ZAL: Zero-Maintenance Address Allocation in Mobile Wireless Ad Hoc Networks

Zhihua Hu, Baochun Li
Department of Electrical and Computer Engineering
University of Toronto
{*zhu,bli*}*@iqua.ece.toronto.edu*

***Abstract*—**
The allocation of IP addresses in hybrid wireless networks is one of the most critical issues in all-IP converged wireless networks. The reason is that centralized IP address allocation mechanisms may not be available in networks comprised of heterogeneous mobile wireless devices. In this paper, we propose Zero-Maintenance Address Allocation (ZAL), a fully distributed address allocation algorithm with extremely low communication overhead. ZAL outperforms existing solutions in many important aspects, and eliminates permanent duplication of IP addresses. ZAL is completely free from periodical maintenance messages, timeouts, delays, and modification of existing network protocols. Theoretically, we prove that ZAL suffers negligible probability of temporary address duplication, while minimizing the usage of address space. In our experiments, we can successfully allocate addresses to a network of 480 nodes with no duplication of IP addresses when the size of available address space is 1024 (2^{10}) or larger. Even for an available address space of size 512, in average, temporary address duplication can be resolved within 60 seconds after the node joins the network.

I. Introduction and Motivation

As the Internet Protocol (IP) emerges as the universal long-term solution to bring diversely different wireless technologies together (e.g., Wireless Local Area Networks (WLAN), Personal Area Networks (PAN) and Cellular Packet Networks), IP-based mobile devices are able to roam across heterogeneous wireless systems transparently and effortlessly [1]. Towards this objective, a few basic questions need to be addressed appropriately, one of which being the allocation of unique IP addresses to all mobile hosts. Though address allocation is straightforward with a centralized authority (e.g., DHCP servers), it is much more complicated when mobile hosts move into a self-organizing *wireless ad hoc network*, where centralized infrastructure does not exist.

As an example illustrated in Fig. 1, the mobile device c_2 is equipped with air interfaces for both cellular (GPRS) and PAN (Bluetooth) connections. However, mobile devices a_1 to a_6 are equipped with air interfaces for WLAN only, with a_0 supporting a PAN connection as well. In this case, nodes a_0 to a_6 have to rely on c_2 to acquire IP addresses and communicate with Internet. The centralized authority from the base station is not available to assign IP addresses to these mobile devices. In this case, devices a_0 to a_6 form an ad hoc network within the hybrid wireless network infrastructure.

The issues of interface incompatibility aside, mobile wireless ad hoc networks may also be used to improve the performance or robustness of cellular packet networks [2], [3], [4], [5], [6].

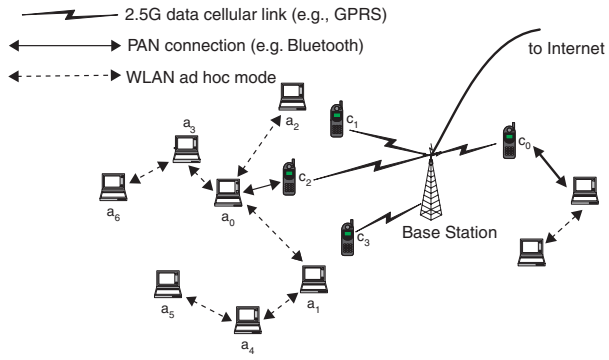

Fig. 1. The issue of IP address allocation in all-IP hybrid wireless networks.

In such situations, some of the mobile devices within the cellular cell may not be able to communicate with the base station directly to acquire IP addresses. The cause is the lack of centralized network management infrastructure to allocate IP addresses in wireless ad hoc networks.

In previous work [7], [8], [9], [10], [11], [12], [13], efforts have been made to assign IP addresses within wireless ad hoc networks. However, there exist major drawbacks in the existing solutions proposed in these papers. These include flooding messages, non-trivial timeout and delay, protocol modification, overhead of periodical maintenance messages, and severe address leaks. In addition, existing solutions suffer from two critical problems: (1) *Address duplication*. Duplicate addresses may be allocated due to inconsistent states and the merging of network partitions (which is henceforth referred to as a *merger*). Strong Duplicate Address Detection (DAD) mechanisms are required to resolve such duplications. Besides the high overhead associated with DAD, Vaidya *et al.* [8] has further shown that it is impossible to guarantee DAD within a bounded time if message delays between at least one pair of nodes in the network are unbounded. (2) *Poor scalability*. In previous work, all messages exchanged in the existing solutions are multi-hop $(\sqrt{n})$ messages that grow with the total number of network nodes n. This results in very poor scalability. The complication of address duplication and poor scalability will eventually paralyze the network and lead to severe address leaks.

In this paper, we propose *Zero-Maintenance Address Allocation* (ZAL), a simple and fully distributed algorithm that solves all the aforementioned problems presented by existing solutions. Particularly, ZAL ensures the uniqueness of the allocated

addresses (except for the situation of address space depletion, which results in *temporary* IP address duplication that happens with low probability and can be resolved within short period of time) with very little overhead. A comparison of scalability between ZAL and representative existing solutions is presented in Table I.

TABLE I
COMPARISONS BETWEEN ZAL AND EXISTING SOLUTIONS (NUMBER OF ONE-HOP MESSAGES)

.

Operation	*MANETconf*[7]	*Buddy*[13]	*ZAL*
Basic joining messages	$\Omega(n\sqrt{n})$	$O(n\sqrt{n})$	$\Theta(1)$
Maintenance messages for stable network	$\Omega(n\sqrt{nt})$	$\Omega(n\sqrt{nt})$	0
Graceful departure	$\Omega(n)$	$\Theta(1)$	$\Theta(1)$
Partition and merger	$\Omega(n)$ per partition and merger	$O(n^2)$ per partition and merger	0
Data structure at each node	$\Omega(n)$	$\Omega(n)$	$\Theta(1)$
Address duplication	permanent and severe	permanent and severe	none
Address leak	severe	severe	negligible

The remainder of the paper is structured as follows. Sec. II discusses related work. Sec. III presents the design and analysis of our algorithm. Sec. IV confirms our claim through extensive simulations and further analysis. Finally, Sec. V concludes the paper.

II. RELATED WORK

Previous work [8], [9], [10], [11], [12] have examined the issue of address allocation in wireless ad hoc networks. All solutions proposed in these papers require heavy maintenance to address the problems of permanent address duplications. We study several prominent examples.

MANETconf [7] has proposed a scheme to allocate addresses in mobile ad hoc networks. In MANETconf, the assignment of an address to a new node must be confirmed by all the nodes in the network. This process may cause significant overhead. Compared with MANETconf, our algorithm is fully distributed and does not require the confirmation from other nodes in the network when an address is assigned. Our strength comes from the fact that we delegate the address space to each node with no overlap. Each node can then accomplish the assignment task independently. Furthermore, as shown in Sec. III-C, [7] suffers from serious complication of duplicate addresses and unscalable protocol overhead. The network will eventually be paralyzed when the number of nodes in the network approaches the total IP address capacity. As a result, a significant number of IP addresses can not be allocated and thus leads to severe address leaks.

