetID to the other header through merge mediator nodes.

The header, whose High_Addr is less than High_Addr of other network header or if the High_Addr of both the networks are equal then the header which has high IP address will do the following: find the new address length as other network Addr_Len + 1, update C-Node list by adding its list to the end of other network list and new header is other network header. After receiving new NetID from other header, it floods an update message to update new NetID and address by adding (new address length – Addr_Len) number of 0s to the left of the address. It also sends new header declaration by sending updated list of C-Nodes to all coordinator nodes and discards its leak addresses list and acts as a coordinator node.

Simulation

The performance our merging scheme is evaluated by using Network Simulator -2 [16]. Our results are compared with the two tree based protocols T-BAAP [8] and T-DAAP [15]. The simulation parameters are shown in Table. 1. The nodes of each network are deployed randomly across simulation area at a time with enough distance and addressing is done separately. Due to mobility of nodes, if the nodes of one network come close to the other network then merging process is performed to make them a single network. The simulation snapshot of our proposed scheme running for the network of total size 50 is shown in Figure 1. The cost and delay of the merging process are calculated to analyze the performance of our scheme.

TABLE.1. Simulation Parameters

Parameter Name	Parameter Value
Communication range	100 m
Simulation area	400 m x 400 m
Pause time	10 s
Routing protocol	AODV
Number of nodes	50 – 100
Maximum speed	10 m/s
MAC protocol	IEEE 802.11
Mobility model	Random Waypoint model

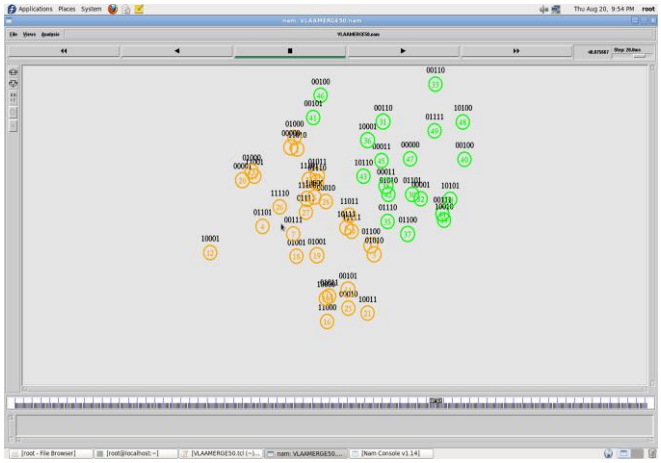

Figure 1: Simulation snapshot executing for network of total size 50 nodes.

A. Merging Cost:

The merging cost is calculated as the total number of control packets used from initiation to till the end of merging process. To find the merging cost of the network of total size, for example 50, we execute all the combinations of networks with sizes 40 & 10, 30 & 20 and 25 & 25 and took the average of these three combinations. The same mechanism is used to find the merging cost of other network sizes. The cost of network merging is shown in Figure 2.

In T-BAAP [8], each node assigns addresses upto S child nodes. The proxy node, which detects merging, has less number of child nodes than S, then it continues merging process, otherwise the merging process is assigned to its father node. The cost of merging process is proportional to the number of nodes in the network, so that the cost increases with the size of the network.

In T-DAAP [15], if merging is detected, then it collects used addresses of other network and finds the address conflicts by comparing with its own used addresses. The conflicted addresses are reassigned from free address pool. If there is less number of free addresses than conflict addresses then it takes more time to resolve conflicts. This merging process increases the cost exponentially with the number of nodes in the network.

In our scheme, the merging process involves finding new address length, updating coordinator nodes list and adjusting address length to new length and does not depend on the size of the network, so that the merging cost is very low as shown in Figure 2.

B. Merging Delay.

The merging delay is calculated as the total time used from initiation to till the end of merging process. To find the merging delay of the network of total size, for example 50, we execute all the combinations of networks with sizes 40 & 10, 30 & 20 and 25 & 25 and took the average of these three combinations. The same mechanism is used to find the merging delay of other network sizes. The delay of network merging is shown in Figure 3.

In T-BAAP [8] and T- DAAP [15] the merging process depends on the size of the network, so that the merging delay is increased with the number of nodes in the network.

In our proposed scheme, the merging process does not depend on the size of the network, hence the merging delay is less when compared with the other schemes as shown in Figure 3.

Conclusion

This paper provides an addressing scheme for network partitioning and merging in mobile ad hoc networks. It handles network partitioning and merging efficiently without depending on the size of the network. The main advantage of this scheme is scalability. The performance of the merging process changes slightly with the size of the network and is based on the tree structure. The future work is to study the security threats and provide the solutions to them.

