Wireless Networks (2022) 28:331–353
https://doi.org/10.1007/s11276-021-02824-7

ORIGINAL PAPER

Plane-separated routing in ad-hoc networks

Doğanalp Ergenç[1] · Ertan Onur[2]

Accepted: 11 October 2021 / Published online: 4 January 2022

Abstract
Control and user (data) plane separation (CUPS) is a concept applied in various networking areas to scale network resources independently, increase the quality of service, and facilitate the autonomy of networks. In this study, we leverage this concept to design a plane-separated routing algorithm, CUPS-based hierarchical routing algorithm (CHRA), as an energy-efficient and low-latency end-to-end communication scheme for clustered ad-hoc networks. In CHRA, while cluster heads constitute the control plane to conduct network discovery and routing, ordinary nodes residing in the user plane forward packets according to the routing decisions taken by the control plane. Exploiting the CUPS, we avoid exhausting cluster heads by offloading packet-forwarding to ordinary nodes and improve the quality of service by utilizing alternative paths other than the backbone of cluster heads. Our simulation results show that CHRA offers a better quality of service in terms of end-to-end latency and data-to-all ratio, and promotes fairness in energy-consumption in both stationary and mobile scenarios.

Keywords Control and user plane separation · CUPS · Ad-hoc networks · Hierarchical routing · Clustering

1 Introduction

Wireless communication has been deployed in a broad range of different networks with significantly varying requirements and conditions. While several systems like cellular mobile networks prioritize the maintenance of service quality for their customers as well as maximizing the profit, other systems such as tactical and emergency networks require reliability and adaptability in challenging environments. Such requirements have a direct impact on the networking architectures and design choices thereof. Military-tactical networks depend on self-organized communication of mobile nodes without any assistance of pre-deployed infrastructure that cannot be easily built on battlegrounds [12]. Comparably, emergency networks in case of a disaster may lack pre-deployed equipment. Therefore, the development of self-organized mobile ad-hoc networks (MANETs) adapting to the challenging demands, e.g., heterogeneity, scalability, and mobility, of such systems has been focused on as a tackling yet promising goal [2, 25]. Apart from these examples, a similar self-organization scheme can also be leveraged in networks with centralized infrastructures. For instance, device-to-device (D2D) communication in 5G cellular mobile networks can be considered as a form of ad-hoc networking, which helps to decrease the load on base stations but still needs further assistance to maintain the end-to-end connections [49]. Similarly, emerging Internet of Things (IoT) and smart city paradigms leverage the self-organized and mobile wireless sensor networks that can be—partially—orchestrated by a controller in a hybrid manner [5]. Eventually, MANETs have been proposed as a prominent solution for infrastructure-less and self-organized connected systems and deployed in the infrastructure-based systems to increase their degree of autonomy and efficiency.

To establish end-to-end communication in MANETs, routing and forwarding can be executed over flat topologies, albeit not scalable. While an existing infrastructure may help to overcome scalability issues, self-organized

Parts of this work are published as a preprint in arXiv:1807.10747.

✉ Doğanalp Ergenç
doganalp.ergenc@uni-hamburg.de

Ertan Onur
eronur@metu.edu.tr

[1] Department of Informatics, Universität Hamburg, Hamburg, Germany

[2] Department of Computer Science, Middle East Technical University, Ankara, Turkey

communication schemes without any assistance of pre-deployed infrastructure can suffer from lack of a (de-)-centralized controller. For such networks, clustering is a well-known approach to achieve scalability and manageability [14, 50]. Figure 1 illustrates a clustered network with the nodes with different roles. In the figure, the network is divided into groups based on some selected clustering criteria, e.g., locations, identifiers, or capabilities of the nodes. Each group, or cluster, is managed by a cluster head, which is elected in a distributed fashion. Cluster heads handle routing, resource scheduling, data aggregation, etc. [11, 31] to manage clusters. In such designs, ordinary nodes do nothing other than delivering packets to their designated cluster heads. In Fig. 1, cluster heads (nodes 2 and 7) and gateways (node 4) run a routing algorithm, where the routing control messaging is indicated with the control plane (CP) edges. Then, the ordinary nodes (nodes 1, 3, 6, and 9) forward packets to the destination over the edges marked as data plane (DP) edges. Note that CP and DP are logical notions; each node may act as a control- or data-plane element depending on their assigned roles. When both routing and forwarding are carried out by those designated nodes, they may be prone to resource exhaustion because of an excessive workload. Besides, when only CP edges are considered to carry data packets, many other potential routes in DP become overlooked.

Control and user plane separation (CUPS) is proposed to cope with such issues distinguishing routing and forwarding as different functions to be performed by particular nodes. It has recently become popular in software-defined networks (SDN), where the programmable nodes on the control plane orchestrate packet-switching nodes on the data plane [8, 17]. The main goal of this separation is to form a logically-centralized controllable and configurable networks, which paves the way for automation and flexible orchestration. Here, a clustered network forms a convenient architecture to leverage CUPS in ad-hoc networks. Cluster heads constitute the control plane having extensive visibility of the network and configuring the ordinary nodes with respective forwarding rules. Other nodes, on the other hand, form the user (or data) plane behaving according to configured forwarding rules. While routing can be handled by cluster heads and gateways, any node may exhibit a role in packet forwarding balancing energy consumption throughout nodes e.g., 1, 3, 6 and 7 in Fig. 1. CUPS has also been found further application areas such as next-generation cellular networks [3, 7, 40] and has started to be partially adapted to ad-hoc networks as well [46, 47].

In this paper, we leverage CUPS architecture to propose a plane-separated routing algorithm for ad-hoc networks, CUPS-based hierarchical routing algorithm (CHRA). By separating those functions, we decrease the energy consumption of control plane elements that are used for both routing and forwarding in the state of the art routing protocols for ad-hoc networks. Besides, as we now deploy ordinary nodes for the forwarding process, it becomes possible to find more efficient end-to-end paths independent from the control plane that may restrict packet forwarding through only the cluster heads. Lastly, we design route repair and recovery techniques that effectively use the control plane nodes and functions. Eventually, CHRA offers fair energy-consumption, and fast and reliable end-to-end communication and is applicable in various types of ad-hoc networks including sensor, military, and backscatter networks.

Numerous routing algorithms are reviewed, compared, and contrasted to CHRA in Sect. 2 qualitatively. Quantitatively, we compare CHRA with two opponents, cluster-based routing protocol (CBRP) and hybrid cluster routing (HCR). CBRP is a well-known algorithm [29] depending

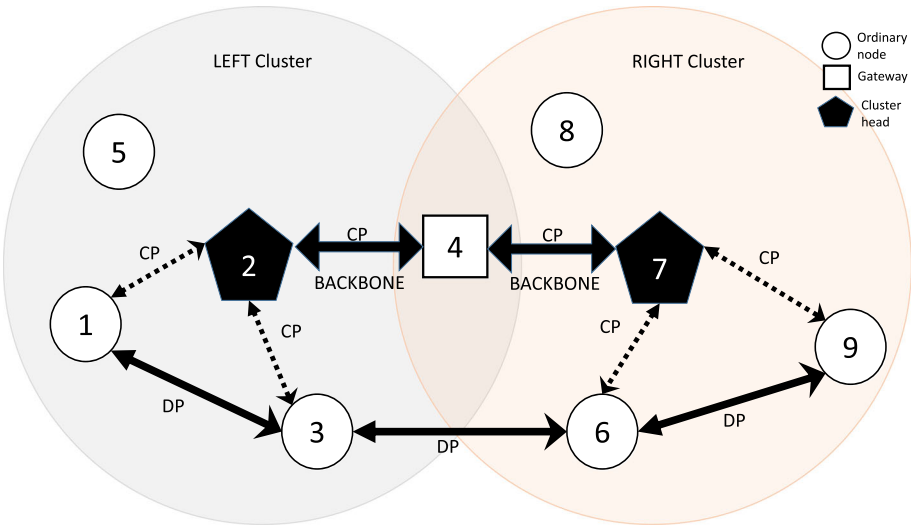


Fig. 1 An illustration of a CUPS-based ad-hoc network where CP and DP represent control and user (data) planes. Cluster heads, gateways and ordinary nodes are shown with pentagons, squares and circles, respectively

on control plane routing and forwarding (i.e., hierarchy without CUPS) and HCR [32, 42] uses a hybrid routing technique on clustered networks that resembles the CHRA the most. Our contributions are listed as follows.

- We adapt the CUPS architecture to ad-hoc networks to improve energy-efficiency and quality of service in end-to-end communication. We present CUPS in Sect. 3.
- We propose a hierarchical routing algorithm, CHRA focusing on the CUPS architecture in Sect. 4.
- We implement CHRA and its opponents in the discrete event-based simulator OMNeT++ and compare them in realistic scenarios under mobility with uniformly and non-uniformly distributed networks in Sect. 5. Our simulation results show that CHRA offloads up to 75% of the forwarded data to ordinary nodes in data plane in comparison to its opponents using controller node for forwarding and achieves a low deviation in energy consumption with the lowest end-to-end delay.
- We present how to handle routing errors mostly due to mobility in CHRA through the examples in "Appendix".

2 Related work

In this section, we first present CUPS and its role in SDN and wireless networks briefly. Then, we review the existing routing algorithms used in hierarchical MANETs as we assume a similar hierarchy to construct plane-separated architecture in ad-hoc networks. Lastly, we discuss the main differences of our work from the presented approaches.

2.1 Plane-separation, SDN, and wireless networks

The plane-separation concept has been leveraged especially with the emerging SDN paradigm. In SDN architecture, *control plane* nodes are responsible for orchestration, management, and configuration of the overall network utilizing higher-level applications [35] from a centralized perspective. In a broader sense, they perform control functions, which can include security mechanisms, e.g., firewall [33], load-balancing [18], resource allocation and scheduling [54], and routing for end-to-end communication [26, 55, 64]. The control plane nodes might be deployed as physical servers or even virtual functions as cloud and edge services [22]. *Data plane*, on the other hand, consists of ordinary nodes that are responsible for data forwarding according to the rules and decisions made by the control plane. SDN-supported data center switches are common examples of data plane entities [51].

Together with SDN, CUPS has been employed in a wide range of application areas, including wireless networks [56]. Especially as they already have separated access and backbone networks in 4G/5G cellular networks, a distinct separation of control functions is quite feasible [41]. For instance, in [60], the authors utilize CUPS architecture to address the coverage and handover issues in high-mobility 5G networks. As the separation of control plane gives further flexibility to reconfigure a network satisfying the quality of service (QoS) requirements for end-to-end communication, it becomes possible to maintain communication with the minimum handover cost and the maximum coverage probability even in ultra-dense scenarios. In [24], routing in a CUPS-enabled wireless mesh network is compared with the traditional wireless routing techniques, and it is shown that the placement of centralized and decentralized controllers increases the routing and forwarding performance in terms of end-to-end delay and packet delivery ratio. Similar results are also shown in [36], which employs a hybrid approach to use centralized control plane entities and distributed routing algorithms together. Lastly, in [47], a CUPS-based SDN approach is adapted to ad-hoc networks deploying decentralized controllers. However, as it is not always possible to deal with the dynamic nature of ad-hoc networks with such infrastructures, the authors capacitate ordinary nodes to make their own routing and forwarding decisions in case of that they cannot be configured by the controller nodes. Although all those approaches distribute the routing and forwarding functions via plane-separation in wireless networks, they are still fully or partially dependent on pre-deployed (de-)centralized controllers.

2.2 Hierarchical routing protocols

Various routing algorithms are designed for the clustered networks. Such algorithms usually fall under the hierarchical routing category [6, 10]. In this section, we briefly introduce some of those algorithms.

