

# Hierarchical Area-Based Address Autoconfiguration Protocol for Self-organized Networks

Mandimby N. Ranaivo Rakotondravelona, Fanilo Harivelo, and Pascal Anelli

Laboratoire d'Informatique et de Mathématiques (LIM)  
University of Reunion Island, France  
{mandimby.ranaivo,fanilo.harivelo,pascal.anelli}@univ-reunion.fr

**Abstract.** Node autoconfiguration is one of the main issues in self-organized networks. One class of approaches relies on hierarchical organization of nodes. This kind of structuration aims to deal with scalability issues, especially for wireless networks. But building and maintaining a hierarchy is generally expensive for these resource-limited networks. We propose a low-cost distributed, hierarchical, location-based address autoconfiguration protocol. Each node infers its address from those of its one-hop neighbors and from its relative position to them. In this way we obtain a globally-consistent organization resulting from local interactions only. This reduces the latency and the overhead generated during address configuration. Moreover this scheme is the first step towards the design of a scalable routing protocol taking advantages of the proposed hierarchical addressing.

**Keywords:** self-organized, ad-hoc, wireless, distributed protocol, address autoconfiguration.

## 1 Introduction

Self-organized networks consist of hosts that rely neither on a central infrastructure, nor on an external intervention to perform necessary configurations of members. *Wireless Sensor Networks* (WSN), *Wireless Mesh Networks* (WMN) and *Mobile Ad-hoc Networks* (MANET) are typical examples. Internet of things, smart cities, disaster recovery emergency communication systems are possible fields of application of such networks. But to be useful and widely adopted in the real world, they need to scale well. By good scaling capability we mean: bounded configuration time, limited traffic overhead ... in other words, good performance despite an increased size of the network.

In self-organized networks, address autoconfiguration is a major issue as it is one of the fundamental prerequisites for communication between hosts. Address assignment takes place before any routing protocol execution. The routing protocol will gain from a suitable organization of nodes obtained by the addressing mechanism. Many address autoconfiguration protocols have been proposed but

those using hierarchical approaches give the better results in term of scalability as they provide benefits to hierarchical routing protocol [5].

Some mechanisms of existing hierarchical approaches rely on the election of special nodes known as cluster heads. These nodes have more responsibility, such as maintaining the hierarchy, than the rest of the network. For this task, they have to maintain more information, for instance addresses of the nodes in their clusters. Thus their failure is difficult to handle. Other approaches use particular address structure, most of the time, an address tree. The trees are constructed in a top-down approach. But with this way, when the address length is fixed, it is difficult to extend the network with larger addresses. This could lead to rapid address exhaustion. Moreover, in high density networks, granularity problems can occur, again because of the top-down construction. In fact, an a priori distribution of addresses is difficult since the topology of the network is unpredictable. A bottom-up approach doesn't suffer from these drawbacks because it is by nature flexible. For instance, in the granularity problem, the top level of the hierarchy would raise instead of address exhaustion. In this paper, we focus on how to get hierarchical organization without having the cumbersome tasks in maintaining the hierarchy.

We propose a hierarchical addressing scheme for self-organized networks. The nodes are treated equally. Each of them runs the same algorithm: local interactions between nodes result in a hierarchical organization of the network. In other terms we obtain a hierarchy from a bottom-up construction. Moreover address acquired by a node not only indicates the hierarchy branch to which it belongs but also reflects its position in the network. To achieve all of this, we designed an address pattern that is naturally followed by the nodes as they run the proposed protocol. We choose to make a distinction between the identifier and the address as in [2]. The identifier is unique and remains unchanged during the node's lifetime. The address gives only information on node's location.

Our two main contributions are:

- *A hierarchical subdivision and labeling of space.* This is a geographic subdivision of the space. Each smallest area is locally labeled with a binary number following a simple rule. A number of those areas form a bigger area that is locally labeled in its turn and so on. The result is a hierarchy of areas with several levels. The global label of an area is a concatenation of its local label and those of the bigger areas it is member of. Another interesting property of the subdivision is that areas sharing a part of their labels (e.g. same  $n$  most significant bits) are geographically close to each other. The details are discussed in Section 3.
- *A distributed address autoconfiguration protocol.* From the previous space partitioning, an address plan is created. In a few words, the labels i.e. the binary numbers representing an area are used as addresses for the nodes. This assignment can be achieved using either the absolute geographic locations of the nodes or their relative positions to each other. In the latter case, only one-hop communications are necessary to determine the address of a new host. By applying the particular labeling of the space to node addressing,

we are able to construct an address hierarchy in a bottom-up way. This protocol is detailed in Section 4.