Vaidya *et al.* [8] proposes to modify the routing table in routing protocols. The solution from Patchipulusu [10] needs the selection of clusterhead nodes and extensive maintenance messages. Boleng [12] proposes to modify the IP header of other protocols. It also uses timeouts in the procedure to detect duplicate addresses. Nesargi *et al.* [11] also needs to modify the IP header of the messages to include unique identifiers. Perkins *et al.* [9] needs to send a 'route request' to test if the randomly selected address has been used.

Mohsin *et al.* [13] proposed Buddy system for IP address allocation. At the first glance, Buddy is similar to ZAL in some basic aspects. However, ZAL is fundamentally different from Buddy in design philosophy and many other important aspects. ZAL is designed to eliminate the need for Duplicate Address Detection (DAD) mechanisms, whereas Buddy relies on strong DAD. ZAL is fine tuned to eradicate address duplication in every aspect of all protocols. Buddy suffers address duplication in address allocation protocol, address leak detection protocol and merger of partitions. For instance, in Buddy system, the server retains the address block assigned to the client if the server fails to receive the Confirm message from the client due to unreliable communication channel. Buddy does not specify the details of how a server will handle Borrow message from servers. If protocols similar to client-server address allocation process is employed, address duplication will occur, exasperated by the potential $\sqrt{n}$-hops messages between servers. Buddy requires heavy maintenance protocols to synchronize state information amongst all network nodes. In contrast, ZAL incurs no overhead in a stable network. Further, in Buddy system, high probability of address duplication and heavy maintenance overhead can lead to severe address leak, as shown in Sec. III-C.

McAuley *et al.* [14], [15] discusses the allocation of address by dividing the address blocks among network nodes. However, these works are not specifically designed to handle the complex situation in mobile ad hoc network.

In addition, most of the existing solutions face significant challenges when addressing the issue of merging partitioned networks. Clean-up and restore messages have to be flooded in each partition to clean and restore the address databases on every node. Such global flooding maintenance messages can be triggered by partition and merger of even a *single* node. If the network is constantly partitioned and repaired, the global flooding messages triggered may eventually paralyze the network. In contrast, in ZAL, there is *no need* for any form of DAD mechanisms, which is the cause of enormous overhead in existing solutions. There exists *zero* overhead if a network is partitioned and then repaired. For networks that have never before met, if sufficient bits can be allocated to represent the 'network number', there is no overhead either.

Grossglauser *et al.* [16] exploits node mobility to improve the throughput of mobile ad hoc networks. The approach of using node movements to facilitate the assignment of unique addresses proposed in this paper is similar, in that we also try to exploit the characteristics of mobile ad hoc networks to diffuse the knowledge of addresses.

III. ALGORITHM: DESIGN AND ANALYSIS

We present the design and analysis of ZAL in this section. Sec. III-A presents the detail of algorithm. Sec. III-B and Sec. III-C present extensive and rigorous analysis of ZAL. Merger of par-

titioned networks and other issues are discussed in Sec. III-D and Sec. III-E.

A. ZAL: Algorithm

In ZAL, the block of available IP addresses are modeled as points on a circle, referred to as the *address space circle*. The addresses are clockwise placed in an incrementing order on the address space circle. Without loss of generality, assume that these IP addresses form a continuous address space; our algorithm can nonetheless be easily extended to the case of multiple non-consecutive address spaces. Such a block (or multiple non-consecutive blocks) of address space may be conceived as centrally allocated by the centralized authority (e.g., DHCP servers) on the gateway or base station to wireline networks (e.g., node c_2 in Fig. 1.

For the situation similar to Fig. 1, the aforementioned address space is then allocated to the mobile nodes through one or more *initial nodes* with connection to base station. It is worth stressing that address spaces assigned to different initial nodes need to be disjoint. A partition ID (same for all initial nodes) will also be assigned by the base stations or access points to all initial nodes. In a mobile network formed ad hoc, any node starts with the complete ownership of the entire address space circle. Since such ad hoc network does not need to communicate with Internet, an unroutable private address pool can be used in such situation. This node will choose a random address for itself. A unique partition ID will also be generated based on unique parameters of the node (such as its unique MAC address), as well as random physical or logical parameters.

The operation of ZAL can be illustrated in an example based on Fig. 1. In Fig. 1, when there is a need to allocate IP addresses to mobile ad hoc devices in the cell, the base station allocates disjoint blocks of IP addresses to cellular devices c_0 to c_2. When mobile device a_0 meets c_2, a_0 acquires a subset or the entire block of IP addresses allocated to c_2. Assume that the total address space acquired by a_0 is 2^m. a_0 assigns the starting address of the IP address block to itself. For the sake of simplicity of presentation, assume (without loss of generality) that this address is 0.

As shown in Fig. 2, a_0's address delegation space is $[0, 2^m-1]$. When node a_1 joins the network, it is assigned address 2^{m-1} and address delegation space $[2^{m-1}, 2^m - 1]$ by a_0. The address space of a_0 is meanwhile reduced to $[0, 2^{m-1}-1]$. When node a_2 joins the network via a_0, it is assigned address 2^{m-2} and address space $[2^{m-2}, 2^{m-1} - 1]$ by a_0. Meanwhile, a_0's address space is reduced to $[0, 2^{m-2} - 1]$. The process of node a_3 joining the network via a_0 and a_4 joining the network via a_1 is similar.

When a node leaves the mobile ad hoc network, its address space is returned to one of its neighbors. The returned address space can then be assigned to other new nodes. If the node moves to a new cell, the node can acquire new IP addresses from the new cell via similar processes.

As discussed hereinabove, when the base station allocates disjoint blocks of IP addresses to cellular devices c_0 to c_2, it can assign the same partition ID to all allocations. Thus all the mobile ad hoc nodes will form one network. As shown in Sec. III-D, there will be zero overhead when the network with one partition ID is partitioned and repaired.

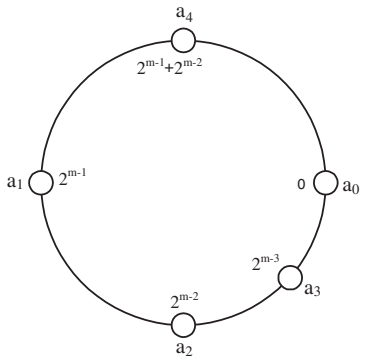

Fig. 2. Illustration of address allocation when new nodes join the mobile ad hoc network.

To handle the depletion of addresses in an address space when unbalanced joining occurs, we can reserve a "global pool" of addresses to be temporarily assigned to the new node, when no additional addresses are available at the existing node who handles the negotiation on behalf the network. The assigned temporary address will be randomly chosen from the global pool. The global pool should be sufficiently large to minimize the collision when multiple temporary addresses are assigned. For the situation similar to Fig. 1, the addresses in global pool need to be routable. The new node is required to release the temporary address as soon as it locates an available address from other nodes as it moves around. Duplication of IP addresses will occur when two or more new nodes hold the same temporary IP address at the same time. However, as shown in Sec. III-C, the probability of such occurrences is very low.

