Computer Communications 190 (2022) 10–23

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

GAR: Gradient assisted routing for topology self-organization in dynamic mesh networks

Andrey Samuylov, Dmitri Moltchanov, Roman Kovalchukov, Anna Gaydamaka *, Alexander Pyattaev, Yevgeni Koucheryavy

Tampere University, Korkeakoulunkatu 10, Tampere, 33720, Finland

ARTICLE INFO

Keywords:
Wireless mesh
Geographical routing
Virtual coordinate system
Topology organization
Topology maintenance

ABSTRACT

Modern mobile handheld devices, such as smartphones and tablets, feature multiple wireless interfaces, some of which can support device-to-device communications, which enable mesh networks on even when the infrastructure is unavailable. One of the key technological challenges hampering the use of multi-hop mesh networks is the extremely high communication overhead of route discovery and maintenance algorithms. The problem is especially pronounced under dynamic network conditions caused by user mobility and nodes joining and leaving the network. In this paper, we propose a fully distributed algorithm for constructing a virtual coordinate system used for geo-like routing by approximating the physical network nodes coordinate. The proposed algorithm, called gradient assisted routing (GAR), builds upon two-hop neighbors' information exchanged in beacons in contrast to conventional geographic routing protocols which rely on external positioning information. We evaluate the proposed solution using algorithmic, topological, and routing-related metrics of interest. We further numerically quantify how the node mobility increases the time needed for topology stabilization, and how network size affects the route discovery success rate. Our comparison also shows that for small to mid-size mesh networks (up to 60 nodes), the performance of the proposed routing procedure is similar to the conventional geographic routing protocols that exploit external positioning information. The proposed solution may efficiently supplement the traditional on-demand routing in small to mid-size mesh systems by independently establishing 50 to 70% of paths and thereby reducing the discovery overheads.

1. Introduction

Modern handheld terminals such as smartphones and tablets feature several wireless interfaces, e.g., Wi-Fi, WiGig, Bluetooth, LTE, NR. Some of these interfaces support peer-to-peer communications links without the need for infrastructure using technologies such as Wi-Fi direct [1,2], and LTE Sidelink [3,4]. The direct communications support is also expected to be introduced in IEEE 802.11ay and 5G NR systems [5,6], thus utilizing millimeter wave frequencies. These terminals may potentially organize an on-demand network for implementing value-added services anytime and anywhere [7,8].

The research on self-organized mesh networks has demonstrated that enabling efficient routing is critical in service provisioning for mesh networks [9,10]. The available approaches can be divided into two broad groups: table-driven (proactive) and on-demand (reactive) routing protocols. The first group is largely derived from fixed network routing protocols and is suitable for low-mobility scenarios such as wireless sensor networks (WSN) [11]. In conditions where nodes are mobile or go through on/off state changes, the amount of signaling overhead grows significantly [12,13], making proactive protocols infeasible.

On-demand routing protocols such as Ad hoc On-demand Distance Vector (AODV) [27] are commonly suggested for these cases. However, these protocols rely upon network flooding to distribute route request messages in the network, which may also lead to extreme overhead growth [28,29] with network size. Furthermore, if the nodes are mobile, the established routes frequently fail, resulting in the need for additional broadcasts to repair the routes [30].

One of the optimizations available for on-demand routing in dynamic mesh networks is to integrate external location information obtained using, e.g., Global Positioning System (GPS) or cellular/Wi-Fi infrastructure. In this case, the route request packets can be sent towards the destination using the gradients induced by node's positions in the network. A well-known example of such routing protocols is Greedy Perimeter Stateless Routing (GPSR) [31], see [32] for an exhaustive survey. Still, precise positioning information may not always

* Corresponding author.
E-mail addresses: andrey.samuylov@tuni.fi (A. Samuylov), dmitri.moltchanov@tuni.fi (D. Moltchanov), roman.kovalchukov@tuni.fi (R. Kovalchukov), anna.gaydamaka@tuni.fi (A. Gaydamaka), alexander.pyattaev@tuni.fi (A. Pyattaev), evgeni.kucheryavy@tuni.fi (Y. Koucheryavy).

https://doi.org/10.1016/j.comcom.2022.03.023
Received 13 January 2022; Received in revised form 21 March 2022; Accepted 31 March 2022
Available online 5 April 2022

Table 1
Classification of topology organization algorithms.

#	Algorithm	Type	Limitations
1	Coordinate algorithm [14]	Anchor-based	Not realistic link layer models and network topologies
2	Position-free solution [15]	Anchor-based	Possible collapses, additional parameterization is needed
3	GSpring [16]	Anchor-based	Significant signaling overheads due to global info exchange
4	Ibrid [17]	Anchor-based	No mobility
5	HVC [18]	Anchor-based	Reliability upon the cellular infrastructure
6	VCSCClockwise [19]	Anchor-based	Problem of holes in the network
12	TPM [20]	Anchor-based	Heavily depends on the anchor selection and their number
13	Anchor selection algorithm [21]	Anchor-based	Due to relaxations the application to real cases is doubtful
14	EATL [22]	Anchor-based	Only range-based localization algorithm is considered
7	DHT algorithms [23]	Anchorless	May cause flooding, high end-to-end latency
8	VCP [24]	Anchorless	Limited domain and mobility
9	VCP-M [25]	Anchorless	Security, link reliability
10	APS-OTBR [26]	Anchorless	Limited range, static networks
11	GrD-OTBR [26]	Anchorless	Limited range, only small networks are considered

be available at network nodes due to various reasons, e.g., the absence of GPS modules, signal and/or infrastructure, indoor environments, or user privacy.

Another approach to enable scalable routing in wireless mesh networks is to use specific algorithms, called virtual coordinate systems (VCS), for building and maintaining topology at all times such that the source immediately has information about the direction towards the destination. Pioneered in [14], several such algorithms have been suggested in the past. Most of these approaches assume "always-on" static nodes serving as topology "anchors" and thus are not suitable for routing in dynamic mesh environments; the overhead induced by these algorithms is often comparable to that of table-driven routing protocols [24].

In this paper, we propose a novel Gradient Assisted Routing (GAR) VCS for multi-hop mesh networks which induce a routing gradient into the network without the need for positioning information received from external sources such as GPS or cellular infrastructure, and approximating physical topology up to the network graph rotation and translation operations. The proposed VCS distributes and dynamically assigns logical addresses to nodes based on two-hop information periodically exchanged in the physical layer beacons or specifically designated packets and (possibly erroneous) direct neighbor distance estimates obtained via received signal strength (RSS). The system dynamically adapts to changing network conditions, including network merging and disjoining, on/off nodes behavior, and nodes mobility.

The main contributions of the work are:

- new GAR VCS capable of maintaining a virtual topology approximating the physical one in the presence of nodes mobility and nodes joining and leaving the network dynamically;
- performance evaluation of proposed GAR approach, using convergence, stability, and routing-related metrics: (i) mean absolute coordinates deviation at successive execution steps; (ii) topology similarity index; (iii) number of undiscovered routes; and (iv) mean route length;
- comparison of the GAR performance to geographical routing GPSR protocol operating over precise location information available at all times and to the shortest path search algorithm with the global network topology knowledge.

The rest of the paper is organized as follows. We review related work in Section 2. Then, in Section 3, we introduce the main contribution of our work — the GAR VCS for routing in dynamic mesh networks. The algorithms for address resolution and routing are introduced in Section 4. Numerical assessment of GAR VCS is performed in Section 5. Conclusions are drawn in the last section.

2. Related work

Starting from [14], the idea of utilizing virtual coordinates has received considerable attention in the literature. The main aim of VCSs is to explicitly introduce topology to the network by assigning virtual coordinates without the use of positioning support from external entities such as GPS or infrastructure. The approaches proposed so far in the literature can be broadly classified into two categories: so-called anchor-based algorithms that use some "always-on" static nodes in the network to compute its relative positions and anchor-independent algorithms. Principally different VCS construction algorithms characterize these categories (see Table 1.)

2.1. Anchor-based approaches

Most of the proposed VCSs target WSNs, where the information needs to be routed to static "always-on" nodes, i.e., sinks. In a variety of VCS proposals, these nodes serve as logical anchor nodes. All the nodes measure their distance in hops towards these anchor nodes. It can be done using specific packets generated by these anchor nodes and flooded to the network. When nodes are mobile or may dynamically join and leave the network, the intensity of these updates needs to be increased appropriately, potentially leading to substantial signaling overhead.

The study in [15] targets static WSN deployments with a small fraction of nodes serving as sinks. Sinks are assigned unique addresses and flood packets to the networks advertising their logical coordinates. The virtual coordinates are determined using the centroid (mean) of neighbors' coordinates. This process needs to be updated regularly to reflect the current virtual topology. There are still open questions related to the algorithm. Particularly, under some algorithm parameters, the coordinates may collapse to a single point, and thus additional attention to parameterization is needed. The authors claim that their proposal can be extended to support the mobility, but no such topology maintenance algorithm has been reported.

The approach suggested in [16] also targets static WSN deployments. However, instead of relying upon a predefined list of sinks acting as anchor nodes for the topology construction process, it introduces a complicated clustering procedure by exchanging information between nodes in local neighborhoods. The algorithms select leaders, further acting as dynamic anchor nodes. The chain-link topology between anchor nodes is built while other nodes are logically attached to these nodes. The unique feature of the proposed algorithm is the built-in support for avoiding convex regions — a well-known problem in geographical routing [33]. While the algorithm explicitly introduces the network adaptation procedure, the negotiation process does not only rely on the local knowledge of the nodes about its direct neighbors; it also requires global state information to be exchanged between all the nodes. Thus, the VCS construction and, especially, maintenance algorithms introduce significant signaling overheads to adapt the protocol to dynamic network conditions.

The study in [17] questions the uniqueness of anchor-based addressing in VCS built for conventional static VCSs. Notably, the authors demonstrate that a single anchor node is not sufficient to identify a

node in the network uniquely. Consequently, they suggest to introduce (possibly artificially) additional "always-on" anchor nodes to the network and use them all together to determine the unique location of the rest of the nodes in the network. The study in [18] further notices that selecting anchor nodes uniformly over the network may improve VCS performance. They advocate introducing anchor nodes according to the hexagonal grid and measure the distance (in terms of hops) to the nearest anchor node. Although the authors do not explicitly specify what kind of anchor nodes are considered, one may rely upon the cellular infrastructure to build VCS. In [19], the authors suggest a clusterization hierarchy to induce the uniqueness of virtual addresses in their VCS. Notably, a node located in the logical center of the network is chosen as an anchor. Then a clockwise structure is built around it by successive application of the proposed algorithm. The VCS performance heavily depends on the choice of the anchor node and its position in the network, making the algorithm suitable for symmetric deployment.