Performance of the address assignment scheme is evaluated qualitatively and through a simulation realized on ns-3 [1] as described in Section 5. Results show that in the worst case (all the nodes starting at the same time in a connected topology network) the convergence delay, i.e. the time needed for all nodes to get valid addresses, is  $O(\log(n))$  where  $n$  is the number of nodes. The number of configuration packets generated during this process is  $O(n)$ . The results demonstrate the scalability of the protocol.

## 2 Related Work

Numerous address autoconfiguration protocols for self-organized networks, mostly for MANETs have been proposed in the literature. They can be classified as stateless, stateful or hybrid. In stateless protocols, nodes have no knowledge about the already used addresses in the network [12,3,11]. The address given to a new node is usually selected randomly from the available set of addresses. Therefore it is necessary to check if the new address is already used. This process is called *duplicate address detection* (DAD). Because nodes don't retain all the used addresses, DAD implies the flooding of the entire network generating high traffic overhead, leading to poor scalability. AROD [8] tries to limit DAD procedures by performing DAD on multiple addresses but broadcast is still necessary.

Stateful schemes tend to be more scalable because there is no need for network-wide DAD. Configured nodes store a state from which addresses for new joining nodes are derived. Usually the protocols rely on some mathematical properties to generate unique sequences of addresses. For instance, Prophet [17] uses a function which generates distinct sequences of addresses according to the seed, each new node receiving a new seed. In [6], properties of prime numbers are used. Other approaches consist in sharing disjoint address pools between configured nodes [18] [13].

The above mentioned addressing protocols produce a flat addressing scheme. Yet according to Hong et al. comparative study [5], hierarchical and geographic routing protocols offer better scalability. And these two approaches are efficient with non flat address structure. Hierarchical addressing can be obtained by clustering the network. Some nodes act as cluster heads like in [7]. A node's address is the sequence of addresses of all its higher levels cluster heads addresses. An adaptation of the IPv6 stateless address autoconfiguration with a hierarchical approach is proposed in [15]. One of the drawbacks of these schemes is that these special nodes potentially become bottlenecks because they have to deal with traffic from lower nodes in the hierarchy. In [2], a hierarchical addressing without clustering is proposed. The link made between positions of the nodes and the branch of the hierarchy to which they belong inspired us. But in this approach, the hierarchy is constructed in a top-down way. So if there are regions with high node density, some nodes can potentially not obtain an address.

With the availability of geographic location information, another way to obtain hierarchical addressing reflecting the positions of the nodes is to partition the network space with structured areas and use the identifiers of these areas as addresses [10]. We propose a similar scheme with elastic partition shape and size depending on the topology and the dynamic of the network.

Finally we want to point out the study in [2] showing that scalability problem in MANETs and in self-organized networks results mainly from the use of the node's address as its identifier at the same time. The authors propose a dynamic addressing in which node's identifier remains unchanged during its lifetime; node's address is only intended for location and routing purposes.

### 3 Hierarchical Areas Organization

This section deals with how we can partition the two-dimensional space into labeled areas in order to get a hierarchical organization. This structure constitutes the address plan used to configure nodes as seen later in Section 4.

#### 3.1 Hierarchy of Areas

From now on, we consider the two-dimensional space. Let's delimit a square and define it as an elementary area. Then we give it a *label*. A label is a binary number of 2 bits, for instance, 00 (Figure 1(a)). This area is a level-1 area. The remaining available labels are 01, 10 and 11. Let's assign them to the 3 areas of the same size next to the first in a way that the whole forms a larger square area (Figure 1(b)). The latter is in turn labeled with a 2-bits number and constitutes a level-2 area. Again, 4 larger areas are grouped together resulting in a much larger area labeled with a 2-bits number (Figure 1(c)) and so on. The process can be repeated indefinitely. The hierarchy is highlighted on Figure 1(d). The elementary areas are uniquely identified by the *concatenation* of the labels of every larger areas to which they belong. To sum up, a level- $n$  area has a 2-bits label appended to a global label resulting from the concatenation of the labels of level- $N$  to level- $n + 1$  areas having it as "child" where  $N$  is the level of the largest area.

