IPv6 Autoconfiguration for Hierarchical MANETs with Efficient Leader Election Algorithm

Safdar Hussain Bouk and Iwao Sasase

Abstract: **To connect a mobile ad hoc network (MANET) with an IP network and to carryout communication, ad hoc network node needs to be configured with unique IP adress. Dynamic host configuration protocol (DHCP) server autoconfigure nodes in wired networks. However, this cannot be applied to ad hoc network without introducing some changes in auto configuration mechanism, due to intrinsic properties (i.e., multi-hop, dynamic, and distributed nature) of the network. In this paper, we propose a scalable autoconfiguration scheme for MANETs with hierarchical topology consisting of leader and member nodes, by considering the global Internet connectivity with minimum overhead. In our proposed scheme, a joining node selects one of the pre-configured nodes for its duplicate address detection (DAD) operation. We reduce overhead and make our scheme scalable by eliminating the broadcast of DAD messages in the network. We also propose the group leader election algorithm, which takes into account the resources, density, and position information of a node to select a new leader. Our simulation results show that our proposed scheme is effective to reduce the overhead and is scalable. Also, it is shown that the proposed scheme provides an efficient method to heal the network after partitioning and merging by enhancing the role of bordering nodes in the group.**

Index Terms: **Address autoconfiguration, duplicate address detection (DAD), leader election, mobile ad hoc network (MANET).**

I. INTRODUCTION

Mobile ad hoc network (MANET) is a spontaneous network of wireless mobile nodes with arbitrary topology. The mobile nodes in MANET function as a router and also as a host (end-points). These networks are popular because of their autonomous, self-healing, self-configuring, extremely flexible, and ease of deployment properties. In [1], MANETs are classified into two categories: Isolated MANETs and MANETs connected with external networks (e.g., hybrid network or global Internet). In the first category; every node can communicate with other nodes within the MANET. In the second category, as shown in Fig. 1, MANETs are connected to the infrastructure networks to extend coverage area of the fixed networks. Typical examples are wireless mesh networks (WMNs) [2] and networks used to extend services at hot spots and cellular network. In this category, a host located anywhere in the fixed network communicates with the node inside the MANET through

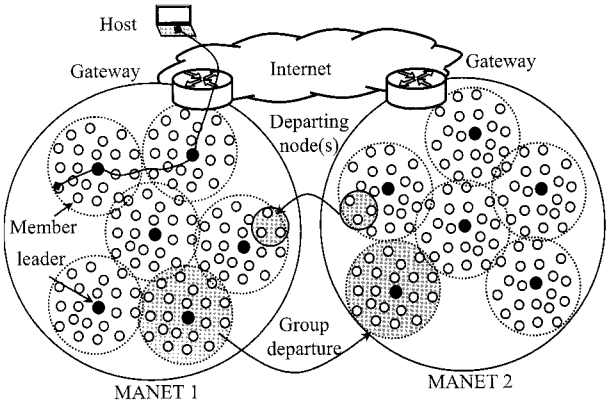


Fig. 1. Hybrid ad hoc networks.

fixed or mobile gateways. To carry out data communication in hybrid networks, a node must be assigned a valid and unique IP-address (IP-autoconfiguration) before transmitting data or routing packets. Also it is difficult to maintain uniqueness of the node identification in a highly dynamic topology network due to the frequent node movement, network partitioning and merging. In wired networks, dynamic host configuration protocol (DHCP) [3] is most commonly used to automatically configure nodes joining the network. The standard methods for IPv6 autoconfiguration are suggested in [4], but these methods can not directly be applied to MANETs due to the lack of central administration as well as highly dynamic and distributed nature of the network.

II. RELATED WORKS

In this section, the existing solutions for IP autoconfiguration in ad hoc networks are discussed in detail. These existing solutions are in Internet draft (I-D) form that are submitted to Internet engineering task force (IETF) [5] or non-ID solutions that were submitted in different journals and conference proceedings. The IETF has formed a working group in the internet area named "AUTOCONF." The main goal of autoconf working group is to standardize the autoconfiguration mechanism that considers the configuration of nodes with unique MANET scope and global scope address. Still the standardization of MANET autoconfiguration is in process and several I-Ds [8]–[15] have been submitted to the working group. Bernardos *et al.* [6] best summarizes the I-Ds submitted to the autoconf working group. Table 1 shows the main features of those I-Ds. To assign a unique IP address to a mobile node in MANET, a mobile node selects a random address or form an address from 48-bit MAC address (EUI-64) [7]. After address formation, a mobile node checks the uniqueness of the address by flooding the message in

Manuscript received June 29, 2007; approved for publication by Chung Gu Kang, Division II Editor, March 20, 2009.

This work is partly supported by Keio University 21st Century Center of Excellence Program on "Optical and Electronic Device Technology for Access Network" and Fujitsu Laboratories.

The authors are with the Department of Information and Computer Science, Keio University, 3-14-1 Hiyoshi, Kohoku-ku, Yokohama-shi, Kanagawa, 223-8522 Japan, email: bouk@sasase.ics.keio.ac.jp, sasase@ics.keio.ac.jp.

Table 1. I-Ds submitted to IETF autoconf working group.

I-D	Address formation	DAD approach	IP family	Scope	MANET type	Partition and merging support	Routing protocol dependency
Perkins *et al.* [8]	Random	Strong DAD	IPv4 & IPv6	MANET-scope	Stand-alone	No	No
Wakikawa *et al.* [9]	EUI-64	No DAD	IPv6	Global-scope	Connected	No	No
Jeong *et al.* [10]	Random	Strong & weak DAD	IPv4 & IPv6	MANET-scope	Stand-alone	Yes	No
Ruffino *et al.* [11]	EUI-64	No DAD	IPv6	Global-scope	Connected	Partial	Yes
Cha *et al.* [12]	Assigned	No DAD	IPv6	Global-scope	Connected	Partial	Yes
Clausen *et al.* [13]	Assigned	No DAD	IPv6	Global-scope	Connected	No	Yes
Jelger *et al.* [14]	EUI-64	No-DAD	IPv6	Global-scope	Connected	No	No
Weniger *et al.* [15]	Random	Strong DAD	IPv6	MANET-scope	Stand-alone	Yes	No

the network.

If any node is already configured with the same address then it will reply. In case of no reply from any node, that address is considered unique and the node configures the interface with this address. This procedure is called strong duplicate address detection (DAD) [8], [10], [15]. In [16], weak DAD was proposed. In weak DAD, a selected address is assigned to the node interface without flooding any query message into the network. In case of any conflict unique key information is used to resolve the conflict [10]. However, if two nodes select duplicate address and key, then conflict can not be detected. In [12] and [13] the address is assigned to the new joining node by the node that is already configured with valid IP address. In case of partitioning the address uniqueness can be guaranteed by the autoconfiguration protocol. However, in case of merging of two networks, address conflict may occur. Hence, the autoconfiguration protocol must mitigate this situation.

In [9] and [11]–[14], autoconfiguration of connected MANET is proposed. The partitioning and merging problem that is caused by the departure and joining of a single or a group of nodes is inevitable either in stand alone or a connected MANET due to the mobility characteristic of ad hoc nodes. However, most of the autoconfiguration solutions [9], [13], [14] for connected MANET do not support partitioning and merging and few of them [11], [12] just partially support merging situation at the cost of high control overhead. Also all of the autoconfiguration schemes for connected MANETs in [6] perform DAD for non-unique local addresses. However, each node in a connected ad hoc network must be configured with a global unique address to carry out communication with node that is located in infrastructure network and unique local address to communicate inside a MANET. To configure each node with a global unique address those schemes send a full broadcast or flood DAD messages in the ad hoc network to test uniqueness of the address [6], [13]. The flooding or full broadcast in an ad hoc network results in a very high overhead and in case of large ad hoc network, full broadcast drastically consumes the network resources. In [12] an infrastructure node (gateway) assigns IP address to nodes that can be the central point of failure.

Several non I-D ad hoc autoconfiguration solutions [17]–[20] have also been proposed. Here, we discuss the non I-D proposals that have been published and presented in several journals and conferences, respectively.

Perkins *et al.* proposed the simple solution for stateless autoconfiguration [17], in which the joining node selects a random address and performs the DAD for uniqueness of the address. Whereas, during the network partitioning and merging, uniqueness of the address is not always guaranteed and full flooding makes this solution not scalable to the large networks.

The passive autoconfiguration for mobile ad hoc networks (PACMAN) [18] uses the cross-layer routing protocol traffic information for address DAD. The distributed maintenance of common address allocation table also depends on the routing protocol traffic. New node generates an address by considering the allocated address information at the lowest probability of conflict. Passive DAD (PDAD) detects the address conflicts based on the anomalies in the routing protocols traffic. Hence, additional control traffic is avoided and in result it saves the energy and bandwidth. PACMAN can easily be implemented with any routing protocol without altering the protocol message format. This protocol only focuses on the MANET scope address and do not consider the global connectivity.

In [19], two autoconfiguration schemes have been proposed: Random address allocation (RADA) method and linear allocation (LiA) method. In RADA, new node picks two random IP addresses from the pre-configured subnet. It uses one as temporary address and other as requested IP address. It broadcasts the address query (AQ) message periodically to detect address conflicts. LiA also broadcasts the control messages to indicate the maximum IP address number used in the network. In both methods, the control packets are broadcasted, and they increase the traffic and consume the bandwidth.