In ZAL, the Address Acquisition (ZAL/AQ) algorithm is used by new nodes to acquire IP addresses. The Address Allocation (ZAL/AL) algorithm is used by existing nodes to allocate IP addresses to new nodes after receiving the initial requests for address allocation. The basic idea of IP address allocation process is for a new node to select the largest offer from the existing nodes who are its immediate neighbours.

Due to space limitation, these two algorithms are not shown. Nevertheless, a few important issues are discussed as follows. The address allocation process in ZAL is fully distributed. Unlike MANETconf [7], there is no need for confirmation from other nodes in the network when an address is allocated to a new node. Second, in the IP address allocation process, there is a probability that the offered IP addresses may be lost during the transfer process. In ZAL, the offering node deletes the offered addresses *before* it sends out the confirmation of the offer. The new node accepts the offered IP addresses only *after* receiving the confirmation of offer. Therefore, if the confirmation message is lost, the offered IP address will be lost or *leaked*. The reason for such design is to eliminate the possibility of IP address duplication.

Nevertheless, as shown in Eq. 1, the probability of IP address leak resulting from transfer process p_{lt} can be reduced to an arbitrarily small value by sending multiple single or multi-hop broadcast messages to confirm the offer.

$$P_{lt} = \prod_{i=1}^{c_{\max}} p_i \quad (1)$$

where $p_i < 1$ is the probability of loss of ith confirmation messages, and $c_{\max}$ is the maximum number of times confirmation messages can be sent.

Due to space limitation, the algorithms for node departure are not shown. They are similar to address allocation process with the design to eliminate address duplication.

B. Address Depletion and Design Parameters in ZAL/DE

Due to unbalanced node joining, it is possible that the IP address space at some of the nodes in the network may be depleted while the rest of the network has ample capacity. The probability of address depletion can be analyzed as follows. For an address space of size 2^m, the address space can be depleted only after $m+1$ nodes join the network. For node a_0, the worst case is that nodes a_1 to a_m all join the network by contacting node a_0. In this case, a_0's address space will be depleted when node a_m joins the network. Assume that when the number of network nodes is n, the probability of a new node joining any of the existing nodes is the same and equal to $\frac{1}{n}$. Therefore, the probability for this case to occur is

$$P_{d,0}^{(m+1)} = \frac{1}{2}\frac{1}{3}...\frac{1}{m} = \frac{1}{m!}$$

where $P_{d,0}^{(m+1)}$ denotes the probability that node a_0's address space is depleted after $m+1$ nodes have joined the network. Similarly, the probability of node a_1 depleting its address space after $m+1$ nodes join the network is $P_{d,1}^{(m+1)} = \frac{1}{m!}$.

For nodes a_i, $i > 1$, more subtleties need to be taken into account. When these nodes join the network, there may be two, rather than one, existing nodes whose address spaces are sufficiently small such that it may lead to address depletion if new nodes joining through them. Therefore, $P_{d,i}^{(m+1)} = \frac{2^{i-1}}{m!}$, $i > 1$. Thus, the total probability of address space depletion after the initial $m+1$ nodes joining the network is

$$P_d^{(m+1)} = \frac{1}{m!} + \sum_{i=1}^{m} \frac{2^{i-1}}{m!} = \frac{2^m}{m!} \quad (2)$$

When additional nodes join the network after the initial $m+1$ nodes, the node with the temporary address may move around and is able to contact other existing nodes with a higher probability within a short period of time, because more nodes are available at this time. Obviously, a larger value of m alleviates the problem of address space depletion.

The ZAL Distribution Equalization (ZAL/DE) algorithm shown in Table II is designed to alleviate the address depletion that may occur after the first $m+1$ nodes join the network. The goal of ZAL/DE is to equalize the distribution of available IP addresses in the network. The challenge is to develop a fully distributed equalizing algorithm with no or negligible overhead. In addition, such an algorithm should not produce any overhead in a stable and reasonably balanced network. A node should make decisions on equalization without participation of all the nodes in the network. In order to achieve this, the algorithm must accurately estimate the IP address capacity that each region of the network needs to have. Being fully distributed, ZAL/DE can achieve all of the aforementioned goals with no or negligible overhead.

TABLE II

THE ZAL ALGORITHM: DISTRIBUTION EQUALIZATION (ZAL/DE)

```
n'.distribute()
    S' ← available IP address space
    target bin capacity S_e ← 2⌈(m−u)/log_2 m⌉
    while (S' > S_e)
        if (no under-capacity bin discovered recently)
            halt equalization process
        if (new node joining activities detected)
            reactivate equalization process
        if (detect other nodes collecting bin capacity)
            reset timer TIMER_dis
        if (TIMER_dis expires)
            send one-hop broadcast to collect bin capacity
            binCapacity ← collected bin capacity
            if (binCapacity < S_e)
                distribute min{S' − S_e, S_e − binCapacity}
                    to the poorest node in the bin
            reset TIMER_dis
```

The operation of ZAL/DE is relatively simple. The complexity lies in the problem of how the design parameters in the algorithm are derived. In a nutshell, a node running ZAL/DE calculates S_e — the target capacity at any small region, referred to as a *bin*. A bin can be explained as follows. When a node joins the network, it can acquire the IP addresses from any of nodes within its transmission range. We refer to such an area within the signal transmission range as a *bin*. As long as there is at least one permanent IP address available in the bin, there will be no address depletion when the new node joins the network in this bin.

As shown in Table II, as long as the capacity of a node is larger than S_e, it periodically broadcasts one-hop *bin capacity discovery messages* to find the bin capacity (excluding the capacity of the node itself). If the actual bin capacity is lower than the target bin capacity S_e, the node compensates the bin capacity via distributing address spaces to the "poorest" node in the bin. After a certain length of time, if the node finds the capacity of all the bins it has checked so far are above S_e, it will stop the periodical broadcast of bin capacity discovery messages. In this way, ZAL/DE will have *zero* overhead in a stable and relatively balanced network. A node will resume the broadcast of bin capacity discovery message only after overhearing messages used in ZAL/AL and ZAL/AQ.

In ZAL/DE, the "poorest" node will be compensated only to the point that the new bin capacity is at most S_e. There are two reasons for this design. First, this approach avoids the loss of large amount of address space as a result of potential packet loss during the transfer process. Second, this approach limits the number of nodes that can broadcast bin capacity discovery message and hence incurs less overhead. The excessive addresses on these nodes can be reduced as new nodes join the network, since the new nodes take the best offers.

ZAL/DE is based on the assumption that the unbalanced address distribution can be gradually alleviated by node move-

ments. This is especially true when there is sufficient time between nodes joining at the same bin. The goal of ZAL/DE algorithm is to handle address depletion as a result of a streak of consecutive node joining events via the same bin before node movements can alleviate the problem.

The ZAL/DE algorithm is designed for *sparse* mobile ad hoc networks. Here, a *sparse* network is defined as a network within which the number of non-overlapping bins is not significantly smaller than the number of nodes in the network. The ZAL/DE algorithm can still be applied to dense networks. However, the need of such a equalizing algorithm is not critical in a dense network. In a dense network, the number of nodes in any bin is so large and the number of bins is so small that probability of unbalanced available IP addresses distribution will be very low. In addition, the relatively more frequent flow of nodes among all the non-overlapping bins can quickly alleviate any unbalanced address distribution.

