Analytical Modeling of Address Allocation Protocols in Wireless Ad Hoc Networks
(Invited Paper)

Ahmad Radaideh and John N. Daigle

University of Mississippi, University, MS, USA 38677
wcdaigle@olemiss.edu, radaidehcie@gmail.com

Abstract. A detailed description of the Internet Protocol Address Assignment (IPAA) is presented and state diagrams for its behavior constructed. Formulae for the expected latency and communication overhead of the IPAA protocol are derived, with the results being given as functions of the number of nodes in the network with message loss rate, contention window size, coverage ratio, and the counter threshold as parameters. The IPAA and MANET Configuration (MANETconf) are compared in detail. The results show that the latency and communication overhead for MANETconf are significantly higher than for the IPAA protocol. Results of extensive sensitivity analyses for the IPAA protocol are also presented.

Keywords: ad hoc network, address assignment latency, address assignment overhead, probabilistic modeling.

1 Introduction

In this paper, a detailed description of the Internet Protocol Address Assignment (IPAA) protocol is presented, and analytical derivations for the expectations of latency and communication overhead are given. Expected values of the performance measures as a function of the number of network nodes at the time the new node joins the network are presented. Analytical and simulation results are presented to show the effect of changes in message loss rate, coverage ratio, and the contention window size on the performance measures. Performance of this protocol is also compared to that of the MANET Configuration (MANETconf).

Baccelli [1] discusses design of address configuration protocols for MANETs, where uniqueness of the assigned address is a major issue. Some proposed solutions are based on duplicate address detection (DAD) mechanism while others use the binary-split idea [2] so that each node has a disjoint subset of network addresses. Most mechanisms are classified in [3].

Perkins et al. [4] configures by first choosing a random address then performing a DAD procedure within the MANET. The protocol performs DAD only when assigning an IP address to a new node, the proposed protocol lacks support for partitioning and merging in MANET. Jeong et al. [5] proposes a protocol

J. Zheng, D. Simplot-Ryl, V.C.M. Leung (Eds.): ADHOCNETS 2010, LNICST 49, pp. 429–446, 2010.

with two address detection mechanisms. A strong DAD, based on the protocol proposed in [4], is performed in the initial phase to verify the uniqueness of the randomly selected address and a weak DAD, based on [6], which is always executed in order to prevent address conflicts with existing nodes. The weak DAD uses a virtual address, which is a combination of an address and a key. The key, which is assumed to be unique in the network, is appended to the address in the routing messages and the routing table. The weak DAD also identifies duplicate addresses, but his protocol monitors and changes the routing messages and therefore is considered routing protocol dependent.

A passive DAD approach [7] has been adopted in some proposed solutions for dynamic address configuration, such as in [8] and [9]. Passive DAD enables nodes to detect duplicate addresses in the network by analyzing received routing protocol messages. One way to detect address conflicts is based on the sequence number of a link-state routing messages. Other ways to detect address conflicts are based on locality and neighborhood, and they are all based on analyzing routing information, which makes these solutions routing-protocol dependent.

Zhou et al. [10] proposed a mechanism based on a stateful function, $f(n)$, to derive addresses with low probability of address duplication and therefore it avoids the use of DAD. The initial state of $f(n)$ is a seed that generates a sequence of unique numbers that can be used as network addresses. The function has to be designed carefully such that the interval between two occurrences of the same number in the generated sequence is extremely long and the probability of generating the same number in finite number of sequences initiated by different seeds is extremely low, which is considered a hard mathematical problem. Additional approaches based on genetic algorithms and a quadratic residue approach are presented in [11] and [12], respectively.

In MANETconf [13], each configured node maintains state information of the currently assigned addresses so it can choose an available address for a new node and verify its uniqueness throughout the network. In case the newly joining node is unable to find a neighbor, joining node concludes that is the first node in the network and performs address configuration for itself; otherwise it asks one of its neighbors to process its address allocation. The selected neighbor chooses an available address and performs the DAD procedure to verify the uniqueness of the chosen address. Mohsin et al. [14] proposed the IPAA protocol which is based on a dynamic configuration of addresses using the concept of binary split. Each node can independently assign a unique address to a new node without consulting any other node in the MANET. Each node in the network has a disjoint subset of the address space. When a new node joins the network, it tries to find a neighbor node that can perform an address configuration on its behalf. If a neighbor is found, the neighbor splits its available address space and sends one half to the new node. The new node then assigns itself the first IP address in the received address space and keeps the rest to configure other nodes in the future. Both MANETconf and IPAA handle network partitioning and merging as well as address recovery due to node departures. A very similar approach to IPAA, which is based on the binary split idea, is proposed in Tayal et al. [15].

The main contributions in this work are building state diagrams for MANETconf [13] and IPAA [14], deriving analytical formulations for the expectations of latency and communication overhead for IPAA [14], and presenting analytical and simulation results for the performance measures of that protocol for different number of existing nodes in the network.[1]

The paper is organized as follows. In Section 2, a detailed description of the proposed address allocation protocol in [14] is presented; a detailed description of the protocol presented in [13] is omitted due to lack of space. For each protocol, two state diagrams are derived based on the protocol specifications. The state diagrams give a whole picture of the protocol behavior, messages, timers and handshakes. One diagram shows the states of the new node during the address allocation process while the other one shows the states of an existing node that performs address allocation for the new node. In Section 3, analytical formulas for latency and communication overhead for the protocol in [14] are derived. The analytical formulas represent the expected values of the performance measures as a function of the number of nodes in the network and with message loss rate, contention window size, coverage ratio, and counter threshold as parameters. Section 4 presents the analytical as well as the simulation results for different values of the message loss rate, contention window size, and the coverage ratio. Section 5 concludes the paper work and suggests future research.

2 Detailed Descriptions of the IPAA Protocol

The IPAA protocol (and also MANETconf) is designed with the following network operating characteristics in mind. Nodes are free to move in the network and to join and leave at any time. Address allocation and maintenance have to be performed whenever the topology changes. The MANET is configured as a private IPv4 network in which the participating nodes are configured in advance to use a specific private address block. At any given time a group of connected nodes forms a network partition that has a universal unique identifier (UUID). As time evolves, a partition could either split or merge with another partition. The nodes in MANET communicate with each other using IP datagrams. Communications between distant nodes of a partition are carried over intermediate nodes running an ad hoc routing protocol.