Weniger and Zitterbart suggested an autoconfiguration scheme for the hierarchical MANET with group mobility [20]. In [20], MANET is divided into logical groups, where group leader maintains the scope of a group within the radius of r-hops. This scheme modifies the neighbor solicitation (NS) message [21], [22] and uses it to perform DAD for a new joining node. The new joining node broadcasts the modified NS message, containing its tentative address or member ID, within the scope of r-hops. In case of address duplication, the joining node receives a reply from the conflicting node and tries with another address, otherwise that address is considered to be unique. When DAD is performed by a leader node to check the uniqueness of its tentative address or leader ID, then the NS message is broadcasted within the MANET. Every leader node in MANET compares the new leader address to check its uniqueness. This scheme reduces the DAD overhead by limiting the broadcast of control messages within the scope, but it does not avoid the flooding of control messages when it performs DAD for leader node. In case of higher node density within MANET, the flooding might cause heavy collision and network congestion. The scheme in [20] uses the k-hop number of nodes as a criterion for electing a new leader. Note that node density may vary time

to time because of joining and departure of nodes, and it results in a frequent change of leader node.

The IETF I-Ds and non-I-D solutions for IP address autoconfiguration for ad hoc networks are briefly summarized in Table 1 and discussed in previous paragraphs. The solutions in [9], [11]–[14], [17], [19] either consider that the nodes are already configured or assigned unique IPs or the randomly assigned IPs to ad hoc nodes are unique. However, due to the dynamic nature of the ad hoc networks these solutions are not suitable for ad hoc networks. Also the solutions in [8] and [10] have significantly high overhead and are not scalable because it considers flat architecture ad hoc network and DAD messages are flooded in whole ad hoc network to perform strong DAD. On the other hand, autoconfiguration scheme in [15], which is similar to [20], consider hierarchical ad hoc networks and significantly reduces the overhead by limiting the broadcast of DAD messages within predefined scope. Our proposed scheme also provides IP address autoconfiguration solution for the hierarchical ad hoc networks. Therefore, the autoconfiguration scheme in [20] considered as a conventional scheme to compare performances with our proposed scheme.

In this paper, we propose an IPv6 address autoconfiguration scheme for hierarchical MANETs by considering the global connectivity. The main difference between our scheme and the related work is that our proposed scheme avoids broadcast of DAD messages in ad hoc network, which saves a lot of network resources. Also, our scheme provides effective method of network merging. The proposed leader election algorithm in this paper considers multiple node parameters and avoids the broadcast of the extra weight messages to up to date other nodes about the weights of each node.

In our proposed scheme, a joining node selects one of the pre-configured nodes for its DAD operation. That pre-configured node is already configured with global address to carry out global Internet communication and also it is capable to send packets over multi-hops within MANET. The pre-configured node performs DAD operations on behalf of the new joining node. We reduce overhead and make our scheme scalable by eliminating the broadcast of DAD messages in the network. Also we propose a new leader election algorithm that takes into account the most important system parameters such as system resources, density and relative position of a node within group. Our simulation results show that the proposed scheme has less overhead and is scalable. The proposed scheme also provides the efficient method to heal the network after partitioning and merging.

III. PROPOSED SCHEME

We propose an IPv6 address autoconfiguration scheme for hierarchical MANETs by considering the global connectivity. In our proposal, we modify the DAD procedure to reduce overhead and make our scheme scalable by eliminating the broadcast of DAD messages in whole or part of the network. A joining node in the proposed scheme selects one of the pre-configured nodes for its DAD, which performs DAD operations on behalf of the new joining node. Here, we consider random way point node mobility, discussed in Section IV, to check scalability and effectiveness of our proposed scheme.

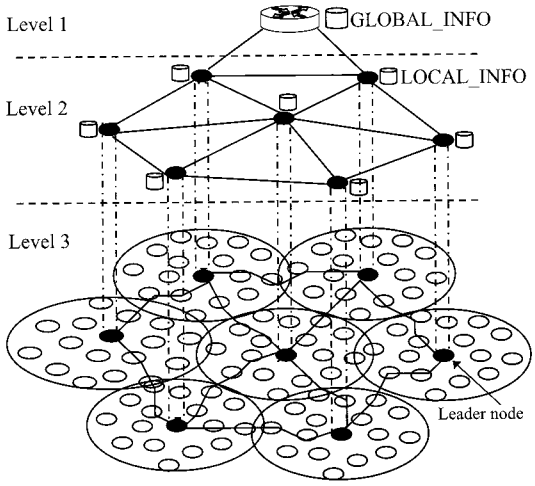


Fig. 2. Hierarchical MANET architecture.

A. Architecture and Message Format

Here, we extend the logical structure of the conventional system in [20] and propose the new message format for our proposed scheme. The network is divided into clusters or groups, where each group has agreed upon a leader. This leader maintains the cluster of r hops, which specifies the size of the cluster. By taking the advantage of the default IPv6 address architecture, we structure the MANET architecture in three level hierarchy as shown in Fig. 2. The nodes that lie at the level 1 of hierarchy are the gateway node(s), which provide connectivity to the MANET with the fixed network. Gateway node also maintains a data structure named GLOBAL_INFO that stores Leader_IDs assigned to every group leader and its validity period. The leader nodes at level 2 control the group activity and DAD traffic for member nodes. Every leader node in MANET maintains a structure, named LOCAL_INFO, which contains member_IDs assigned to the group members and their validity duration. The third level of hierarchy consists of the member nodes. Our main objective to divide MANET into different level hierarchies is to minimize the DAD overhead and efficiently handle the network partitioning and merging.

In our scheme, we logically divide the interface ID of IPv6 address into two sections, leader_ID and member_ID, where the sizes of leader_ID and member_ID are implementation dependent. Each node in our proposed scheme performs DAD for verifying the uniqueness of its logical ID. After confirming the uniqueness, a mobile node forms an interface ID by concatenating leader_ID and member_ID. For example, a leader node with leader_ID "2EAFF" forms a link local address as "FE80::2.EAFF.0.0/64" and a member node with member_ID "3E33" and above mentioned leader_ID forms its link local address as "FE80::2.EAFF.0.3E33/64." Similarly, every mobile node in our proposed scheme configures its MANET-local or global IPv6 address by attaching the respective network prefixes to the 64-bit interface ID.

Our proposed autoconfiguration scheme allows a new node to select a pre-configured member node, called coordinator node,

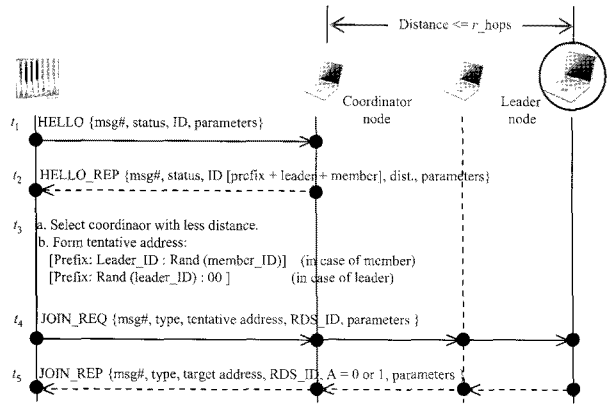


Fig. 3. Proposed DAD message sequence.

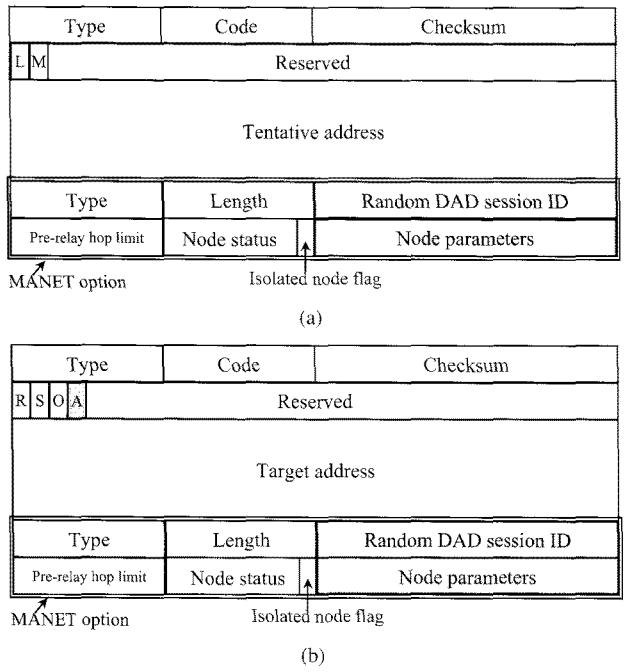


Fig. 4. Message formats with MANET option: (a) The ADR_REQ message with MANET option and (b) the ADR_REP message with MANET option.

for its autoconfiguration, because new node can not send packets on multi-hop path without a MANET scope address. The coordinator node is responsible for performing DAD on behalf of new node. For selecting a coordinator, a new joining node sends a HELLO message [23]–[25] to its 1-hop neighbors, as shown in Fig. 3 as t_1. All the replies for that HELLO message are stored in the cache for short period of time and these entries are deleted from the cache when their time expires. Here, we assume that the neighboring nodes reply for that HELLO message containing IP address, distance d-hops from the leader node and status of a node. The node status in the HELLO message indicates the sender of the message. The new node will select the coordinator by considering the distance and status information to differentiate between router, leader and member nodes' HELLO_REP messages, as sequence t_2 in Fig. 3. After selecting a coordinator, the status of new joining node becomes a requester, which requests for its IP.