Cluster-based routing protocol (CBRP) [29] is one of the first hierarchical routing algorithms. It utilizes lowest ID-based clustering to form two-hop clusters. In CBRP, a source node broadcasts route requests within its cluster. The request is forwarded through the neighbor clusters via gateway nodes. The cluster head of the destination node eventually sends a routing response back through the neighbor cluster heads, and the data packets are forwarded through the (reverse) path, which the routing response follows. Eventually, routing and forwarding are handled by the cluster heads and gateways. Many other routing protocols take a similar approach with CBRP and accordingly, they take CBRP as reference for comparison and analysis [43, 62, 63].

Some routing protocols have a more holistic approach involving the clustering, e.g., hierarchy building and cluster head selection, process. Cross-CBRP [15] evaluates the ratio between power levels of two successive signals to elect more reliable cluster heads and then both routing and forwarding are performed through those cluster heads. While the clustering phase differs, routing and forwarding embrace the same method with CBRP. Although Cross-CBRP performs slightly better than CBRP in terms of packet delivery ratio and control overhead, it falls short in end-to-end delay. Energy-efficiency is not evaluated at all in this study. In weighted clustering algorithm (WCA) [53], the authors focus on different parameters considering mobility and energy consumption for clustering, and routing and forwarding through the cluster heads. Since the cluster heads are selected among least-mobile and higher-energy nodes, an end-to-end paths are more reliable, e.g., with least moving nodes and having sufficient energy. One of the key observations of the study is that routing constantly through particular paths leads to depletion of energy and thus, cluster heads should be changed by selecting the ones with the highest energy. However, reclustering also requires an extra energy consumption and control overhead. Moreover, since the cluster heads are still used for both routing and forwarding, they are at risk of drained quickly in case of high control and data traffic. Optimized link state routing (OLSR) [13] does not directly involve a clustering process but relies on the use of multipoint relays (MPRs), which are selected by each individual node. In OLSR, nodes discover their 1- and 2-hop neighborhoods by flooding hello packets and keep their topology database updated proactively. To limit flooding, only a set of MPRs per node forwards respective control packets. Although MPRs utilize packet forwarding and decreases the control overhead of link-state routing, each node should still maintain its own routing table for each destination. Furthermore, this maintenance complicates detection and recovery of broken routes as it requires the reiteration of flooding process. Both problems can be easily addressed by using an efficient control plane design, which is responsible for the maintenance of topology through the controller nodes with broader visibility as we propose in this work. A recent study [23] utilizing OLSR improves that approach with another clustering scheme, which results in the formation of a backbone to be used for both control and data packets. Although its design perspective is flexible as it distinguishes overlay and underlay networks for maintenance and aims to isolate physical changes from the application-level operations, it still relies on a single backbone of cluster heads for the whole communication.

Apart from fully ad-hoc approaches where there is not a pre-deployed infrastructure, there are some protocols assisted by external mechanisms. For example, clustering routing protocol based-on segmentation (CRPBS) [61] is designed for vehicular ad-hoc networks, where mobility is the main concern. To cope with wider areas and increasing vehicle density, CRPBS divides clusters into the segments and examines them to find the farthest segment with minimum packet forwarding time. The segmentation process requires global positioning system (GPS) information, which might not be suitable for various scenarios such as sensor networks. Then, a cluster head is selected for each segment and end-to-end communication is completed through the cluster heads. Even though the results show that CRPBS performs better than its opponents in terms of average end-to-end delay, it leads to exhausting particular nodes in the long term due to their common use for control and data traffic. Zone-based Hierarchical Link State (ZHLS) [30] protocol also uses GPS for (i) clustering to group closer nodes and (ii) routing to detect farther nodes and obtain related routes. In contrast to the cluster-based hierarchical networks, the network is initially divided into zones and each node knows its ID, location, and the zone it belongs beforehand. Note that there is not any cluster head to form backbone, but inter-zone communication is established via gateways. The zone structure should be updated continuously due to node mobility. Although ZHLS is evaluated in terms of path lengths under increasing mobility and increasing number of nodes, its energy-efficiency is not discussed in the paper. Cluster based location-aided routing protocol (CLACR) [59] takes advantage of central infrastructures (managers and servers) for topology discovery and routing. Therefore, their applicability depends on the existence of such infrastructure. In CLACR, routing and forwarding are handled in a cluster-by-cluster manner. It also proposes a local repairing method that handles the failures of only the source and destination nodes. Besides, CLACR has an ad-hoc route optimization mechanism that modifies paths on the fly, e.g., during forwarding, by recording traveled nodes and updating the related segment of the path if there is another shortcut from one of the traveled nodes. Such cases are relevant, especially in mobile scenarios. In contrast, in cluster-based inter-domain routing (CIDR) [65], the topology discovery is performed by dividing the network into domains. Cross-domain communication is handled via cluster heads and gateway nodes similar to CBRP. CIDR targets an Internet-like architecture, where the domains represent autonomous systems (AS) and network discovery is handled by a Border Gateway Protocol-like (BGP) approach. To limit the control overhead and increase scalability, CIDR embraces a fish-eye approach for the domain discovery. The authors also propose a domain splitting scheme for the maintenance of the hierarchical network structure. Even though a detailed analysis of the study is presented, it is not compared by any opponent protocol. In [34], the authors take

advantage of more capable network nodes, such as tanks and UAVs in an army-tactical network, to form a multi-layer hierarchy. They propose a routing scheme where those nodes with less or no power-consumption limitation act as gateways, or cluster heads in a traditional manner, to forward the traffic. It consequently creates a multi-level backbone and restrains whole data traffic within specific nodes.

As an alternative approach without using any infrastructure, virtual grid architecture (VGA) [4] uses a simple power control scheme capturing differences in signal transmission powers to map the physical network topology onto a virtual grid topology. Then, it combines such virtual grid topology and hierarchical clusters for the routing process to enhance the packet delivery ratio in both homogeneous and heterogeneous networks. In their follow-up work [1], they extended VGA to provide end-to-end quality of service guarantees using Open Shortest Paths First (OSPF) protocol. In this work, GPS information is assumed available to build virtual grid topology. VGA and its extensions do not have any further energy-efficiency discussion.

There are hybrid routing algorithms that effectively use cluster-based hierarchy. Hybrid cluster routing (HCR) [32, 42] proactively updates routes for intra-cluster communication. For inter-cluster communication, it behaves like all other algorithms presented here and uses cluster heads and gateways directly. Hybrid optimized link state routing (HOLSR) enhances OLSR by using a hierarchical structure. It decreases control overhead using the fish-eye technique, which sends routing control messages to farther nodes less frequently. This technique also increases bandwidth utilization by decreasing the number of control messages. The hierarchy definition of HOLSR is not directly related to the roles as in clustered structure, it is represented by the number of hops between nodes instead [39]. Group adaptive hybrid routing algorithm (GAHRA) [66] addresses military-tactical scenarios where different army squats form groups with similar mobility behaviors. While the nodes in a squat perform table-driven routing for intra-squat communication, they use an AODV-like flooding mechanism to connect to the further nodes in different squats. In GAHRA, there is not a particular hierarchy but homogeneity within naturally-constructed groups with respect to their mobility. Although it performs better than AODV in terms of end-to-end latency and packet delivery ratio, energy-efficiency of the algorithm is not evaluated in the study. Similarly, in [58], the authors discuss the applicability of border gateway protocol (BGP) on multi-domain MANET with the assistance of airborne nodes acting as gateway nodes between domains and forming a backbone for inter-domain communication. Their results outperform traditional BGP and OSPF, yet the design mostly depends on the existence of such particular backbone nodes. In another flexible scheme, the authors of [16] proposes a set of different routing methods to be used adaptively depending on a stability metric. While relatively stable nodes form groups and share reliable topology information in-between to fill their routing tables, other nodes perform routing by flooding data to the overall network. Even though such a hybrid approach can cope with the heterogeneity of networks, it does not offer further in terms of energy-efficiency and resource utilization.

Lastly, Greedy forwarding limited flooding routing (GLFR) [57] is a recent study that focuses on energy-efficiency, likewise CHRA. It combines the Greedy peripheral stateless routing (GPSR) and AODV for routing in flying ad-hoc networks (FANETs). In GLFR, nodes periodically broadcast hello packets to announce their positions and those packets are forwarded further in the network greedily to limit flooding. The greedy node selection to forward packets is optimized by using particle swarm optimization (PSO). Although it does not consider any hierarchy that might enable the advantages of CUPS, the results show that GLFR is more energy-efficient than AODV thanks to its restricted forwarding mechanism. However, considering the high-mobility nature of FANETs, it is quite challenging to obtain stable routes in the control plane and guarantee fair energy consumption in the data plane.

2.3 Main differences from the state of the art

In comparison to the presented routing algorithms, our algorithm, CHRA, uses cluster heads (in the control plane) only for finding the routes, and ordinary nodes (in the data plane) and gateway nodes for forwarding data packets. As a consequence, CHRA does not drain cluster heads and leads to an energy-efficient communication scheme by offloading forwarding burden from control plane nodes to (ordinary) data plane nodes. Besides, including ordinary nodes effectively to the communication, CHRA becomes able to utilize further alternative paths for a better quality of service. Furthermore, CHRA does not assume the existence of any external entity assisting to clustering or routing processes such as GPS and central infrastructure in comparison to the other SDN-based plane-separated approaches in wireless networks. Furthermore, we do not assume any special node that can lead the communication without any resource limitations. Therefore, it is easily adaptable to different scenarios with minimum assumptions. Lastly, even though clustering is a required to build a hierarchical network, where plane-separation can take place, CHRA does not necessarily depend on the dynamics of such a clustering mechanism. That is, CHRA can be used to establish end-to-end communication seamlessly without

regarding the technique to deploy plane-separation. We leave the analysis (of the potential impact) of clustering algorithms out of the scope as it is quite a broad topic. Interested reader can find further details about clustering in [14, 20, 21, 50].

3 Control-user plane separation (CUPS) in ad-hoc networks

Our CUPS approach in ad-hoc networks assumes that a network is clustered using some approach such as [21] to form a control plane by cluster heads. In CUPS, control packets that carry routing or topology information are exchanged among cluster heads via gateways as shown in Fig. 1 instead of flooding the whole network. Except for isolated clusters, all cluster heads are connected to the other cluster heads through gateways, i.e., cluster heads and gateways form a connected backbone. Since cluster heads have topology information of their clusters, e.g., cluster members and their connectivity, they can easily make routing decisions for the intra-cluster communication. Through periodic inter-cluster control signaling over the backbone, inter-cluster routing is handled. The maintenance and discovery of the routes are narrowed down to the backbone nodes that form the control plane.

Let us assume nine nodes are grouped into left and right clusters as shown in Fig. 1. Both clusters have five ordinary nodes in the data plane and one cluster head and one gateway in the control plane. Where node 4 is the gateway; and nodes 2 and 7 are the cluster heads of left and right clusters, respectively, those three nodes form a backbone. Assume node 1 sends a packet to node 9. In legacy clustered ad-hoc networks, ordinary nodes send their packets to the responsible cluster head. The cluster head then runs the control plane functions (i.e., performs routing) and then forwards the packet over gateways and cluster heads, i.e., over the path 1–2–4–7–9.

The approach in CUPS-based ad-hoc networks is different. In CUPS, node 1 sends its communication request (not the data packet) to its cluster head. The cluster head (node 2 in this example) runs CHRA that we present in Sect. 4 and obtains the route to node 9, i.e., the path 1–3–6–9. Then, node 1 transmits the packet to node 3 over the route determined in the logically centralized control plane, i.e., cluster heads. Note that all nodes are equal in the data plane while they have different roles (cluster head, gateway, or ordinary node) in the control plane.