For sparse networks, the probability of a streak of nodes joining the same bin can be summarized in the following theorem.

Theorem 1. New nodes join a sparse network until the number of network nodes is n_e. The number of nodes in the network before any new nodes is $2m_e$. During the node joining process, for any bin, the probability of a streak of consecutive nodes joining whose length is $q\lceil\log_{m_e} n_e\rceil$ is upper bounded by $1/n_e^{q-1}$.

Proof: The proof process is similar to the analysis of streak probability given in [17]. Due to lack of space, the proof is not presented here. □

Theorem 1 shows that a shorter streak has a higher probability to appear. In fact, for a sufficiently large n_e, the probability diminishes quickly with larger q. In ZAL/DE, we choose $q = 2$ as the design parameter. In this case, we define $S_e = 2\lceil\log_{m_e} n_e\rceil$ as the target bin capacity, and ZAL/DE will compensate any bin whose bin capacity is below S_e.

Further, the design parameters for m_e and n_e are chosen as $m_e = m$ and $n_e = 2^{m-u}$, respectively, where 2^m is the total IP address space, $2^u - 1$ is the average address space per node after 2^{m-u} nodes join the network, and u is the minimum positive integer that satisfies

$$\frac{2(m-u)}{\log_2 m} + 2 \le 2^u \tag{3}$$

Therefore, S_e becomes

$$S_e = 2\lceil\frac{m-u}{\log_2 m}\rceil \tag{4}$$

The reason to choose $m_e = m$ is relatively straightforward. As analyzed before, the address depletion can occur only after $m + 1$ nodes join the network. In fact, the probability of address depletion when the $m + 1$th node joins the network is very low. The reason to choose $n_e = 2^{m-u}$ can be further explained as follows.

(1) When the number of nodes is larger than 2^{m-u}, the probability of a streak of consecutive nodes joining is very low, as shown in the following equation:

$$q\lceil\log_{2^{m-u}} 2^m\rceil < 1/2^{m(q-1)} \tag{5}$$

Eq. 5 shows that even a much shorter streak will have very low probability. Therefore, we should optimize ZAL/DE against the streak of higher probability.

(2) To minimize the unnecessary protocol overhead and eliminate the possibility of the "oscillating" phenomenon when excessive IP addresses are distributed, ZAL/DE prohibits a node to compensate under-capacity bins when the available address space in the node itself is no larger than the desired bin capacity. In other words, we need to stop the equalization process when the average address space per node in the network is below S_e. That is, we need to choose u such that

$$2^u - 1 \ge S_e = 2\lceil\log_{m_e} n_e\rceil$$

If we choose u such that

$$2^u - 1 \ge 2\log_{m_e} n_e + 1 \tag{6}$$

then we can guarantee $2^u \ge S_e$. From Eq. 6, we have

$$\begin{aligned} 2^u &\ge 2\log_{m_e} n_e + 2 \\ &= 2\log_m 2^{m-u} + 2 \\ &= 2\frac{m-u}{\log_2 m} + 2 \end{aligned}$$

We have thus proved the need for Eq. 3.

Without the constraint of Eq. 3, the "oscillating" phenomenon can occur and extra protocols have to be developed to address this problem. The "oscillating" phenomenon can be explained in the following scenario. Assume that there are two nodes a and b in a bin. The available address space of a and b is $S_a = S_e$ and $S_b < S_e$, respectively. If a offers space $S_e - S_b$ to b, then b's space will be increased to S_e and a's space will be reduced to $S_b < S_e$. As a result, b will compensate a since a's space is less than S_e. This process will continue until a and b are apart. Additional protocols have to be developed to prevent this "oscillating" from occurring. Furthermore, even if protocols are developed to prevent such oscillation, from the point of view of the network, the distribution of IP addresses is not improved after a compensate b. The reason is that both a and b have equal probability to join any other bin in the network at a later time. Due to these reasons, ZAL/DE prohibits a node to compensate under-capacity bins when the available address space in the node itself is not larger than the desired bin capacity.

C. Address Duplication

In ZAL, duplicate addresses may only occur when two or more new nodes are assigned the same temporary addresses as the result of address space depletion. In addition, these nodes must share the same address at the same time. There is one important distinction between address duplication in ZAL and existing solutions for address allocation. The duplication in ZAL is *temporary* as long as the available address space is sufficiently larger than the number of nodes in the network. With node movements, the new nodes in ZAL with temporary addresses will eventually acquire unique permanent addresses from nodes with a non-empty address space. Further, such temporary address duplication affect only the *new nodes* with temporary addresses. All old nodes assigned the permanent addresses are free from such problems forever.

In existing solutions, the duplication of addresses is permanent in the sense that any node in the network is always subject to IP address duplication, even after the duplication is detected

and new addresses are reassigned. The causes of duplication include inconsistent states among the network nodes or merger of partitions. For example, even in existing solutions with heavy maintenance protocols to prevent IP address duplication (e.g., MANETconf [7] and Buddy system [13]), address duplication is inevitable when the network is partitioned.

In MANETconf, assume that the network is partitioned into multiple sub-networks. Also assume that there is no inconsistency within the same partition due to problems such as message loss. The IP address allocated to a new node a joining one of the partitions will potentially be in conflict with IP addresses used by nodes in other partitions. In fact, a will potentially be in conflict with all new nodes joining other partitions as well. In Buddy system, the address allocation process, address leak detection process and merger of partitions can all cause address duplication.

In ZAL, the probability of address duplication can be summarized in the following theorem.

Theorem 2. The probability of address duplication in ZAL, P_u, satisfies

$$P_u = 1 - \frac{S_g!}{(S_g - \lambda_o P_d^{(n)} T_s)! S_g^{\lambda_o P_d^{(n)} T_s}} \tag{7}$$

where S_g represents the size of the global temporary address space used to assign temporary addresses. λ_o is the raw node arrival rate. $P_d^{(n)} \leq 1$ is the effective depletion probability when the number of nodes in the network is n. $T_s < \infty$ represents the *average reassignment time*, *i.e.,* the time elapsed before a node with temporary address can be assigned a unique permanent address by another node whose address space is not empty.

Proof: The duplication of addresses in ZAL can be modeled as a server with service time T_s. Fig. 3 illustrates the server model for duplication in ZAL. In this model, nodes with temporary addresses are modeled as jobs for the server. These jobs arrive at the server with the arrival rate λ. The average number of jobs in the server $\bar{n}_u$ represents the average number of nodes with temporary addresses at any time. Within the context of ZAL, the service time T_s represents the average time elapsed before a node with a temporary address can be assigned a unique permanent address by another node whose address space is not empty.