In our proposed scheme we introduce address request (ADR_REQ) and address reply (ADR_REP) messages, shown in Figs. 4(a) and 4(b), respectively. The ADR_REQ is the extended neighbor solicitation message and ADR_REP is the extended neighbor advertisement (NA) message [21], [22]. The ADR_REQ with MANET option is similar to the one used by the conventional scheme [20]. Here, we introduce two new flags L and M that indicate the type of joining nodes. In ADR_REQ message if flag L is set high indicates that the DAD is performed for the new leader node and ADR_REQ must be forwarded to the gateway node. Similarly if flag M is set high denotes that the DAD is being carried out for the new joining member node and must be forwarded to the group leader. In MANET option, the random DAD session ID (RDS-ID) identifies the DAD session for the particular requester. Also we introduce a node weight field, which represent its overall weight of the new node. This weight information is a main criterion for electing a new group leader, discussed in Section IV. The MANET option is transmitted along with the ADR_REQ and ADR_REP messages to distribute information about the node within the group. The ADR_REP message is shown in Fig. 4(b), where flags R, S, and O are the default flags of the NA message [22]. In ADR_REP message we introduce a new flag named A. The flag A indicates the type of reply for the ADR_REQ message, where $A = 0$ means positive reply and $A = 1$ means negative reply for the ADR_REQ.

B. Modified DAD Procedure

In our proposed scheme, every new node joining the network performs the following modified DAD procedure for verifying the uniqueness of their respective IDs. Our proposed DAD procedure mainly differs from the conventional one in [20] in terms of message transmission. In the conventional DAD procedure, a joining member node performs duplicate address detection by broadcasting the DAD message within the scope. If the new joining node is a leader node then the conventional scheme performs duplicate address by flooding the whole network with the DAD message. On the other hand, in our proposed scheme, the DAD messages for a new member or a leader node are forwarded in a unicast manner.

Before initiating the DAD procedure, the requester forms a tentative address by combining a randomly selected member_ID, with the leader_ID and local prefix information from coordinator node's HELLO_REP. Requester also sets the tentative address, RDS-ID, node weight, and flag values in the ADR_REQ and forwards to the coordinator. The coordinator forwards the ADR_REQ in a unicast manner to the leader node. Upon receiving the ADR_REQ from the coordinator, the leader node checks the address within the LOCAL_INFO. If that address is not assigned to any other node within the group then the leader node sends ADR_REP by setting Target Address similar to tentative address, and set flag $A = 0$. Also leader node stores the address and RDS-ID in the LOCAL_INFO. The RDS-ID is stored till DAD session time (t_{DS}) and discarded when t_{DS} expires. Here, we assume that the RDS_IDs generated by the requester nodes between t_{DS} time period are random. In case of duplicate address, the ADR_REP with $A = 1$ is sent by the

leader node to the coordinator. When the coordinator receives ADD_REP, it forwards to the requester node. In case of positive reply, requester initializes its interface with the new IP address, and then, it takes part in data communication and control message forwarding within MANET. In case, if requester receives negative reply, then it generates another tentative address and repeats the same DAD procedure.

In case of a new leader node, the ADR_REQ is forwarded by the coordinator to the gateway node. When gateway node receives ADR_REQ (with $L = 1$), it matches the leader_ID within the GLOBAL_INFO repository. If address is unique, then gateway node stores leader node parameters in the repository and sends ADR_REP with $A = 0$. If that leader_ID is already assigned, then the gateway node sends negative reply for the ADD_REQ message.

C. Packet Loss, Coordinator and Leader Node Failure

After sending ADR_REQ by the member node, it initiates DAD reply timer (t_{DR}) and wait for the ADR_REP message till t_{DR} expires. If node does not receive ADR_REP and t_{DR} expired, then it sends another ADR_REQ message with same RDS_ID and other parameters up to $T_{\mathrm{max\_retry}}$ times.

$$t_{\mathrm{DR}} = \frac{t_{\mathrm{DS}}}{T_{\mathrm{max\_retry}}} \tag{1}$$

The conditions in which a requester fails to get ADR_REP are discussed below.

- Loss of ADR_REQ: The case where ADR_REQ has been lost then in result requester does not get ADR_REP from leader or gateway node till t_{DR} expires. The requester sends another ADR_REQ with same parameters.
- Loss of ADR_REP: The situation where ADR_REQ successfully arrives at the leader or gateway node, however the ADR_REP fails to arrive at the requester. The message loss may be due to link break, node failure or node departure. If t_{DR} expires and requester fails to receive reply, therefore in this situation the requester sends the same ADR_REQ message again. When the leader or gateway node receives ADR_REQ message then, it searches the tentative address and relative entries from the repository. If the RDS-ID is same for the tentative address and still the $t_{\mathrm{DS}}$is not expired, then the leader or gateway node sends positive ADR_REQ to the requester. But if the RDS-ID is different or t_{DS} expires, then the leader or gateway node sends negative ADR_REQ. Therefore, RDS-ID must be same for the tentative address for $t_{\mathrm{max\_retry}}$ times.
- Coordinator failure: Let new node u selects node v as coordinator and node v fails before it communicates ADR_REP to node u. In this situation, node u waits until t_{DR} expires, and sends another ADR_REQ up to $t_{\mathrm{max\_retry}}$ times. If node u does not receive any ADR_REP up to $t_{\mathrm{max\_retry}}$ times, then it will perform neighbor discovery again and select another coordinator for its configuration process.
- Leader node failure or departure: In case when a leader node fails or abruptly departs from the ad hoc networks then the leader election is triggered by the node that detects the leader failure, as discussed in Section IV-B. The newly elected leader node will perform DAD for its leader_ID and register itself the gateway node. Also, the newly elected leader node has no prior information regarding the addresses assigned to the group members. In this case, each member of that group has to register their addresses by sending a unicast ADR_REQ message to the new leader node. During address registration process, if a leader node detects any duplicate address then it sends a negative ADR_REP message to the member node. If a member node receives negative ADR_REP then it selects another address and performs DAD

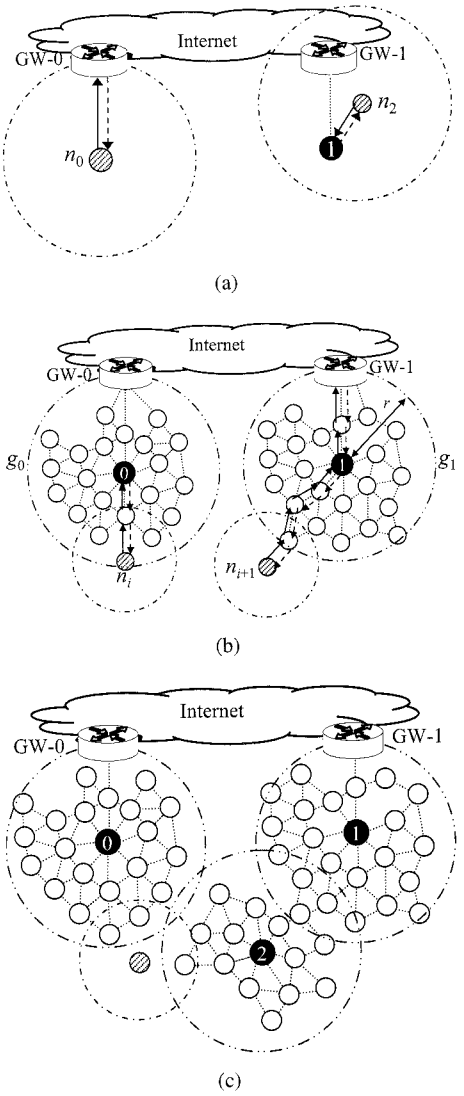


Fig. 5. Network formation: (a) Very first node n_0 joined network and n_2 who discovered leader and gateway node, (b) node n_i is in scope and and n_{i+1} joined out of scope, and (c) multi-homed node.

D. Network Formation

In this section we discuss the network formation when nodes join the network and perform DAD for their IP configuration. Here we consider all different situations of node joins and they are shown in Figs. 5(a)–(c).