In CUPS architecture, control plane elements should have a broader network view to orchestrate data plane nodes. Here, cluster heads convey the topology information of their clusters to the neighbor clusters over the backbone. However, as sending such information through the whole network is costly in wireless medium, we define *cluster sight area* (CSA) of a cluster, where the cluster head has the topological view of a limited area, e.g., covering a few neighbor clusters (which is selected as two neighbor clusters). Figure 2 describes an example CSA. In the figure, a cluster-hop represents the distance between two clusters in terms of the number of cluster heads. Assuming that cluster heads are connected via gateways, n-cluster-hop contains $(4n+2)$-hop paths at most. Within CSA, the whole topology is known by all cluster heads, i.e., any link between nodes is known by each cluster head.

To maintain their CSA view, cluster heads exchange their local cluster topology information with *sight area messages* (SAMs) via gateways. Those messages are sent in different periods with a fish-eye approach [44] depending on the distance between clusters to reduce the control overhead. That is, while the control packets are sent in every T_{SAM} seconds (which is chosen as 3s for simulations) to 1-cluster-hop neighbors, they are sent to n-cluster-hops neighbors in $n \times T_{SAM}$ seconds. Each cluster head stores this CSA view constructed by SAMs in its *visibility matrix*. Eventually, each cluster head has more fresh topology information about closer clusters, and CSAs are maintained proactively by cluster heads, as a natural extension of clusters.

The main reason for the construction of CSA is creating a sense of a smaller network that is relatively easy to maintain from the viewpoint of a cluster head. Since proactive maintenance of the network-wide routes is costly, CSA is considered as an effective and easy-to-maintain approach in CUPS. Note that the size of CSA and the frequency of SAMs, i.e., T_{SAM} are the design parameters to be selected depending on the use case.

4 CUPS-based hierarchical routing algorithm (CHRA)

In this section, we present the dynamics of CHRA. Besides, we briefly discuss the fundamental differences between CHRA and its opponent hierarchical routing algorithms comparing the presented dynamics with the alternative methods.

CHRA consists of two routing techniques:

- *In-CSA communication* represents the communication inside a CSA using a proactive link-state approach.
- *Out-of-CSA communication* means end-to-end communication outside the CSA, where the number of hops between source and destination nodes is relatively higher. In this case, we follow a reactive fish-eye approach.

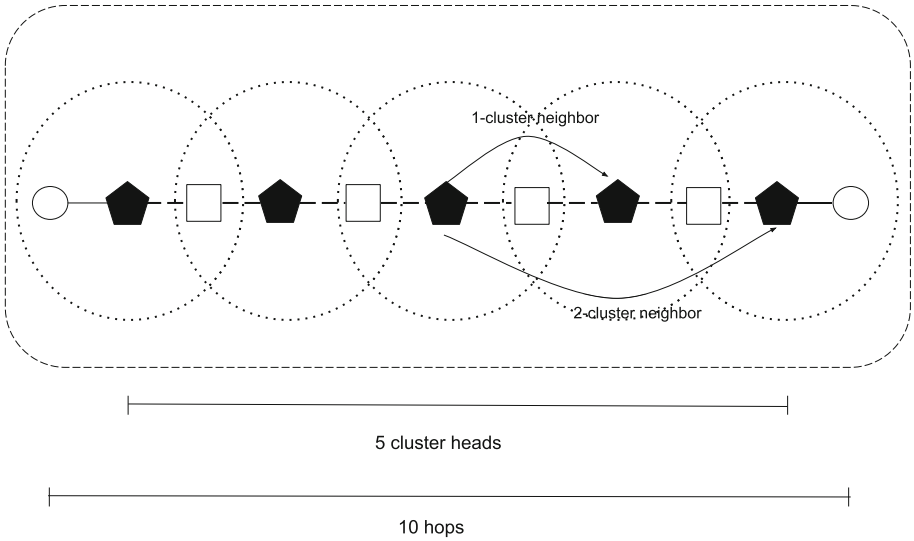


Fig. 2 Cluster sight area covers maximum 10-hop and cover 2-cluster-hop

Algorithm 1 Hybrid route discovery process.

```
procedure RouteDiscovery
    Source node sends an RREQ to cluster head containing destination address
    ▷ Cluster head checks;
    if Destination node in the visibility matrix then
        Cluster head sends RREP containing the shortest path to RREQ-source
    else if Destination node in distance table then
        Cluster head sends RREP containing next-hop information to RREQ-source
    else
        Cluster head forwards RREQ to other cluster heads via gateways
    ▷ Source node checks;
    if No RREP received until timeout then
        Source node initiates route discovery again
    else
        if RREP contains full path then
            Source node updates routing table with the shortest path
        else if RREP contains only next-hop then
            Source node updates distance table
```

A source node initiates the route discovery process sending a route request (RREQ) to its cluster head when it sends or forwards a packet to a destination node (i) for the first time or (ii) after a predetermined timeout duration. There are two possible outcomes: (i) If the destination node is in the CSA of the cluster head, then it can determine the route using a shortest-path algorithm within CSA, and then populates the routing table of the source node, e.g., using software-defined networking approaches (in-CSA communication). (ii) If the destination node is not in the CSA of the cluster head, the cluster head forwards RREQ to the neighbor cluster heads over gateways on the control plane backbone (out-of-CSA case). This forwarded RREQ is processed by each cluster head using the above approach by checking whether or not the destination node is the CSA. The RREQ is forwarded until a path is found by one of the cluster heads. This process is briefly summarized in Algorithm 1. In this section, we introduce those two approaches in detail.

4.1 In-CSA communication

In-CSA communication is illustrated in Fig. 3. Here, (*a*) and (*f*) are the source and destination nodes, respectively. To find a route to (*f*), (*a*) sends an RREQ to its cluster head (*g*) in step (1). When the cluster head receives this RREQ, it first checks its visibility matrix if (*f*) is visible, i.e., whether or not in CSA. If it is visible, the cluster head runs Dijkstra's shortest path algorithm and finds the shortest path on the data plane.

However, (*f*) is not in the visibility matrix of (*g*) in Fig. 3. Consequently, the RREQ in step (1) cannot be answered directly. Instead, (*g*) forwards the RREQ to neighbor cluster (*h*) via the gateways through the

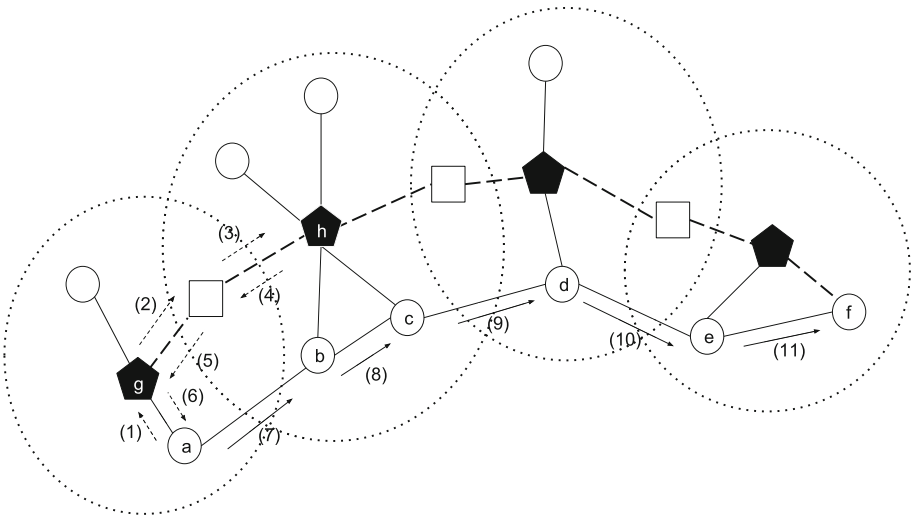


Fig. 3 In-CSA communication scenario. Cluster head (h) knows the presence both source (a) and destination (f) in its visibility matrix and finds a path

backbone, i.e., the control plane in steps (2)–(3). As (h) is aware of the whole topology shown in Fig. 3 having the 2-cluster-hop large CSA, it can find a route from (a) to (f) on the data plane, i.e., not on the backbone.

Afterwards, (h) sends back the path [a–b–c–d–e–f] to (g) in steps (4)–(5). (g) notifies the source (a) with a routing response (RREP) containing the demanded route and (a) stores this path in its routing table in step (6). A routing table contains the ID of the destination node, the path to the destination node and the length of this path; and is constructed for only in-area communication. The routing process is then finalized in the control plane.

Finally, (a) forwards the data packets tailing path (i.e., using source routing) to (b), and the forwarding process is continued hop-by-hop through the steps (7)–(11) in the data plane that consists of ordinary nodes. If an intermediary node is not aware of that particular path, it caches the path and forwards the packet. Otherwise, it assumes that the path is used before and the next nodes in this path are aware of this path as well, and removes the path from the data packet before forwarding to decrease the size of the packet. Note that intermediary nodes can use the cached paths for only data forwarding. It means that they do not use an indirectly obtained path for initiating an end-to-end communication as a source node. The reason behind the restriction in caching is keeping track of the path configurations of the data plane so that they can be alerted in case of a routing error due to a broken path, which is discussed in more detailed in the "Appendix".

4.2 Out-of-CSA communication

When there is no cluster head knowing both source and destination nodes in its CSA, data packets are directly forwarded through the backbone. In Fig. 4, the distance between the source node (a) and destination node (f) is $p + 1$, where $p \geq 4n + 2$. When (a) sends an RREQ to its cluster head in step (1), it cannot find a path. Therefore, the cluster head forwards the RREQ to the neighbor cluster heads via gateways. Since no intermediary cluster head has both source and destination nodes in its visibility matrix, the RREQ is forwarded until it reaches to the cluster where (f) belongs to, in steps (1)–(p). Afterwards, the respective cluster head sends an RREP back to the originator of the RREQ in steps (p + 1)–(2p). In each step (p + i), the receiver backbone node records where the packet comes from and its distance from the destination node to its distance table. For instance, the node which receives (p + 1), becomes aware of that it can send packets to (f) through the cluster head in two hops. Similarly, the receiver of RREP (p + i) knows that the destination node is $i + 1$ hops away. Note that when an RREP offering a shorter path is received, the distance table is updated with this shorter distance and the related next-hop node. Eventually, the route is defined through the backbone and (a) forwards its data packets accordingly.