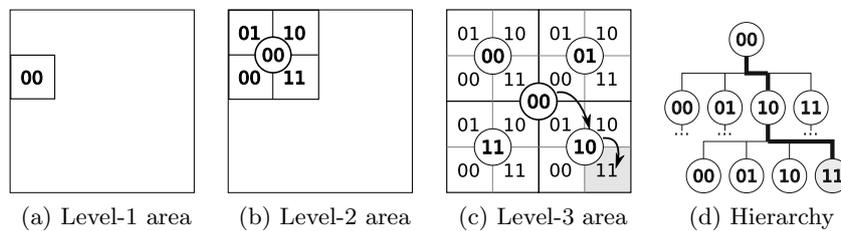


Fig. 1. Area hierarchy

How about the hierarchy? Each 4 siblings of areas belongs to a particular level of the hierarchy. One area of level  $n$  is composed of 4 areas of level  $n - 1$  and so on. The equation (1) gives the global labels of any 4 siblings of the space with the desired number of levels of hierarchy represented by  $n$ . The two-dimensional space can be considered as covered by a numerical matrix ( $M(n)$ ) whose elements correspond to the global labels of each area.

$$M(n) = \begin{pmatrix} (10 - n\%4)||M(n-1) & (01 - n\%4)||M(n-1) \\ (01 - n\%4)||M(n-1) & (00 - n\%4)||M(n-1) \end{pmatrix} \quad (1)$$

With  $M(0) = ()$

*Example.* On Figures 1(c)(d) we have a hierarchy with  $n = 3$  levels. Let's consider the area colored in gray. It is labeled with 11. The larger area to which it belongs is labeled with 10 that is itself part of the largest area 00. Therefore the complete global label of the considered area is the concatenation of these labels from top to down (indicated by arrows on (c)). The result is 001011.

### 3.2 Properties

The area hierarchy described above has some interesting properties for the design of the address configuration mechanism:

- *Unlimited.* Theoretically, with equation (1), we can partition the entire two-dimensional space with an infinite number of levels of hierarchy.
- *Quasi-isotropic.* If we calculate values of equation (1) with great  $n$ , and if we draw the path from label 00 to the highest value, we would get a spiral. In other words, the construction of the structure tends to grow equally towards all directions. Moreover, as there is no privileged direction, the lengths of the labels of two opposite areas with respect to the center of the considered space tend to be the same.
- *Geographic and hierarchical property.* One of the most important property of this area organization is the relation between the geographic location of an area and its hierarchical position. In fact areas with the same parent belong to the same level of hierarchy but are also close to each other. For instance, if we have a look at Figures 1(c) (d), the areas 001011 (the gray one) and 001010 (on “top” of the gray one in (c), on its “left” in (d) ) share the same parent level-2 area 0010 but are also geographically in the level-2 area corresponding to this label. In other words, the more bits two areas share (from the high-order bit), the more they are geographically close to each other.
- *Easy neighboring identification.* From equation (1) we know how global labels are geographically distributed to all areas. Hence for a given area, it is easy to find the labels of the surrounding areas (from the elements of  $M(n)$  in (1)).

## 4 Address Autoconfiguration Protocol

In this section we give details about the address autoconfiguration protocol that we designed based on the previous two-dimensional space subdivision. The main

idea is to use the labels of areas as addresses for nodes. In other words, the previous labels distribution serves as an *address plan*. This way, the address assignment scheme inherits the above mentioned properties:

- The maximum number of addressable nodes depends only on the maximum supported address length. It is possible to start with a small address length and increase it as the network grows in scale.
- The more bits two addresses share (from the high-order bit), the more geographically close the hosts are.
- Neighbors addresses of a given node can be calculated from its own address. Inversely, a node's address can be deduced from those of its neighbors, provided the information about its relative position to them.