- Initial: In this situation the joining node is the only node in the network, as shown in Fig. 5(a) where node n_0 joins the GW-0. The joining node gets only one HELLO_REP

from the gateway router in response to HELLO message, because it is the first node that joined the network. In this case, the gateway router is the coordinator and the joining node is the requester. Here the requester requests for the unique leader_ID and sends the ADR_REQ message with tentative address formed by concatenating the randomly selected leader_ID with MANET-local prefix (i.e., fec0:0:0:ffff::/64 for IPv6) [17]. The requester is considered as a group leader. Therefore, while sending the ADR_REQ, the flag "L" is set high to indicate that the leader node sends the request. Also the isolated node flag in MANET option is set high to indicate the isolation state of the node. The gateway node matches the leader_ID in the GLOBAL_INFO. If there is no entry in the list, then gateway node stores node parameters and sends positive ADR_REP ($A = 0$) message. In case of duplicate Leader_ID, gateway node transmits ADR_REP with $A = 1$ to indicate the conflict. When new node receives positive ADR_REP then it configures its interface with address parameters. In case of conflict, the requester sends another ADR_REQ. The new leader node initializes its d-hop parameter with 0.

- Near the leader and gateway: Here, we describe the situation where new joining node gets HELLO_REP message directly from the leader node and gateway node simultaneously. This instance is shown in Fig. 5(a), when node n_2 joins the network and node 1 is already configured as a leader node. In such a case, the node must select the leader node as its coordinator, because to make this protocol work efficiently, number of leader nodes must be less than the number of member nodes. Hence, the new node sends ADR_REQ to the leader node with M and isolated node flag high to indicate that it is a new member node. The leader node checks the selected member_ID within its LOCAL_INFO table and sends ADR_REP accordingly. The node n_2 sets its d-hop parameter to 1.
- Within the group: Here, we consider the situation where the new node receives HELLO_REP messages from the multiple group members. That state of a new node (n_i) is shown in Fig. 5(b), where node n_i joins the group g_0. In this case, the node n_i selects one of the member nodes as its coordinator by considering the shortest distance "d-hops" (value of d-hops in HELLO_REP) from the group leader. After selecting a coordinator, the requester sends the ADR_REQ to the coordinator with flag $M = 0$ and isolated node flag set to high. The coordinator forwards unicast ADR_REQ message to the leader node. When leader node receives this ADR_REQ message, it performs same operations as described in the situation "*near the leader and gateway*" and replies accordingly. If node ni receives positive ADR_REQ, then it initializes itself with address parameters and sets its d-hop parameter by incrementing distance value from its coordinator.
- Out of scope: When the new node gets HELLO_REPs from the members with $d = r_{\max}$, as shown in Fig. 5(b), where requester (n_{i+1}) joins g_1, it indicates that the requester is out of scope. Hence the requester selects the coordinator, forwards ADR_REQ to the coordinator with random leader_ID, RDS-ID, and sets L and isolated node flag high. When this request arrives at the gateway, the rest of the DAD process is the same as the scenario named "*Initial*."
- Multi-homing: Multi-homing situation is detected by a requester when it receivers HELLO_REPs from member nodes of multiple groups. That situation is shown in Fig. 5(c). In this case, the requester can join any of the group with minimum value of d or the requester can form multiple MANET scope addresses by getting leader_ID from the neighboring groups.

So far, we consider all the possible situations of the joining node and the DAD procedure in all scenarios. In every scenario, the unicast communication of ADR_REQ & ADR_REP messages on multi-hop links during the DAD process is transparent to the requester. Here, we emphasis that the GLOBAL_INFO is periodically exchanged between the gateway nodes to ensure the uniqueness of leader_IDs in the MANET.

E. Global and MANET Scope Address

After performing a successful DAD, a node forms a unique MANET scope address by combining its member_ID, leader_ID with local prefix advertised by the group leader or neighboring node. Leader node advertises that information in reply to solicitation message or periodically within the scope of the group (r-hops). The uniqueness of the address is guaranteed, because all the leader_IDs, in GLOBAL_INFO, are unique in the ad hoc network. Therefore, every node can participate in routing information within the MANET, however they cannot send or receive information from the Internet. To send or receive information from the IP based network, every node must have global scope unique IP.

The gateway router is configured with one or more global prefixes. These global prefixes are advertised periodically or in response to the solicitation message from a node, so that every node forms a global scope IP. The gateway advertisement message contains one or more global prefixes, which are assigned to the gateway router. Every node forms its global scope address from the global prefix and the MANET scope address. After assigning the global scope IP to its interface, a node can send and receive Internet messages.

The member_ID and leader_ID in LOCAL_INFO and GLOBAL_INFO are stored for a time period of $(t_{\mathrm{DS}} \times 3)$s and $(t_{\mathrm{DS}} \times 5)$s at leader and gateway node, respectively. If leader or gateway node does not receive any message or ADR_REQ message within specified time period then these addresses are termed as obsolete and are deleted from the LOCAL_INFO or GLOBAL_INFO. These addresses are free and can be reused again. On the other hand, if a leader or gateway node receives any message from that node whose ID is stored in repository, then its timer is reinitialized.

F. Partitioning and Merging Support

A node detects its partition from the group, when it does not receive any advertisement from its leader node or any response for HELLO message. In this case, that node considers itself partitioned and its MANET scope and global scope addresses are no more valid for data and control message communication. When this node joins or merges with the same or different ad hoc net-

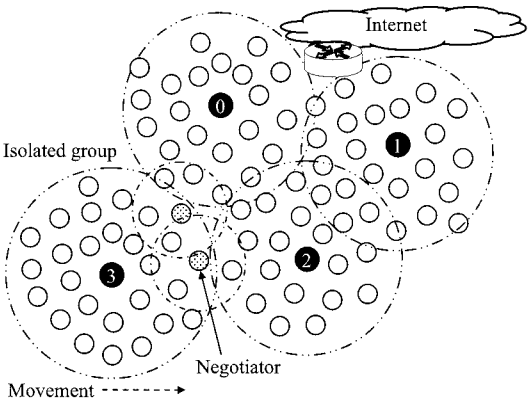


Fig. 6. Isolated group of nodes joining a MANET.

work, then it has to get a new IP by performing the autoconfiguration from scratch.

Group partitioning is determined, when the leader node does not receive periodic or solicited global advertisement from the gateway router but it gets messages from its member nodes. Also partitioning situation is detected when bordering nodes do not encounter any echo or message transmission in neighboring groups. When this isolated group of nodes joins or merges with other MANET, as shown in Fig. 6, the border nodes of the isolated group perform a major role. While merging with the other ad hoc network, border nodes of the isolated group get router advertisement, global advertisement or echo of the transmitted messages from the other neighboring MANET nodes. The border nodes that detect this situation inform the group leader. In this case, the leader node of the isolated group selects one of its bordering nodes, which encounters more number of neighbors, to perform DAD. This node is called Negotiator.

To perform DAD for a unique leader_ID of the group leader, negotiator selects a coordinator from one of the bordering nodes of the MANET to which the isolated group joins. After that ADR_REQ message is generated by the leader node with $L = 1$ and isolated node flag is set to be low. Then, this message is forwarded to gateway through negotiator to the coordinator node. This isolated flag indicates that the ADR_REQ contains leader_ID of the new leader node for DAD instead of negotiator's address for DAD. The ADR_REQ message is forwarded to the gateway node in a unicast manner. In case of unique Leader_ID the gateway node send ADR_REP message to the new leader node through negotiator. In case of unique leader_ID, the leader node announces its leader state to the whole group. After getting this leader_ID every node in the group renumbers its IP, and forms the new MANET scope unique IP. Every node in the group forms global scope address by receiving solicited or un-solicited gateway advertisements.

IV. GROUP LEADER ELECTION

A. System Metrics

We propose a leader election algorithm by considering the important metrics of a mobile node, such as available resources, degree of a node and relative distance from the center of the group. An available resource is an important criterion to select a leader because a group leader consumes ample amount of resources to accomplish the group management tasks. Also the degree and relative distance of a node from leader of the cluster are considered to avoid the division of a group area into two more groups after changing the group leader [26], [27].

The unit graph $G(V, E)$ is a best model for ad hoc networks, where V represents wireless mobile nodes and E denotes the links between mobile nodes. Node v is in r-hop neighborhood of node $u\{\delta(u)\}$, if the distance between u and $v\{d(u, v)\}$ is less than or equal to r-hops.

$$\delta(u) = \{v \in E | d(u, v) \leq r\}. \tag{2}$$

The number of r-hop neighbors of node u is calculated as follows:

$$\lambda_r(u) = |\delta(u)|. \tag{3}$$

The r-hop density of node u, $D_r(u)$ in (4), is defined as "the ratio of edges between node u, its neighbors and edges between its r-hop neighboring nodes and the number of nodes in its r-hop neighborhoo" [28].

$$D_r(u) = \frac{|e = (v, w) \in E | v \in \delta(u) \text{ and } w \in \delta(u)|}{\lambda_r(u)} \tag{4}$$

where $\lambda_r(u)$ represents number of nodes in r-hop neighborhood of node u. To calculate the node density every node sends the list of its neighbors to the neighboring node during neighbor discovery process.