4.3 Decision for in- and out-of-CSA communication

It is not always possible to determine instantaneously if the destination node is in CSA. Besides, even though the source node sends only a single RREQ to its cluster head, it is then forwarded to all adjacent clusters. Therefore, it is quite common that multiple routes, which reside inside or outside of a CSA, are obtained. In Fig. 5, two alternative routes are found using two different methods: packets (2)–(n) lead to a backbone-out-of-CSA communication and

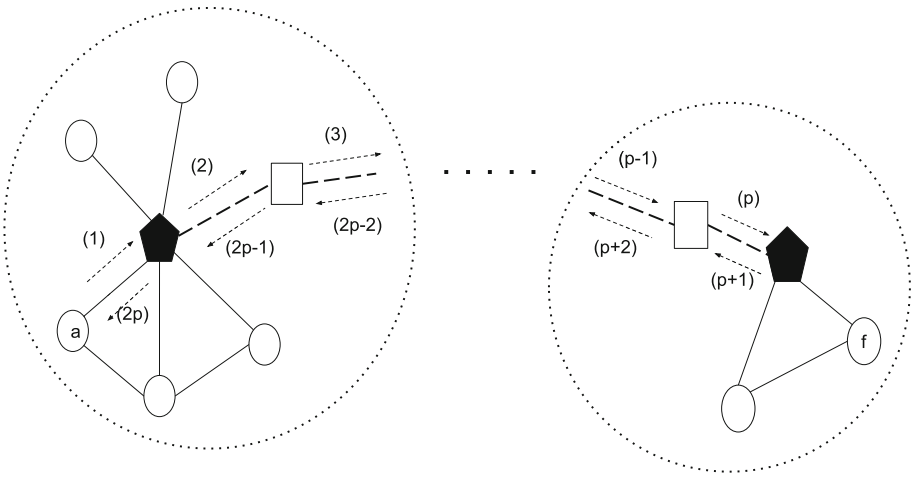


Fig. 4 Communication in longer distances is constructed on the backbone

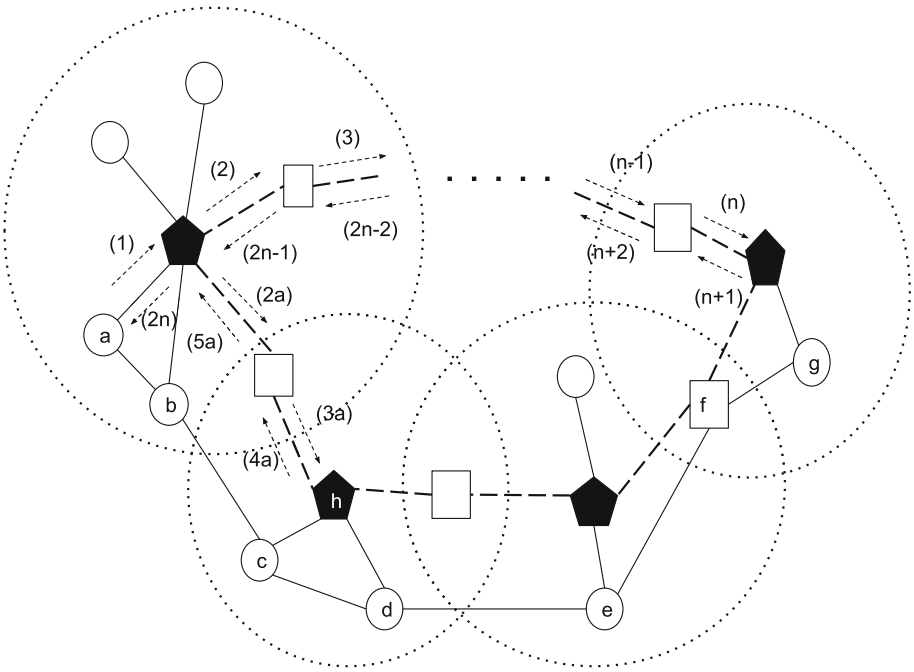


Fig. 5 End-to-end path in data plane is priorly preferred over the backbone-dependent one

packets (2a)–(3a) provide the data plane path since the cluster head (*h*) has both source and destination nodes in its visibility matrix. In this case, for both reducing the traffic load on the backbone and selecting the shorter path, the path [*a–b–c–d–e–f–g*] is used to forward the data packets.

Lastly, even though the maintenance of the topology is mostly handled by clustering mechanisms in hierarchical ad-hoc networks, the repair/recovery of broken routes are still required to ensure communication reliability. Therefore, we introduce two techniques, local repair and global repair that optimize control overhead and the route maintenance time. As we mostly focus on the CUPS concept and its performance gains, we briefly describe those techniques at the end (in "Appendix") to keep the definition of CHRA here simpler.

4.4 Differences between hierarchical routing methods

In this section, we present a qualitative comparison between CHRA and its two opponents, CBRP and HCR, to emphasize the differences of CUPS-based dynamics in CHRA. The comparison is also briefly presented in Table 1. The quantitative comparison as a result of our simulations is also presented in Sect. 5.

For network discovery in CBRP, nodes broadcast their identifiers to their neighborhood. In CHRA and HCR, apart from individual nodes, cluster heads attempt to a wider discovery sharing cluster information. However, the topology discovery of cluster heads is limited by the size of CSA in CHRA, while HCR performs a global discovery.

Table 1 Differences in route request and repair processes between CHRA, CBRP and HCR

Phase	Algorithm		
	CHRA	CBRP	HCR
Bootstrapping	Only CHs send their cluster formation to the neighbor CHs	All nodes broadcast their neighborhood information	All nodes broadcast their neighborhood information, also CHs spread the backbone structure
Discovery	Only CHs and gateways forward RREQ packets	Only CHs and gateways forward RREQ packets	Only CHs and gateways forward RREQ packets
	RREQs are forwarded until (i) first CH that finds a path in in-area communication or (ii) CH of destination node's cluster in long-distance communication	RREQs are forwarded to destination node	RREQs are directly responded by the first CH
Forwarding	Data packets are forwarded on data plane	Data packets are forwarded through backbone via CHs	Data packets are forwarded through backbone via CHs
Repair	Link failures are notified to direct CH. If there is no local repair possibility, RERRs are sent through backbone	Link failures are broadcast to source node	Link failures are unicast to source node

Moreover, CHRA utilizes a fish-eye approach to decrease the control overhead for the discovery of farther clusters. Therefore, HCR is the most costly algorithm in terms of network discovery. Note that the routing technique of HCR mostly depends on this discovery scheme. Accordingly, the routing overhead of HCR becomes higher in comparison to CHRA and CBRP.

For data forwarding, CBRP and HCR have similar dynamics: Data packets are forwarded through the backbone. The main difference in routing is that while CBRP employs a distance vector-based reactive approach, HCR performs source routing using proactively maintained topology information to find routes on the backbone. On the other hand, CHRA offloads most of the data forwarding to the data plane nodes, which leads to a fairer overall energy consumption as well as the use of alternative paths via ordinary nodes.

For route maintenance, CHRA quickly reacts to perform local repair with minimum overhead in case of a broken route and extend it to a global repair scheme to simplify maintenance. In contrast, CBRP and HCR directly attempt global repair using broadcasting and unicasting to the source nodes of the broken routes, respectively.

Except AODV, the other routing algorithms are built upon clustered networks. CHRA and CBRP are using a basic ID-based clustering algorithm [37]. In that class of clustering algorithms, each node has a unique identifier, and after exchanging control messages during a bootstrapping period, the nodes with the lowest (or highest) identifiers are selected as cluster heads. HCR, on the other hand, combines the ID-based approach with another degree-based approach, where the nodes with higher connectivity have priority to be selected as cluster heads. Although routing is generally relevant to the underlying clustering technique, the evaluation of its impact is not in the scope of this study.

5 Results and discussion

In this section, we evaluyate the performance of CHRA. All tests are conducted in OMNeT++ using the simulation parameters in Table 2. To validate the advantages of plane-

Table 2 The values of the simulation parameters

Parameter	Value
Area size	200 m × 200 m
Runs per batch	200
Scenario duration	200 s
Transmission radius per node	40 m
Node density	0.001–0.0015 node/m^2
Ratio of mobile nodes	30%
Speed of nodes	2–10 km/h
Path loss model	Free space
Power consumption model [38, 52]	150 mW Tx
	60 mW Rx
	2 mW Idle
Background noise	− 90 dBm
Mobility model	Random Waypoint [27]

separation, CHRA is compared with (i) Cluster-Based Routing Protocols (CBRP), which employs the common routing method used in almost all hierarchical routing algorithms, (ii) Hybrid Cluster Routing (HCR) [32, 42] as a superior of CBRP and a more recent approach to routing in clustered topologies and (iii) AODV on flat topologies. Therefore, CBRP and its superior HCR are the important comparison elements, while AODV is used for benchmarking for a non-hierarchical approach.

The simulations are conducted in both mobile and stationary scenarios with uniformly and non-uniformly distributed topologies. The triangular distribution is used for non-uniformly distributed network scenarios. It represents the topology, where the majority of the nodes tend to gather around a particular area and some other nodes are spread as outliers. Ideal link layer and physical layer models are considered.

For traffic generation, a UDP application is implemented. It periodically (every 2s) sends UDP packets between randomly selected source and destination nodes. The size of a data packet is defined as 300B. For routing, the size of control packets flowing through the backbone is selected as 64B as it is shown to be the optimum size for AODV discovery packets as well [28]. The same packet size is also chosen for the standard AODV algorithm. Apart from that, the cost of topology discovery to form cluster sight area is considered in CHRA for a fair comparison.

For mobility scenarios, the random waypoint mobility model is employed, where the nodes move towards a random direction in a given speed range periodically. It represents a heterogeneous network model that nodes (or a certain group of nodes) can move individually with different patterns, i.e., varying direction and speed whether following an inter-related pattern or not. It also triggers route update and recovery mechanisms more often due to such randomness, which imposes further challenges for route maintenance. Lastly, signal transmission radius per node is selected as 40 m being proportional to the network size, i.e., 200 m $\times$ 200 m, for medium coverage. Note that similar proportions, e.g., being scaled up, in terms of network coverage for hop-to-hop communication, are also used to evaluate different ad-hoc routing protocols [9, 45], including military tactical scenarios under certain mobility models [48].

In the rest of this section, the performance measures to compare algorithms and the performance results of all four cases (different distributions in stationary and mobile scenarios) are presented and discussed.

5.1 Performance measures

We use six performance measures to evaluate CHRA and its opponents.

1. Average energy consumption ratio per node: It measures the average of the total energy consumption of a node during the simulation time. To get more realistic results, the radio state-based power consumption model is adjusted according to well-known chips Microchip RN1810 [38] and SparkLAN WSDB-102GN [52]. While 150 mW is consumed for packet transmission, it is 60 mW and 2 mW for reception and idle state, respectively. In the respective figures for this measure, we note the standard deviation in energy consumption of nodes for each scenario via the error bars to represent if some nodes consume significantly more energy than the other, i.e., higher deviation from the average.
2. Number of routing control packets: It measures the total number of routing control packets, which is the control overhead.
3. Data-to-All ratio (DAR): DAR is defined as the ratio of the total size of successfully delivered data packets to the total size of all packets including CSA overhead for CHRA and global discovery for HCR, i.e., data packets and control packets together.
4. Average end-to-end delay: This measure represents the average length of paths used for forwarding to evaluate the quality of service.
5. Data plane gain (DPG): It is the ratio of data packets that are forwarded through to data plane to the all data packets. It represents how effectively data plane takes role in data forwarding. DPG is defined as the percentage of data flowing (i) through both gateways and ordinary nodes (excluding only cluster heads) and (ii) through only ordinary nodes (excluding both CHs and gateways).

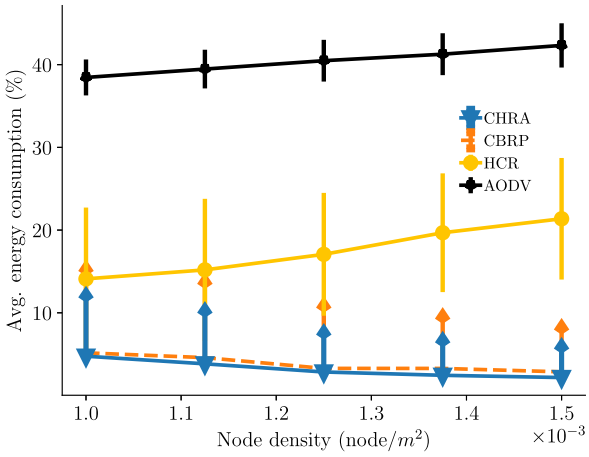
The listed measures are selected to evaluate different aspects such as (1) to measure energy-efficiency, (2) and (3) to measure control overhead, (4) to measure the quality of service, and (5) to measure the effectiveness of plane separation.

5.2 Results in stationary scenarios

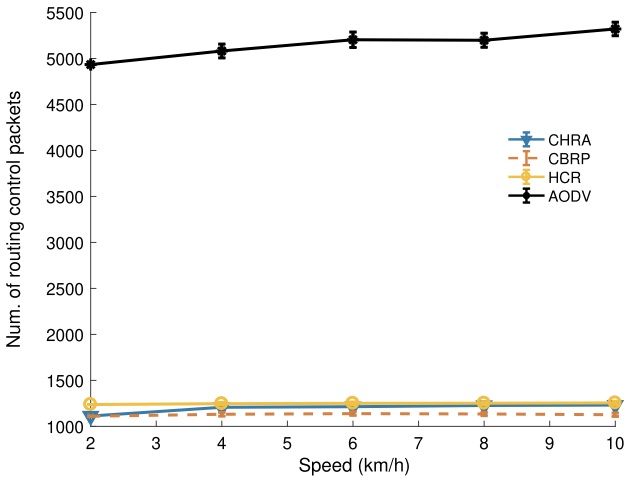
Figures 6 and 7 show the effects of node density (i.e., increasing number of nodes) in uniform and non-uniform node distributions.