Because ZAL can always find the unique permanent address for a node with a temporary address, we have

$$T_s < \infty$$

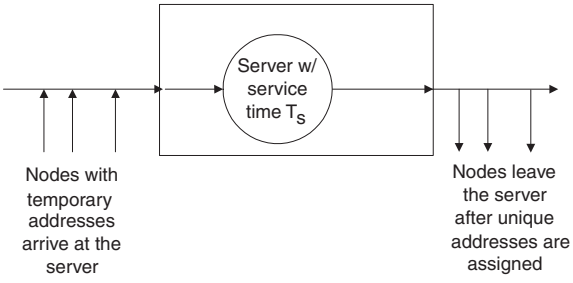

Fig. 3. Illustration of a server model for analyzing addresses duplication in ZAL.

For a server with service time T_s, according to *Little's Law*, we have

$$\bar{n}_u = \lambda T_s \tag{8}$$

where the job arrival rate $\lambda = \lambda_o P_d^{(n)}$ (λ_o being the raw node arrival rate). $P_d^{(n)} \leq 1$ is the effective depletion probability when the number of nodes in the network is n. The *effective depletion* is the depletion that results in the assignment of a temporary global IP address to a new node.

The probability for two out of a group of $\bar{n}_u$ to share the same address from an address pool of size S_g is

$$P_u = 1 - \frac{S_g!}{(S_g - \bar{n}_u)! S_g^{\bar{n}_u}} \tag{9}$$

where S_g represents the size of global temporary address space used to assign temporary addresses.

Substitute Eq. 8 into Eq. 9, we have

$$P_u = 1 - \frac{S_g!}{(S_g - \lambda_o P_d^{(n)} T_s)! S_g^{\lambda_o P_d^{(n)} T_s}}. \qquad \square$$

From Eq. 7, we have observed that when other factors remain constant, P_u increases as T_s increases. T_s depends on many other factors such as the ratio of the size of the potential address space S_a and the number of nodes in the network.

The implications of Theorem 2 are profound. It gives us significant insights towards an efficient address allocation algorithm. For example, Eq. 9 suggests that a small increase of $\bar{n}_u$ will produce a much larger increase of P_u. In fact, Fig. 3 can also be used to model IP address duplication in existing solutions, where $\bar{n}_u \gg 0$. Table III shows the value of P_u under different values of S_g and $\bar{n}_u$. From this table, we can conclude that a small value of $\bar{n}_u$ is critical to achieve low duplication probability. Simulation in Sec. IV indicates ZAL can achieve $\bar{n}_u < 2$ when 512 unique addresses are allocated to a network of 480 nodes. Therefore, in ZAL, even for a smaller global address pool of 50, the duplication probability is still around 0.02. In contrast, for a network of 480 nodes, as shown at the beginning of this section, existing solutions have $\bar{n}_u \gg 2$, which will produce much larger duplication probability.

TABLE III
DUPLICATION PROBABILITY P_u UNDER DIFFERENT VALUES OF S_g AND $\bar{n}_u$

S_g	$\bar{n}_u$	P_u
1024^2	240	0.027
4296	480	$1 - 8.32 \times 10^{-13} \simeq 1$
4296	240	0.999
200	2	0.005
50	2	0.02

In some of the existing solutions (e.g., MANETconf [7] and Buddy system [13]), the effective number of nodes involved in addresses duplication may only be certain portion of the entire network nodes. Suppose that the ratio between the available address space[1] and the number of both existing nodes and new nodes involved in address duplication is $r = S_g/n_u$. Fig. 4 shows the duplication probability under different values of n_u and r. As more nodes attempt to join the network, n_u will increase while S_g decreases. Therefore, r will diminish rapidly. As a result, the behavior of the network will converge to the upper-left corner of Fig. 4, where duplication probability approaches 1.

[1]Note that it is the space still available, not the total address space, since some of the total space has already been allocated to existing nodes.

The network will eventually be flooded with node join messages and be paralyzed. Such protocols are obviously not scalable. As a result, a significant number of IP addresses can not be allocated, which leads to severe address leaks.

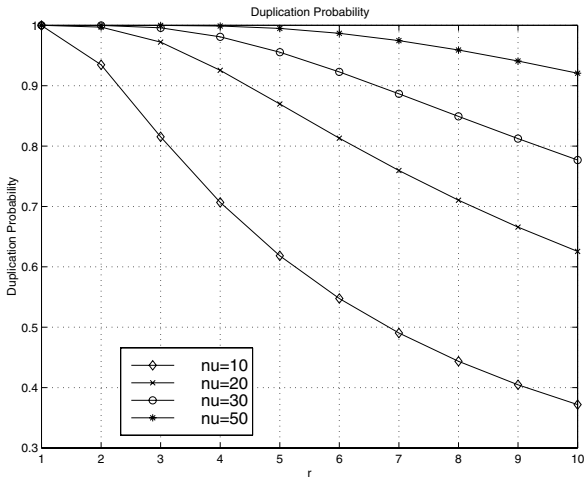

Fig. 4. Duplication probability under different values of r and n_u.

D. Merging of Partitioned Networks

The merging of partitioned networks can be further divided into two sub-problems. In the first sub-problem, the partitioned networks are parts of the same larger network and share same partition ID. In the second sub-problem, the partitioned networks have never met before and differ in their partition IDs.

The ZAL solution to the first sub-problem is trivial. *Nothing* needs to be done when two sub-networks of the same larger network meet. This is true even when there are new nodes joining any or both of the sub-networks after they separate. The reason is that the address spaces at different sub-networks are disjoint. The two sub-networks will be able to recognize that they belong to the same larger partition by comparing the partition ID. For a typical scenario shown in Fig. 1, the base station can assign the same partition ID to IP address blocks allocated via all cellular nodes.

For the second sub-problem, the ZAL solutions are more involved. In this paper, we propose to convert the addresses of nodes in smaller networks to address space of larger networks. Only addresses in one of the partitioned networks can be preserved. The others have to forgo their addresses and convert to new addresses. Of course, the conversion is a gradual process in which the nodes at the boundaries of small networks will be converted. Even though the conversion is inevitable, it is desirable to minimize the overhead. Especially, the number of conversions should be minimized. In addition, such conversions must be based on distributed algorithms.

One simple technique to minimize the number of conversions is to assign a *network number* to each network. That is, each small network will randomly pick a network number and append the allocated address to the network number and use the result as the address. Because the number of networks is small, unlike the number of nodes in the network, the bits needed for network number can be rather small. The authors expect that a 4-bit network number will be sufficient for most cases. However, the coversion of the partition IDs is still necessary.

In the unfortunate case that two networks select the same network number, a conversion is inevitable. The following algorithm implements the conversion process. When two nodes from different small networks meet, the node in the network with fewer nodes needs to be converted to the larger network. Thus, the challenge is to ensure each node has the up-to-date information about the number of nodes in its original small network. The number of nodes in the network can be easily estimated from the ratio between the total size of the address space and the size of the remaining address space owned by each node. A smaller ratio generally indicates a smaller number of nodes in the network.

Certain subtleties of conversion process call for more detailed discussions. A converting node must return its address space to the network it is departing from. The converted node will join the new network as a new node. The communication among the nodes from two merging networks will be gradually established as more nodes from the smaller network convert to the larger network. At the end, all nodes will belong to one merged network. The following scenario illustrates this process.