Additionally, flooding of signaling packets is needed to facilitate VCS construction. Unfortunately, the authors did not provide any topology maintenance algorithm, leaving the question of VCS applicability to the case of dynamic network conditions open. Furthermore, using a chain-like structure to connect all the nodes in the network makes the use of resources inefficient. The studies in [20], [21], [22] further address the issue of the optimal number of anchors for a network and how they are chosen.

2.2. Distributed approaches

An alternative approach when anchor nodes are not available is to create structured topology using algorithms like distributed hash tables (DHT) [23]. There are a few inherent problems associated with this approach, including the selection of initiating topology construction nodes, conflict resolution, and, especially, topology maintenance algorithms in dynamic network conditions. The latter is a critical problem as DHT-like solutions are known to be characterized by topology update delays when nodes join or leave the network or change its position in the structure.

A fully distributed approach for routing in mesh networks is proposed in [24] specifying VCS, data lookup, and routing functionalities in a single bundle. The cornerstone of the proposal is an algorithm forming a chain-like topology. The algorithm is fully distributed and is based on one-hop neighbor availability information exchanged in beacon packets. Besides, it also specifies maintenance algorithms in case of nodes leaving and joining the VCS. The resulting structure guarantees the existence of a route between any two nodes in the network and is based on local knowledge of the node's neighborhood. Nevertheless, the major shortcoming of the proposed solution is that the path is far from the shortest one discovered by conventional on-demand routing protocols. This property has drastic consequences in terms of mesh network performance. Notably, network throughput becomes bounded by the capacity of a single link; nodes in the logical chain center spend more of their resources, including both bandwidth and energy, compared to other nodes. They may run out of the operation quickly, negatively affecting user experience and compromising the overall network performance. A similar approach has been proposed in [25].

In [26] authors suggest a new anchorless algorithm, named Organized Topology Based Routing (OTBR), which can be applied in both static (APS-OTBR) and dynamic (GrD-OTBR) use cases. Simulations show that static algorithm shows good results in sense of node utilization frequency, while dynamic has stable performance when network changes.

2.3. Summary and open problems

Most of VCSs proposed to date target static networks such as WSNs, where there is the need to route information towards one or few "always-on" static sinks. This scenario affects the choice of the VCSs construction logic, relying on sinks or infrastructure as anchor nodes. Thus, nodes in VCSs estimate their position in the network with respect to those anchors using, e.g., the number of hops towards one or few anchors as a locally unique address. The core logic of anchor-based VCS construction algorithms cannot be adapted to the case when all the nodes may act as potential destinations since there are no guarantees that nodes addresses are unique. Address uniqueness, however, can be induced by adding more anchor nodes to the network. Finally, anchor-based VCS systems usually rely upon broadcast transmissions to infer the number of hops to the destination and thus induce significant overhead for topology maintenance in meshes with on/off and mobile nodes behavior.

Anchor-independent VCSs create DHT-like topologies in the network using a local exchange of beacons packets between adjacent nodes. These algorithms are often fully distributed, do not require anchor nodes in the network, and support features for topology maintenance in case of node dynamics. Contrarily to the anchor-based solutions, these algorithms route packets between any nodes in the network. Yet, due to the specifics of these VCSs, the path between arbitrarily chosen nodes might be far from the shortest one. They are also characterized by inefficient resource utilization at network nodes.

None of the proposed VCSs allows creating and maintaining virtual network topology simultaneously providing the following features: (i) any-to-any communications, (ii) shortest path in the logical network closely approximating that of physical topology, and (iii) supporting nodes leaving/joining the network and their mobility. In the next section, we report the necessary algorithms capable of creating VCS that provides these features.

The overall rationale behind the proposed solution is to reduce the routing overhead in arbitrary mesh networks in any location such that it may scale linearly with the number of nodes in the network as compared to other routing solutions, which typically scale quadratically. Geographical routing is known to have this scaling behavior. However, since nodes may not have geographical coordinates available at all times, we propose a topology formation and maintenance algorithm that relies upon only two hop information about its neighbors that can be exchanged in beacon (keep-alive) packets. Importantly, and in contrast to distance vector protocols, such beacon packets would have fixed size, independent of network size. Such an approach can be useful not only on conventional terrestrial mesh networks but also for unmanned aerial vehicle (UAV) fleets performing missions indoors or search and rescue operations in forests or mountains. Thus, the main advantages of the proposed approach can be summarized as: (i) linear overhead scaling for both VCS construction and routing as the number of nodes in a network grows, (ii) ability to perform in environments, where GNSS system is not available.

The research interest in the context of VCS is expected to grow in the coming years. The rationale is the appearance of new application areas including (i) infrastructure-free mesh networks [34–36] and (ii) unmanned aerial vehicles (UAV) fleets performing missions in complex environments such as indoor or search and rescue operations in deep woods or mountains, where global navigation positioning systems (GNSS) are not available [37–39]. Infrastructure-free mesh networks mainly rely upon AODV-like solutions preventing these systems from scaling to a large number of nodes. At the same time, the problem of topology maintenance of UAV fleets in complex environments has not received serious attention so far as most of the studies presume availability of GNSS information or resort to specific methods. For example, the authors in [37] propose to use a method of dead reckoning, i.e. calculate new coordinates using known initial coordinates and motion parameters. Infrastructure-based solutions are discussed

in [40]. In the context of infrastructure-less environments only inertial-based solutions have been considered so far [41]. However, these solutions are inherently biased due to accumulation of positioning errors especially at high speeds.

3. Proposed coordinate system

In this section, we describe the proposed GAR VCS for a multi-hop mesh network. First, we outline the general idea, introduce a mathematical model of the system and then proceed to specify a candidate algorithm for constructing and maintaining topology under on/off and mobile node dynamics.

3.1. Requirements

The primary purpose of the proposed GAR VCS is to provide a capability for a node to calculate its virtual GAR address based on the locally available knowledge only to minimize the overheads of information dissemination over the whole network.

We consider a network of n mobile devices located in some 2D space, $n < \infty$. The device is unable to determine the geographic coordinates of its location. Each device (network node) has a unique identification number i, $i = 1, 2, \dots, n$. Node i is capable of receiving a signal from other nodes within some radius. Based on a received signal strength (RSS), node i can estimate the physical distance $d_{i,j}$ between itself and another node j.

We denote a set of one-hop neighbors of the node i as N_i. Here, one-hop neighbors are the adjacent nodes directly available for communication via radio interface without any intermediate relay nodes. Two-hop neighbors of node i are the nodes that require a one-hop neighbor relay to transmit data to node i. A set of two-hop neighbors of the node i is denoted as $M_i = (\bigcup_{j \in N_i} N_j) \setminus N_i$. The sets of one-hop and two-hop neighbors do not overlap.

Given that the geographic coordinates of nodes are unknown, we aim at building a virtual topology that reflects the physical one up to the rotation, translation, and scale operations. In virtual topology a node i has its virtual coordinates $s_i = (x_i, y_i)$. Since the nodes are constantly moving in the network, their physical coordinates change with time, and so do the virtual ones. For this reason we introduce another index — time step, t. The aim is to find virtual coordinates $s_{t,i} = (x_{t,i}, y_{t,i})$ of node i at step t.

To build a virtual topology that reflects the physical one up to the rotation, translation, and scale operations, we assume the following information is available at each node:

- list of unique destinations identifiers (ID);
- list of one-hop neighbor nodes' unique IDs, their GAR addresses and distance assessment to them obtained via, e.g., RSS;
- lists of two-hop neighbors, i.e., for each one-hop neighbor, a list of its neighbor nodes' IDs and their GAR addresses.

Here, one-hop neighbors are the adjacent nodes directly available for communication via radio interface without any intermediate relay nodes, and two-hop neighbors are the nodes available via only a one-hop neighbor relay. Therefore, these two sets do not overlap. We note that more than two-hop neighbor information can be used by the algorithm specified below. In general, the higher the amount of information supplied to the algorithm, the better its accuracy. We limit it by two-hop neighbors providing the trade-off between the overhead and accuracy of the algorithm.

Most of the abovementioned information is received from neighboring nodes through exchanging signaling packets, while distance assessment may be obtained by measuring RSS and then applying free-space propagation loss model or by any other means available, including empirical assessment based on the number of common neighbors. IDs must be stored beforehand in a phone book style. The rest of the parameters should be predefined by a per-node basis or globally based on the type and characteristics of a radio interface in use and general considerations regarding the network in question.

No specific restrictions are imposed on the nodes or the purpose of the network. For instance, nodes can move freely, temporarily switch on and off, etc.

3.2. Address allocation algorithms

In the section, we propose two algorithms for establishing the GAR virtual addresses. Those are supposed to be used in a distributed manner by the nodes in the network, enabling nodes to build a virtual topology for communication or support of a separate routing protocol on top of the virtual topology.

3.2.1. Single-step address allocation

The first algorithm, called *single-step allocation*, attempts to minimize the time it takes for the whole network to establish acceptable quality GAR addresses for further use. It is achieved by solving a local multilateration problem [42]. The algorithm constructs a target optimization function that incorporates distance assessments to one-hop neighbors and penalties based on the two-hop neighbors. The optimization problem is solved to obtain the most fitting virtual address. This algorithm best suits for nearly static networks with limited mobility.

Consider the following optimization function

$$F_o(s_{k,j}) = \sum_{j \in N_i} \left[F_1(\rho(s, s_{k,j}), d_{i,j}) + \sum_{j \in M_i} F_2(\rho(s, s_{k,j}), d_{i,j}) \right], \tag{1}$$

where $\rho(s_i, s_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$.

The main idea of the optimization function (1) is to find the virtual coordinates of node i by minimizing the penalty assigned to it when a set of rules is not obeyed. The first rule is that the distance between the node i and one-hop neighbors should be around a certain predefined limit d_n. Node i should not be closer than a physical distance estimation, and at the same time should not move away too far, since the one-hop neighbors be definition. The second rule is that the distance between the node i and two-hop neighbors should be more or equal a certain predefined limit d_m. Otherwise a two-hop neighbor falls inside the area where node i is capable to receive a signal from node j and node j becomes a one-hop neighbor by definition. The described rules are reflected in (2) and (3).