As stated before, we make a distinction between the *identifier* (ID) and the *address* of a node. The ID uniquely identifies the node and the address gives an information about its geographic position. This scheme is interesting for networks of mobile nodes. As a matter of fact it is not easy to keep track of a node whose ID is changing frequently. Hence it is better to keep the ID unchanged. But the ID has then to be resolved into an address. To achieve that, distributed lookup service exists [2] [10]. ID-address separation allows also the assignation of the same address to two or more nodes in the same limited geographic location. This helps to solve granularity problem (case of very high number of nodes in the same location) and spares the address space.

#### 4.1 Basic Idea

We propose a mechanism that assigns addresses to nodes depending on their geographic position in the network. There are two cases:

- If the nodes know their absolute geographic position (using GPS), the entire region where they are located is mapped with our address plan. This leads to a grid partitionning of the space. The size of the elementary squares is chosen to be close to the radio range of the nodes. Each node chooses as address the label of the region where it is located using its absolute coordinates.
- If the nodes have only access to their relative geographic position [16], the address assignement needs communication between nodes. From now on, we will focus on this second case.

A new arriving node gets its address from an already configured node. By configured we mean having a valid address. Thanks to a distributed algorithm, nodes acquire addresses according to the address plan. After the configuration of all nodes in the network, the distribution of the addresses looks like the distribution of the labels of areas as seen previously.

There are two types of configured nodes :

- A *standard* node doesn't respond to address requests from new joining nodes.
- An *Address Agent (AA)* node responds to address requests of incoming nodes with its own address.

It is important to notice that AA nodes aren't elected. They become AA in order to satisfy unconfigured nodes with no AA in their vicinity.

## 4.2 Conventions and Assumptions

From now on, we will consider a wireless ad-hoc network. The *neighborhood* of a node is the set of nodes within its communication range. In other terms, the neighborhood is formed by all nodes reachable within one hop. The *relative position* of a node to one of its neighbor must be taken in a geographic sense and can be obtained using techniques similar to those surveyed in [16]. We call *prefix* at level  $n$  the two bits corresponding to the label of an area of level  $n$ . The address of a particular node is then the concatenation of all prefixes of all levels. For example, given the address 1011, the prefix of level 1 is 11 and that of level 2 is 10.

The following assumptions are made and are necessary for the protocol to work properly:

- A node is uniquely identified with a separate identifier than the address (the MAC address, the EUI-64 identifier, . . .)
- A node is aware of its neighborhood. Nodes exchange "hello" messages periodically with the neighbors.
- The nodes are aware of their relative position to their neighborhood.
- Each connected group of nodes (or one isolated node) is identified by a network ID.

## 4.3 Address Allocation

*New Node.* To join a network, a new node sends an address request. If present, existing AA in the neighborhood provide the node an address. If this is the first node in the network, it assigns itself the address 00 and then becomes an AA while a random network ID is generated.

*Becoming an AA.* When it "hears" repeated address requests from a new joining node indicating that there is no response, a standard node becomes AA to satisfy the address solicitation. For this purpose, it takes the addresses of all AA in its neighborhood and calculates a new address in accordance with the address plan. This new address will be used to configure incoming nodes.

*Example.* On Figure 2(a), a node A starts. It sends address requests but receives no reply. So it assigns itself the address 00 and becomes an AA. Later (Figure 2(b)), node B arrives, wants to join and sends address requests. A replies so B is configured as standard node with the same address as A. Then comes node C (Figure 2(c)). We suppose that it is out of the range of A. It sends address requests. B "hears" but doesn't reply as it is not AA. C continues to send requests. B assumes that C has no AA in its vicinity. So B configures itself as AA and uses A's address and its relative position to infer the address 11 (see Figure 1(b)). Finally B responds to C.

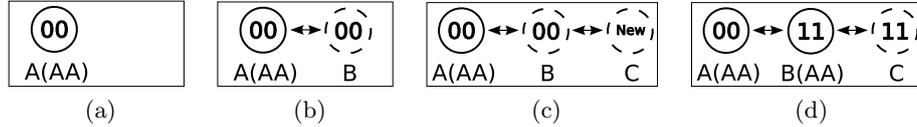


Fig. 2. Joining nodes

#### 4.4 Topology Changes Issues

*Mobility.* From the moving node point of view, when it reaches a new position, it acts like a new node: it requests new address whether it was a standard node or an AA. From the neighborhood perspective, there is no action to take. A node is aware of its moves (or those of its neighbors) by observing changes in its neighbors table.