IPAA presents a distributed dynamic address allocation protocol for a standalone MANET. The protocol avoids the DAD process by employing a proactive approach using the binary-split idea. The binary-split idea is that each node has a disjoint subset of the address block and it can independently allocate a unique address and hand half of its address space to a newly joining node without getting an agreement from every other node in its partition.

When a new partition is formed, the only node in that partition reserves the whole address block and assigns itself the first address in that block. A newly joining node, a requester (or client), asks an existing neighbor node, an initiator

[1] The protocol proposed in [16] is compared to MANETconf via simulation with latency reported at about one half of that of MANETconf.

(or server), for a network address. The server divides its address space in two and gives one half to the requester. The requester assigns itself the first address from the received address space and keeps the rest of it to serve other nodes in the future. If the initiator has no space left, it borrows an address space from an existing node and forwards it to the requester. This procedure avoids flooding the whole partition to verify the uniqueness of the selected address as DAD based protocols do.

Maintaining the whole Address_Block is the key issue in this protocol. The protocol performs maintenance procedures to avoid address leak and conflict problems. The address leak problem occurs when a node abruptly departs the network or moves out its partition without returning its address space. Without cleaning up departed nodes addresses, these addresses will be considered allocated to existing nodes and can not be allocated to newly joining nodes. Nodes keep track of allocated address blocks to resolve the address leak problem. Graceful departure is provided where nodes that want to leave the network return their address blocks and confirm their departure. Address conflict problem could happen when two partitions merge together since each partition reserve the whole address block for itself. Nodes with the same address in both partitions are allocated the same address space when configured. The conflicting node that has a bigger address space should give up its allocated space and ask other nodes for new allocation.

New Node Address Allocation. Figures 1 and 2 show the state diagrams of a client and a server nodes during address allocation process, respectively. When a client joins the network it broadcasts a one-hop `REQUEST` message. A neighbor replies with a `REPLY` message to the client. The client selects one of its existing neighbors who replied to its request message to be its server. The client sends an `ACK` message to the selected server asking for a unique address. When receiving the `ACK` message, the server starts the allocation process for the requested client. Neighbors are expected to reply to the client request within `REPLY_TIMEOUT` amount of time. If the timer expires without receiving any reply, the client repeats searching for neighbors for `NEIGHBOR_REQUEST_THRESHOLD` number of times. If all trials have failed, the client reserves the whole address block for itself, allocates itself the first address in that block and sets a UUID for that partition.

If reply messages have been received, the client selects one of its neighbors to perform the address allocation process by sending an `ACK` message and starting a timer with a timeout value `ADDRESS_BLOCK_TIMEOUT`. When a node receives the `ACK` message, it divides its available address space into two disjoint subsets and sends one subset to the client in the `ADDRESS_BLOCK` message. The client receives the `ADDRESS_BLOCK` and the partition UUID, configures itself the first address in that block, and keeps the rest of its address space to configure newly joining nodes in the future. The client confirms a successful address allocation by sending `CONFIRM` message back to the server. If the client timer expires before receiving the `ADDRESS_BLOCK` message, the client considers that the server is no longer

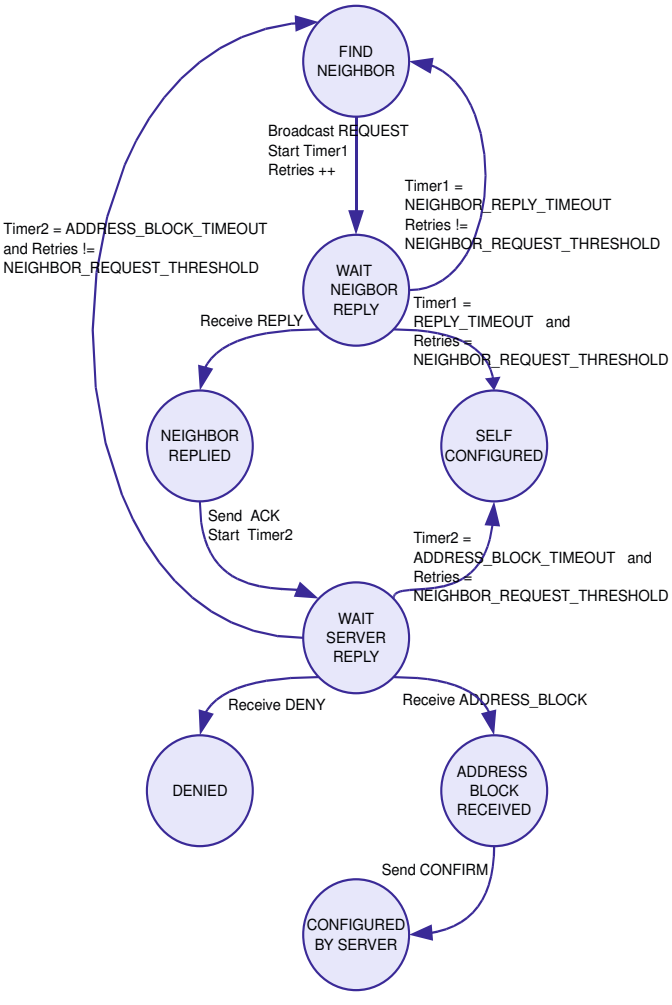

Fig. 1. State diagram of a client node during the address allocation process of IPAA protocol

exist and searches for another server to perform the address allocation process again up to `NEIGHBOR_REQUEST_THRESHOLD` number of times. If `ADDRESS_BLOCK` message has not been received in all trials, the client performs self allocation to configure itself an address as described above.

In case that the selected server has no available addresses to serve the client, the server searches for an existing node in the partition that has an available addresses. For this purpose, each node maintains state information about the allocated address blocks in Allocated_Blocks data set. The Allocated_Blocks set lists the configured nodes in the partition with their available Address_Blocks. The server selects the node with the largest available Address_Block and sends it