Every node has different amount of available resources and every available resource has relative importance in every distinct application. In our leader election algorithm we consider, r-hop node density $D_r(u)$, residual power $P(u)$, node speed $S(u)$, and distance $d(u)$ from the leader node to calculate node weight. Due to the varying scales of node parameters they must be normalized before calculating the overall node weights. There are two methods to normalize node weights: *Benefit criteria*, where maximum value is the best, i.e., $D_r(u)$ and $P(u)$ and *cost criteria*, where minimum value is best, i.e., $S(u)$. The node parameters are normalized as follows:

$$n_{ij} = \begin{cases} \dfrac{x_{ij} - x_j^{\min}}{x_j^{\max} - x_j^{\min}} & \text{for benefit criteria} \\ \dfrac{x_j^{\max} - x_{ij}}{x_j^{\max} - x_j^{\min}} & \text{for cost criteria} \end{cases} \tag{5}$$

where n_{ij} represents the normalized value of jth resource of neighboring node i. Also $x_j^{\min}$ and $x_j^{\max}$ represent the minimum and maximum value of resource j of neighboring nodes, respectively. The x_{ij} is the actual value of parameter j (i.e., $D_r(u)$, $P(u)$, or $S(u)$) of ith neighbor of node u.

The weighting factors w_i assigned to each individual node resource are application specific. For our leader election scheme, the relative weighting factors of $D_r(u)$, $P(u)$, and $S(u)$ are 0.4, 0.4, and 0.2, respectively. We assign higher values to the resources that are important to keep a node to act as a leader for

longer time. The overall weight of a node is calculated as:

$$W_i(u) = \frac{\sum_{j=1}^{m} w_j n_{ij}}{d_i(u) + 1} \quad (6)$$

where $\sum w_j = 1$ and $m = 3$, because we consider only 3 node parameters in our election algorithm. In (6), d represents the distance of a node from the center of a group area or from the previous group leader. As the distance increases from the center of the group area, the overall weight decreases for every node, even if it has more resources. This information prioritizes the nodes, which are near the center of the group and increase their chances to be selected as a group leader. This criterion also avoids impairing the group structure and decreases the splitting chances of a group into two or more groups.

B. Group Leader Election Algorithm

In this section, we discuss our proposed leader election algorithm in contrast with the existing weighted clustering algorithms (WCA) in [31], [32] and also the conventional election algorithm in [20]. Also a good comparison and discussion of different clustering algorithms is discussed in [33]. The weighted WCA algorithm considers multiple node parameters to calculate node weights. However, WCA considers physical distance between neighboring nodes as one of the election parameter and as a criteria to trigger the election procedure. In result, it causes high overhead and frequent changes in clustering structure of ad hoc network. Similarly, the clustering algorithm in [20] is triggered periodically after every t_{DS}, which significantly increases the network traffic and may cause the impairment of the group structure. The proposed election algorithm in [20] also considers number of nodes in the scope as a clustering criteria. The ad hoc network has dynamic architecture where node density may vary time to time because of joining and departure of nodes. Therefore it may result in a frequent change of leader node.

Our proposed leader election algorithm considers multiple node parameters to calculate node weights. The node with highest weight in scope is elected as a clusterhead. It considers r-hop node density, residual battery power and node speed. These parameters are highly dynamic in ad hoc network and the election decision based on these parameters result in stable hierarchical network architecture. The relative distance between previous leader node and member node is also considered to prioritize the nodes near the center of cluster and avoid the division of a group area into two more groups after election of a new group leader. The algorithm notations and our proposed clustering algorithm are shown in Tables 2 and 3, respectively. Leader election algorithm elect leaders based node multiple node parameters and the node with highest weight in r-hop scope is assigned the functionality of a leader node. These node parameters are propagated via ADR_REQ, ADR_REP, neighbor discovery, or separate message within r-hops. This algorithm is triggered when:

- Leader node departs gracefully.
- A node detects the leader node failure.
- A group of nodes detects the partition.

When a leader node gracefully departs from the group then it selects one of its members with highest $W(u)$ as a group leader. The leader node sends a unicast leader election message (ADR_REQ with $E = 1$) to the newly selected node. Also leader node sends the member_ID list to the new leader node. The newly selected leader node uses the same leader_ID that was used by the previous leader node and sends a multicast message to announce its leader state. While using the old leader_ID, the other nodes do not need to renumber their addresses and they can use their old addresses for the data communication. The graceful departure also includes the situation when its overall weight $W(u)$ becomes less than the predefined system threshold $TH_{\min}$.

In case when leader node fails and does not select another node as a leader node, then the leader election is triggered by the node that detects the leader failure. This node broadcasts a leader election message (ADR_REQ with $E = 1$ and "node parameters" field of ADR_REQ with node's current residual battery power, r-hop density, current speed, and distance from previous leader node) within group radius of r-hops and initialize t_{LE}. The other nodes, when they receive this message, they also generate the similar election message with their respective node parameters and initialize t_{LE}. When t_{LE} expires, every node will calculate overall weight $W_i(u)$ based on the shared node parameters using (5) and (6). If node itself has highest weight, then announce as leader node otherwise send ADR_REQ to the node with $\max\{W_i(u)\}$. The node with highest weight will use the same leader_ID to prevent the renumbering or it can get a new leader_ID by performing the DAD. All the members send unicast message to register their member_IDs to the new elected leader node.

Leader election algorithm is also initiated when a group of nodes departs from a MANET. If leader node also departs with this group, then they do not need to perform leader election

Table 2. Algorithmic notation.

Symbol	Description
E	E flag denoting leader election
W_i	Node i's overall weight
x_{ij}	Actual values of node parameters in the "node parameters" field of ADR_REQ / ADR_REQ message
t_{LE}	Leader election times

Table 3. Group leader election algorithm.

i. When leader node gracefully departs:
1. Select node with $\mathrm{MAX}\{W_i(u)\}$, [where $i = 1, 2, \cdots,$ length (neighbor_list)]
2. Send unicast message with $E = 1$
3. Reset all parameters and depart from the group.

ii. When member node u detects leader node failure/group partitioning:
1. Broadcast node parameters in r-hops and wait for t_{LE}.
2. Calculate weights of neighboring node and node itself using (5) and (6).
3. If $\{W_i(u) ==$ node itself$\}$ then [where $i = 1, 2, \cdots,$ length(neighbor_list)]
4. Set *Status = Leader*
5. Send *Leader Advertisement* message.
6. Else
7. Send ADR_REQ to the node with $\mathrm{MAX}\{W_i(u)\}$.
8. *Repeat this process until max_retry.*
9. If {Receive ADR_REP} then
10. *Initialize Interface with new ID*
11. Else
12. *Select another node and repeat the process again until node join any cluster or form a new cluster.*
13. End If
14. End If

whereas if there is no leader node within the isolated group, then one or more nodes that detect the departure state initiate the leader election. The rest of the election procedure is same as the leader node failure state in the previous paragraph.

V. PERFORMANCE EVALUATION

A. Simulation Setup

Simulation is performed in NS2.33 [29] for a varying network size of 50, 100, 150 to 500 nodes connected with fixed network via 2 gateway nodes. We consider a network with random way Point mobility model [30] where every node randomly selects a new x, y coordinates and moves at randomly selected speed between 0 to Max_Speed. After arriving at the randomly selected location, a node waits for a pause time of 2 s and selects another new location within the simulation area of 1000 m $\times$ 1000 m. The network performance is measured for different group radiuses of r-links. Table 4 shows the simulation parameters. In initial phase of simulation, 25 mobile nodes randomly form the network by performing DAD to verify their address uniqueness and forming the group architecture by electing group leaders among it selves. These 25 nodes are called pre-configured nodes and rest of the network nodes will join this preconfigured network at random intervals during simulation time of 200 s.

Table 4. Simulation parameters.

Parameter	Value
Simulator	NS 2.33
Network size	50,100, 150, $\cdots$, 500 nodes
Pre-configured nodes	25
Gateway nodes	2
$r_{\max}$	1, 2, and 3
MAC type	802.11
Network interface	Wireless_802_11_Phy
Propagation model	Two ray ground
Antenna type	Omniantenna
Channel bandwidth	1 Mbps
Transmission range	150 m
Simulation area	1000 m $\times$ 1000 m
Max_Speed	0.1, 0.2, $\cdots$, 2.5 (m/s)
Mobility model	Random way point
Pause time	2 s
Simulation time	200 s
$t_{\max\_retry}$	3
t_{DS}	6 s
t_{LE}	5 s

The simulation results are the average of 10 simulations per scenario and we found that the outcome doesn't vary during all those iterations. The simulation results measure the network overhead in terms of number of packet processed in the network. It is very important to note that our scheme considers ad hoc networks connected with fixed networks via gateways whereas proposed scheme just considers isolated ad hoc networks. Therefore the control overhead of the proposed scheme includes the gateway discovery messages also. However, the conventional scheme doesn't count the number gateway discovery overhead. The other simulation parameters are: Average energy consumption per node, address allocation latency, number of address conflicts, number of clusterheads, and clusterhead updates, re-affiliation count, leader election, and network merging overhead and latency.