*Merging.* Merging of networks can be detected when nodes with different network IDs are in contact. One solution makes the node with lower (or higher) network ID reach the other network by requesting for new address just like a new node. An optimized solution would compare the size of the two networks: the nodes from the smaller join the bigger. Estimating the size can be done in a distributed way [9]. In any case, the merging process goes gradually, node per node, avoiding a potential explosion of traffic in the network. At the end, the nodes form a single network with the same network ID.

*Partitioning.* Partitioning of networks doesn't affect the validity of addresses in the resulting partitions. In fact, every partition corresponds to a part of the address plan. Therefore, there is no action to take. And the address space is not reduced because nodes arriving at the previously used partition will use the same addresses as the previous holders.

## 5 Evaluation

### 5.1 Qualitative Analysis

To evaluate the quality of an address autoconfiguration protocol, common metrics are considered [14]:

*Uniqueness.* It used to be the most important criteria for address assignment in ad-hoc networks: two or more nodes aren't allowed to share an address. This is only relevant in the case where addresses serve also as identifiers. In our proposition, addresses are only intended to give location information for routing purpose. The nodes which are located in a nearby area are allowed to share the same address. It is analog to people living in the same house who share the same postal address. This contributes to the savings of address space. It solves also the granularity problem which happens when a level- $n$  of the hierarchy is saturated and can't take more level- $n - 1$ . It happens when all possible addresses within a given area are allocated and a new node starts.

*Protocol Overhead.* How much bandwidth is consumed during configuration processes? As we have seen previously, all the protocol's operations require local communication only. This reduces traffic overhead compared to other mechanisms with multi-hop broadcast communications. But our protocol needs neighboring discovery, so, hello packets are advertised periodically. The bandwidth consumed by these packets increases linearly with the number of nodes in the network. Nevertheless hello protocols can be improved [4], generating less overhead. Comparing to stateful approaches our solution generates fairly the same amount of traffic as these protocols need also periodic updates.

*Latency.* Latency is the time between the joining of an unconfigured node and when it is fully configured with a valid address. The notion of AA reduces latency. If there is an AA in the neighborhood of a new node, the configuration process ends after the exchange of only two messages (assuming a reliable radio connection). If there is no AA, the latency is increased with the time needed for one of the one-hop neighbors to become an AA. This shows that latency increases as much as the distance to the closest AA. Because all configured nodes had an AA in their vicinity prior their address assignation, the probability to not find an AA is equivalent to the probability that every AA in a given region crashes at the same time. It decreases with the size of the region. Hence the probability to not find an AA decreases with distance (again assuming a reliable radio connection). Therefore latency is bounded at least in a reliable network.

*Integration with Routing Protocol.* The efficiency of routing process can be improved with a well designed address assignation mechanism. In our scheme, addresses have a hierarchical structure giving advantages for hierarchical routing protocols which are known to scale well [5]. Another property of the address structure is the relation between addresses and geographic position. Therefore, our protocol can be used along with geographic routing protocols that consume fewer bandwidth [5]. A hybrid approach can also be chosen, combining the hierarchical and geographic aspects.

*Scalability.* Local communication means lower communication overhead and shorter latency leading to good scalability.

*Comparison with Similar Approach.* Compared to [2] the traffic overhead is merely on the same order. But in our solution, nodes don't have to maintain any address range. [10] assigns addresses to nodes with only geographic position, hence without bandwidth cost. Author make the assumption that every node is aware of the global partitioning of the world, what we don't.

*Compatibility with IP Applications.* Our approach needs some adaptations in order to run IP applications. This can be done by using nodes identifiers as "IP addresses". This way the autoconfiguration of the nodes addresses (in the sense of our protocol) and the routing through them is hidden to the application layer. Another approach consists in using our addressing scheme for geocasting,

i.e, multicasting with groups composed with nodes geographically close to each other. These adaptations are necessary for our protocol to be compatible with IP-based trending protocol stack like 6LowPan.