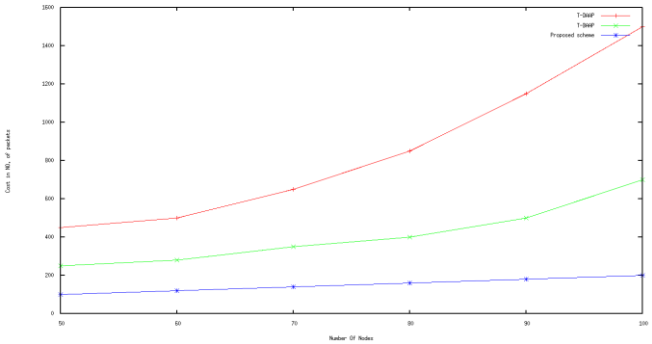

Figure 2: Merging Cost.

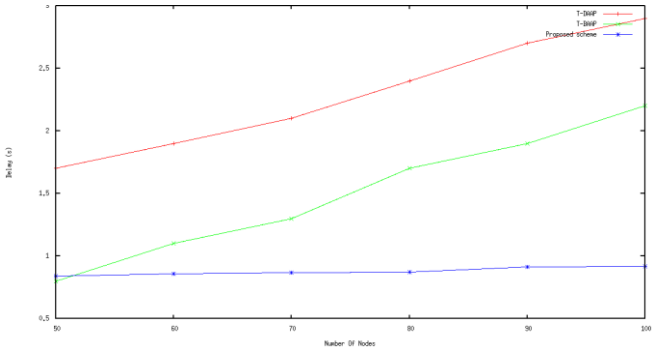

Figure 3: Merging Delay.

References

[1]. Perkins C, Malinen JT, Wakikawa R, Belding-Royer EM, Sun Y, 2001, "IP address autoconfiguration for ad hoc networks", IETF draft.

[2]. Droms R, 1997, "Dynamic Host Configuration Protocol", RFC 2131.

[3]. Thomson S, Narten T, Jinmei T, 2007, "IPv6 Stateless address autoconfiguration", RFC 4862.

[4]. Sehoon K, Jinkyu L, Ikjun Y, 2008, "Modeling and Performance analysis of address allocation schemes for mobile ad hoc networks", IEEE Transactions on Vehicular Technology, Vol.57, No.1, PP.490-501.

[5]. Sonia Mettali G, Elabidi A, Farouk K, 2010, "Distributed address auto configuration protocol for Manet networks", Telecommunication Systems, Vol.44, No.1, PP.39-48.

[6]. Longjiang L, Yunze C, Xiaoming X, 2009, "Domain-based autoconfiguration framework for large-scale MANETs", Wireless communications and Mobile computing, Vol.9, No.7, PP.938-947.

[7]. Jang-Ping S, Shin-Chih T, Li-Hsiang C, 2008, "A distributed IP address assignment scheme in ad hoc networks, Int. J. Ad Hoc and Ubiquitous Computing, Vol.3, No.1, PP.10-20.

[8]. Wang X, Qian H, 2014, "A tree-based address configuration for a MANET', Pervasive and Mobile Computing, Vol.12, PP.123-137.

[9]. Vaidya N. H, 2002, "Weak duplicate address detection in mobile ad hoc networks", In: Proc. of

ACM MobiHoc 2002, PP.206-216, Lausanne, Switzerland.

[10]. Weniger K, 2003, "Passive duplicate address detection in Mobile ad hoc Networks", In: Proc. of IEEE WCNC 2003, Vol.3, PP.1504-1509, New Orleans, LA.

[11]. Sanket N, Ravi P, 2002, "MANETconf: Configuration of Hosts in a Mobile Ad Hoc Network", In: Proc. of IEEE INFOCOM 2002, Vol.2, PP.1059-1068.

[12]. Jeff B, 2002, "Efficient network layer addressing for Mobile Ad Hoc Networks", In: Proc. of IEEE MILCOM 2002, PP.271-277, Las Vegas, NV.

[13]. Mansi R. T, Ravi P, 2006, "A distributed protocol for dynamic address assignment in Mobile Ad hoc networks", IEEE Transactions on Mobile Computing, Vol.5, No.1, PP.4-19.

[14]. Stephen T, Donal M, 2003, "Self-organizing node address management in ad-hoc networks", Personal Wireless Communications, Vol.2775, PP.476-483.

[15]. Mamoun F A, Mohammad A, Ahmad Q, 2011, "Tree based dynamic address autoconfiguration in mobile ad hoc networks", Computer Networks, Vol.55, PP.1894-1908.

[16]. Kevin F, Kannan V, "The ns Manual", www.isi.edu/nsnam/nsdocumentation.html.