Suppose the number of nodes in the two partitioned networks a_1 and a_2 are n_1 and n_2 where $n_1 < n_2$. Without loss of generality, also assume that the conversion starts at time $t = 0$ and there are $b(t)$ pairs of nodes at the boundary of two networks. Due to the unbalanced address distribution within each network, the conversion rate from network a_1 to network a_2 is $0.5 < c(t) \leq 1$. So the number of conversion at time $t = 0$ is $b(0)c(0)$ from a_1 to a_2 and $b(0)(1 - c(0))$ from a_2 to a_1. As time elapses, the average address space per node in the smaller network will increase because of the address space returned by the converted nodes. In the meantime, the average address space per node in the larger network will decrease because more address space are delegated to converted nodes. As a result, $c(t)$ will increase over time. The conversion process is similar to the process of *erosion*. Layer after layer of nodes in the smaller network will be converted to the larger network over time.

Most of the existing solutions experience significant difficulties when addressing the issue of merging partitioned networks. Some existing solutions [7] can only handle the first sub-problem discussed above. In the worst case, every address in each merged network must be checked to detect duplicate addresses. To address the first sub-problem, [7] and [12], unlike ZAL, have to clean-up the addresses from address databases on each node when network is partitioned. When the two sub-networks meet again, messages have to be flooded in both sub-networks to restore the databases. This will occur even when only one of the sub-networks contains only one node. As a result, if two sub-networks are 'switching' between meeting and departing all the time, the two sub-networks will be overwhelmed by flood messages sent for cleaning up and restoring address databases.

E. Further Discussions

1) Reassignment Time: As shown in Eq. 7, when other factors remain constant, P_u increases as reassignment time T_s increases. In [16], it is shown that for two arbitrary nodes i and j, at any time t, the probability that they are within the transmission range of each other is at least $\Theta(1/n)$. The assumption here is that the

location of each node is a stationary and ergodic process with stationary distribution uniform within the node movement area. Since a new node can acquire IP address from any node whose address space is not empty, the reassignment time T_s is bounded by following equation:

$$P\{T_s > \gamma t_\Delta\} \leq (1 - 1/n)^{\gamma n_{ne}} \tag{10}$$

where n_{ne} is the number of nodes whose IP address spaces are not empty. t_Δ decreases as node velocity increases.

2) Address Leak: In essence, ZAL guarantees zero address duplication at the cost of potential address leaks. However, as we can observe, the probability of address leaks is negligible.

Assume that the lifetime of a node is exponentially distributed with the cdf as follows.

$$F_X(t) = 1 - e^{-t/T_l} \tag{11}$$

where $t \geq 0$, T_l is the expected lifetime of the node.

Also assume that node arrival follows Poisson distribution with the following pmf:

$$P\{N = k\} = \frac{\alpha^k}{k!} e^{-\alpha} \tag{12}$$

Suppose that the total space is S at $t = 0$, then the probability of average address leak of equal or greater than $l_m = S/(k + 1)$ is

$$P_k(t) < k(\sum_{i=0}^{k} \frac{(\alpha t)^i}{i!} e^{-\alpha t})(1 - e^{-t/T_l}) \tag{13}$$

Fig. 5 illustrates $P_k(t)$ for $T_l = 14$ days and $\alpha = 1/120$ 1/s.

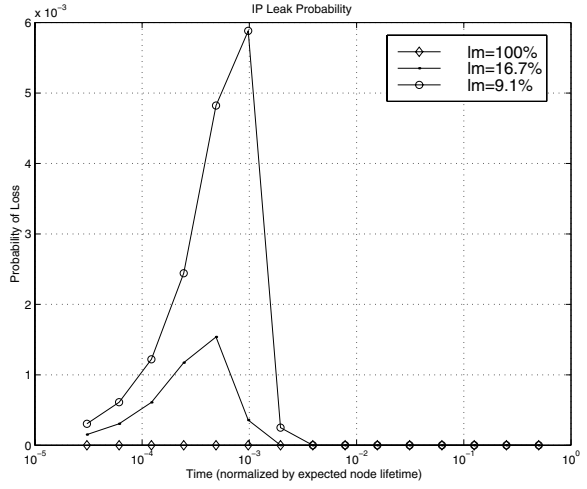

Fig. 5. IP address leak probability.

From Fig. 5, we find that the probability is negligible to lose more than 9.1% of the total address space. The reason for this is that when t is small, the probability of node failure is very low. When $t \gg 0$, the probability of very few nodes joining the network is very low. Realistically, the value for T_l and α is typically much larger than the values used to illustrate Fig. 5. It is worth stressing that in existing solutions such as MANETconf [7] and Buddy system [13], high probability of address duplication and heavy maintenance overhead can lead to severe address leak, as shown in Sec. III-C. Such address leak are caused by the heavy overhead that can paralyze the network when address duplication is significant.

In contrast, address leak in ZAL will not affect the operation of network until all the available permanent addresses in the network have been assigned to network nodes. Even after all the addresses have been depleted, the operation of nodes with permanent addresses will not be affected. The only problem in such situation is the address duplication amongst the new nodes with temporary addresses. We argue that address leak is not a major problem for application of ZAL. The very reason of using ZAL is lack of central coordination. Applications of such nature usually do not need to be operative for too long. Nevertheless, provisions can be made to deal with address leak in very unfavorable operation environment (e.g., significant portion of nodes depart ungracefully). For the situation similar to Fig. 1, the base stations or access points can allocate the addresses with limited lease life, after which alternative address blocks can be allocated with new lease life. Alternative address blocks are necessary in order to facilitate the continuous operation of mobile nodes during the migration. The aforementioned process can be applied to mobile networks formed ad hoc as well. In such situation, every node in the network can choose alternative address blocks that will not conflict with the address blocks in operative because each node has the knowledge of address blocks being used.

IV. Performance Evaluation

The performance of ZAL is evaluated extensively using a simulator developed in Java that implements all aspects of ZAL. The node mobility model used in the simulation is the Random Waypoint (RWP) mobility model. In RWP, a node randomly selects a position in a predefined area. The node then moves towards the destination with a velocity v. v is uniformly distributed between $[0, V_{\max}]$. $V_{\max}$ is the maximum speed a node can achieve. After the node arrives at the destination, the node will remain at the destination for a period of t_s. t_s is exponentially distributed with a mean $1/\lambda_s$. The node transmission range is 250 m.

There are two phases in the simulation. At the beginning of the simulation, there is only one node in the mobile ad hoc network. In the first phase, new nodes join the mobile ad hoc network based on Poisson distribution described in Eq. 12. A total of $480 > 2^8$ nodes join the network in the first phase. All nodes move according to the aforementioned mobility profile. When each node joins the network, it is assigned an address based on the ZAL algorithm. In the second phase, all 480 nodes continue to move according to the same mobility profile without new nodes joining the network.