The function $F_1(\rho(s_i, s_j), d_{i,j})$, $j \in N_i$ operates with the current node i and its one-hop neighbors N_i. It compares the virtual distance $\rho(s_i, s_j)$ between node i and its one-hop neighbor j (calculated via Euclidean distance) and the estimation of physical distance between i and j and assigns a certain penalty based on its difference. In general form the function is defined:

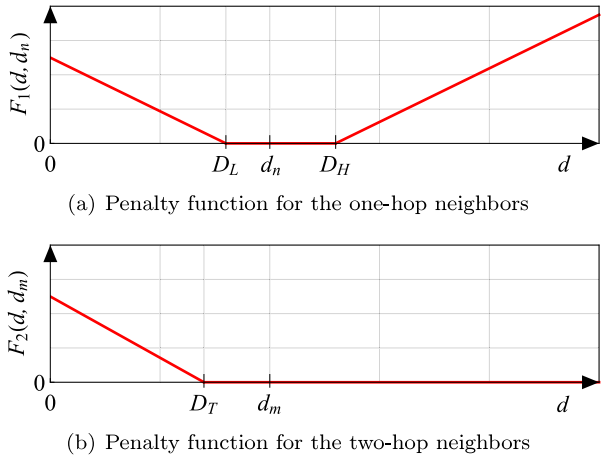
$$F_1(d, d_n) = \begin{cases} \|d - D_L\| & d \le D_L(d_n), \\ 0 & D_L(d_n) < d \le D_H(d_n), \\ \|d - D_H\| & D_H(d_n) < d, \end{cases} \tag{2}$$

Similarly to (2), function $F_2(d, d_m)$ penalizes the current node i for being too close to any of its two-hop neighbors $j \in M_i$, as they are not in the coverage. It is given by

$$F_2(d, d_m) = \begin{cases} \|d - D_T\| & d \le D_T(d_m), \\ 0 & d > D_T(d_m). \end{cases} \tag{3}$$

An example of the algorithm application is illustrated in Fig. 2. Here, nodes $s_{t,j}$, as one-hop neighbors, influence the coordinates of node $s_{t+1,i}$ and force it to move in the regions with minimum penalty (depicted with blue color).

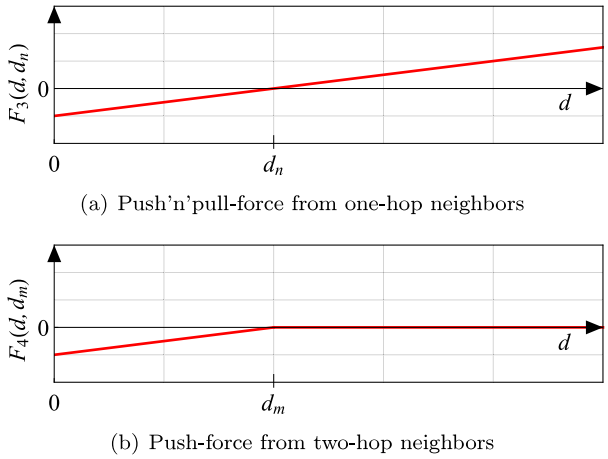
In the formalization (1)–(3), $F_o(s)$ is the optimization function, where $s = \vec{s} = (x, y)$ is the resulting pair of virtual coordinates for the node in question to take for the current step; N_1 is the set of all one-hop neighbors, while $N_{2,n}$ is the set of their corresponding two-hop neighbors, except those that are direct one-hop neighbors. Vector



(a) Penalty function for the one-hop neighbors

(b) Penalty function for the two-hop neighbors

Fig. 1. Penalty functions $F_1(d, d_n)$ and $F_2(d, d_m)$.



(a) Push'n'pull-force from one-hop neighbors

(b) Push-force from two-hop neighbors

Fig. 3. Force $F_3(d, d_n)$ and $F_4(d, d_m)$ as functions of virtual and physical distances.

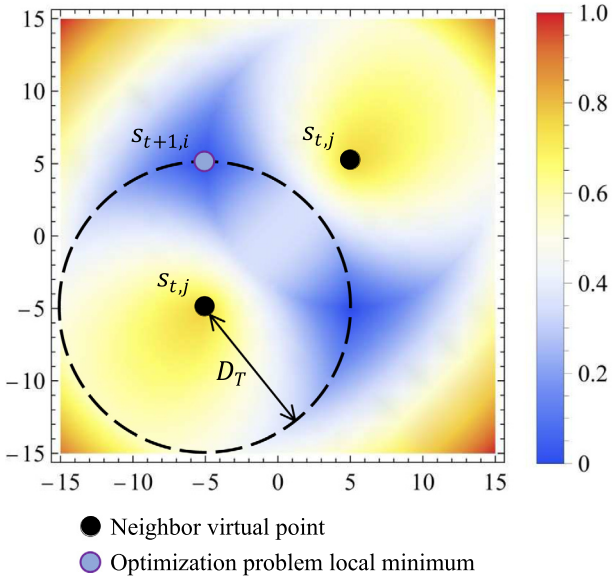


Fig. 2. Example of single-step address allocation algorithm. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$c = \vec{c}_n = (x_n, y_n)$ holds the virtual coordinates currently known for node n; d_n is the assessed distance between the current node and node n; while D_L, D_H, and D_T are the fine tuning parameters controlling the "looseness" of the optimization problem.

Two functions $F_1(\vec{d}, \vec{d}_n)$ and $F_2(\vec{d}, \vec{d}_m)$ in (1) are the penalty functions. The function $F_1(\vec{d}, \vec{d}_n)$ is being used for multilateration purposes to place the current node into a point that complies with the distance measurements to one-hop neighbors. Function $F_2(\vec{d}, \vec{d}_n)$ penalizes the current node for being too close to any of its two-hop neighbors, as they are not in the coverage. The structure of functions $F_1(\vec{d}, \vec{d}_n)$ and $F_2(\vec{d}, \vec{d}_n)$ is illustrated in Fig. 1. We note that the structure of these functions used in different algorithm implementations may differ from the depicted ones. Particularly, nonlinear functions can be used.

3.2.2. Gradual address convergence

The algorithm considered in the previous section attempts to determine the positions of nodes in the GAR network in one step and is best suited for networks with no or low mobility. However, if the nodes forming the network are characterized by high or mixed mobility or on/off dynamics, abrupt changes in the network topology may lead to temporal divergence from the current optimal virtual topology. To address these cases, we propose the second algorithm. This algorithm induces gradual convergence into the address allocation process and forces the virtual network to follow the physical topology gradually. On top of this, we employ the simplified formalization of the optimization problem, specifying the current gradient.

The second algorithm, called *gradual convergence*, favors a slower approach to establish a node's virtual address by allowing nodes to push and pull each other with different forces depending on the virtual and physical distances between them. It enables nodes to create a topology in a more organized and relaxed manner. This algorithm, characterized by slower convergence, can be applied to dynamic mesh networks, where nodes may join and leave the network as well as move. Updated virtual coordinate for the current node is computed as follows

$$s_{k+1,i} = s_{k,i} + \left[\sum_{j \in N_i} \frac{s_{k,j} - s_{k,i}}{\rho(s_{k,i}, s_{k,j})} F_3(\rho(s_{k,i}, s_{k,j}), d_{i,j}) + \right.$$
$$\left. + \sum_{j \in M_i} \frac{s_{k,j} - s_{k,i}}{\rho(s_{k,i}, s_{k,j})} F_4(\rho(s_{k,i}, s_{k,j}), d_{i,j}) \right] \Delta, \quad (4)$$

where $F_3(d, d_n)$ and $F_4(d, d_n)$ are given by

$$F_3(d, d_n) = \frac{A}{d_n}(d - d_n),$$
$$F_4(d, d_n) = \begin{cases} \frac{A}{d_n}(d - d_n) & d \le d_n, \\ 0, d > d_n. \end{cases} \quad (5)$$

In (4), $s_{k+1,i}$ is a virtual coordinate of node i in iteration $k+1$ updated from $s_{k,i}$ recursively, and Δ is the parameter defining the relative convergence step size. Functions $F_3(d, d_n)$ and $F_4(d, d_n)$ control the force with which the current node is pushed or pulled by its neighbors. The structure of functions $F_3(d, d_n)$ and $F_4(d, d_n)$ is illustrated in Fig. 3. We note that the structure of these functions used in the actual algorithm may differ from the presented ones. For both algorithms, in the cases when there are no neighbors and a node attempts to establish its GAR virtual address for the first time, it can be initialized with a pseudorandom address, which may be modified later as neighbors come into range.

The proposed VCS algorithm can be extended to multi-interface operation. In this case, the distance estimation to the neighbor nodes can be performed using all the available interfaces, improving its accuracy. Furthermore, no specific requirements on the GAR address format are introduced. For example, one may use a two-digit address (i.e., X and Y coordinates as in our case) that may not be individually unique.

The application of the algorithm is illustrated in Fig. 2. Here, we calculate new virtual coordinates for the node i. Before calculation it

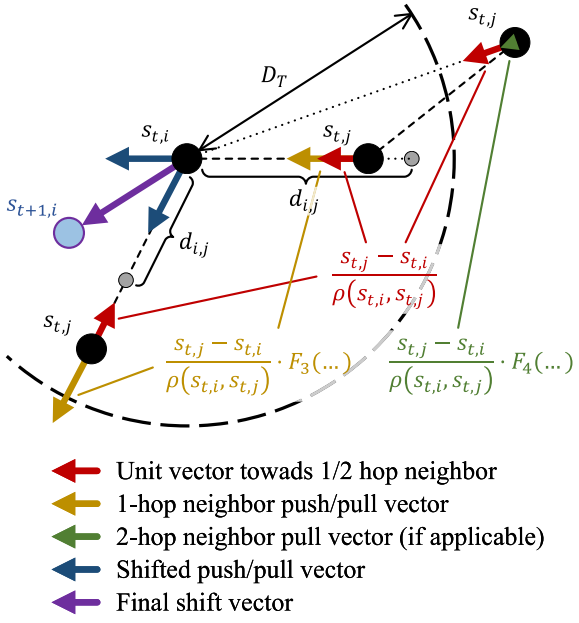


Fig. 4. Example of gradual address convergence algorithm.

has virtual coordinates from the previous iteration, $s_{t-1,i}$. We aim to find virtual coordinates at the iteration t, i.e. $s_{t,i}$. For it, we consider two sets: one-hop and two-hop neighbors of node i. In our case there are two one-hop neighbors, and one two-hop neighbor. The resulting forces are as shown in Fig. 4.

3.3. Procedure and algorithm

The step-by-step procedure for VCS construction is illustrated in Fig. 5. Note that this procedure is executed at individual nodes upon reception of update messages from its neighbors. The block "Update Virtual Coordinates" presumes execution of single-step or gradual convergence algorithms described previously in this section. The pseudo-code of the algorithm is presented in Algorithm 1. Recall that its aim is to find virtual coordinates $s_{k,i} = (x_{k,i}, y_{k,i})$ of node i at step k of algorithm iteration.