Figures 6a and 7a show the average energy consumption together with the deviation in energy consumption between nodes (with error bars). Note that here the standard deviation is averaged calculating the deviation of the energy consumption of nodes in each simulation run.

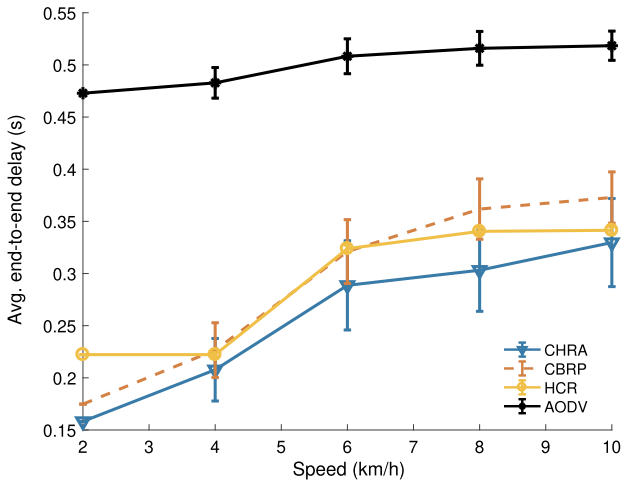
In those figures, the average energy consumption is nearly the same for CHRA and CBRP. The deviation in energy consumption of nodes utilizing CHRA is nearly



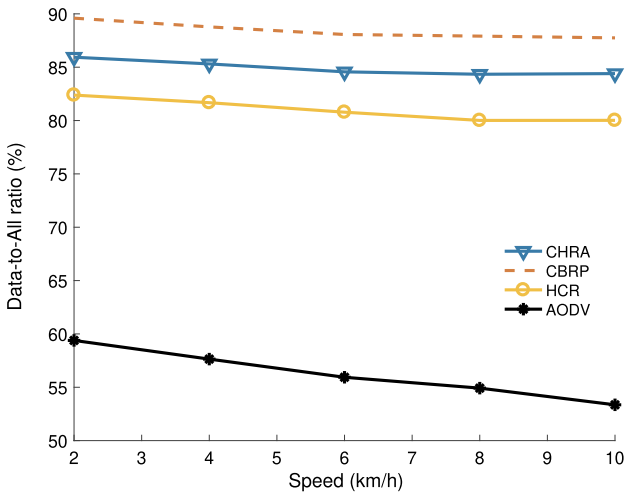
(a) Energy efficiency.



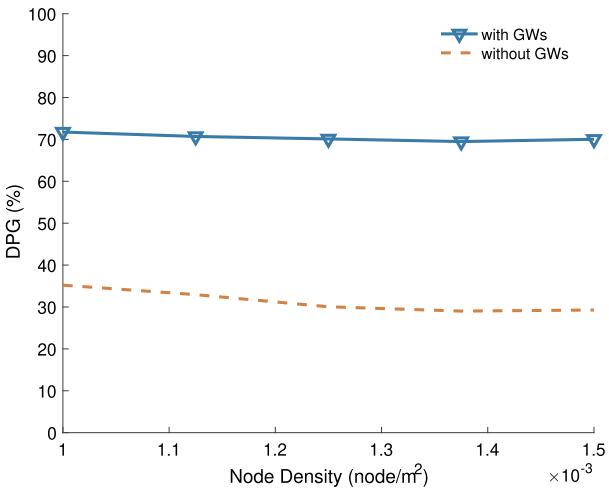
(b) Number of control packets.



(c) Average end-to-end delay.

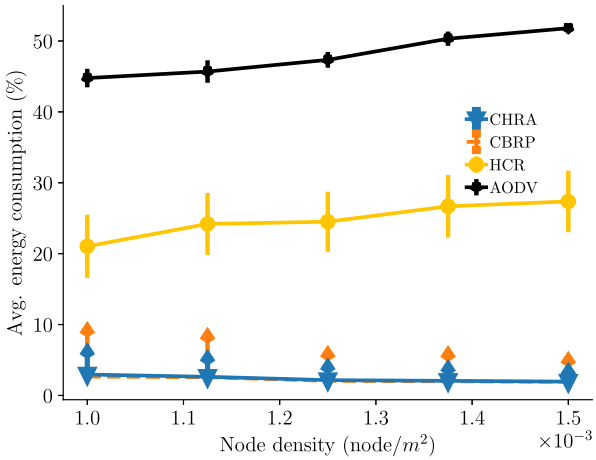


(d) Data-to-All ratio.

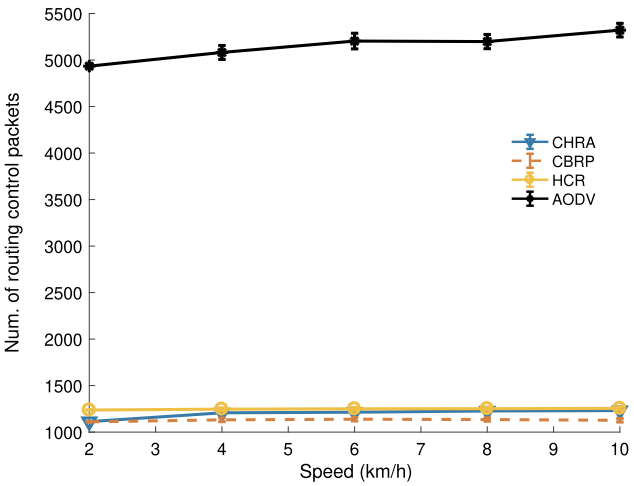


(e) Data plane gain.

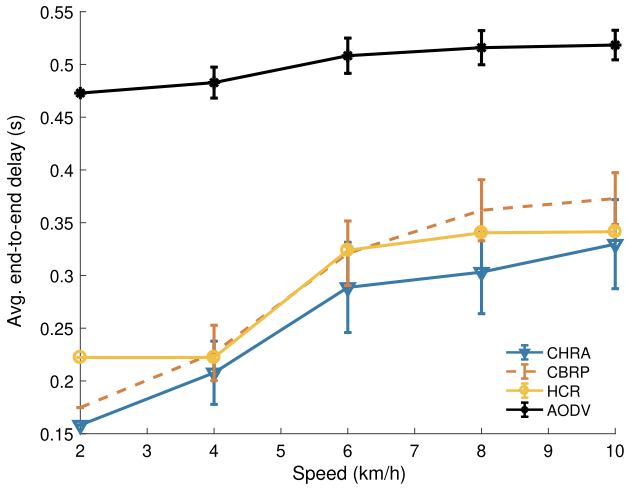
Fig. 6 The effects of the network density in uniformly distributed scenarios



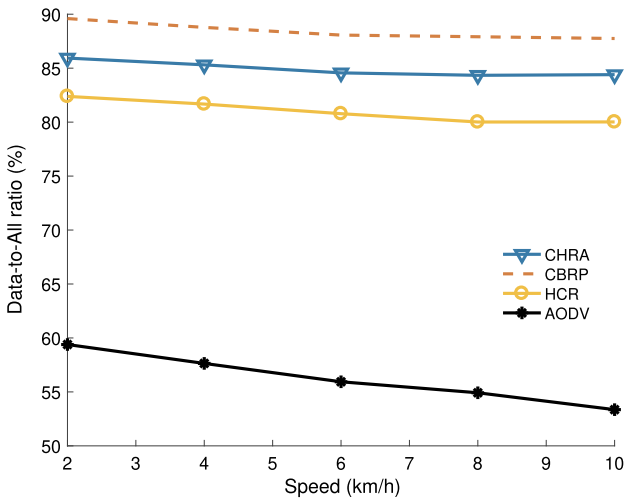
(a) Energy efficiency.



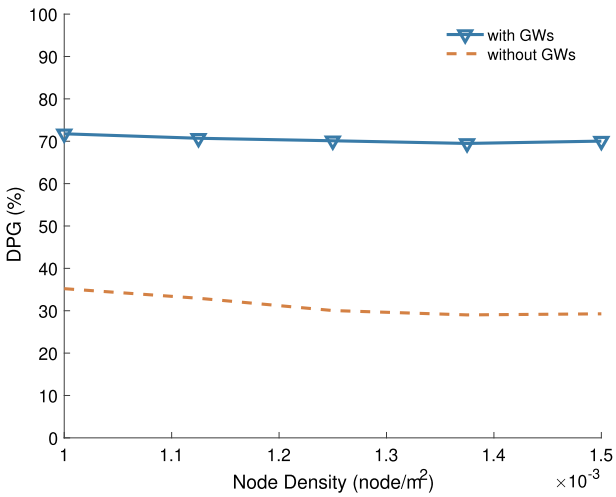
(b) Number of control packets.



(c) Average end-to-end delay.



(d) Data-to-All ratio.



(e) Data plane gain.

Fig. 7 The effects of the network density in non-uniform scenarios

zero as it distributes the load, e.g., control and data traffic, fairly via control and user-plane separation. As nearly all nodes tend to broadcast control packets via flooding in AODV, they consume similar amount of energy with low deviation even though it is much higher in total consumption than CBRP, CHRA, and HCR. Similarly, HCR depends on network-wide discovery packets where cluster heads discover the full backbone structure including cluster topologies. Moreover, the source routing technique of HCR does not use the shortest paths but finds the path from source to destination by forwarding the packets through the backbone instead. Therefore, topology discovery and forwarding process cost two-times energy than CBRP and CHRA with a large deviation as the backbone nodes are loaded much higher than the ordinary nodes.

Figures 6b and 7b show the number of control packets including route request and reply packets, and also repair, recovery and CSA packets for CHRA and HCR. The overhead of CHRA is slightly higher than that of CBRP due to repair, recovery and CSA packets. In contrast, AODV has the highest overhead in terms of the number of the control packets due to its flooding mechanism. HCR has higher control overhead in terms of the number of packets than the other hierarchical routing protocols due to its global discovery scheme. However, since a number of control packets contain topology information for clusters and all inter-cluster data packets have piggy-backed backbone information, the size of routing control packets vary, which directly affects the efficiency—or DAR—of communication.

The quality of service is evaluated in terms of DAR and end-to-end delivery delay. In Figs. 6d and 7d, the DAR generally remains above 90% for CHRA and CBRP while AODV's is much lower, 75% at maximum. Since the control overhead in CHRA is slightly higher due to CSA maintenance and route recovery, the DAR in CHRA is lower than CBRP, around 2–5%. Such difference is much higher for HCR due to its global discovery and source routing scheme. On the other hand, AODV shows quite poor performance in terms of DAR due to high control overhead. Figures 6c and 7c show that CHRA has the lowest end-to-end delay as it finds the shortest path inside CSAs. In contrast, CBRP has very limited alternative paths because they are constructed through only cluster heads and gateways. HCR has a lower latency having a proactive routing scheme through the backbone. Note that HCR does not propose the use of shortest paths and data packets are forwarded through the backbone until they are reached to the destination cluster. Eventually, CHRA outperforms other algorithms in terms of the end-to-end delay. Moreover, the difference between uniform and non-uniform scenarios is notable. Since non-uniform deployment causes a denser formation, many nodes are placed around a certain area and this topology decreases average end-to-end delay in communication.