The parameters of the simulation are as follows. The total simulation time for the two phases is 7600 seconds. $1/\lambda_s = 3s$. The moving speed is uniformly distributed within $[0, 20.0]$ m/s. The node arrival rate is Poisson distributed with $\alpha = 0.07$ 1/s. The size of the simulation area is a square of 6km by 6km. The results to be shown are averages of 5 simulation runs.

Fig. 6 shows the average length of time during which duplicate addresses exist. We hereafter refer to this as *duplication active time*. The x-axis is the size of available address space S_a. There are four lines in the figure, representing results with different sizes of the global temporary address pool (S_g in Eq. 7). From this figure, we may observe that ZAL performs very well for $S_a = 1024$, 2048 and 4096. Only for $S_a = 512$, when there is only 32 spare addresses for the network with 480 nodes,the

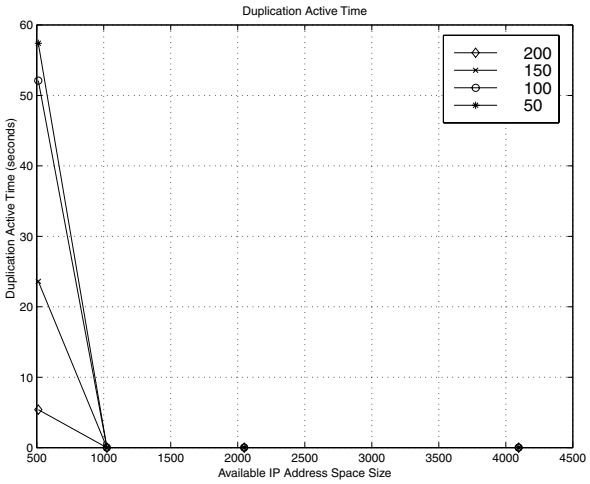

Fig. 6. Duplication active time for different values of available address space size S_a and global temporary address pool size S_g. The 'diamond', 'x', 'o' and '*' lines are results for S_g=200, 150, 100 and 50, respectively. The value of S_a (x-axis) is 512, 1024, 2048, and 4096, respectively.

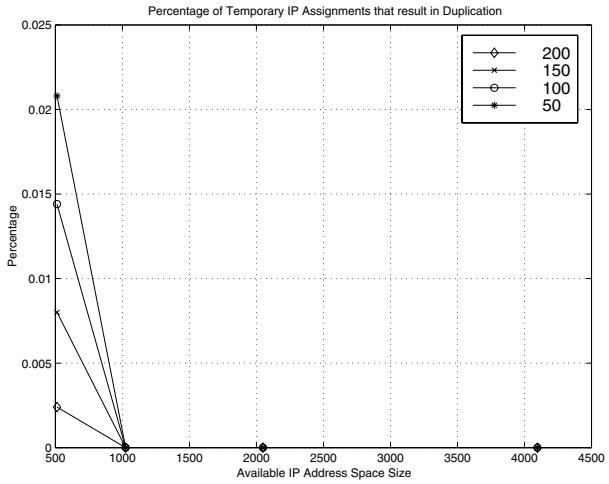

Fig. 7. Percentage of temporary addresses that results in the duplication of addresses for different values of available address space size S_a and global temporary address pool size S_g.

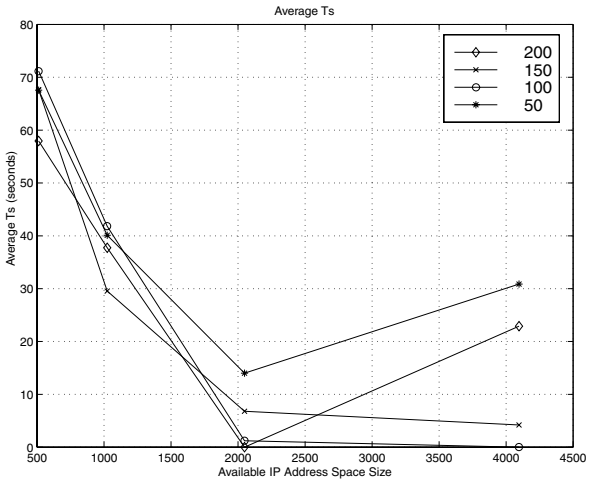

Fig. 8. Average reassignment time T_s for different values of available address space size S_a and global temporary address pool size S_g.

performance degrades. From this figure, we can see that by using one or two more bits in address space to increase the number of bits from the minimum requirement of 9 bits to 10 or 11 bits, we can essentially eliminate address duplication with ZAL.

Even with $S_a = 512$, *all* the nodes with temporary addresses are reassigned with new unique addresses within, in average, 70 seconds or less from the time the nodes join the network (Fig. 8). In other words, even for $S_a = 512$, after a period during which duplication addresses may occur, all nodes receive unique addresses. The importance of this fact is that because there is no permanent duplication of addresses, *no* Duplicate Address Detection (DAD) algorithm is needed for ZAL. In fact, the numerical values for $S_a = 4096$, 2048 and 1024 are all zeros even for a smaller global temporary address space $S_g = 50$, which signals that there is no duplication at all.

Fig. 7 illustrates the low percentage of temporary addresses that results in the duplication of addresses, which is the key success factor for ZAL.

The reason for Fig. 7 can be explained in Fig. 8, which shows the average reassignment time T_s. In this figure, we find that even though T_s increases as the size of available address decreases, it still remains at a relatively small value of 70s or less. It is worth noting that the simulations are performed in a *sparse* network. In a dense network, T_s will be much smaller. The small value of T_s is critical to the success of ZAL. Because for a duplication of addresses to occur, two or more nodes must share the same temporary address *at the same time*. A small value of T_s makes the probability very small for the second conditions to be satisfied. It is worth noticing that *when a new node joins the network, it will receive an IP address instantly*, albeit may from global temporary address pool. The reassignment time T_s is the time the new node needs to find a permanent address. During T_s, the new node can communicate with other nodes and Internet using its temporary IP address.

To further analyze ZAL, we illustrate the duplication intensity of ZAL in Fig. 9. Duplication intensity represents the average number of duplications at any time. From this figure, we find that in most of cases, there is only one pair of duplication involved.

Fig. 10 shows n_u, the average number of nodes holding temporary addresses at any time. As can be seen from this figure, n_u in ZAL is very small. It is smaller than 2. For larger values of available address space size, the value is close to 1 or less. According to Eq. 9, a value of 1 for n_u will produce 0 for P_u. A value of 2 for n_u will produce 0.005 for $S_g = 200$ and 0.02 for $S_g = 50$. These theoretical results match well with our experimental results. Again, the small value for n_u is the key success factor of our ZAL algorithm.