4. Name resolution and routing

In this section, we introduce the upper layer functionality necessary to establish routes and to perform GAR address resolution for multi-hop mesh networks. These functions rely on the previously proposed VCS, which provides the topology construction and maintenance.

4.1. GAR address allocation

The proposed solution assumes a global address system, such as the one defined in ITU-T Recommendation E.164 [43]; however, GAR addresses may also be used locally during a local network existence time. From the user perspective, the system operates over unique global IDs, for example, "Alice" and "Bob".

In this paper, we assume that the communication process relies on a IDs distribution system which provides unique identifiers in the form of a unicode string to those parties who want to communicate to each other. The particular implementation of the names distribution system may depend on many side factors and is thus out-of-the-scope of this paper. Specifically, these IDs can be based on email addresses, phone books, URIs, hierarchical naming schemes, social network usernames, etc.

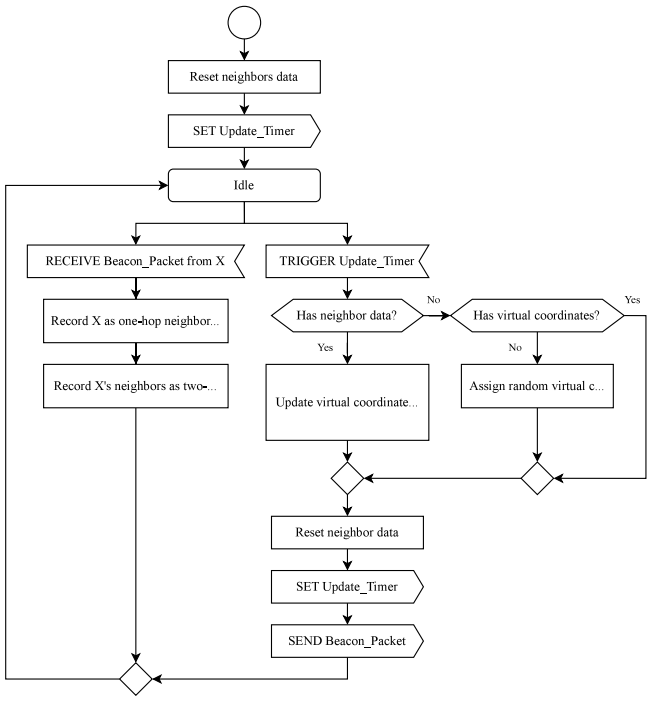


Fig. 5. VCS coordinates update algorithm.

Algorithm 1: VCS Update Algorithm

Input:

- signaling data packets $p_j \in P_i$ with data:
  - j - ID of the one-hop neighbor of node i
  - $s_{k,j}$ - virtual coordinates of node j
  - N_j - a list of one-hop neighbors' IDs of node j
  - a set of virtual coordinates of N_j
- $d_{i,j}$ - an estimate of the physical distance between nodes i and j

Result: virtual coordinates $s_{k+1,i} = (x_{k+1,i}, y_{k+1,i})$ of node i at iteration k of the algorithm

1. During a specified time interval Δt node i may receive signaling data packets p_i;
2. Calculate $s_{k+1,i}$;

if $P_i = \emptyset$ **then**
  if $s_{k,i} \neq NULL$ **then**
    $s_{k+1,i} = s_{k,i}$
  else
    $s_{k+1,i} = rand()$
else
  if *low mobility* **then**
    Single-step Address Allocation
  else
    Gradual Address Convergence

3. Broadcast p_i, $p_i = \{i, s_{k,i}, N_i, s_{k,j}, j \in N_i\}$;
4. Repeat from point 1

Specific names distribution systems implementations may also depend on the specific use cases and applications that will be communicating over the GAR network. Particular applications may be working only if users know each other's name (e.g. video calls, email-like services), while other applications may rely on centralized, decentralized, or distributed IDs search mechanisms (e.g., content delivery services).

Also, some of the applications that can work using GAR may not need IDs at all. For example it might be "Infrastructure as a Service" recommendation systems, proximity based services or underlay for Network as a service and Security as a service.

4.2. GAR address resolution

GAR address resolution corresponds to the so-called *location service* [44] in geographic routing protocols; in the considered system, the resolution can be performed using two options. According to the first option, when cellular infrastructure is available, it can be assigned the role of an address resolution service. All nodes in the network shall register and regularly update their address at a specially designated database. The update intervals are implementation-specific and may depend on the node velocity. In this case, to discover Bob's GAR address, Alice sends a RETRIEVE message to the database and receives a RESOLVE response with the current address associated with Bob's name.

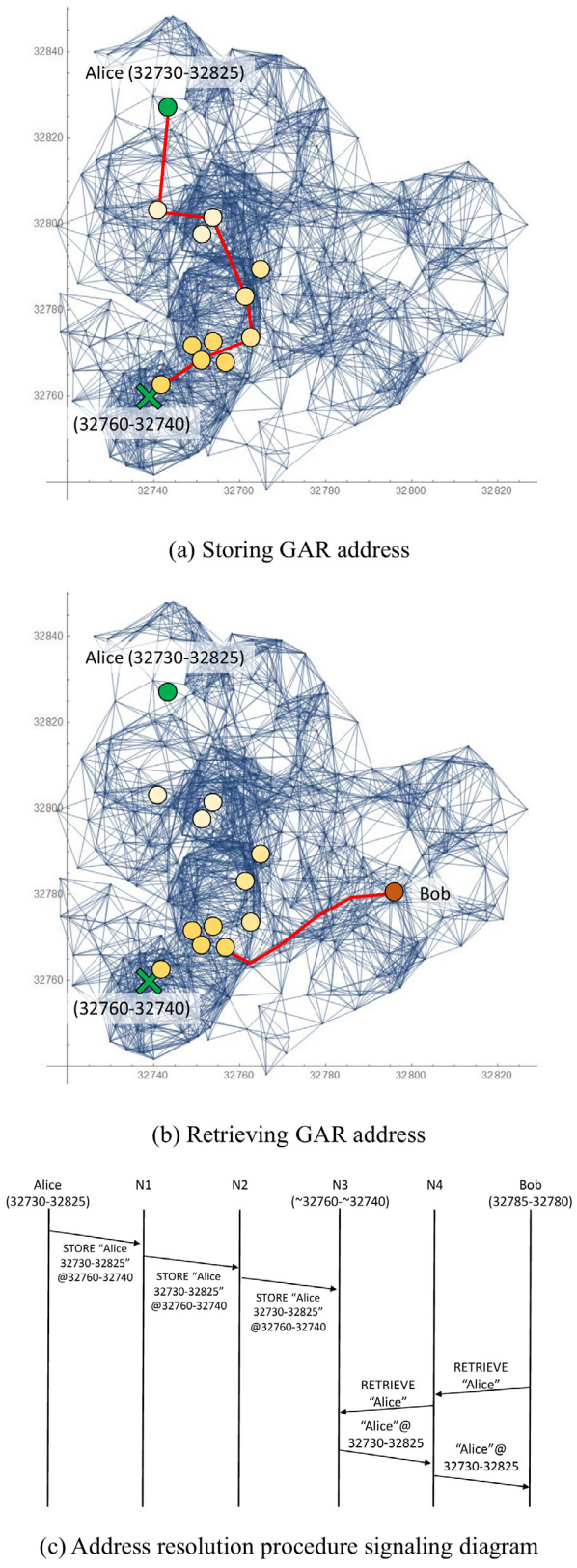
The second option, when infrastructure is unavailable, is to utilize a DHT [45] address storage system. The idea is to use a hash function to convert any ID to a GAR address, which points to where the GAR address of this node must be stored. Although the choice of the hash function is implementation-specific, it must return the GAR addresses in the same format as used in the network.

Consider an example of communications between two nodes: Alice and Bob. For instance, we assume that Bob wants to start communication with Alice. Before Bob can start sending data to Alice two address resolution operations should be completed. First, as soon as Alice joins the GAR network she needs to send a STORE message to the network. Second, Bob needs to send a RETRIEVE message to the GAR network. Both STORE and RETRIEVE messages have the same destination, based on a common for all GAR nodes hash function that translates unicode string values (IDs) to GAR VCS addresses. If Bob successfully retrieves Alice's virtual coordinates address from GAR network, he will be able to start communication.

A typical example of the storage and resolution processes is illustrated in details in Fig. 6. First, Alice uses a hash function to convert its name "Alice" to a GAR address, e.g., 32760-32740. Alice then sends a STORE message (see Fig. 6(a)) using GAR topology towards 32760-32740 containing the mapping of the ID "Alice" to its current GAR address, e.g., 32730-32825. To summarize, in this case, a STORE message will be: "Alice 32730-32825"@32760-32740. One can read this message as "Alice's current address is 32730-32825, please store it at the node closest to address 32760-32740, which is her ID's hash result". The intermediate nodes, which will receive or relay this message, may also decide to temporarily store the mapping for faster future address resolution.

Fig. 6(b) considers the case of address retrieval. Assume Bob wants to discover the route to Alice. In this case, he hashes (using the same hashing function as Alice) her ID "Alice" and obtains the GAR address 32760-32740. He then sends a RETRIEVE message containing ID entry "Alice" using the GAR topology towards this address. Any node that discovers the GAR address corresponding to the entry "Alice" responds with a RESOLVE message containing Alice's GAR address. Storage and retrieve progress in time is shown in Appendix A and as signaling diagram on Fig. 6(c)).

Both infrastructure-based and DHT storage of the IDs to GAR address mappings can be used simultaneously. For example, if the infrastructure is temporarily inaccessible, the DHT address storage system shall be utilized.



(a) Storing GAR address

(b) Retrieving GAR address

(c) Address resolution procedure signaling diagram

Fig. 6. In-network address resolution approach.

4.3. Applications of GAR VCS

Two applications corresponding to different operational regimes are envisioned for the proposed system. In stand-alone GAR mode, the GAR VCS can be used as is in the "opportunistic" mode without any additional functionality to provide specific services such as instant messaging. In this case, once Bob obtained Alice's GAR address, he may

immediately send a packet containing a short message to Alice without the need for route establishment.

Furthermore, the proposed GAR VCS can be used in the "routing underlay" mode, to support on-demand routing protocols. Specifically, assuming the use of AODV protocol, once Bob obtains Alice's address as in Fig. 6, the route request packet (RREQ) of AODV is not broadcasted to all nodes as it usually happens. Instead, RREQ is sent towards Alice's GAR address utilizing unicast communications based on the principles of geographical routing [46].