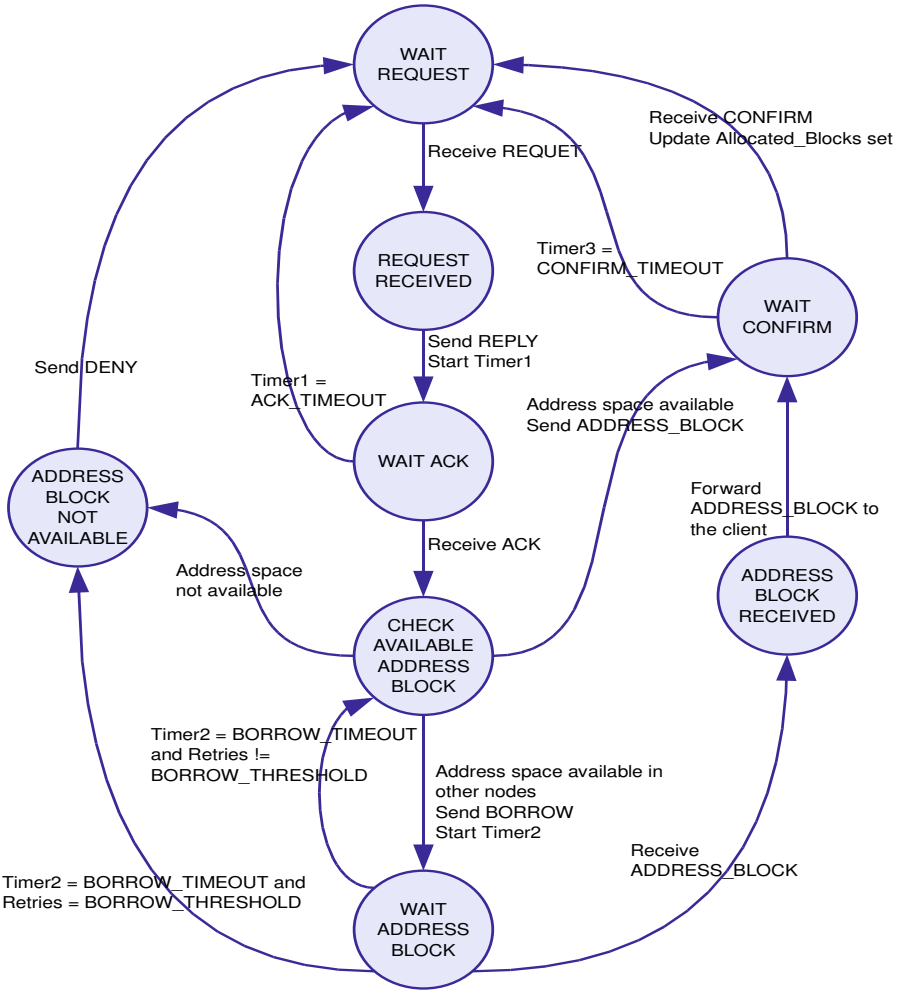

Fig. 2. State diagram of a server node during the address allocation process of IPAA protocol

a `BORROW` message requesting half of its available space. Once the `ADDRESS_BLOCK` message is received from the requested node, the server forwards the message to the client. If all nodes in the server Allocated_Blocks set have no available address space, the server sends a `DENY` message to the client indicating that addresses are not available in this partition.

Nodes could depart the network or move out their partitions at any time. If a node does not respond to the `BORROW` message within a timeout period of `BORROW_TIMEOUT`, the server sends the borrow message to the node who has the second largest address space. The borrow process is repeated up to `BORROW_THRESHOLD` number of times. If all trials have failed or no address space

is available in the rest of the partition nodes, the server sends `DENY` message to the client indicating that addresses are not available in this partition.

Network Partitioning and Merging. Nodes may depart the network or move out their partitions at any time. Departure of a node leads to address leak problem where the nodes address block will not be used by other existing nodes. Each node is responsible for cleaning up the Address_Block of its missing buddy node. To achieve this, the Allocated_Blocks set in each node has to be updated regularly. Each node in the partition broadcasts its Allocated_Blocks set to every other node in the partition. A node updates its Allocated_Blocks set when receiving other nodes sets to keep its information up to date.

Since the state information a node has is assumed up to date, each node looks up its Allocated_Blocks set from time to time to check the existence of its buddy node. If the buddy node is missing from the set, a node claims that its buddy has departed the network. Therefore, it merges its buddy node Address_Block with its block. Each partition has a UUID to be identified from other existing partitions. Partitioning is detected if the node with the lowest address is missing. When detecting the partitioning event, each node sets the partition UUID to the lowest address currently allocated to a node in their partition.

It is possible that a partition merges with another partition in the network. When two configured nodes get close to each other, they exchange their partition UUIDs. If the nodes UUID are different, a merging event is detected. The nodes that detect the merging event exchange their Allocated_Blocks sets. Each node broadcasts the other node's set to all nodes in its partition. When a node receives The Allocated_Blocks set, it searches the set to determine whether a node with the same address exists in the other partition. If an address conflict is detected, the node with the larger address space gives up its address space and asks existing nodes for a new allocation. Merging of two partitions ends when all address conflicts are resolved. The partition UUID maintained by each node is updated to the lowest address allocated in the resulting partition.

3 Analytical Modeling of Address Allocation Protocols

In this Section, we derive analytical expressions of the expected values of latency and communication overhead that are used to evaluate the performance of IPAA [14]. The main objective of such analytical derivations is to obtain mathematical formulations that can clarify the impact of network characteristics and the number of existing nodes in the network on the performance measures of the address allocation process under consideration. The network characteristics that have an impact on the performance measures are the network area, the node's coverage area, collisions and message loss rate. The derived formulas also show the impact of the protocol parameters which are the timeout values and the counter values on the performance measures of the selected protocol.

The derivations of the expected latency and communication overhead are first carried out for small number of existing nodes in the network and then

generalized for an arbitrary number of nodes. Section 3.1 presents the network model under consideration. It defines the network boundary, the node's coverage area, the message loss as well as collision for the incoming traffic at the new node. In Section 3.2, derivations for the expected latency and communication overhead are conducted.

3.1 The Network Model

Let A represent the total area of the network. Each node in the network is located at a random position and nodes are assumed to be uniformly distributed over the network area. Nodes are assumed to have a coverage area of A_x. Let $\tilde{n}$ denote the number of existing nodes in the network at the time a new node wishes to join. Let $\tilde{x}_i$ indicate the presence of node i, $i = 1, 2, \cdots, n$, within the transmission range of the new node; that is, $\tilde{x}_i = 1$ is a neighbor of the newly entering node, and $\tilde{x}_i = 0$ otherwise.