## 5.2 Simulation

In this subsection we focus on a quantitative study of the scalability of the proposed protocol. The simulation was done on ns-3 [1] (ns-3.22 release).

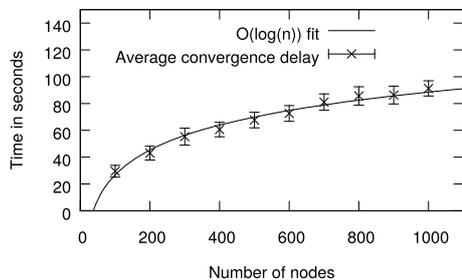
**Scenario and Metrics.** We consider the worst case scenario for the protocol, consisting in unconfigured nodes starting at the same time in a topology connected network. In fact, joining nodes receive addresses quickly when there are already configured nodes in the network. In the case of concurrent starts, address requests are unsatisfied due to unconfigured neighbors. Each node "thinks" that it is alone then assigns itself the 00 address and chooses a random network ID while becoming an AA. After this first configuration, high number of merges occur as each node receives hello packets from different network IDs. At the end of this transient state, the entire network should share the same network ID and addresses should respect the address plan.

To study the scalability of the protocol in this worst case, we chose two metrics and observed their behavior with different network sizes:

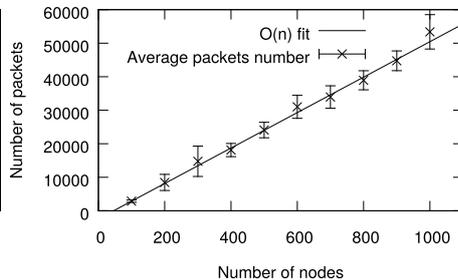
- The *convergence delay* which is the amount of time between when the nodes start and when the last node acquires a valid address with respect to the address plan. In other words, it shows the maximum latency for acquiring a network-wide valid address in a concurrent start case.
- The *number of configuration packets* which is exchanged from the start till the end of the configuration of the last node with a valid address. These packets include address requests, address offers etc. The hello packets are not considered because they are periodically exchanged in any situation.

**Simulation Parameters.** The above metrics are measured with increasing nodes number from 100 to 1000. The nodes are randomly distributed in a square region with a density of 0.375 node per unit  $\times$  unit, each node having a communication range of 3 units. These values ensure a global connected topology network. The density is kept constant for all simulations. The nodes are also kept in static positions. From the point of view of the protocol, mobility is a crash followed by a start because, as a node moves, it must acquire a new address reflecting its new location. Therefore very high mobility can be considered as a massive concurrent start as studied here. The ns-3 wifi model is 802.11b with non-Qos MAC, disabled rate control and set to ad-hoc mode. UDP is chosen as transport protocol. All the means in the data are presented with their respective 95% confidence interval.

**Results.** Figure 3 shows the evolution of the average convergence delay for different simulations, each with different number of nodes  $n$ . Here  $n$  varies from 100 to 1000 nodes. One finds that for a given density, the delay is proportional to  $\log(n)$ . The reason is that each configured node can in turn configure all



**Fig. 3.** Average convergence delay



**Fig. 4.** Average number of configuration packets

its neighbors. Thus the number of "configurable" nodes grows exponentially with the number of already configured nodes. In other words, the number of configured nodes grows exponentially in time even if "slowed" by the merges implying reconfigurations.

In Figure 4 we see that the average total number of configuration packets grows linearly with  $n$ . In fact, the more we have unconfigured nodes, the more configured packets are sent. This linearity shows that for a single node, the average necessary number of configuration packets it has to exchange doesn't depend on the total number of nodes in the network.

To resume, in the worst case of massive concurrent starts in a global connected topology network, the average convergence delay is proportional to  $\log(n)$  and the average number of configuration packets to  $n$ . These values can be considered as bounds for other less constraining scenarios, showing the scalability of our approach in terms of configuration delay and traffic overhead.

## 6 Conclusion and Future Work

We propose a hierarchical addressing protocol for self-organized networks. A node's address reflects both the branch of the hierarchy to which it belongs and its geographic position in the network. Address assignment needs only local information from one-hop neighborhood. Qualitative performance evaluation and simulations results show the scalability of the scheme. In the worst case, the average convergence delay which is the time needed for all nodes to acquire a valid address is  $O(\log(n))$  and the average total number of configuration packets exchanged is  $O(n)$ .