For example, one can use a greedy algorithm to broadcast the message, i.e., when the distance to the destination is shorter than that of the previous hop [31]. In addition, similarly to GPSR extensions [47,48], one may utilize multicast communications at each hop by sending the routing packets to few nodes that are geographically closer to the destination to improve the performance of the GAR VCS in the "routing underlay" regime.

Besides, the proposed solution may not be the only option in operational wireless mesh networks. For delay-tolerant applications such as data transfer, one may always opportunistically utilize GAR VCS first, potentially reducing the amount of routing overhead in the network and use conventional on-demand protocols such as AODV as a fallback option [49,27].

4.4. Routing complexity on top of GAR

To quantify routing complexity on top of GAR and compare it with other protocols, consider an example of a hypothetical network graph with N nodes and L edges. Define overheads in terms of rate of packets that need to be sent by a all nodes in the network. Thus, asymptotic best-case overheads will be defined as $o(f)$, and worst-case overheads as $O(f)$ where f is some function.

OSPF-like protocols will immediately initiate flooding across the entire network, ultimately using at least one packet on every link. Since each state update is critically important for correct operation of OSPF, broadcasting can only be used if all neighbors send ACK packets in reply, which would be almost the same size as link state advertisements, thus reducing the benefit of broadcasting. The intensity of updates propagating through the network at any time is directly proportional to L (assuming fixed intensity of link state changes), and thus the cost of running OSPF is $o(NL)$. On-demand routing protocols have essentially the same issues with scaling, since any new route construction still causes flooding consuming resources as $o(L)$, with total cost of running scaling as $o(NL)$ (for a fixed intensity of connection requests). As a result, on-demand routing provides benefits only when connection intensity is low.

In contrast, a distance vector protocol has a fixed intensity of updates (which is set by the administrators). Unfortunately, the size of DV updates grows with N as distance vector should include all reachable nodes in the network, and thus their overhead is $o(NL)$ for a given update intensity.

Now, consider a virtual coordinate system such as GAR. Similarly to DV protocols it has a fixed intensity of updates (which is set by the administrators), but on top of that it also has fixed update size which does not depend on the size of the network. As a result, its total overhead as seen by a link is $O(1)$ and does not depend on the size of the network, or its traffic pattern, with total network-wide cost proportional to $O(N)$, as desired. Clearly, when GAR cannot resolve a path on its own, additional overheads are introduced, which are no worse than in the case of conventional on-demand routing.

5. Numerical assessment

In this section, we evaluate the performance of the proposed GAR VCS system using algorithmic, topological, and routing-related metrics of interest. We first introduce the performance metrics we use to compare the physical and virtual topologies. Then, assess the performance of the proposed GAR VCS under various levels and types of network dynamics. Finally, we compare performance of different geographical routing algorithm on top of GAR.

We evaluate performance by concentrating mainly on GAR VCS performance. The rationale is to test whether the proposed topology organization and maintenance algorithm shows promising results and can be considered as a candidate for geographical routing on top of it. Nevertheless, in our simulations we also captured routing-related metrics by benchmarking the performance of the proposed protocol to those of GPSR and perfect Dijkstra algorithm as discussed below. The performance evaluation campaign has been carried out by utilizing specifically developed Matlab code implementing the GAR VCS system on the graph of the physical network topology. The simulator has been equipped with tools capturing system dynamics, i.e., ability to dynamically remove and add nodes as well as move them according to random direction mobility (RDM) model.

5.1. Performance metrics

In this section, we introduce three types of metrics we will utilize to assess GAR performance. These are: (i) algorithmic metrics reflecting convergence and stability properties, (ii) topological metrics describing similarity between virtual and physical topologies and (ii) routing metric reflecting the routing performance.

5.1.1. Algorithmic metrics

To assess convergence and stability properties of GAR VCS, we utilize the *mean absolute coordinates deviation* at each step of the algorithm at all the nodes. In our example, the period of exchange between network nodes is equal to the beacon interval in IEEE 802.11n technology, which is 102.4 ms. The mean absolute deviation is estimated by averaging the difference between the previous coordinates of nodes and the current coordinate over all the nodes in the network. When plotted against time, this feature visualizes the convergence and stability of the algorithm. Note that the mean absolute deviation does not directly characterize the quality of the obtained virtual topology.

5.1.2. Topological metric

As the primary indicator of the similarity between the network topology and the virtual one, we utilize the so-called *topology similarity index*. This index is defined as the Pearson correlation coefficient [50] between pairwise distances in the virtual and physical topologies.

5.1.3. Routing metric

We also investigate and compare the performance of the proposed solution and conventional routing algorithms. For comparison purposes, we select the Dijkstra's algorithm and GPSR [31] protocol. First, Dijkstra's algorithm represents a perfect model of a routing protocol as it always finds the shortest path to the destination using the complete global knowledge of the network topology. Second, GPSR represents a typical geographical routing protocol that operates using accurate positioning information.

GPSR is a geographical routing method, where the route discovery packets are relayed to the node that is geographically closest to the destination. For GPSR protocol, we assume full knowledge of real geographical coordinates via, e.g., GPS. Both GPSR and GAR use a greedy path discovery algorithm that broadcasts the message further if the distance to the destination is shorter than that of the previous hop.

We use the following routing metrics:

- *Fraction of undiscovered routes* reflects the ability of the routing protocol to discover paths in a network. We compute it as a fraction of routes that cannot be discovered using the algorithm to all the routes in the network.

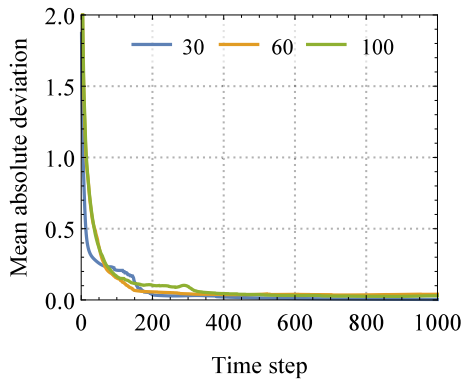


Fig. 7. Mean absolute deviation for static scenario.

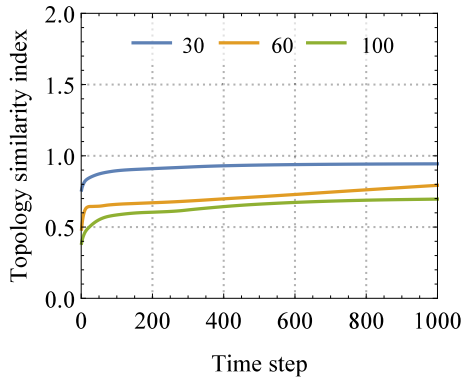


Fig. 8. The topology similarity index for static scenario.

- *Mean route length* reflects the performance of routing over the maintained topology, comparing the distances in the number of hops between all nodes in the network. Particularly, we compare the average distance of discovered paths in a network using the proposed GAR VCS to GPSR, and the Dijkstra's algorithm.

Note that for both Dijkstra and GPRS we capture only the algorithmic part that evaluates the shortest path to the destination. In this context, Dijstra algorithm provides an upper bound on the performance that cannot be attained by any other algorithm, while GPSR (or more, specifically, its algorithmic component forwarding packet only if the distance to the destination from the receiving node is smaller than the distance in the packet) provides an upper bound for geographical routing algorithm. The proposed algorithm based on GAR VCS should always perform worse then these two, but the quantitative difference between GPSR and geographical routing on top of GAR VCS indicates the performance loss due to unavailability of the GNSS information and relying upon VCS for topology management.

5.2. Scenarios and parameterization

To perform the comparison, we consider 3 network sizes with $N = 30$, 60, 100 nodes in square area of $160\,\text{m} \times 160\,\text{m}$, $210\,\text{m} \times 210\,\text{m}$ and $275\,\text{m} \times 275\,\text{m}$, respectively. The coverage of each node is assumed to be 50 m, which corresponds to modern Wi-Fi systems operating distance in the microwave ISM 2.4 GHz band. To assess the performance of the proposed VCS, we consider three following scenarios:

- *Static scenario.* In this case, we assume that nodes are randomly and uniformly distributed in the considered area and remain static over time. All the nodes are immediately available in the system at the initial time. This scenario aims to illustrate the network formation process and assess GAR VCS performance for nearly static scenarios such as crowded public events.
- *Nodes churn scenario.* At the start of the simulation, there are no nodes in the network. Nodes appear every 10 s one-by-one up until all N nodes are on. Once all the nodes are in on state, the network remains static for 10 s. Then, every 10 s, a randomly selected node is removed from the network, and after another 10 s, a new node with a randomly and uniformly chosen position is added to the network. The procedure repeats with the next random node until the simulation is over.
- *Mobile scenario.* The initialization period in this scenario is similar to the previous one. However, all the nodes, as soon as they appear, start moving according to the random direction mobility (RDM) [51] model, which has been widely used for ad hoc network performance analysis [46,52,4]. Wrap-around is used to handle the boundary problem [51].

5.3. Representative results

5.3.1. Static scenario

We study the basic static scenario concentrating first on the stability properties of the proposed solution. Fig. 7 illustrates the mean absolute coordinates deviation at successive algorithm time steps at network nodes as a function of the number of nodes. Namely, this metric measures the coordinate updates' stability and evaluates the network convergence time. As one may observe, for two considered cases with 30 and 60 nodes, the metric of interest decreases exponentially, reaching negligible deviations already after 100–200 time steps. As the beacon interval is set to 102.4 ms, this results in the network convergence time of approximately 20 s. Note that for 100 nodes, the convergence time is significantly larger and may take up to 400 time steps.

We further analyze the structural similarity of the physical and virtual topologies. Fig. 8 shows the topology similarity index as a function of the successive algorithm time steps at network nodes. Analyzing the presented data, one may observe that despite small coordinate deviations starting from time step 200 (see Fig. 7), the algorithm continues to improve the VCS topology gradually in time. Particularly, for 30 nodes, the correlation coefficient improves from around 0.8 to almost 0.95 in 2000 time steps (approximately 200 s with coordinates update exchange interval set to 102.4 ms), while for 60 nodes case, the increase is from 0.5 to 0.87. However, we note that the topology similarity index may not fully converge.

Recall that the aim of the proposed VCS is to decrease the amount of routing information by using geo-like routing. However, the topology similarity index does not fully characterize this functionality as there might be many specific VCS graphs with high similarity index but drastically different routing performance. To quantify the performance of the proposed routing solution, Fig. 9 illustrates the number of undiscovered routes for 30, 60, and 100 nodes and three routing strategies. Overall, there are 900, 3600, and 10000 (i.e., N^2) distinctive paths for 30, 60, and 100 nodes, respectively, in the considered cases. A red line on Fig. 9, 13, 16 shows the maximal number of possible routes and serves as an upper limit for the number of undiscovered paths. At the same time, the blue line is the shortest route found using Dijkstra's algorithm, serving as a lower limit.