Since the existing nodes are uniformly distributed over the network area independent of the placement of all other nodes, $\tilde{x}_i$, $i = 1, 2, \cdots, n$ is set of identical, independent, Bernoulli trials with success probability $\frac{A_x}{A}$. Thus, the number of existing nodes that are within the transmission range of the new node, denoted as $\tilde{x}$, is a binomial random variable with parameters $\left(n, \frac{Ax}{A}\right)$ and

$$\mathrm{E}[\tilde{x}|\tilde{n} = n] = n\,\mathrm{P}\left\{\tilde{x}_i = 1|\tilde{n} = n\right\} = n\,\frac{A_x}{A}. \tag{1}$$

The new node communicates with a neighbor node over a wireless channel. A bad channel condition results in a low signal to interference plus noise ratio (SINR) for the received message and the message is considered lost if its SINR is below a given threshold. Assume that the wireless channel between the new node and a neighbor node has a message loss rate equals to ϵ. To avoid collision, each node in the network has a contention window of size W. A node that has data to send chooses a random slot number uniformly from $\{1, 2, \cdots, W\}$ and sends its data in that selected slot. Collision in a given slot could happen if two or more nodes transmit in the same slot despite the SINR value of their messages. A message is received successfully if it does not collide with other messages in the transmission slot and has a good SINR value.

3.2 Expectations of the Performance Measures

The amount of time for the new node to be configured with a network address is denoted as $\tilde{\ell}$ and the number of messages sent during the address allocation process is denoted as $\tilde{c}$. The derivations for the expected values of latency and communication overhead for a given number of nodes, $\tilde{\ell}|\tilde{n} = n$ and $\tilde{c}|\tilde{n} = n$, respectively, are first constructed for the case $\tilde{n} = 0$ and then generalized to an arbitrary number of nodes $\tilde{n} = n$. Due to space limitations, only a sketch of the developments are presented here, and the interest reader is referred to [17].

3.3 The $\tilde{n} = 0$ Case

In this case the new node is the only node in the network. The new node starts by broadcasting a `REQUEST` message. Since there are no nodes in the neighborhood, the timer for receiving the `REPLY` message will reach the timeout value $T_{\texttt{NR}}$ without receiving any `REPLY` message. Since no `REPLY` messages have been received during the first timeout period, the new node sends another `REQUEST` message and waits for timeout. The new node will repeat sending the `REQUEST` message and waits for timeout for a maximum of $K_{\texttt{NR}}$ trials. After the last timeout period, the new node considers itself the first node in the network and performs address configuration itself. It allocates the whole address space for itself and assigns itself the first address in that space.

The expected latency for a new node to be allocated a network address, denoted as $\mathrm{E}\left[\tilde{\ell}|\tilde{n}=0\right]$, and the expected communication overhead, denoted as $\mathrm{E}\left[\tilde{c}|\tilde{n}=0\right]$, for this case are easily found to be

$$\mathrm{E}\left[\tilde{\ell}\,|\,\tilde{n}=0\right] = K_{\texttt{NR}}\,T_{\texttt{NR}} \quad \text{and} \quad \mathrm{E}\left[\tilde{c}\,|\,\tilde{n}=0\right] = K_{\texttt{NR}}. \tag{2}$$

3.4 The General Case $\tilde{n} = n$

In this section we derive formulas for the expectations of latency and communication overhead for the IPAA protocol in the general case where the number of nodes in the network is $\tilde{n} = n$. Since this protocol performs address allocation through local communications with the new node neighbors, the expectations are derived by conditioning first on the number of nodes that are within the transmission range of the new node. The general formulas for the expected latency and communication overhead are then given by

$$\mathrm{E}\left[\tilde{\ell}\,|\,\tilde{n}=n\right] = \sum_{x=0}^{n} \mathrm{E}\left[\tilde{\ell}\,|\,\tilde{x}=x\right]\ \mathrm{P}\left\{\tilde{x}=x\,|\,\tilde{n}=n\right\}, \tag{3}$$

and

$$\mathrm{E}\left[\tilde{c}\,|\,\tilde{n}=n\right] = \sum_{x=0}^{n} \mathrm{E}\left[\tilde{c}\,|\,\tilde{x}=x\right]\ \mathrm{P}\left\{\tilde{x}=x\,|\,\tilde{n}=n\right\}, \tag{4}$$

where $\tilde{x}$ is a binomial random variable with parameters $\left(n, \frac{A_x}{A}\right)$.

For the special case when there is no node within the transmission range of the new node, that is $\{\tilde{x}=0\}$, the expected latency and communication overhead are obtained from (2) and are given by

$$\mathrm{E}[\tilde{\ell}\,|\,\tilde{x}=0] = K_{\texttt{NR}}\,T_{\texttt{NR}} \quad \text{and} \quad \mathrm{E}[\tilde{c}\,|\,\tilde{x}=0] = K_{\texttt{NR}}. \tag{5}$$

For all other cases, where $\tilde{x} > 0$, the expectations for the latency and communication overhead for a given number of neighbors are calculated by conditioning

on the number of trials, denoted by $\tilde{\kappa}$, required to successfully allocate a network address to the new node. Conditioning on the value of $\tilde{\kappa}$ yields

$$\begin{aligned}\mathrm{E}[\tilde{\ell} \mid \tilde{x} = x] = & \sum_{\kappa=1}^{K_{\mathtt{NR}}} \mathrm{E}[\tilde{\ell} \mid \tilde{x} = x, \tilde{\kappa} = \kappa] \, \mathrm{P}\{\tilde{\kappa} = \kappa \mid \tilde{x} = x\} + \\ & \mathrm{E}[\tilde{\ell} \mid \tilde{x} = x, \tilde{\kappa} > K_{\mathtt{NR}}] \, \mathrm{P}\{\tilde{\kappa} > K_{\mathtt{NR}} \mid \tilde{x} = x\}, \end{aligned} \tag{6}$$

and

$$\begin{aligned}\mathrm{E}[\tilde{c} \mid \tilde{x} = x] = & \sum_{\kappa=1}^{K_{\mathtt{NR}}} \mathrm{E}[\tilde{c} \mid \tilde{x} = x, \tilde{\kappa} = \kappa] \, \mathrm{P}\{\tilde{\kappa} = \kappa \mid \tilde{x} = x\} + \\ & \mathrm{E}[\tilde{c} \mid \tilde{x} = x, \tilde{\kappa} > K_{\mathtt{NR}}] \, \mathrm{P}\{\tilde{\kappa} > K_{\mathtt{NR}} \mid \tilde{x} = x\}. \end{aligned} \tag{7}$$