Fig. 11 shows the ratio of duplication active time to the total simulation time. This figure is an indicator of duplication probability P_u in Eq. 7. Since all the reassignment of unique addresses to the nodes with temporary addresses have been resolved during the current simulation time, the increase of simulation time will dilute the values in this figure and results in lower values. In our experiment, we deliberately choose the simulation time that is slightly larger than the amount of time needed for all 480 nodes to join the network. So the values shown in this figure are good indicators of P_u. When we compare the numerical values of this

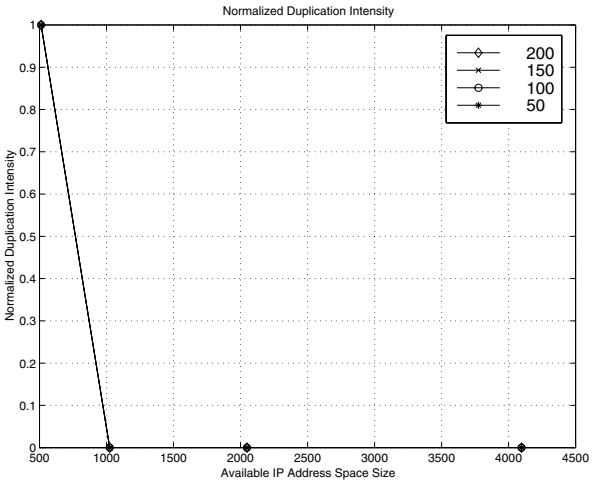

Fig. 9. Duplication intensity for different values of available address space size S_a and global temporary address pool size S_g.

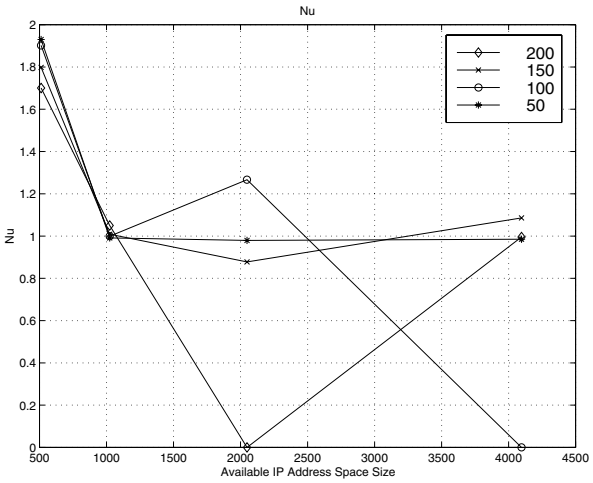

Fig. 10. N_u for different values of available address space size S_a and global temporary address pool size S_g.

figure, we observe that they approximately agree with the theoretical results produced in Sec. III-C, within the same orders of magnitude.

V. Conclusions

From the extensive analysis and performance evaluations presented in this paper, the Zero-Maintenance Address Allocation (ZAL) algorithm proposed in this paper outperforms existing solutions in many important aspects. Among them, ZAL does not need Duplicate Address Detection (DAD) mechanisms, which can impose significant message exchange overhead. ZAL does not need to flood networks with maintenance messages. ZAL is maintenance free — no maintenance messages are needed at all. ZAL is also very fast, no timeouts needed and thus no delays. ZAL does not need to modify components of other protocols such as the IP header or the routing table. In addition, there is no or very low overhead involved in ZAL to merge the partitioned networks. We believe that ZAL represents a firm step towards realizing all-IP wireless networks, by solving basic problems of address allocation without centralized authorities.

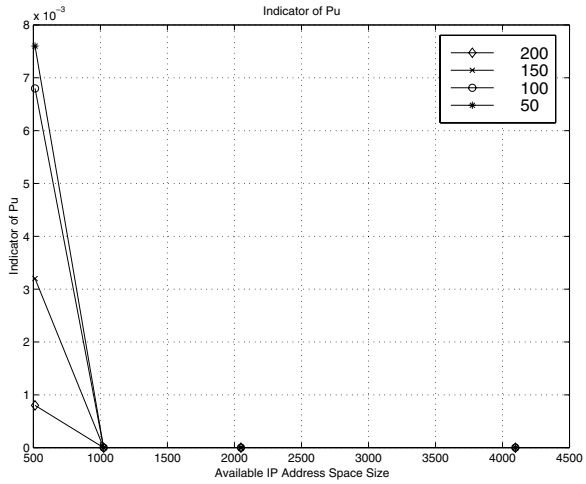

Fig. 11. Indicator of P_u for different values of available address space size S_a and global temporary address pool size S_g.

VI. Acknowledgments

We greatly appreciate all the valuable suggestions from the anonymous reviewers of this paper. Some of the suggestions for future improvements, unfortunately, are not accommodated in this paper due to space and time constraints. They include performance comparisons between ZAL and existing solutions, as well as detailed merging protocols and their evaluations. We leave them as future work.

References

[1] M. EI-Sayed and J. Jaffe, "A view of telecommunications network evolution," *IEEE Communications Magazine*, vol. 40, December 2002.
[2] H. Hsieh and R. Sivakumar, "On using the ad-hoc network model in wireless packet data networks," in *Proc. ACM MOBIHOC 2002*, 2002.
[3] X. Wu, S.-H. Chan, and B. Mukherjee, "Madf: A novel approach to add an ad-hoc overlay on a fixed cellular infrastructure," in *Proc. IEEE WCNC 2000*, 2000.
[4] C. Qiao and H. Wu, "icar: An integrated cellular and ad-hoc relay system," in *Proc. IC3N 2000*, 2000.
[5] Y.-D. Lin and Y.-C. Hsu, "Multihop cellular: A new architecture for wireless communications," in *Proc. IEEE INFOCOM 2000*, 2000.
[6] G. Aggelou and R. Tafazolli, "On the relaying capacity of next-generation gsm cellular networks," *IEEE Personal Communications Magazine*, vol. 8, no. 1, pp. 40–47, Feburary 2001.
[7] R. Prakash S. Nesargi, "Manetconf: Configuration of hosts in a mobile ad hoc network," in *Proc. INFOCOM 2002*, 1997.
[8] N. Vaidya, "Weak duplicate address detection in mobile ad hoc networks," in *Proc. ACM MobiHoc'02*, 2002.
[9] C. E. Perkins, E. M. Royer, and S. R. Das, *IP Address Auto-configuration for Ad Hoc networks*, 2000, Internet Draft, IETF Working Group MANET.
[10] P. Patchipulusu, "Dynamic address allocation protocols for mobile ad hoc networks," M.S. thesis, Computer Science, Texas A&M University, 2001.
[11] S. Nesargi and R. Prakash, "Issues pertaining to service discovery in mobile ad hoc networks," in *ACM Workshop on Principles of Mobile Computing*, 2001.
[12] J. Boleng, "Efficient network layer addressing for mobile ad hoc networks," Tech. Rep. MCS-00-09, The Colorado School of Mines, 2000.
[13] Mansoor Mohsin and Ravi Prakash, "Ip address assignment in a mobile ad hoc network," in *Proc. MILCOM 2002*, 2002.
[14] A. J. McAuley and K. Manousakis, "Self-configuring networks," in *Proc. MILCOM 2000*, 2000.
[15] A. McAuley, A. Misra, L. Wong, and K. Manousakis, "Experience with autoconfiguring a network with ip addresses," in *Proc. MILCOM 2001*, 2001.
[16] M. Grossglauser and D. Tse, "Mobility increases the capacity of ad-hoc wireless networks," in *Proc. IEEE INFOCOM 2001*, 2001.
[17] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, MIT Press, 2 edition, 2001.