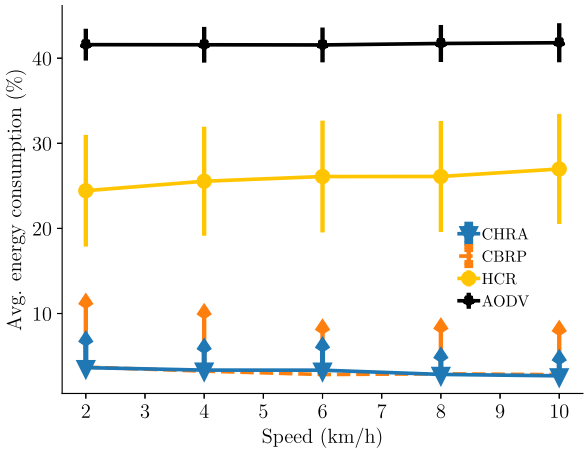
DPG is shown for stationary scenarios in Figs. 6e and 7e. In uniformly distributed scenarios, 45–50% of forwarding burden is offloaded to ordinary nodes. In comparison, the same amount of data is forwarded through only the control plane nodes, i.e., cluster heads and gateways, in CBRP and HCR. In total, 70–80% of data forwarding is handled together by ordinary nodes and gateways. While DPG with GWs is the same for uniform and non-uniform scenarios, DPG for only ordinary nodes (without GWs) decreases to 20–30%. Because higher number of nodes become gateways in the denser formation of non-uniform distribution and it directly affects the number of gateways in the data plane. Similarly, while node density is increasing in both scenarios, the activity of the data plane ascends with the increasing possibility of in-CSA communication.

Eventually, Figs. 6 and 7 shows that in stationary scenarios, CHRA has the lowest average energy consumption with a minimum standard deviation, and offers both efficient and fair energy usage. While causing slightly higher overhead than CBRP to maintain the cluster sight areas, CHRA still has more than 90% data-to-all ratio. However, it is seen that leveraging CSA with a certain overhead, CHRA can utilize end-to-end paths with lower delay than its opponents.

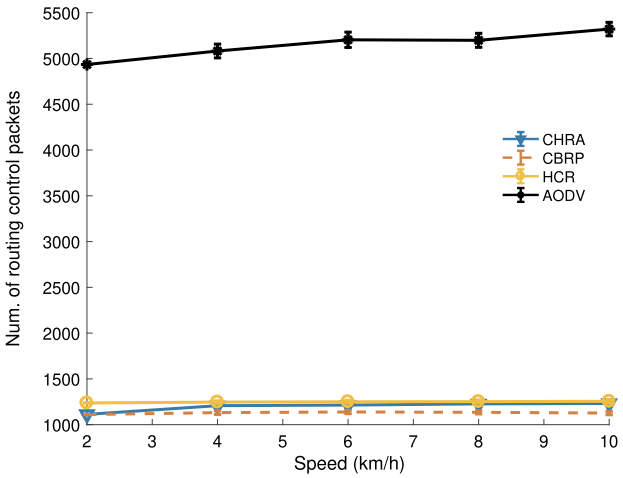
5.3 Results in mobile scenarios

Figures 8 and 9 show the effect of increasing speed of nodes in mobile scenarios with uniform and non-uniform node distributions. For such scenarios, the node density is fixed to 0.00125 node/m^2.

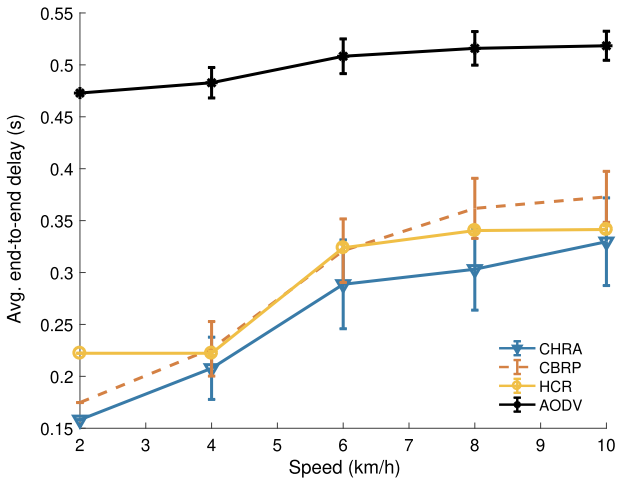
The average energy consumption is slightly higher than the results in stationary scenarios as shown at the lines in Figs. 8a and 9a as re-routing, recovery, and repair processes are more frequent in mobile scenarios. As seen in Figs. 8a and 9a, CHRA has a minimal deviation in energy consumption even in higher mobility with increasing speed. HCR, similar to stationary scenarios, show a high deviation in uniform and mobile scenarios with moderately high average consumption. In uniform scenarios, the increasing speed of nodes observably affects the deviation in energy consumption for each routing technique since the mobility strongly changes the already-sparse network distribution. In contrast, it is not affected especially for CHRA since the tolerance to mobility in a denser area that is mostly covered by CSAs is much higher. Therefore, routes can be maintained more effectively in CHRA thanks to the CSA.



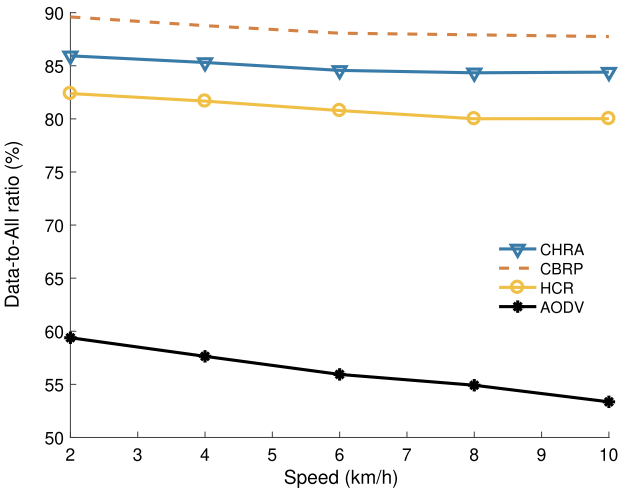
(a) Energy efficiency.



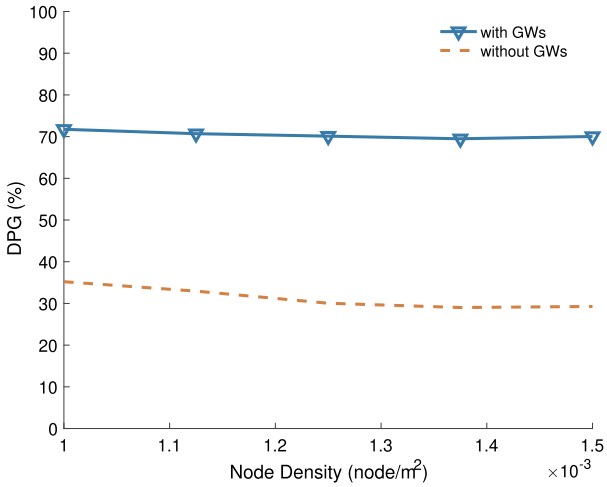
(b) Number of control packets.



(c) Average end-to-end delay.

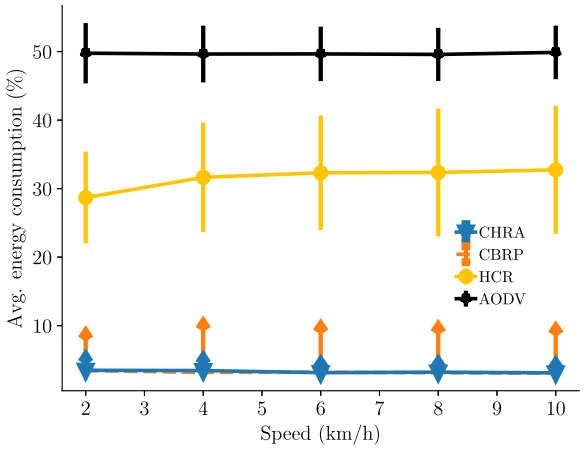


(d) Data-to-All Ratio.

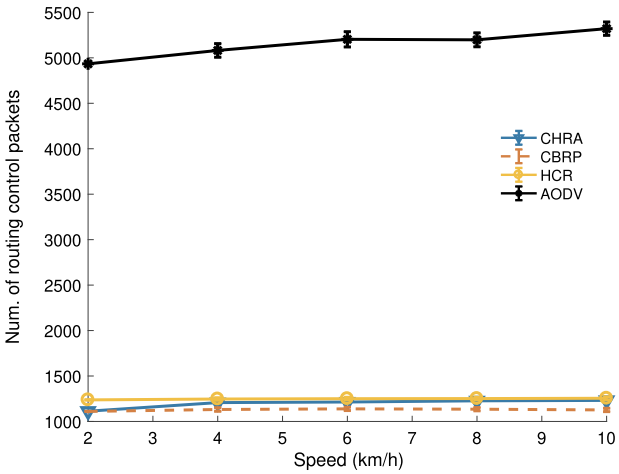


(e) Data plane gain.

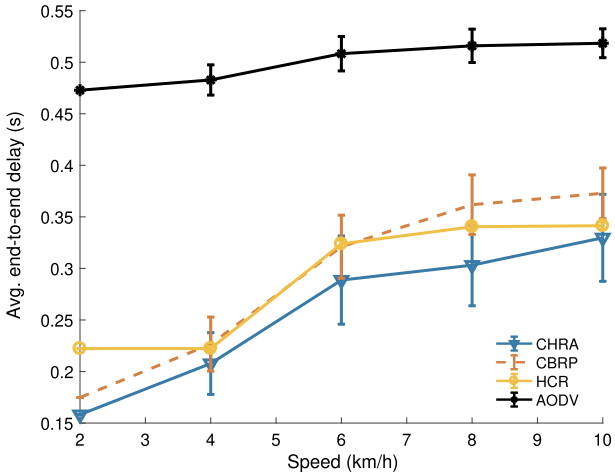
Fig. 8 The effects of the speed in uniformly distributed scenarios



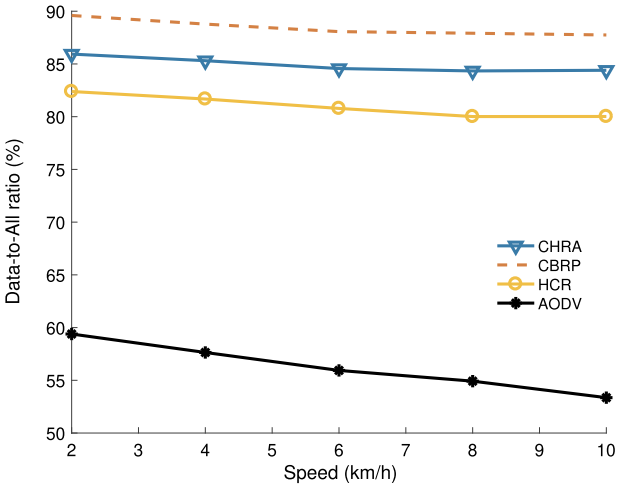
(a) Energy efficiency.



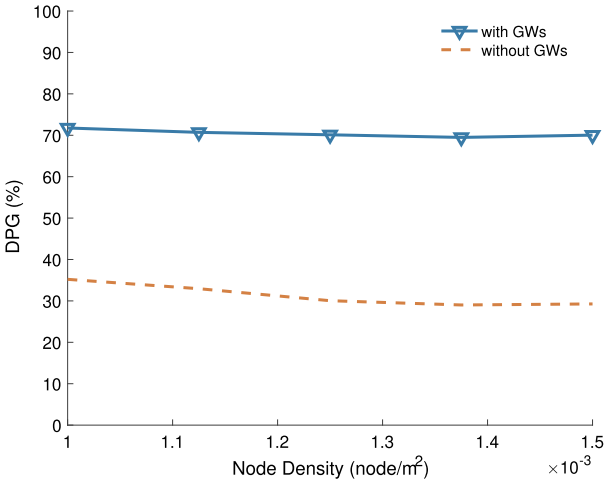
(b) Number of control packets.