The κ-th trial is successful if an `ADDRESS_BLOCK` message is received from a neighbor node in response to the `ACK` message. The `ADDRESS_BLOCK` message contains a disjoint subset of the address space where the new node selects the first address from that space for itself and keeps the rest for serving other joining nodes in future. Assuming that successive trials are independent and identical, the probability that a successful address allocation occurs on the κ-th trial is given by

$$\mathrm{P}\{\tilde{\kappa} = \kappa \mid \tilde{x} = x\} = (\mathrm{P}\{\tilde{a} = 0 \mid \tilde{x} = x\})^{\kappa - 1} \, \mathrm{P}\{\tilde{a} = 1 \mid \tilde{x} = x\}, \tag{8}$$

where $\tilde{a} = 0$ is the indicator random variable for the event of a successful allocation. As discussed earlier in previous sections, the trial fails if the `ADDRESS_BLOCK` message is not received when at least one `REPLY` message is received or when no `REPLY` message is received. The probability of a failed trial in presence of x neighbors is given by

$$\begin{aligned}\mathrm{P}\{\tilde{a} = 0 | \tilde{x} = x\} &= \mathrm{P}\{\tilde{a} = 0 | \tilde{x} = x, \tilde{s} = 0\} \, \mathrm{P}\{\tilde{s} = 0 | \tilde{x} = x\} + \\ &\quad \mathrm{P}\{\tilde{a} = 0 | \tilde{x} = x, \tilde{s} = 1\} \, \mathrm{P}\{\tilde{s} = 1 | \tilde{x} = x\} \\ &= \mathrm{P}\{\tilde{s} = 0 | \tilde{x} = x\} + (\epsilon + (1 - \epsilon)\, \epsilon) \, \mathrm{P}\{\tilde{s} = 1 | \tilde{x} = x\}, \end{aligned} \tag{9}$$

where $\tilde{s}$ is the indicator random variable for the event of the reception of at least one reply message from a neighbor node and that message is transmitted with no other messages in a slot and has a good SINR value.

Define $\tilde{r}$ to be the number of responders who received a `REQUEST` message and $\tilde{r}_i$ to be the number of responders who responded in slot number i. $i \in \{1, 2, \cdots, W\}$. Then the probability of receiving at least one `REPLY` message successfully given that there are x neighbors is given by

$$\mathrm{P}\{\tilde{s} = 1 | \tilde{x} = x\} = \sum_{r=1}^{x} \mathrm{P}\left\{ \bigcup_{i=1}^{W} \{\tilde{r}_i = 1\} | \tilde{r} = r \right\} \mathrm{P}\{\tilde{r} = r | \tilde{x} = x\}. \tag{10}$$

The first term of the equation represents the probability that at least one transmission slot has exactly one `REPLY` message and that message has a good SINR value. The second term represents the probability that $\tilde{r}$ neighbors received the `REQUEST` message out of the total number of neighbors. Since message transmissions are assume to have identical failure probabilities of ϵ and messages are transmitted independently, $\tilde{r}$ is a binomial random variable with parameters $(x, (1-\epsilon))$.

The probability that at least one slot has exactly one reply message given that there are r responders is equal to [18]

$$\mathrm{P}\left\{\bigcup_{i=1}^{W}\{\tilde{r}_i = 1\} \mid \tilde{r} = r\right\} = \sum_{i} \mathrm{P}\left\{\tilde{r}_i = 1 \mid \tilde{r} = r\right\} - \sum_{i<j} \mathrm{P}\left\{\tilde{r}_i = 1, \tilde{r}_j = 1 \mid \tilde{r} = r\right\} + \sum_{i<j<k} \mathrm{P}\left\{\tilde{r}_i = 1, \tilde{r}_j = 1, \tilde{r}_k = 1 \mid \tilde{r} = r\right\} - \cdots + (-1)^{W+1} \mathrm{P}\left\{\tilde{r}_i = 1, \tilde{r}_j = 1, \tilde{r}_k = 1, \cdots, \tilde{r}_W = 1 \mid \tilde{r} = r\right\}. \tag{11}$$

The first term of (11) represents the probability that slot i has exactly one message, in other words, exactly one responder transmitted the message in slot number i and that message when received had a good SINR value. That is

$$\mathrm{P}\left\{\tilde{r}_i = 1 | \tilde{r} = r\right\} = \binom{r}{1} \frac{1}{W} \left(1 - \frac{1}{W}\right)^{r-1} (1-\epsilon). \tag{12}$$

The other terms of (11) represent joint probabilities of having more than one slot with exactly one message. In general, the joint probability of having exactly one reply in each of the slots $i, j, \cdots, z$ where the $\tilde{r}_i + \tilde{r}_j + \cdots + \tilde{r}_z = h$, is equal to zero for $h > r$ and for the $h \leq r$ case it is calculated as

$$\mathrm{P}\left\{\tilde{r}_{i_1} = 1, \cdots, \tilde{r}_{z_h} = 1 | \tilde{r} = r\right\} = \binom{r}{h} h! \, \frac{(W-h)^{r-h}}{(W)^r} (1-\epsilon)^h. \tag{13}$$

Substituting this result in (11) and performing some algebra yields

$$\mathrm{P}\left\{\bigcup_{i=1}^{W}\{\tilde{r}_i = 1\} \mid \tilde{r} = r\right\} = \sum_{h=1}^{\min\{r,W\}} (-1)^{h+1} \binom{W}{h} \binom{r}{h} h! \, \frac{(W-h)^{r-h}}{W^r} (1-\epsilon)^h. \tag{14}$$

Now, we consider the conditional expectations of the latency and communication overhead on the number of trials required for the new node to be allocated a network address. Recall that $\tilde{\kappa}$ represents the number of trials that has been performed, the expected latency when the address allocation succeeded on the κth trial, where $\kappa \leq K_{\mathtt{NR}}$ is the sum of of the timeouts for the first $(\kappa - 1)$ failed trials plus the timeout of receiving a `REPLY` message and the timeout of receiving the `ADDRESS_BLOCK` message in the last successful trial. The latency for a failed trial is the sum of the timeout to receive the `REPLY` message plus the timeout to

receive the ADDRESS_BLOCK message if at least one reply is successfully received. That is

$$\begin{aligned}\mathrm{E}[\tilde{\ell}|\tilde{x}=x,\tilde{\kappa}=\kappa] &= (\kappa-1)\,(T_{\mathtt{NR}}+\ \mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\}\ T_{\mathtt{AR}})+T_{\mathtt{AR}}+T_{\mathtt{AR}}\\ &= \kappa\,T_{\mathtt{NR}}+T_{\mathtt{AR}}\,(1+(\kappa-1)\,\mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\}),\end{aligned}\tag{15}$$

where $\mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\}$ is presented in (10).