We simulate the address autoconfiguration scheme with leader election algorithm and the results are shown in Figs. 7–17. The simulation of address autoconfiguration and the performances of the proposed scheme and the conventional scheme [20] are compared for the network scenario where new node(s) join (merge) and leave (depart) the network. The performances of address autoconfiguration are shown in Figs. 7–10 and 16–17 to compare the effectiveness of both the proposed and conventional autoconfiguration schemes. The performance of the proposed leader election algorithm is compared with the conventional scheme and WCA and shown in Figs. 11–15. The WCA scheme [31] is a clustering technique only, which considers multiple node parameters, as our proposed scheme, to elect leader nodes and does not consider any autoconfiguration mechanism. Therefore the performance of WCA is not included in Figs. 7–10 and 16–17. Also, WCA does not support any partitioning and merging, hence the performances of the proposed, conventional and WCA leader election scheme are compared for the fixed network size to fairly justify the effectiveness of the proposed leader election scheme. Almost all the clustering techniques that consider multiple node parameters [33] do not propagate node weight to multiple hops to elect a leader election. WCA is the only scheme that propagates the node weight information to multiple hops to elect a potential leader node. Therefore, we compare the effectiveness of the proposed leader election algorithm with the WCA and conventional scheme and the results are shown in Figs. 11–15.

B. Overhead and Scalability

In our simulation, we calculate the network overhead in terms of number of unicast, multicast and broadcast messages processed by every node that participates in DAD session. As our proposed scheme is for the ad hoc networks connected to the IP network through gateway(s) and compared with the conventional scheme that is an autoconfiguration scheme for the isolated ad hoc networks. Therefore, the control overhead for our proposed scheme includes the gateway advertisement messages, whereas the gateway advertisement overhead is not counted in the conventional scheme. Fig. 7 shows the control overhead in ad hoc network for varying network size and r-hops to check the scalability and stability of ad hoc network. To perform DAD in the proposed scheme, messages are transmitted in unicast manner, whereas in the conventional scheme for performing DAD, messages are broadcasted within a group. It is shown that the increase in the network size results in a slight increase in control overhead in our proposed scheme compared to the conventional scheme. Also for different group radius of r-hops, there is small increase control overhead in the proposed scheme in contrast with the conventional scheme. Because in the proposed scheme, DAD messages are transferred in unicast manner from coordinator to the leader node, and due to increase in r has not significant impact on the control overhead. On the contrary, in the conventional scheme these messages are flooded from one

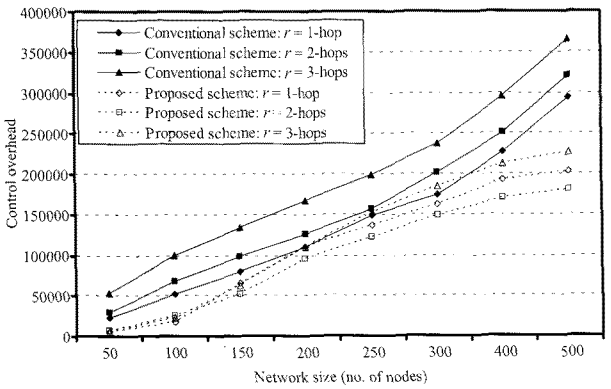


Fig. 7. Control overhead vs. network size for scope $r = 1, 2$, and 3 and Max_Speed=5 m/s.

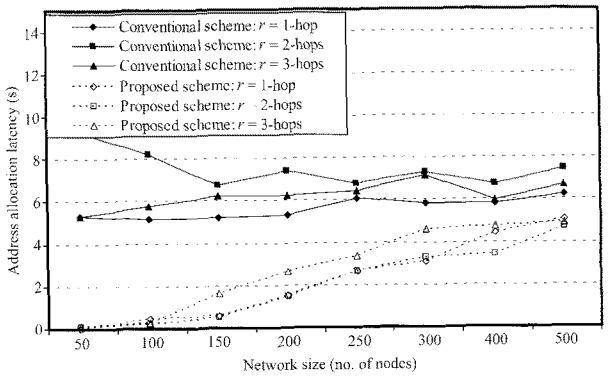


Fig. 9. Average address allocation latency vs. network size for $r = 1, 2$, and 3 and Max_Speed=5 m/s.

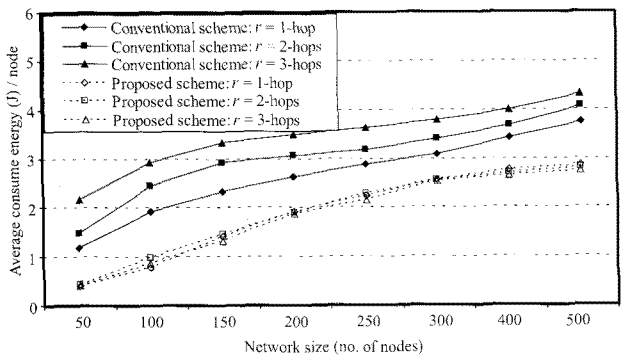


Fig. 8. Average energy consumption per node vs. network size $r = 1, 2$, and 3 and Max_Speed=5 m/s.

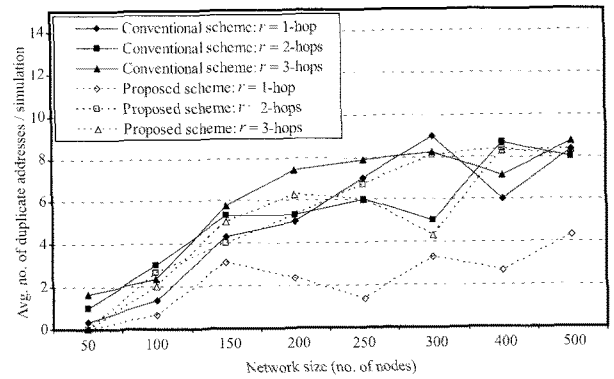


Fig. 10. Average address conflicts vs. network size for $r = 1, 2$, and 3 and Max_Speed=5 m/s.

boundary node to the whole group, that's why it causes high control overhead in conventional scheme.

Fig. 8 shows the average energy consumption per node for different network sizes and group radius for node speed of 5 s. It shows that our scheme has less consumed energy per node compared to the proposed scheme. The average consumed energy per node has no effect of different scope size in proposed scheme. Also our scheme has small increase in energy consumption when network size increases, because of the gateway advertisement messages are broadcasted in the network along with the DAD messages, which cause increase in the control overhead and also the energy consumption. On the other hand, the conventional scheme only considers the isolated ad hoc networks, therefore the energy consumption is only result of DAD messages. It is evident from the results that even though in the presence of multiple gateways and their redundant advertisement messages, our scheme has less energy consumption per node.

Figs. 9 and 10 show the average address allocation latency and the average number of address conflicts for different r-hops versus the network size. Fig. 9 shows that the average address allocation latency of the proposed scheme is less than the conventional scheme. Since the conventional scheme initializes the DAD timer, which is 5 s, and waits for the address reply message. If there is no reply message during that period then it assumes that the address is unique. However, if there is an address conflict then it sends another request message and again initializes the timer. Conversely, the proposed scheme just sends an ADR_REQ and waits for the ADR_REP from either leader node or the gateway node. It also initializes the timer t_{DAD} that is equal to 2 s. If node doesn't receive till t_{DAD} expires, then another ADR_REQ is sent to the gateway of leader node. In result when packet loss ratio is less, in case of small network sizes, the address allocation latency is almost less than 0.5 s otherwise the latency slightly increases in the proposed scheme. Fig. 9 shows that the average improvement in the address allocation latency of our proposed scheme is from 57% to 71% for different network sizes and values of r-hops. Fig. 10, shows that average number of conflicts versus the network size for 10-bits size of Member_ID and Leader_ID. The number of address conflicts for both the schemes increase when there is increase in network size.

The performance comparison of the proposed leader election algorithm and the conventional election algorithms in [20] and WCA [31], [32] are depicted in Figs. 11–13. These leader election performances are calculated for network of 200 nodes, scope of 2 hops and varying node speed from 0.1 to 2.5 m/s. Fig. 11, show that the proposed election algorithm maintains less number of clusters or groups to avoid that unnecessary cluster maintenance overhead. It is only because our clusterhead election scheme considers the 2-hop node density as one of the criteria to elect leader nodes. Our leader election algorithm also

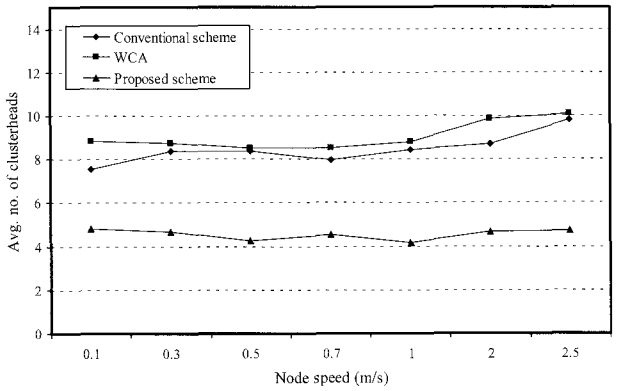


Fig. 11. Average no. of clusterheads vs. node speed for network size of 200 nodes and $r = 2$.