(c) Average end-to-end delay.



(d) Data-to-All ratio.



(e) Data plane gain.

Fig. 9 The effects of the speed in non-uniform scenarios

Figures 8c and 9c show that the overall end-to-end delay for all routing algorithms is lower in non-uniform scenarios than uniform scenarios. CHRA has the lowest end-to-end delay in any case, even if it is increasing with the increasing speed of nodes for the other algorithms. Interpreting Figs. 8d and 9d, it is seen that CHRA decreases the end-to-end delay preserving a high DAR as 85%. The DAR is decreasing with the increasing speed of nodes since the high mobility triggers routing process and eventually increase routing control overhead. HCR has the lowest DAR among hierarchical routing algorithms due to the high overhead of proactive maintenance of routes, especially in mobile scenarios. Note that CHRA is affected by mobility more than CBRP and HCR because the data plane routes depend on mobile ordinary nodes instead of the backbone that is relatively easy to fix during the clustering maintenance. When a new cluster head is selected, the cluster neighborhood can be recovered easily and the traffic flows through the backbone. However, thanks to the route repair and recovery process for the data plane-routes in CHRA, even if it is more sensitive to mobility, there is not a DAR decrease due to the increasing speed of nodes. The difference of DAR results between CHRA and CBRP is caused by the extra recovery process and CSA maintenance in CHRA. This difference is even larger between CBRP and HCR for proactive route maintenance. Figures 8b and 9b show such extra overhead in terms of the number of control packets. Besides, they explain the low-DAR results of AODV that demonstrates a high control overhead.

Figures 8e and 9e show DPG for mobile scenarios. In uniformly distributed scenarios, 30–35% of total data is forwarded by ordinary nodes, while this ratio is 70% when gateways are included. The gain is less than stationary scenarios since (i) maintenance of paths are getting harder and in case of broken paths due to mobility, long-distance paths through the control plane are used as backup routes, (ii) finding paths also becomes harder and more stable control plane nodes are—relatively—more frequently preferred and (iii) role changes occur more frequently and they eventually affect route stability. Expectedly, the increasing mobility of nodes negatively affects the gain since the stability of data plane decreases proportionally. However, there is at least 30% of data burden offloaded from cluster heads and gateways to the ordinary nodes in all scenarios. When the participation of gateways is also considered, DPG is significantly higher in both stationary and mobile networks.

In short, Figs. 8 and 9 shows that the performance of CHRA in mobile scenarios is very close to the results in the stationary scenarios. In terms of energy consumption, CHRA still offers the lowest energy consumption with better fairness. As it uses repair and recovery mechanisms more actively in mobile scenarios, its DAR is slightly lower than the stationary scenarios. However, as those mechanisms bring reliability for the end-to-end communication, CHRA can deliver more data packets and therefore, the difference in DAR between CHRA and CBRP is even lower for mobile scenarios. Lastly, CHRA provides more than 70% data plane gain Independent of mobility.

6 Discussion

In this section, we discuss some additional points on the application of CHRA.

Inspired by the configuration scheme of SDN networks, the routing process of CHRA can be adapted to be more flexible and distributed. Instead of relying on paths offloaded to data plane nodes, a cluster head could configure the nodes in the respective cluster to be aware of only the next hop. This approach would prevent ordinary nodes from knowing about end-to-end paths and ease the routing process in the control plane, e.g., not returning back the whole path to the source node. Besides, route maintenance becomes relevant only between a cluster head and the related ordinary nodes instead of maintaining paths. This improvement will be examined as an extension of this study.

Out-of-CSA, or long-range communication, is a practical yet still costly method for the control plane. To overcome that, instead of relying on a single cluster head that has discovered both source and destination nodes, i.e., in the same CSA, it is also possible to merge multiple segments in sequence to construct an end-to-end path using the vision of multiple cluster heads. This approach would increase the probability of finding paths on the data plane before using the backbone as we have done in Out-of-CSA technique. Another improvement would be using the backbone partially for forwarding. That is, after forwarding data packets through the backbone for a certain number of

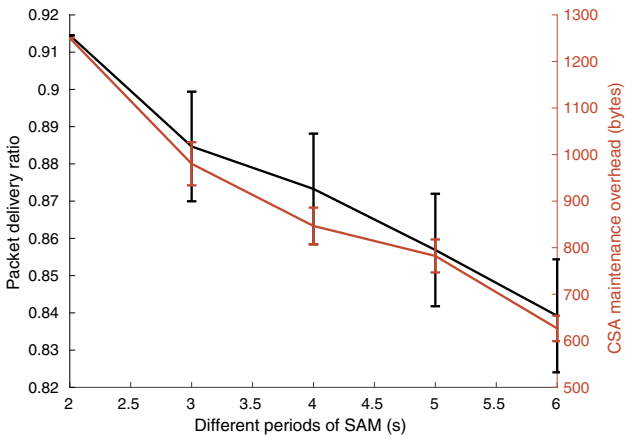


Fig. 10 The effects of T_{SAM} on packet delivery ratio and overhead

hops in case of the absence of an initial path, it is still possible to switch to a path on the fly whenever it is found. Both alternative approaches would strengthen the separation of planes by decreasing the use of control plane for forwarding.

Lastly, the control overhead for CSA maintenance is another issue to analyze. As seen in Fig. 10, PDR is decreasing with increasing period of SAM packets, T_{SAM}. In contrast, control overhead is getting less with more infrequent SAM packets as expected. The reason is, the infrequent SAM packets directly lead to routing based-on obsolete topology information. In this case, packets cannot be forwarded through destination when the route repair is not possible. In this manner, T_{SAM} need to be decided based on mobility characteristics of the network.

CSA directly affects the efficiency of paths for the in-CSA communication. Therefore, the size of CSA will be investigated in more detail possibly for larger networks in our future works.

7 Conclusion

Mobile ad-hoc networks have been deployed in a variety of scenarios such as tactical, emergency, and sensor networks as well as the emerging ecosystems like IoT and 5G cellular networks. While their self-organized nature, e.g., without any pre-deployed (or only partially deployed) infrastructure, procures a degree of freedom in design, the lack of an orchestration mechanism, e.g., a centralized controller, complicates their maintenance and establishing end-to-end communication between increasing number of nodes in such networks. Although several routing protocols have been proposed to establish communication autonomously, they usually bring excessive control overhead and thus, a significant energy consumption due to the costly network discovery, maintenance, and packet forwarding mechanisms that usually rely on the utilization of a limited set of nodes or flooding in a flat topology.

In this work, we presented a plane-separated hierarchical routing algorithm, CHRA, in ad-hoc networks to address those problems. In CHRA, we employ control and user plane separation (CUPS), which has recently become popular with the emerging software-defined networking (SDN) paradigm to separate routing and forwarding as two distinct functions of different network elements. We utilize this concept to construct a control plane that can efficiently discover and maintain a network in a self-organized manner. The separation of the control plane and the data plane leads to finding alternative routes in the data plane that are not dependent on the backbone, i.e., formed by the control plane nodes, in contrast to many other hierarchical routing algorithms presented in Sect. 2. The utilization of data plane routes provides a fair energy-consumption scheme since a significant data forwarding burden is taken from the control plane and distributed to ordinary nodes in the data plane.

We discuss the whole picture and the major dynamics of the algorithm in different scenarios, which are stationary and mobile scenarios in uniform and non-uniform distribution. Our results show that CHRA can utilize the paths on the data plane effectively to offload up to 75% of data packets and achieve a low deviation in energy consumption ensuring the lowest end-to-end delay. It however has a slightly higher control overhead to maintain the information for its route discovery mechanism in stationary scenarios. Furthermore, CHRA offers a very similar performance in mobile scenarios with slightly higher control overhead due to the cost of route repair and recovery mechanisms considering broken links due to the mobility.

Appendix: Routing errors and route maintenance

In this appendix, we present how to handle routing errors and repair them.

Routing errors

A broken link in the backbone is easy to detect since cluster heads have a periodic message exchange scheme for clustering control packets. However, it is not always possible to detect a broken link between two ordinary nodes. In such cases, any path containing the broken link loses its validity. Other nodes using related invalid routes need to be informed about the broken links with minimum overhead. Therefore, a route recovery and maintenance mechanism is required to continuously manage routes in the data plane.

Figure 11 shows a routing error scenario that occurs in the data plane. In the scenario, (e) is not available anymore due to mobility, or a node crash. When (a) uses the route obtained in steps (1)–(2), data packets are forwarded through (d), which should detect (e) is lost/down and the path is broken. Accordingly, it deletes any recorded route in which (e) is included and sends a route error (RERR) packet to its cluster head in step (5a). This RERR packet contains the source node of the route, ID of the lost/down node, and a timestamp. When (h) receives the RERR packet, it deletes the lost node from its visibility matrix and all recorded routes containing that node from its routing table(s). Then, it constructs a list of source nodes, which is called *source notification list*, that have requested any of the deleted routes before. Note that the destination nodes of those routes also added to the list since they record reverse routes (i.e., route from destination to source node) as well.

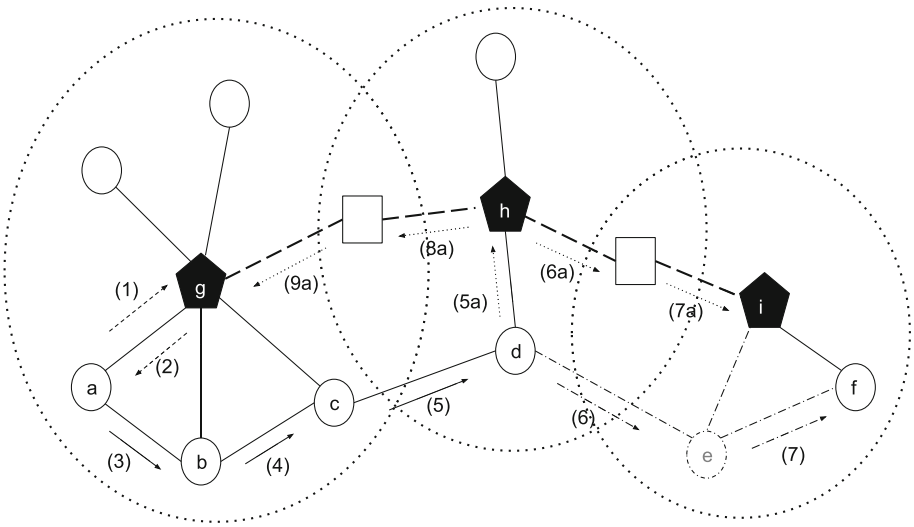


Fig. 11 Routing error in data plane. Absence of node (e) breaks the path constructed between node (a) and node f

Adding the source notification list, it forwards the RERR packet to all neighbor cluster heads. After the first RERR packet, each cluster head applies the same procedure updating the source notification list and forwarding RERR. Besides, cluster heads forward RERR packets to the nodes in the source notification list to update their routing tables as well. Note that this method is only applicable when intermediary nodes are not allowed to use a cached route to initiate a connection. If they do so, source nodes for related routes cannot be tracked and RERR messages need to be broadcast frequently. It creates a significant overhead especially for high-mobility networks.

As cluster heads may receive the same RERR packets multiple times over the backbone, they record the sequence number of RERRs and directly discards duplicates. Besides, since paths are defined in maximum $(4n + 2)$-hop (where CSA has a n-cluster-hops radius), TTL of RERR packets for source nodes is limited to $(4n + 2)$. Eventually, discarding duplicates and the TTL limitation minimize the flooding of RERR packets.

Route repair

There is also an alternative method to overcome excessive number of RERRs that may be an issue in high-mobility networks, that is route repair. When a node detects a broken link, it is able to repair such link before sending a RERR packet.