The expected communication overhead for a successful address allocation on the κ-th trial is the sum of messages sent during the first $(\kappa-1)$ trials and they are one REQUEST message, the expected number of REPLY messages and one ACK message that is sent if at least one reply is received correctly. In the last trial, the new node sent a REQUEST and received at least one REPLY so it sent out an ACK message to one of the responders and received an ADDRESS_BLOCK message that contains a portion of the address space. Therefore, the expected communication overhead is equal to

$$\begin{aligned}\mathrm{E}[\tilde{c}|\tilde{x}=x,\tilde{\kappa}=\kappa] &= (\kappa-1)\,(1+\mathrm{E}[\tilde{r}|\tilde{x}=x]+\ \mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\})+\\ &\quad (1+\mathrm{E}[\tilde{r}|\tilde{x}=x]+2)\\ &= \kappa\,(1+\mathrm{E}[\tilde{r}|\tilde{x}=x])+(\kappa-1)\,\mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\}+2.\end{aligned}\tag{16}$$

In the case where all trials have failed, that is $\kappa > K_{\mathtt{NR}}$, the expected latency and communication overhead are given by

$$\mathrm{E}[\tilde{\ell}|\tilde{x}=x,\tilde{\kappa}>K_{\mathtt{NR}}] = K_{\mathtt{NR}}\,(T_{\mathtt{NR}}+\ \mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\}\ T_{\mathtt{AR}}),\tag{17}$$

and

$$\mathrm{E}[\tilde{c}|\tilde{x}=x,\tilde{\kappa}>K_{\mathtt{NR}}] = K_{\mathtt{NR}}\,(1+\mathrm{E}[\tilde{r}|\tilde{x}=x]+\ \mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\}).\tag{18}$$

Substituting (8), (15), (16), (17) and (18) into (6) and (7) gives

$$\begin{aligned}\mathrm{E}[\tilde{\ell}\mid\tilde{x}=x] = \sum_{\kappa=1}^{K_{\mathtt{NR}}} &\kappa\,T_{\mathtt{NR}}+T_{\mathtt{AR}}\,(1+(\kappa-1)\,\mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\})\\ &(\mathrm{P}\,\{\tilde{a}=0\mid\tilde{x}=x\})^{\kappa-1}\,\mathrm{P}\,\{\tilde{a}=1\mid\tilde{x}=x\}+\\ &K_{\mathtt{NR}}\,(T_{\mathtt{NR}}+\ \mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\}\ T_{\mathtt{AR}})\,(\mathrm{P}\,\{\tilde{a}=0\mid\tilde{x}=x\})^{K_{\mathtt{NR}}},\end{aligned}$$

and

$$\begin{aligned}\mathrm{E}[\tilde{c}\mid\tilde{x}=x] = \sum_{\kappa=1}^{K_{\mathtt{NR}}} &(\kappa\,(1+\mathrm{E}[\tilde{r}|\tilde{x}=x])+(\kappa-1)\,\mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\}+2)\\ &(\mathrm{P}\,\{\tilde{a}=0\mid\tilde{x}=x\})^{\kappa-1}\,\mathrm{P}\,\{\tilde{a}=1\mid\tilde{x}=x\}+\\ &K_{\mathtt{NR}}\,(1+\mathrm{E}[\tilde{r}|\tilde{x}=x]+\ \mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\})\,(\mathrm{P}\,\{\tilde{a}=0\mid\tilde{x}=x\})^{K_{\mathtt{NR}}},\end{aligned}\tag{19}$$

where the probabilities $\mathrm{P}\,\{\tilde{a}=0\mid\tilde{x}=x\}$ and $\mathrm{P}\,\{\tilde{s}=1|\tilde{x}=x\}$ are derived in (9) and (10), respectively.

4 Numerical Results

In this section, we present numerical results for latency and communication overhead based on analysis and simulation, and in addition, we consider the sensitivity of these measures to message loss rate, contention window size and coverage ratio. in Subsection 4.1, a description of the simulation carried out for the IPAA and MANETconf protocols is presented. Then, Subsection 4.2 presents a comparison of the IPAA and MANETconf protocols. Subsection 4.3 presents the analytical and simulation results, which show good agreement, for the IPAA protocol as functions of network size with different parameter values. A summary of results is deferred to the conclusion section.

4.1 Simulation Description

The network dimensions are set to $100m \times 100m$. Each node is placed at a uniformly distributed location within the network area. Results are collected for node populations from 0 to 50. For a given number of nodes in the network, the number of nodes within the transmission range of the new node depends on its coverage area. Results are collected for coverage ratios 10%, 15%, and 20%. Each message sent from one node to another is subject to loss with rate ϵ; loss rates examined were 0, 0.1, and 0.2. Contention window sizes, W, considered were 10, 20, and 30. For the IPAA protocol, the timeout value to find a neighbor $T_{\texttt{NR}}$ was set to 0.2 seconds and the timeout to receive the address_block message $T_{\texttt{AR}}$ was set to 0.02 seconds, and $K_{\texttt{NR}}$ was varied from 3 to 5 to show the effect of the counter threshold on the performance measures of the protocol. For MANETconf protocol $T_{\texttt{NR}}$ was set to 0.2 seconds and $T_{\texttt{AR}}$ was set to 2 seconds, and $K_{\texttt{NR}}$ was set to 3 and $K_{\texttt{AR}}$ was set to 5.

The simulation results presented are the average values of 10000 simulation runs. For large number of simulation runs, the sample mean of a performance measure for any network size follows the normal distribution $N(\mu, \frac{s}{\sqrt{n}})$, where μ is the true population mean, s is the sample standard deviation, and n is the sample size. Throughout simulation, we have found that the maximum value of the sample standard deviation is 0.002 for latency samples and it is 0.1 for communication overhead samples for all values of network size. The 95% confidence interval, that is likely to include the true population mean of a performance measure, is equal to $(\bar{a} \pm 1.96\ \frac{s}{\sqrt{n}})$, where $\bar{a}$ represents the sample mean for the performance measure. Therefore, the 95% confidence interval for latency samples is $(\bar{x} - 3.92 \times 10^{-5},\ \bar{x} + 3.92 \times 10^{-5})$ and for communication overhead it is $(\bar{y} - 0.002,\ \bar{y} + 0.002)$.