Analyzing the data presented in Fig. 9, one may observe that in the considered cases, similarly to the Dijkstra's algorithm, GPSR is capable of finding nearly all the routes in the network. Such success is not always the case in general due to so-called communication voids in network topologies [32], i.e., a situation when the destination cannot be reached following the greedy approach once a packet gets stuck in a dead-end. To solve this problem, one can use the void handling techniques. Note that these techniques also must be beneficial for GAR performance and can be used as one of its extensions.

For the 30 nodes case, the proposed GAR VCS solution that operates without any external positioning information has been capable of showing similar performance after 1500 time steps. Initially, however,

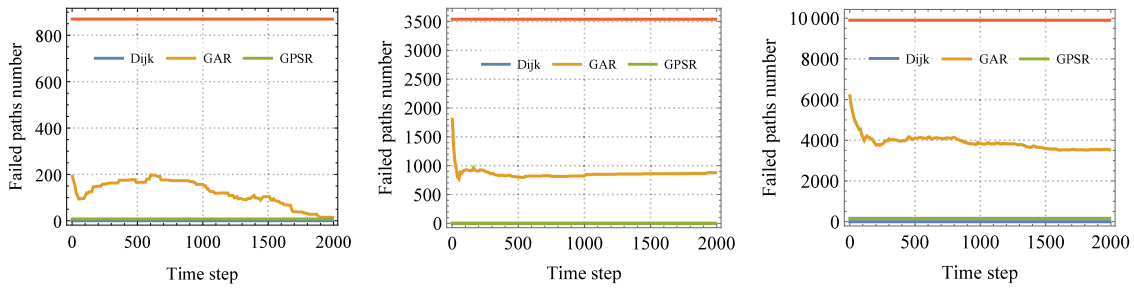


Fig. 9. Mean number of undiscovered paths for static scenario. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

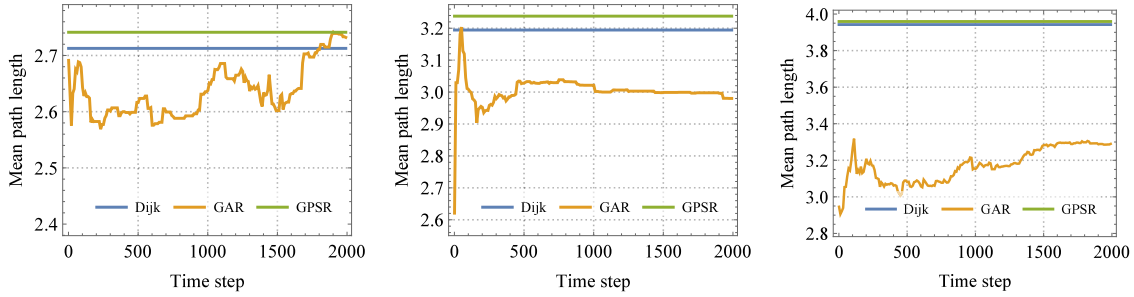


Fig. 10. Mean route length for static scenario. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

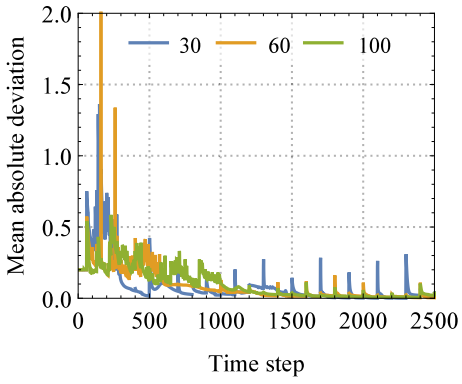


Fig. 11. Mean absolute deviation for churn scenario.

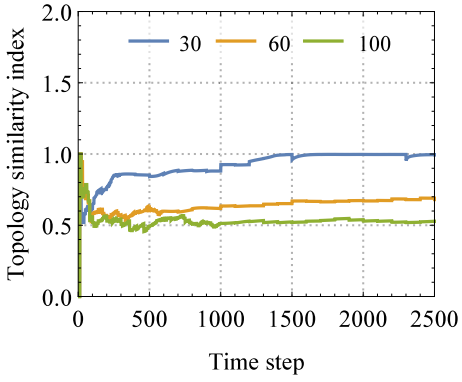


Fig. 12. The topology similarity index for churn scenario.

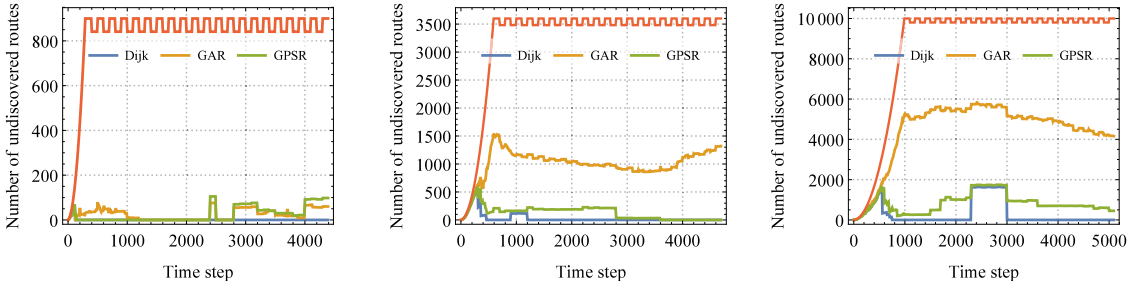
only 30% of all the routes have been discovered. Nevertheless, this fraction reduces fairly quickly and reaches 3% already for time step 500 ($\approx$50 s). This behavior remains qualitatively similar to the 60 nodes case plotted in Fig. 9(b). However, after 2000 time steps ($\approx$200 s) the virtual topology is still not perfect. At this time instant, 16% of available routes remain undiscovered.

Another metric of the quality of routing is the mean path length along the route between source and destination. This metric is demonstrated in Fig. 10 for Dijkstra's algorithm, GPSR, and the proposed GAR VCS algorithm. Analyzing the data, we can observe that the proposed solution, similarly to the FishEye routing algorithm [53], tends to find shorter paths between source and destination, providing routes in the closer neighborhood while keeping distant nodes unreachable. This effect becomes more pronounced when more nodes are added to the network. Thus, the mean length of discovered paths shown in Fig. 10 is smaller than that of GPSR and Dijkstra algorithms. This effect becomes more pronounced when more nodes are added to the network. One of the ways to optimize the solution is to use destination address distance to decide whether to utilize the GAR VCS solution or to use the standard on-demand route discovery procedure.

Nevertheless, the absolute deviation between GAR VCS and GPSR solutions is relatively high, even in the first few time steps (approximately 0.3 and 0.5 for 30 and 60 nodes cases, respectively). One may observe that as the time progresses, the topology is becoming closer to the physical one (see Fig. 8) and more routes are discovered (see Fig. 9), and the mean path length of discovered routes tends to that of GPSR and Dijkstra's algorithm.

5.3.2. Nodes churn scenario

Having studied the critical characteristics of the proposed solution in a static environment, we now proceed with analyzing the peer churn scenario, where nodes may join and leave the network. Similarly to the static scenario, we assess stability and convergence properties first. Fig. 11 shows the mean absolute coordinates deviation at successive steps of the algorithm for node churn scenario with 30, 60, and 100 nodes. Recall that in this scenario, nodes first appear one after another, separated by a constant time interval. Once the number of nodes reaches the maximum value, nodes start to leave and join the network at regular intervals as well.



(a) 30 nodes (b) 60 nodes (c) 100 nodes

Fig. 13. Mean number of undiscovered routes for nodes churn scenario.

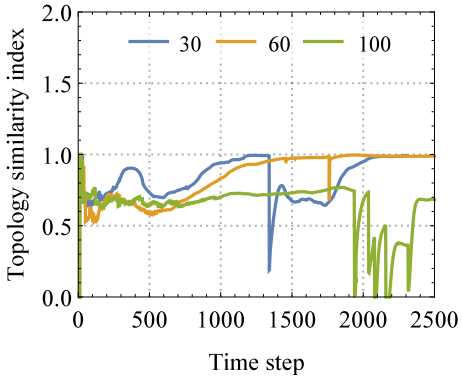


Fig. 14. The topology similarity index for mobile case.

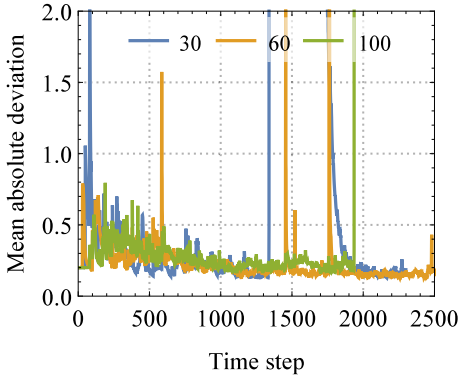


Fig. 15. Mean absolute deviation for mobile scenario.

Analyzing the presented data, we observe that nodes joining one by one induce inconsistencies in the mean absolute deviation pattern prolonging the convergence time. It takes approximately 500 and 1000 time steps (50 and 100 s.) for 30 and 60 nodes, respectively, which is around two times higher compared to static cases, see Fig. 7. Similarly, nodes leaving and joining affect the performance of the algorithm, which is indicated by regular spikes in the presented data. However, these spikes are well localized, implying that the algorithm handles these situations.

The topology similarity index, shown in Fig. 12, illustrates that the dynamic behavior of nodes also affects the topology convergence time. Particularly, as one may observe, for 60 and 100 nodes, the convergence is extremely slow. The topology similarity index does not reach 0.8, even for 4000 times steps (400 s.). However, this behavior can also be attributed to specific network topologies that the proposed solution fails to capture well. In our experiments, the fraction of such topologies remains below 10%.

Finally, Fig. 13 shows the number of undiscovered by the proposed GAR VCS solution routes, GPSR protocol operating over physical coordinates, and the Dijkstra's algorithm. Before we proceed with the analysis of the results, there are a few critical notes. First, the observed periodic fluctuation of the overall number of paths is explained by nodes joining and leaving at regular intervals and the number of routes fluctuating between, e.g., 100^2 and 99^2 for the 100 nodes case. Second, the gradual increase in the overall number of paths available in the network is caused by delayed nodes joining at regular intervals at the beginning of simulations. Third, the slope is different for a different considered number of nodes in the network as nodes appear in the network one by one separated by a fixed time interval. As opposed to the static case inspected previously, there are situations where Dijkstra's algorithm fails to find routes. It only happens when one of the few nodes is physically disconnected from the network. To capture routing behavior in realistic deployments, we allow these situations to happen.