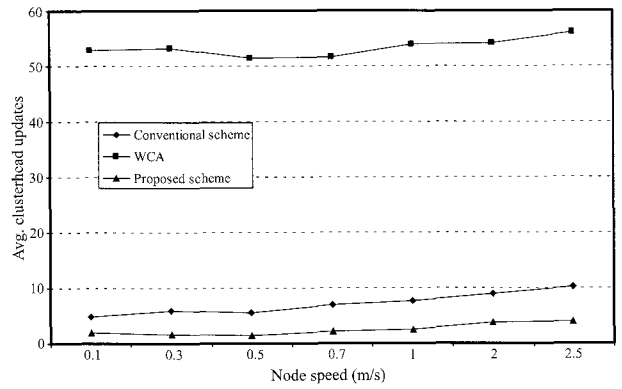


Fig. 12. Average no. of clusterhead updates vs. node speed for network Size of 200 nodes and $r = 2$.

takes into account the node mobility and residual battery power to avoid the frequent deformation of clusters due to clusterhead failure or clusterhead departure, refer Fig. 12 for the clusterhead updates.

In contrast, the WCA and conventional election algorithm consider overall distance between a node and its neighbors and number of nodes in scope, respectively, as an election criterion and to trigger the election algorithm. Therefore, node movement in an ad hoc network results in frequent change in these parameters, which triggers clustering algorithm frequently. In conventional election algorithm there are chances that a node with high speed and/or less residual battery power with larger number of neighbors can be a clusterhead. That type of clusterhead can only form a cluster for a very short time due to abrupt departure or depletion of battery power. Hence, the conventional election algorithm has also a very frequent leader election updates compared to the proposed scheme. It is evident in Fig. 12 our clusterhead election algorithm has less number of clusterhead updates compared to other clusterhead election schemes.

The cluster re-affiliation count is the number of times a node moves from one cluster to another or joins another cluster as a member due to failure or migration of its pervious clusterhead. Fig. 13 shows that our clustering algorithm has less number of re-affiliations than the WCA and conventional algorithm. Fig. 14 illustrates the overhead comparison of the proposed, conventional and WCA clusterhead election algorithm. When

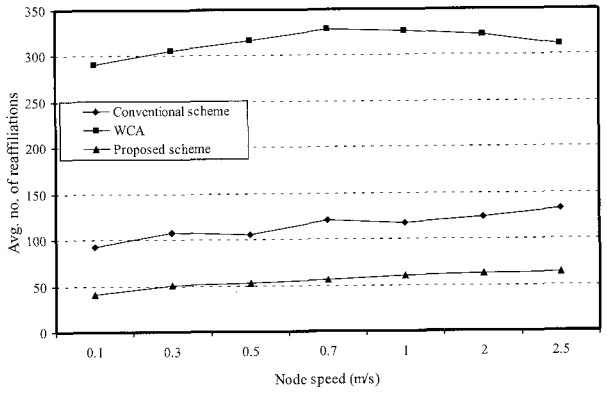


Fig. 13. Average no. of re-affiliations vs. node speed for network size of 200 nodes and $r = 2$.

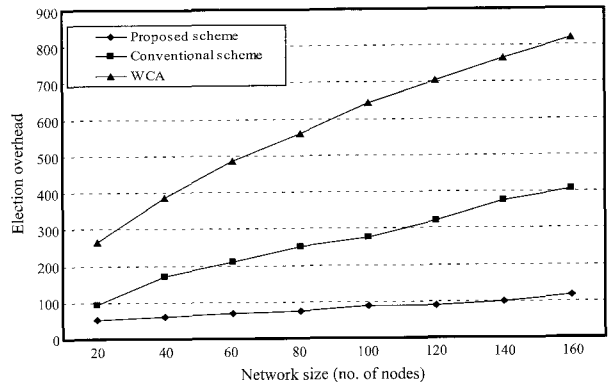


Fig. 14. Leader election overhead vs. network size for node speed of 0.5 m/s and $r = 2$.

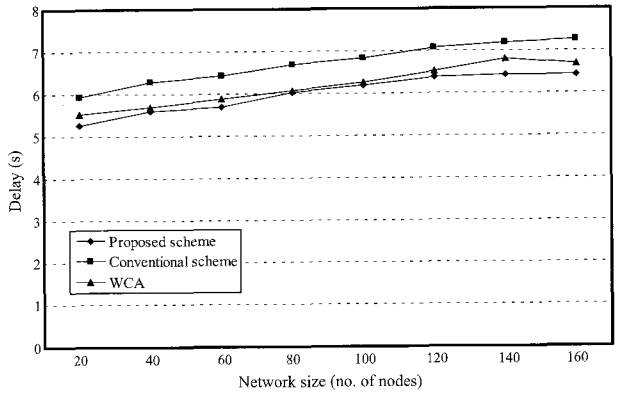


Fig. 15. Leader election delay vs. network size for node speed of 0.5 m/s and $r = 2$.

WCA election algorithm is triggered then every node calculate its own weight based on its node parameters and broadcast in whole network. Similarly, in conventional scheme when leader election is triggered due to significant difference in node density parameter of leader and member nodes. Then, node checks its ID by broadcasting its random ID and its r-hop scope density. On the otherhand if leader election is triggered in proposed scheme, the node parameters are broadcasted within the scope and every node in the scope calculate the weight of its neighbors and itself. The node with higher weight announces itself as

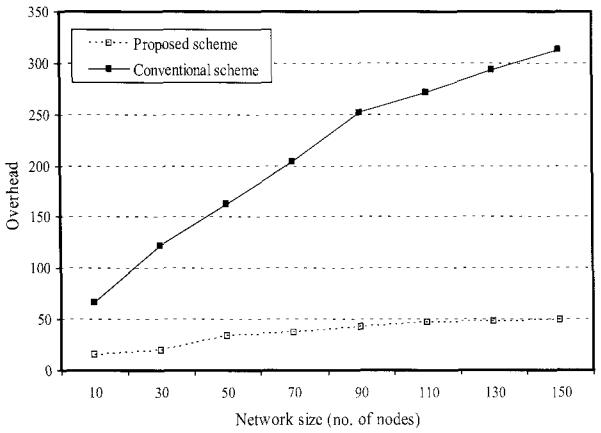


Fig. 16. Merging overhead vs. network size for speed = 0.5 m/s and $r = 2$.

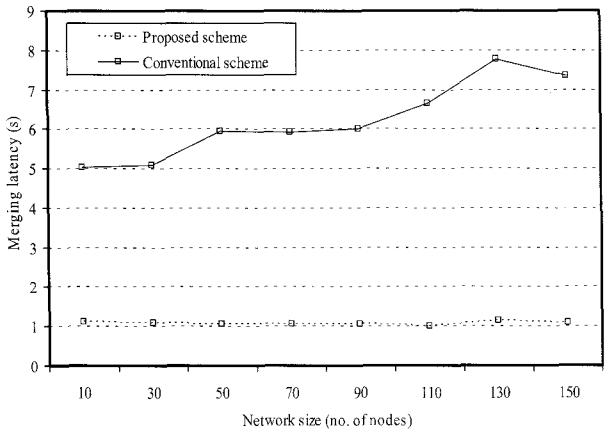


Fig. 17. Merging latency vs. network size for speed = 0.5 m/s and $r = 2$.

a leader and all the other nodes join that cluster. Due to avoidance of any full flooding of node weights or node parameters within whole network in proposed scheme, our leader election algorithm has less overhead than the WCA and conventional scheme.

The leader election latency is time period between detection of leader failure and the leader ID assignment to the new leader. Fig. 15 represents that the leader election latency in proposed scheme and conventional scheme is almost same. Since in our proposed scheme after leader election failure detection all nodes in that group share the node parameters within scope and initialize the timer t_{LE}. After the expiry of t_{LE}, nodes calculate their weights and the node with highest overall weight in scope is elected as a leader node. Afterward that newly elected leader node either uses same leader_ID or sends ADR_REQ for new leader_ID to the gateway. Similarly, after leader node failure detection in WCA the weights are broadcasted in whole network and timer is initialized. In conventional leader election algorithm, soon after leader election failure a node with higher number of neighbors and with status as candidate, become the leader node and will broadcast the DAD messages in the network and initializes the t_{DAD} timer. When timer expires and there is not any reply from any other node, then that node is considered as leader node.

Next, we compare the overhead of both schemes when an isolated group of nodes merges with the configured network. Fig. 16 illustrates the merging overhead of an isolated network of 10 nodes with the network of size 10 to 150 nodes. In case of network merger, our scheme selects one of its bordering nodes that perform DAD for the leader node and forwards the DAD messages to the gateway node. Once, the unique leader_ID is assigned to the leader node then all members just renumber their addresses and can forward traffic over the network. On the other hand, the conventional scheme broadcast messages for testing the uniqueness of the leader node, which in turn increases the overhead. Also the delay due to broadcast and leader ID uniqueness has higher latency than performing uniqueness test for leader ID using unicast messages. The results in Figs. 16 and 17, show that the overhead and latency of network merger for conventional scheme increases when the isolated network merges with the larger networks. This proves that the efficiency of the proposed scheme is better than the conventional scheme is scalable.

C. Summary of Evaluations

After analyzing the performances of our proposed in contrast to the proposed scheme and WCA leader election algorithm, we found that: Our scheme has reduced average of 63% in address allocation latency, 40% less network overhead even after taking into account the gateway advertisements, saved 40% of average energy consumption per node and 80% less overhead when network merges. In case of leader election: our proposed leader election maintains 45% and 49% less clusters, has 65% and 91% less clusterhead updates and has around 50% and 81% less number of re-affiliations compared to conventional scheme and WCA respectively. Therefore, we conclude that our scheme has better performance than the conventional and WCA schemes and is scalable and effective for varying network sizes and node movement speeds. The performance of the proposed autoconfiguration scheme depends upon the efficiency of the clustering algorithm. Because frequent re-clustering and changes in clusterheads will slightly degrade the performance of the proposed scheme. In case of conventional scheme, the performance will be drastically degraded if there is frequent re-clustering or clusterhead changes because it broadcasts the DAD messages in the network.