4.2 IPAA vs. MANETconf Protocol

Figure 3 compares the latency of IPAA protocol to the MANETconf protocol latency. In MANETconf, the initiator node selects an address and performs DAD process throughout the network. Therefore, the latency of the allocation process

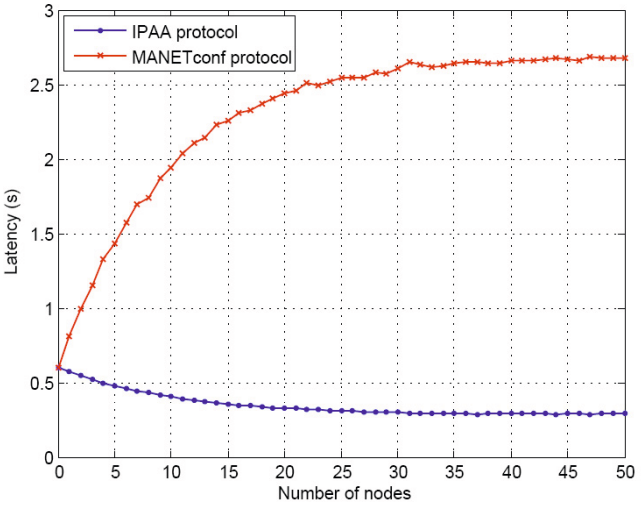

Fig. 3. Latency of the IPAA and MANETconf protocols as a function of network size

increases as the number of node in the network increases. In IPAA protocol, the address allocation process is carried out by an existing neighbor or it may be carried out by the new node itself in case no neighbor exists. For large number of nodes, the probability of finding a node within the transmission range of the new node is high so that address allocation can be carried out by a neighbor node and therefore it decreases as the number of nodes increases.

Figure 4 presents the communication overhead for IPAA and MANETconf protocols. Since MANETconf performs the DAD process throughout the network, the number of messages sent increases as the number of nodes increases. The IPAA protocol performs address allocation through local communications with neighbor nodes only. Therefore, the number of messages sent is less than that for the MANETconf protocol.

4.3 Analytical and Simulation Results for the Performance Measures of the IPAA Protocol

Figures 5 shows the latency at loss rates $\epsilon = \{0, 0.1, 0.2\}$. The contention window size is set to $W = 30$ and the coverage ratio is set to $\frac{A_x}{A} = 10\%$. The results show that the latency increases as the loss rate increases since a loss of the protocol messages may results in a failed trial and force the new node to start the process again. Communications overhead was also analyzed. For relatively small number of nodes in the network, the communication overhead increases as the loss rate increases since there exists a small number of neighbor nodes within the transmission range of the new node so the probability of loss of the protocol message is high. For large number of neighbors, the probability of loss for all messages is lower and the number of messages becomes closer to the case

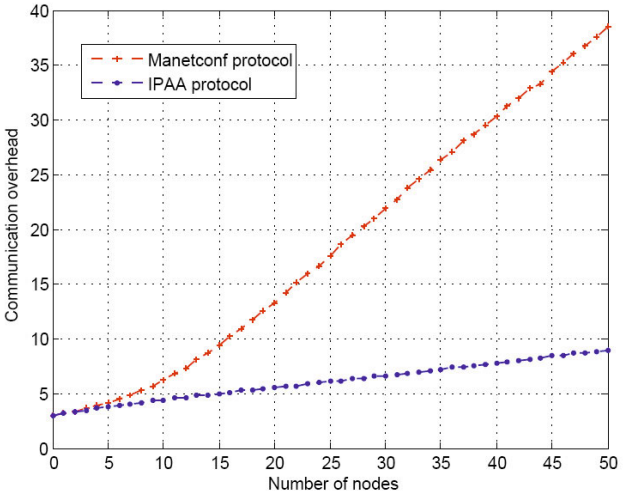

Fig. 4. Communication overhead of the IPAA and MANETconf protocols as a function of network size

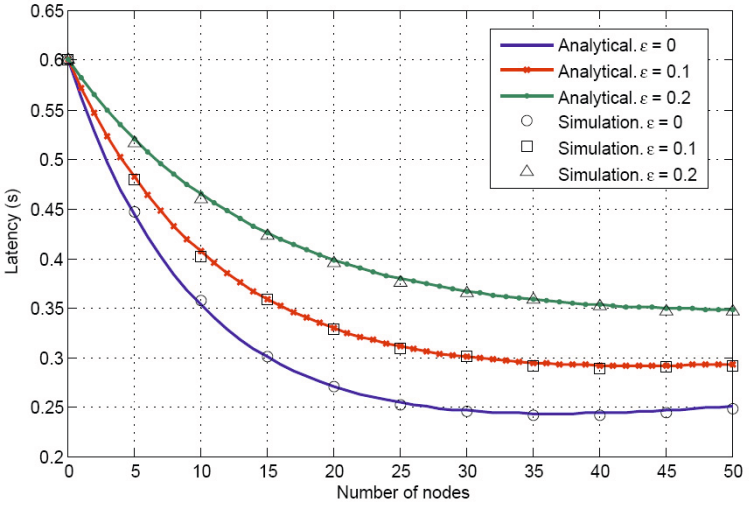

Fig. 5. Latency of the IPAA protocol as a function of network size with message loss rate as a parameter

where no loss is assumed. The graph showing these results is omitted because it is so similar to the results shown in Figure 4.

Figures 6 and 7 show the effect of the contention window size W on the latency and the communication overhead, respectively when $\epsilon = 0.1$, $\frac{A_x}{A} = 10\%$ and $K_{\mathtt{NR}} = 3$. The window size has a greater effect on latency and communication

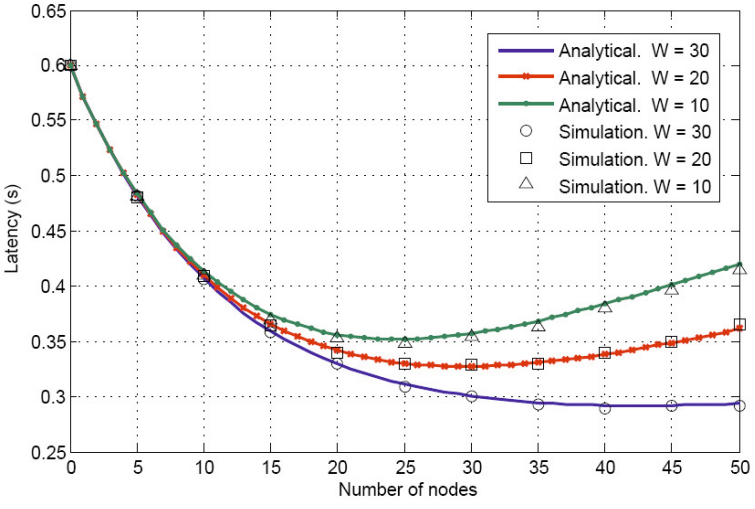

Fig. 6. Latency of the IPAA protocol as a function of network size with contention window size as a parameter