The addressing scheme has been thought to allow the future design of a scalable routing protocol taking advantages from the hierarchical and geographic addresses. A distributed location service will be also necessary in order to follow the identifier-address separation paradigm. Regarding the addressing scheme, further research will be conducted to adapt the pattern to the three-dimensional space.

**Acknowledgments.** Mandimby N. Ranaivo Rakotondravelona's PhD study is supported by Région Réunion and European Union (ERDF).

## References

1. ns-3: a discrete event simulator. <https://www.nsnam.org/>
2. Eriksson, J., Faloutsos, M., Krishnamurthy, S.: DART: Dynamic address routing for scalable ad hoc and mesh networks. *IEEE/ACM Transactions on Networking* 15(1), 119–132 (2007)
3. Fazio, M., Villari, M., Puliafito, A.: AIPAC: Automatic IP address configuration in mobile ad-hoc networks. *Computer Communications* 29(8), May 2006
4. Giruka, V., Singhal, M.: Hello protocols for ad-hoc networks: overhead and accuracy tradeoffs. In: *IEEE WoWMoM*, pp. 354–361, June 2005
5. Hong, X., Xu, K., Gerla, M.: Scalable routing protocols for mobile ad hoc networks. *IEEE Network* 16(4), 11–21 (2002)
6. Hsu, Y.Y., Tseng, C.C.: Prime DHCP: a prime numbering address allocation mechanism for MANETs. *IEEE Communications Letters* 9(8) (2005)
7. Iwata, A., Chiang, C.C., Pei, G., Gerla, M., Chen, T.W.: Scalable routing strategies for ad hoc wireless networks. *IEEE Journal on Selected Areas in Communications* 17(8), 1369–1379 (1999)
8. Kim, N., Ahn, S., Lee, Y.: AROD: An address autoconfiguration with address reservation and optimistic duplicated address detection for mobile ad hoc networks. *Computer Communications* 30(8), 1913–1925 (2007)
9. Le Merrer, E., Kermarrec, A.M., Massoulié, L.: Peer to peer size estimation in large and dynamic networks: A comparative study. In: *IEEE Symposium on International High Performance Distributed Computing*, pp. 7–17 (2006)
10. Li, J., Jannotti, J., De Couto, D., Karger, D., Morris, R.: A scalable location service for geographic ad hoc routing. In: *International Conference on Mobile Computing and Networking*, pp. 120–130. ACM (2000)
11. Nesargi, S., Prakash, R.: MANETconf: Configuration of hosts in a mobile ad hoc network. In: *IEEE INFOCOM*, pp. 1059–1068 (2002)
12. Perkins, C., Malinen, J.T., Wakikawa, R.: IP address autoconfiguration for ad-hoc networks. *IETF draft-ietf-manet-autoconf-01* (2001)
13. Sheu, J.P., Tu, S.C., Chan, L.H.: A distributed IP address assignment scheme in ad hoc networks. *International Journal of Ad Hoc and Ubiquitous Computing* 3(1), 10–20 (2008)
14. Wangi, N., Prasad, R., Jacobsson, M., Niemegeers, I.: Address autoconfiguration in wireless ad hoc networks: protocols and techniques. *IEEE Wireless Communications* 15(1), 70–80 (2008)
15. Weniger, K., Zitterbart, M.: IPv6 autoconfiguration in large scale mobile ad-hoc networks. In: *European Wireless*, pp. 142–148 (2002)
16. Zekavat, S.A., Kansal, S., Levesque, A.H.: *Wireless Positioning Systems: Operation, Application, and Comparison*, pp. 3–23. John Wiley & Sons, Inc. (2011)
17. Zhou, H., Ni, L., Mutka, M.: Prophet address allocation for large scale MANETs. In: *IEEE INFOCOM*, vol. 2, pp. 1304–1311, March 2003
18. Zimmermann, A., Hannemann, A., Schleinzer, B.: IP address assignment in wireless mesh networks. *Wireless Communications and Mobile Computing* 11(3), 321–337 (2011)