VI. CONCLUSION

We have proposed an autoconfiguration scheme for hierarchical topology MANETs by considering the global connectivity. In our scheme, we eliminate the broadcast of DAD messages in the network to reduce overhead and to make our scheme scalable. The joining node selects a coordinator node from the preconfigured members of a group, which performs DAD operations on behalf of the joining node. By computer simulation it is shown that our proposed scheme can reduce overhead in terms of broadcast and unicast messages per address allocation, and is scalable for large networks. Also the computer analysis represents that the proposed leader election algorithm can perform efficiently in the large network scenarios.

REFERENCES

[1] S. Ruffino, P. Stupar, T. Clausen, and S. Singh, "Connectivity scenarios for MANET," Draft-ruffino-conn-scenario-01.txt, July 2005.

[2] I. F. Akyildiz and W. Xudong, "A survey on wireless mesh networks," *IEEE Commun. Mag.*, vol. 43, no. 9, pp. S23-30, Sept. 2005.

[3] R. Droms, "Dynamic host configuration protocol," RFC 2131, Mar. 1997.

[4] R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," RFC 3315, July 2003.

[5] IETF Ad-Hoc Network Autoconfiguration. [Online]. Available: http://www.ietf.org/html.charters/autoconf-charter.html

[6] C. Bernardos, M. Calderon, and H. Moustafa, "Survey of IP address autoconfiguration mechanisms for MANETs," draft-bernardos-manet-autoconf-survey-04, Nov. 2008.

[7] Guidelines For 64-bit Global Identifier (EUI-64). [Online]. Available: http://standards.ieee.org/db/oui/tutorials/EUI64.html

[8] C. E. Perkins, J. T. Malinen, R. Wakikawa, E. M. Belding-Royer, and Y. Sun, "IP address autoconfiguration for ad hoc networks," draft-perkins-manet-autoconf-01.txt, Nov. 2001.

[9] R. Wakikawa, J. T. Malinen, C. E. Perkins, A. Nilsson, and A. J. Tuominen, "Global connectivity for IPv6 Mobile Ad Hoc Networks," draft-wakikawa-manet-globalv6-05.txt, Mar. 2006.

[10] J. Jeong, J. Park, H. Kim, H. Jeong, and D. Kim, "Ad hoc IP address autoconfiguration," draft-jeong-adhoc-ip-addr-autoconf-06.txt, Jan. 2006.

[11] S. Ruffino and P. Stupar, "Automatic configuration of IPv6 addresses for MANET with multiple gateways (AMG)," draft-ruffino-manet-autoconf-multigw-03, June 2006.

[12] H. W. Cha, J. S. Park, and H. J. Kim, "Extended support for global connectivity for IPv6 mobile ad hoc networks," draft-cha-manet-extended-support-globalv6-00.txt, Oct. 2003.

[13] T. Clausen and E. Baccelli, "Simple MANET address autoconfiguration," draft-clausen-manet-address-autoconf-00.txt, Jan. 2005.

[14] C. Jelger, T. Noel, and A. Frey, "Gateway and address autoconfiguration for IPv6 adhoc networks," draft-jelger-manet-gateway-autoconf-v6-02.txt, Apr. 2004.

[15] K. Weniger and M. Zitterbart, "IPv6 stateless address autoconfiguration for hierarchical mobile ad hoc networks," draft-weniger-manet-addressautoconf-ipv6-00.txt, Feb. 2002.

[16] N. H. Vaidya, "Weak duplicate address detection in mobile ad hoc networks," in *Proc. ACM MobiHoc*, Lausanne, Switzerland, June 2002.

[17] C. E. Perkins, J. T. Malinen, R. Wakikawa, E. M. Royer, and Y. Sun, "IP Address Autoconfiguration for Ad Hoc Networks," Internet Draft, draft-perkins-manet-autoconf-01.txt, Nov. 2001.

[18] K. Weniger, "PACMAN: Passive Autoconfiguration for Mobile Ad Hoc Networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no.3, pp. 507–519, Mar. 2005.

[19] N. Choi, C. K. Toh, Y. Seok, D. Kim, and Y. Choi, "Random and Linear Address Allocation for Mobile Ad Hoc Networks," in *Proc. WCNC*, New Orleans, USA, Mar. 2005.

[20] K. Weniger and M. Zitterbart, "IPv6 autoconfiguration in large scale mobile ad-hoc networks," European Wireless-2002, 2002.

[21] T. Narten, "Neighbor Discovery and stateless Autoconfiguration in IPv6," *IEEE Internet Comput. Mag.*, vol 3, pp. 54-62, 1999.

[22] T. Narten, E. Nordmark, and W. Simpson "Neighbor Discovery for IP Version 6 (IPv6)," RFC 2461, Dec. 1998.

[23] I. D. Chakeres and E. Belding-Royer, "The utility of hello messages for determining link connectivity," in *Proc. 5th WPMC*, 2002.

[24] E. Baccelli, T. Clausen, and J. Garnier, "OLSR Passive Duplicate Address Detection," draft-clausen-olsr-passive-dad-00.txt, July 2005.

[25] K. Weniger and K. Mase, "PDAD-OLSR: Passive Duplicate Address Detection for OLSR," draft-weniger-autoconf-pdad-olsr-01, June 2006.

[26] S. Sivavakeesar and G. Pavlou, "Stable clustering through mobility prediction for large-scale multihop intelligent ad hoc networks," in *Proc. IEEE WCNC*, Mar. 2004.

[27] Q. Liang, "Clusterhead election for mobile ad hoc wireless network," in *Proc. PIMRC*, Sept. 2003, pp. 1623–1628.

[28] N. Mitton and E.Fleury, "Self-organization in ad hoc networks," Research Rep. INRIA, RR-5042.

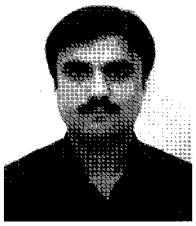
[29] The Network Simulator - NS-2. [Online]. Available: http://www.isi.edu/nsnam/ns/

[30] C. Bettstetter, G. Resta, and P. Santi, "The node distribution of the random waypoint mobility model for wireless ad hoc networks," *IEEE Trans. Mobile Comput.*, pp. 257–269, July–Sept. 2003.

[31] M. Chatterjee, S. K. Das, and D. Turgut, "An on-demand weighted clustering algorithm (WCA) for ad hoc networks," in *Proc. IEEE GLOBECOM*, 2000, pp. 1697–1701.

[32] M. Chatterjee, S. Das, and D. Turgut, "WCA: A weighted clustering algorithm for mobile ad hoc networks," *J. Cluster Comput.* vol. 5, 2002, pp. 193-204.

[33] J. Y. Yu and P. H. J. Chong, "A survey of clustering schemes for mobile ad hoc networks," *IEEE Commun. Surveys Tuts.*, vol.7, no.1, pp. 32-48, First Quarter 2005.



Safdar Hussain Bouk received the B.E. degree in Computer Systems from M.U.E.T., Jamshoro, Pakistan, in 2001 and M.Sc. degree from department of Information and Computer Science, Keio University, Yokohama, Japan, in 2007. Currently, he is a Ph.D. student at Keio University. His research interests include autoconfiguration, clustering, and trust management in ad hoc networks.

Iwao Sasase was born in Osaka, Japan in 1956. He received the B.E., M.E., and D.Eng. degrees in Electrical Engineering from Keio University, Yokohama, Japan, in 1979, 1981, and 1984, respectively. From 1984 to 1986, he was a Post Doctoral Fellow and Lecturer of Electrical Engineering at University of Ottawa, ON, Canada. He is currently a Professor and Chairman of Information and Computer Science at Keio University, Yokohama, Japan. His research interests include modulation and coding, broadband mobile and wireless communications, optical communications, communication networks, and information theory. He has authored more than 256 journal papers and 364 international conference papers.

He received in 1984 IEEE Communications Society Student Paper Award (Region 10), in 1986 Inoue Memorial Young Engineer Award, in 1988 Hiroshi Ando Memorial Young Engineer Award, in 1988 Shinohara Memorial Young Engineer Award, and in 1996 Institute of Electronics, Information, and Communication Engineers (IEICE) of Japan Switching System Technical Group Best Paper Award. He was the Vice President of the IEICE Communications Society (2004–2006), the Chair of the IEICE Network System Technical Committee (2004–2006), the Director of the IEEE ComSoc Asia Pacific Region (2004–2005), the Chair of the IEEE ComSoc Satellite and Space Communications Technical Committee (2000–2002), the Chair of the IEICE Communication System Technical Committee (2002–2004), and Director of the Society of Information Theory and Its Applications in Japan (2001–2002). He is a Senior Member of IEEE, a Member of IEICE, SITA and Information Processing Society of Japan, respectively.