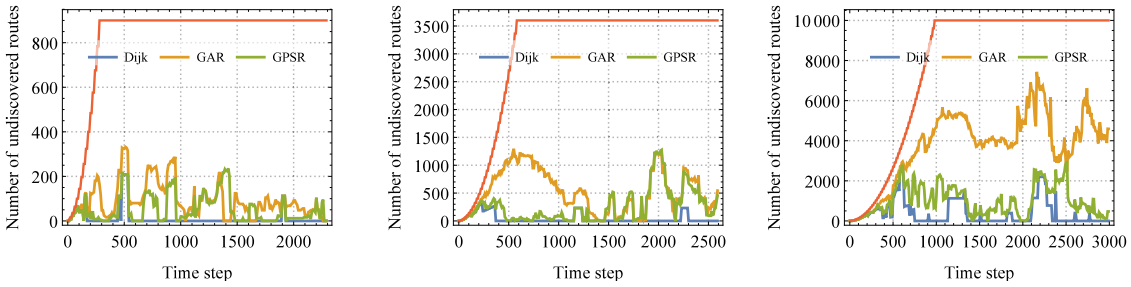
Analyzing the data presented in Fig. 13, one may notice that the proposed solution performs as well as GPSR protocol for 30 nodes case, even showing some minor improvements over it. However, the performance changes for the higher number of nodes, 60 and 100. Notably, we observe that the nodes leaving and joining may deteriorate the efficiency of the proposed solution such that 30% and 50% of routes are undiscovered in the GAR VCS approach. We note that this behavior is typical for the proposed solution, implying that in dynamic networks with node churn, it can only be used as a complementary solution for standard on-demand routing protocols such as AODV.

5.3.3. Mobile scenario

The most demanding scenario from the topology organization and maintenance point of view is when nodes are mobile. We characterize the performance of the proposed GAR VCS for the most generic case when all the nodes are mobile and may also join and leave the network. In the examples reported below, the RDM speed is constant and set to 1 m/s while the mean run length to 20 m. Wrap-around is used to handle the boundary problem [51] and to simulate realistic situations when a node leaves a network and joins another one.

Similarly to the static and dynamic scenarios, we study the performance of the proposed solution for the mobile scenario with stability and convergence time indicators. Fig. 15 illustrates the mean absolute coordinates deviation for mobile scenario for 30, 60, and 100 nodes. As one may observe, the proposed solution converges quickly, maintaining the topology in the presence of mobility. The occasional spikes in the graphs are explained by the wrap-around procedure when nodes disappear at one side of the modeled area and instantaneously appear at another. We specifically note that those wrap-around cases cause large mean one-step deviations. Besides, these deviations are handled well by the algorithm, implying that their effect on the routing performance is not long-lasting.

The topology similarity index for mobile scenario with 30, 60, and 100 nodes is illustrated in Fig. 14. Observing the presented data,



(a) 30 nodes (b) 60 nodes (c) 100 nodes

Fig. 16. Mean number of undiscovered routes for mobile scenario. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

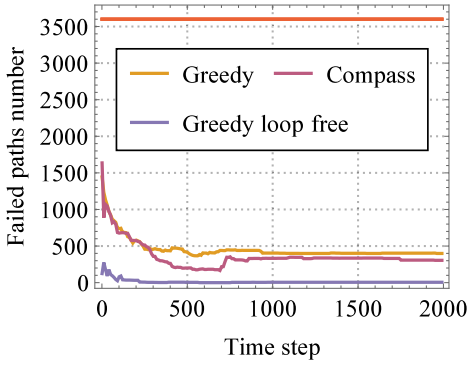
one may deduce that contrarily to the static case, where we observed gradual converge of physical and virtual topologies, the mobility causes smooth changes in the topology similarity index. In all our experiments, the considered metrics fluctuate between 0.6 and 1.0, implying a decent match between topologies. The specific behavior of the topology similarity index is observed for the case of 100 nodes, where there are extreme fluctuations caused by nodes wrap-around during the simulation run. However, as one may observe, the proposed GAR VCS algorithm is capable of quickly recovering from these occasional mismatches, improving the similarity index.

Finally, we investigate the number of discovered routes by the algorithms in the mobility scenario, see Fig. 16. Similarly to the case of the node churn scenario, the mobility of the nodes leads to undiscovered paths, even for the Dijkstra's algorithm. Likewise, the movement causes significantly more failures for GRSP protocol as well compared to the node churn scenario, see Fig. 13. Analyzing the presented data, one may deduce that the increase in the number of nodes leads to a higher number of undiscovered paths, for both GAR VCS and GPSR solutions. While the proposed solutions perform quantitatively similar to GPSR for 30 and 60 nodes discovering the majority of routes, GAR effectiveness drops for 100 nodes case. However, even in this case, at least half of the paths are discovered at all times.

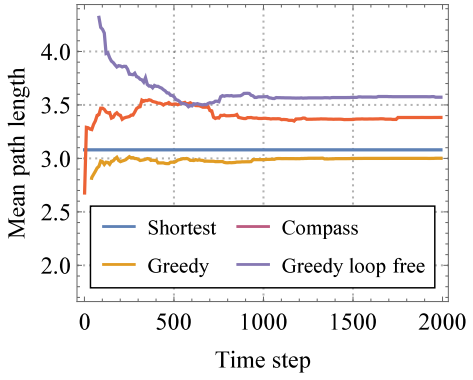
5.4. Comparison of routing strategies

Finally, we compare the performance of different routing on top of GAR VCS. To this aim, Fig. 17 shows the mean number of undiscovered paths and average path length for static case and three different protocols, two variants of GPSR called "greedy" and "loop-free" and "Compass" geographical routing. Compass routing makes a decision on the next hop based on the angle between the current node and its neighbors. The node with the minimal angular difference as compared to the angle with the current node and destination is selected [54]. The greedy version of GPSR operates similarly to the protocol considered in previous parts of this section, while loop-free protocol implements additional checks to avoid loops in the network.

By analyzing the data presented in Fig. 17, one may conclude that the modified version of GPSR works significantly better as compared to both the greedy version of GPSR and Compass. In terms of the mean path length, the greedy version of GPSR is characterized by a slightly smaller route length than the shortest path length provided by the Dijkstra algorithm. However, two other versions of geographical routing produce mean route length that are higher on average. Logically, the loop-free version of the GPSR is characterized by the maximal mean path length. These results indicate that significant routing gains can be achieved on top of the conventional greedy version of the GPSR algorithm.



(a) Mean number of undiscovered paths



(b) Mean route length

Fig. 17. Comparison of routing strategies for static scenario.

6. Conclusions

To reduce the overhead associated with on-demand routing in dynamic wireless mesh networks, we have proposed a new approach for logical topology organization and maintenance in distributed wireless mesh networks. The proposed GAR VCS does not require global network information and relies only on updates from the nodes in the two-hop proximity of the nodes that are exchanged periodically. A critically important advantage of the proposed solution is that GAR scales linearly with network size. We have developed a completely distributed algorithm for building GAR topology, which is robust to network dynamics caused by nodes joining/leaving the network and nodes' mobility.

To assess the performance of the proposed GAR VCS solution, we have analyzed algorithmic, topological, and routing-related metrics of interest. From the stability viewpoint, the solution has been found to quickly adapt to network changes. At the same time, the convergence

time heavily depends on network nodes' characteristics, e.g., mobility and frequency of nodes joining/leaving the system. In general, the proposed GAR VCS solution is most suited to small-to-mid sizes networks with static or mobile nodes. The worst performance is observed in crowded conditions when network nodes regularly join and leave the system. However, even in this case, the proposed solution may discover 50 to 70% of the paths in the network. Furthermore, the proposed solution performs better at finding shorter paths and shows worse results for longer routes.

CRediT authorship contribution statement

Andrey Samuylov: Software, Writing. **Dmitri Moltchanov:** Writing. **Roman Kovalchukov:** Conceptualization, Methodology, Software, Writing. **Anna Gaydamaka:** Methodology, Software, Writing. **Alexander Pyattaev:** Conceptualization, Methodology. **Yevgeni Koucheryavy:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was founded by the Academy of Finland [project number 339519]; project ACCESS: Autonomous Communication Converged with Efficient Sensing for UAV Swarms. The authors thank Prof. Valery Naumov (Service Innovation Research Institute, Finland) for the fruitful discussions on the problem statement and solution method.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.comcom.2022.03.023.

References

[1] C. Funai, C. Tapparello, W. Heinzelman, Enabling multi-hop ad hoc networks through WiFi direct multi-group networking, in: 2017 International Conference on Computing, Networking and Communications, ICNC, IEEE, 2017, pp. 491–497.

[2] A. Ometov, A. Orsino, L. Militano, D. Moltchanov, G. Araniti, E. Olshannikova, G. Fodor, S. Andreev, T. Olsson, A. Iera, et al., Toward trusted, social-aware D2D connectivity: Bridging across the technology and sociality realms, IEEE Wirel. Commun. 23 (4) (2016) 103–111.

[3] R. Molina-Masegosa, J. Gozalvez, LTE-V for sidelink 5G V2X vehicular communications: A new 5G technology for short-range vehicle-to-everything communications, IEEE Veh. Technol. Mag. 12 (4) (2017) 30–39.

[4] A. Orsino, D. Moltchanov, M. Gapeyenko, A. Samuylov, S. Andreev, L. Militano, G. Araniti, Y. Koucheryavy, Direct connection on the move: Characterization of user mobility in cellular-assisted D2D systems, IEEE Veh. Technol. Mag. 11 (3) (2016) 38–48.

[5] Y. Ghasempour, C.R. da Silva, C. Cordeiro, E.W. Knightly, IEEE 802.11 ay: Next-generation 60 GHz communication for 100 Gb/s Wi-Fi, IEEE Commun. Mag. 55 (12) (2017) 186–192.

[6] C. Campolo, A. Molinaro, F. Romeo, A. Bazzi, A.O. Berthet, 5G NR V2X: On the impact of a flexible numerology on the autonomous sidelink mode, in: 2019 IEEE 2nd 5G World Forum, 5GWF, IEEE, 2019, pp. 102–107.

[7] Q. Chen, X.J. Zhang, W.L. Lim, Y.S. Kwok, S. Sun, High reliability, low latency and cost effective network planning for industrial wireless mesh networks, IEEE/ACM Trans. Netw. 27 (6) (2019) 2354–2362.