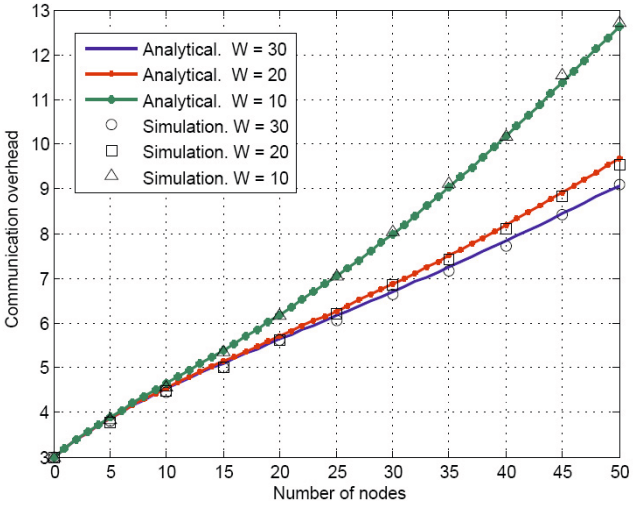

Fig. 7. Communication overhead of the IPAA protocol as a function of network size with contention window size as a parameter

overhead for a large number of nodes since the collision probability increases as the number of nodes increases. For a given large number of nodes, decreasing the window size results in more collisions for the incoming reply messages at the new node which may results in failed allocation trial. More failed trials results in more latency and more communication overhead as shown in the figures.

Numerous additional analytical and simulation results were carried out and are given in [17], but are omitted here due to lack of space. Also analyzed were the the effect of the coverage ratio $\frac{A_x}{A}$ on the latency and communication overhead, effect of the protocol parameter $K_{\mathtt{NR}}$ on the latency and communication overhead, and effect of the loss ratio on the number of neighbor allocations. In all cases analytical and simulation results showed close agreement.

5 Conclusions

Two address allocation protocols in wireless ad hoc network were discussed. The MANETconf protocol [13], represents the DAD-based address allocation scheme while the IPAA protocol [14], represents the neighbor-based scheme. A state machines that shows the behavior was shown for the IPAA protocol while that for the MANETconf was deferred to [17]. The state machine gives a complete picture of the states, handshakes, timers, and types of messages for a protocol. For IPAA, analytical formulas for the expectation of latency and communication overhead are carried out as a function of the number of nodes in the network with message loss rate, contention window size, coverage ratio, and the counter threshold as parameters.

Extensive numerical results were collected. The results show that the latency and communication overhead for MANETconf are higher than the measures of the IPAA protocol and that the increase results from performing the DAD process in MANETconf. For IPAA, it has been shown that the latency and communication overhead of the allocation process increase as the message loss rate increases. The contention window size has an effect on the performance measures for a relatively large number of nodes in the network. Extensive additional numerical results and conclusions are given in [17]. As an example, numerical results not presented here show that for loss rate of 0.1, contention window size of 30, and coverage ratio of 10%, address allocation is carried out by neighbor nodes for almost 95% of the time when the number of nodes in the network more than 30.

The address allocation problem for ad hoc networks is still unsolved. The work done in this paper provides understanding of the nature of the problem and the way to evaluate the performance of an allocation protocol. What is needed is a protocol that handles all types of exceptions such as message losses, node departures, network partitioning, and merging. The performance of the protocol should be analyzed and compared to the performance of existing protocols.

References

1. Baccelli, E.: Address autoconfiguration for MANET: Terminology and problem statement, IETF draft-ietf-autoconf-statement-02 (September 2007)
2. Knowlton, K.C.: A fast storage allocator. Commun. ACM 8(10), 623–624 (1965)
3. Bernardos, C., Calderon, M., Moustafa, H.: Survey of IP address autoconfiguration mechanisms for MANETS, IETF draft-bernardos-manet-autoconf-survey-03 (April 2008)

4. Perkins, C.: IP address autoconfiguration for ad hoc networks, IETF draft-perkins-manet-autoconf-01 (November 2001)
5. Jeong, J.: Ad hoc IP address autoconfiguration, IETF draft-jeong-adhoc-ip-addr-autoconf-06 (January 2006)
6. Vaidya, N.: Weak duplicate address detection in mobile ad hoc networks. In: Proc. of ACM MOBIHOC, Lausanne (2002)
7. Weniger, K.: Passive duplicate address detection in mobile ad hoc networks. In: Proc. of IEEE WCNC (March 2003)
8. Mase, K., Adjih, C.: No overhead autoconfiguration OLSR, IETF draft-mase-manet-autoconf-noaolsr-01 (April 2006)
9. Weniger, K.: PACMAN: Passive autoconfiguration for mobile ad hoc networks. IEEE Journal on Selected Areas in Communications 23(3), 507–519 (2005)
10. Zhou, H., Ni, L., Mutka, M.: Prophet address allocation for large scale MANETs. In: Proceedings of INFOCOM (2003)
11. Yong, L., Ping, Z., JiaXiong, L.: Dynamic address allocation protocols for mobile ad hoc networks based on genetic algorithm. In: Proc. of 5th International Conference on Wireless Communications, Networking and Mobile Computing (2009)
12. Chu, X., Sun, Y., Xu, K., Sakander, Z., Liu, J.: Quadratic residue based address allocation for mobile ad hoc networks. In: Proc. IEEE ICC (2008)
13. Nesargi, S., Parakash, R.: MANETconf: Configuration of hosts in a mobile ad hoc network. In: Proc. IEEE INFOCOM. IEEE Press, New York (2002)
14. Mohsin, M., Parakash, R.: IP address assignment in a mobile ad hoc network. In: Proc. IEEE MILCOM, IEEE Press, New York (2002)
15. Tayal, A.P., Patnaik, L.M.: An address assignment for the automatic configuration of mobile ad hoc networks. Personal Ubiquitous Comput. 8(1), 47–54 (2004)
16. Xu, T., Wu, J.: Quorum based ip address autoconfiguration in mobile ad hoc networks. In: ICDCSW 2007: Proceedings of the 27th International Conference on Distributed Computing Systems Workshops, p. 1. IEEE Computer Society, Washington (2007)
17. Radaideh, A.M.: Analytical modeling of address allocation protocols in wireless ad hoc networks, Master's thesis, The University of Mississippi (December 2008)
18. Ross, S.M.: Introduction to Probability Models, 9th edn. Academic Press, London (2008)