[8] R. Pirmagomedov, A. Ometov, D. Moltchanov, X. Lu, R. Kovalchukov, E. Olshannikova, S. Andreev, Y. Koucheryavy, M. Dohler, Applying blockchain technology for user incentivization in mmWave-based mesh networks, IEEE Access (2020).

[9] M. Güneş, D.G. Reina, J.M.G. Campos, S.L. Toral, Communication protocols for multi-hop ad hoc networks, in: Mobile Ad Hoc Network Protocols Based on Dissimilarity Metrics, Springer, 2017, pp. 19–24.

[10] G. Ramezan, C. Leung, Z.J. Wang, A survey of secure routing protocols in multi-hop networks, IEEE Commun. Surv. Tutor. 20 (4) (2018) 3510–3541.

[11] N. Sabor, S. Sasaki, M. Abo-Zahhad, S.M. Ahmed, A comprehensive survey on hierarchical-based routing protocols for mobile wireless sensor networks: Review, taxonomy, and future directions, Wirel. Commun. Mob. Comput. 2017 (2017).

[12] D. Murray, M. Dixon, T. Koziniec, An experimental comparison of routing protocols in multi hop ad hoc networks, in: 2010 Australasian Telecommunication Networks and Applications Conference, IEEE, 2010, pp. 159–164.

[13] R. Draves, J. Padhye, B. Zill, Comparison of routing metrics for static multi-hop wireless networks, ACM SIGCOMM Comput. Commun. Rev. 34 (4) (2004) 133–144.

[14] A. Rao, S. Ratnasamy, C. Papadimitriou, S. Shenker, I. Stoica, Geographic routing without location information, in: Proceedings of the 9th Annual International Conference on Mobile Computing and Networking, ACM, 2003, pp. 96–108.

[15] T. Watteyne, I. Augé-Blum, M. Dohler, S. Ubéda, D. Barthel, Centroid virtual coordinates–A novel near-shortest path routing paradigm, Comput. Netw. 53 (10) (2009) 1697–1711.

[16] B. Leong, B. Liskov, R. Morris, Greedy virtual coordinates for geographic routing, in: 2007 IEEE International Conference on Network Protocols, IEEE, 2007, pp. 71–80.

[17] N. Filardi, A. Caruso, S. Chessa, Virtual naming and geographic routing on wireless sensor networks, in: 2007 12th IEEE Symposium on Computers and Communications, IEEE, 2007, pp. 609–614.

[18] J.-P. Sheu, K.-Y. Hsieh, M.-L. Ding, Routing with hexagonal virtual coordinates in wireless sensor networks, Wirel. Commun. Mob. Comput. 9 (9) (2009) 1206–1219.

[19] A. Karima, B. Mohammed, B. Azeddine, New virtual coordinate system for improved routing efficiency in sensor network, Int. J. Comput. Sci. Issues 9 (3) (2012) 59.

[20] D.C. Dhanapala, A.P. Jayasumana, Anchor selection and topology preserving maps in WSNs — A directional virtual coordinate based approach, in: 2011 IEEE 36th Conference on Local Computer Networks, 2011, pp. 571–579, http://dx.doi.org/10.1109/LCN.2011.6115519.

[21] T. Bouchoucha, Z. Ding, Anchor selection for topology inference and routing in wireless sensor networks, J. Commun. Inf. Netw. 5 (3) (2020) 318–323, http://dx.doi.org/10.23919/JCIN.2020.9200895.

[22] Y. Fan, X. qi, B. Yu, L. Liu, A distributed anchor node selection algorithm based on error analysis for trilateration localization, Math. Probl. Eng. 2018 (2018) 1–12, http://dx.doi.org/10.1155/2018/7295702.

[23] S. Ratnasamy, I. Stoica, S. Shenker, Routing algorithms for DHTs: Some open questions, in: International Workshop on Peer-to-Peer Systems, Springer, 2002, pp. 45–52.

[24] A. Awad, R. German, F. Dressler, Exploiting virtual coordinates for improved routing performance in sensor networks, IEEE Trans. Mob. Comput. 10 (9) (2010) 1214–1226.

[25] R. Anwit, P. Kumar, M.P. Singh, Virtual coordinates routing using VCP-m in wireless sensor network, in: 2014 International Conference on Computational Intelligence and Communication Networks, IEEE, 2014, pp. 402–407.

[26] J. Shen, C. Wang, A. Wang, X. Sun, S. Moh, P. Hung, Organized topology based routing protocol in incompletely predictable ad-hoc networks, Comput. Commun. 99 (2016) http://dx.doi.org/10.1016/j.comcom.2016.07.009.

[27] C. Perkins, E. Belding-Royer, S. Das, Ad Hoc on-Demand Distance Vector (AODV) Routing, Rfc 3561, IETF, 2003.

[28] L. Guo, Y. Peng, X. Wang, D. Jiang, Y. Yu, Performance evaluation for on-demand routing protocols based on OPNET modules in wireless mesh networks, Comput. Electr. Eng. 37 (1) (2011) 106–114.

[29] A. Zehni, S. Zolfaghari, M. Fathy, M. Shahverdy, M. Asgari, Wireless mesh network routing: A comparative survey, Rev. QUID (1) (2017) 412–421.

[30] S. Mohapatra, P. Kanungo, Performance analysis of AODV, DSR, OLSR and DSDV routing protocols using NS2 simulator, Procedia Eng. 30 (2012) 69–76.

[31] B. Karp, H.-T. Kung, GPSR: Greedy perimeter stateless routing for wireless networks, in: Proceedings of the 6th Annual International Conference on Mobile Computing and Networking, 2000, pp. 243–254.

[32] D. Chen, P.K. Varshney, A survey of void handling techniques for geographic routing in wireless networks, IEEE Commun. Surv. Tutor. 9 (1) (2007) 50–67.

[33] A. Nayak, I. Stojmenovic, Wireless Sensor and Actuator Networks: Algorithms and Protocols for Scalable Coordination and Data Communication, John Wiley & Sons, 2010.

[34] S. Bhattacharjee, S. Kanta, S. Modi, M. Paul, S. DasBit, Disaster messenger: An android based infrastructure less application for post disaster information exchange, in: 2016 IEEE International Conference on Advanced Networks and Telecommunications Systems, ANTS, IEEE, 2016, pp. 1–5.

[35] F. Mirsadeghi, M.K. Rafsanjani, B.B. Gupta, A trust infrastructure based authentication method for clustered vehicular ad hoc networks, Peer Peer Netw. Appl. 14 (4) (2021) 2537–2553.

[36] H. Chu, J.Z. Yang, Building disaster resilience using social messaging networks: The WeChat community in Houston, Texas, during hurricane harvey, Disasters 44 (4) (2020) 726–752.

[37] M. Asadpour, K.A. Hummel, D. Giustiniano, S. Draskovic, Route or carry: Motion-driven packet forwarding in micro aerial vehicle networks, IEEE Trans. Mob. Comput. 16 (3) (2016) 843–856.

[38] S. Moon, Y. Choi, D. Kim, M. Seung, H. Gong, Outdoor swarm flight system based on rtk-gps, J. KIISE 43 (12) (2016) 1315–1324.

[39] W. Power, M. Pavlovski, D. Saranovic, I. Stojkovic, Z. Obradovic, Autonomous navigation for drone swarms in gps-denied environments using structured learning, in: IFIP International Conference on Artificial Intelligence Applications and Innovations, Springer, 2020, pp. 219–231.
[40] M. Campion, P. Ranganathan, S. Faruque, UAV swarm communication and control architectures: A review, J. Unmanned Veh. Syst. 7 (2) (2018) 93–106.
[41] A. Kohlbacher, J. Eliasson, K. Acres, H. Chung, J.C. Barca, A low cost omnidirectional relative localization sensor for swarm applications, in: 2018 IEEE 4th World Forum on Internet of Things, WF-IoT, IEEE, 2018, pp. 694–699.
[42] N. Sirola, Closed-form algorithms in mobile positioning: Myths and misconceptions, in: 2010 7th Workshop on Positioning, Navigation and Communication, IEEE, 2010, pp. 38–44.
[43] The International Public Telecommunication Numbering Plan, E.164, ITU-T, 2012.
[44] M. Mauve, J. Widmer, H. Hartenstein, A survey on position-based routing in mobile ad hoc networks, IEEE Netw. 15 (6) (2001) 30–39.
[45] E.K. Lua, J. Crowcroft, M. Pias, R. Sharma, S. Lim, A survey and comparison of peer-to-peer overlay network schemes, IEEE Commun. Surv. Tutor. 7 (2) (2005) 72–93.
[46] A. Boukerche, B. Turgut, N. Aydin, M.Z. Ahmad, L. Bölöni, D. Turgut, Routing protocols in ad hoc networks: A survey, Comput. Netw. 55 (13) (2011) 3032–3080.
[47] A. Hinds, M. Ngulube, S. Zhu, H. Al-Aqrabi, A review of routing protocols for mobile ad-hoc networks (MANET), Int. J. Inf. Educ. Technol. 3 (1) (2013) 1.
[48] B.B. Maqbool, M.A. Peer, Classification of current routing protocols for ad hoc networks - A review, Int. J. Comput. Appl. 7 (8) (2010) 26–32.
[49] C.E. Perkins, E.M. Royer, Ad-hoc on-demand distance vector routing, in: Proceedings WMCSA'99. Second IEEE Workshop on Mobile Computing Systems and Applications, IEEE, 1999, pp. 90–100.
[50] L. Sachs, Applied Statistics: A Handbook of Techniques, Springer Science & Business Media, 2012.
[51] P. Nain, D. Towsley, B. Liu, Z. Liu, Properties of random direction models, in: Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies, Vol. 3, IEEE, 2005, pp. 1897–1907.
[52] M. Abolhasan, T. Wysocki, E. Dutkiewicz, A review of routing protocols for mobile ad hoc networks, Ad Hoc Netw. 2 (1) (2004) 1–22.
[53] G. Pei, M. Gerla, T.-W. Chen, Fisheye state routing: A routing scheme for ad hoc wireless networks, in: 2000 IEEE International Conference on Communications. ICC 2000. Global Convergence Through Communications. Conference Record, vol. 1, IEEE, 2000, pp. 70–74.
[54] S. Medjiah, T. Ahmed, F. Krief, Agem: Adaptive greedy-compass energy-aware multipath routing protocol for WMSNs, in: 2010 7th IEEE Consumer Communications and Networking Conference, IEEE, 2010, pp. 1–6.