In CHRA, there are two types of route repair mechanisms. *Local repair* aims for minimum control overhead and modification in an existing route for repair. *Global repair* aims for route reliability with a more controllable approach.

Local route repair

Figure 12 shows an example of local repair where only a single alternative next-hop node is searched to fix the route locally instead of finding a new route or a route segment, i.e. a partial route completing broken route. In this sense, related route is patched with minimum effort and it is not required to spread RERR packets through the backbone. In Fig. 12, [a–b–c–e–f] is constructed between (a) and (f). When (c) detects the broken link to (e), it sends a repair request (RPREQ) to its cluster head in steps (4a)–(5a). Here, (g) directly looks for an alternative (next-hop) node between (c) and (f), instead of an end-to-end path between (a) and (f). Eventually, the only update in the route is forwarding through (d) instead of (e). Whether (a) is aware of the loss of (d) or not, (c) repairs the path without announcing it to the whole network but its cluster head. During repair, the data packets are cached in the node that detects the broken link. As a result, a minimum number of routing control packets is generated and it leads to the higher resource utilization and lower delay communication with a quick fix. However, it is not always possible to perform local repair considering an alternative segment. In such cases, global repair is performed.

Global route repair

Figure 13 shows the global repair that searches for a partial route (or a route segment) after a broken link. Instead of finding a new end-to-end path, it only completes the path after the broken link. In the figure, when (d) detects the broken link, it sends an RPREQ packet to its cluster head in steps (5a)–(6a). Since cluster head (h) cannot find a local repair alternative (an alternative node instead of (e), it draws a totally different path to be replaced with *only* the

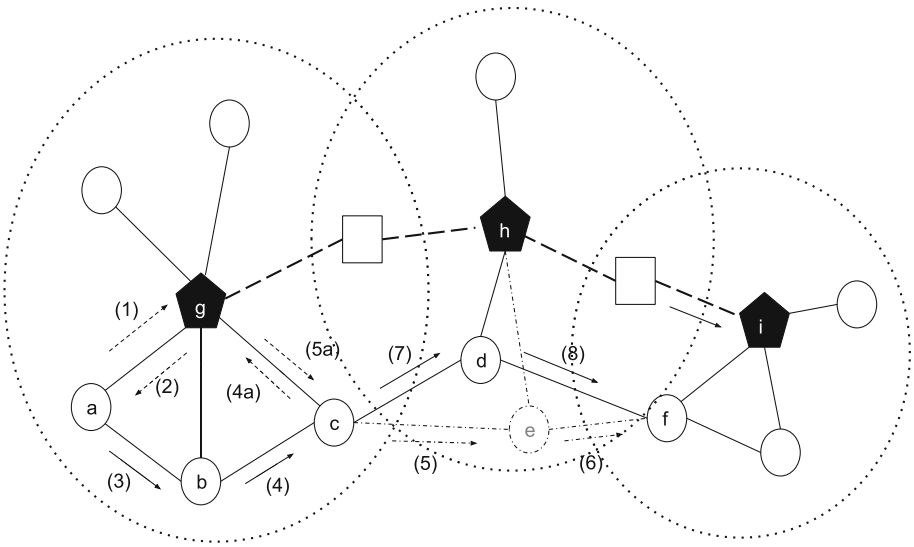


Fig. 12 Local repair is completed when a single next-hop alternative is found. Node (*c*) detects the absence of (*e*) and sends a RPREQ to its cluster head. An alternative node, (*d*), is found and the path is repaired locally

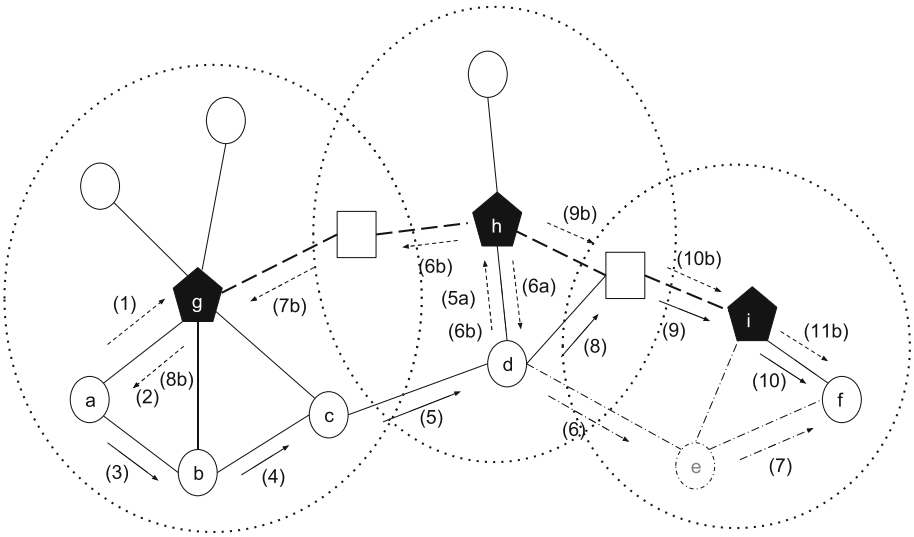


Fig. 13 Global repair is performed when local repair cannot find an alternative next-hop node instead of the lost node. In the figure, (*d*) detects a broken link to (*e*) and asks for repair to its cluster head. Since local repair fails, a new route is drawn and announced to the rest of the network

broken part. For instance, the data packets are forwarded from (*d*) to (*f*) through steps (8)–(10) rather than steps (6)–(7) substituting (*e*). Additionally, cluster head (*h*) sends RERR packets through the backbone including updated path and the identifier of broken path in steps (6b)–(11b) to announce such update to the source and destination nodes. Note that while the invalid route is announced to the network in global repair, it is not a case in local repair. Because the maintenance of a one-hop updated path is relatively easier than a multi-hop path.

Lastly, there could be such scenarios where any type of route repair is not possible at all. However, RPREQ packets are sent in any case since repair cannot be performed without cluster heads that manages the CSAs. Therefore, nodes are waiting for RPREQ response (RPREP) for a limited time, then drops the cached data packets if related RPREP is not received. RERR packets are triggered by the cluster heads that receive RPREQ but cannot repair the broken part.

Acknowledgements This work is partially supported by ASELSAN and TUBITAK Project Number 215E127. A cross-layer framework for OMNeT++, which is implemented through this study, is available at [19].

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Al-Karaki, J. N., & Kamal, A. E. (2005). End-to-end support for statistical quality of service in heterogeneous mobile ad hoc networks. *Computer Communication, 28*, 2119–2132.
2. A new type of battlefield network is in development. *The Economist* (2018). https://www.economist.com/science-and-technology/2018/06/14/a-new-type-of-battlefield-network-is-in-development.
3. Abe, S., Hasegawa, G., & Murata, M. (2018). Effects of C/U plane separation and bearer aggregation in mobile core network. *IEEE Transactions on Network and Service Management, 15*(2), 611–624.
4. Al-Karaki, J. N., & Kamal, A. E. (2008). Efficient virtual-backbone routing in mobile ad hoc networks. *Computer Networks, 52*(2), 327–350.
5. Alam, T., & Benaida, M. (2019). The role of cloud-MANET framework in the Internet of Things (IoT). *CoRR*. arXiv:1902.09436.
6. Alotaibi, E., & Mukherjee, B. (2012). A survey on routing algorithms for wireless ad-hoc and mesh networks. *Computer Networks, 56*(2), 940–965.
7. Arnold, P., Bayer, N., Belschner, J., & Zimmermann, G. (2017). 5G radio access network architecture based on flexible functional control/user plane splits. In *Proceedings of the European conference on networks and communications (EuCNC)* (pp. 1–5).
8. Bannour, F., Souihi, S., & Mellouk, A. (2018). Distributed SDN control: Survey, taxonomy, and challenges. *IEEE Communications Surveys Tutorials, 20*(1), 333–354.
9. Belding-Royer, E. M. (2003). Multi-level hierarchies for scalable ad hoc routing. *Wireless Networks, 9*, 461–478.
10. Boukerche, A., Turgut, B., Aydin, N., Ahmad, M. Z., Bölöni, L., & Turgut, D. (2011). Routing protocols in ad hoc networks: A survey. *Computer Networks, 55*(13), 3032–3080.
11. Chen, Y. C., & Wen, C. Y. (2009). Adaptive cluster-based scheduling management for wireless ad-hoc sensor networks. In *Proceedings of the third international conference on sensor technologies and applications* (pp. 256–263).
12. Cheng, B. N., & Moore, S. (2012). A comparison of MANET routing protocols on airborne tactical networks. In *IEEE military communications conference (MILCOM)* (pp. 1–6). https://doi.org/10.1109/MILCOM.2012.6415798.
13. Clausen, T., & Jacquet, P. (2003). *RFC3626: Optimized link state routing protocol (OLSR)*. RFC Editor.
14. Cooper, C., Franklin, D., Ros, M., Safaei, F., & Abolhasan, M. (2017). A comparative survey of VANET clustering techniques. *IEEE Communications Surveys& Tutorials, 19*(1), 657–681.
15. Dana, A., Yadegari, A., Hajhosseini, M., & Mirfakhraie, T. (2008). A robust cross-layer design of clustering-based routing protocol for MANET. In *10th International conference on advanced communication technology* (vol. 2, pp. 1055–1059). https://doi.org/10.1109/ICACT.2008.4493948.
16. Danilov, C., Henderson, T. R., Goff, T., Brewer, O., Kim, J. H., Macker, J., & Adamson, B. (2012). Adaptive routing for tactical communications. In *IEEE military communications conference (MILCOM)* (pp. 1–7). https://doi.org/10.1109/MILCOM.2012.6415825.
17. Eichhorn, F., Corici, M. I., Magedanz, T., Du, P., Kiriha, Y., & Nakao, A. (2017). SDN enhancements for the sliced, deep programmable 5G core. In *Proceedings of the 13th international conference on network and service management (CNSM)* (pp. 1–4).
18. Ejaz, S., Iqbal, Z., Azmat Shah, P., Bukhari, B. H., Ali, A., & Aadil, F. (2019). Traffic load balancing using software defined networking (SDN) controller as virtualized network function. *IEEE Access, 7*, 46646–46658. https://doi.org/10.1109/ACCESS.2019.2909356
19. Ergenç, D., & Onur, E. (2018). *Cross-layer stack design for OMNeT++*. https://wins.ceng.metu.edu.tr:8085/gitlab/doganalp1/CrossLayerFramework.
20. Ergenç, D., Eksert, L., & Onur, E. (2018). Density-aware probabilistic clustering in ad hoc networks. In *2018 IEEE international black sea conference on communications and networking (BlackSeaCom)* (pp. 1–5). https://doi.org/10.1109/BlackSeaCom.2018.8433605.
21. Ergenç, D., Eksert, L., & Onur, E. (2019). Dependability-based clustering in mobile ad-hoc networks. *Ad Hoc Networks, 93*, 101926.
22. Espinel Sarmiento, D., Lebre, A., Nussbaum, L., & Chari, A. (2021). Decentralized SDN control plane for a distributed cloud-edge infrastructure: A survey. *IEEE Communications Surveys Tutorials, 23*(1), 256–281. https://doi.org/10.1109/COMST.2021.3050297
23. Gelil, W. A., & Kunz, T. (2019). A hierarchical P2P overlay for hierarchical mobile ad hoc networks (MANETs). In *IEEE 10th annual ubiquitous computing, electronics mobile communication conference (UEMCON)* (pp. 0640–0646). https://doi.org/10.1109/UEMCON47517.2019.8993022.
24. Gilani, S. S. A., Qayyum, A., Rais, R. N. B., & Bano, M. (2020). SDNMesh: An SDN based routing architecture for wireless mesh networks. *IEEE Access, 8*, 136769–136781. https://doi.org/10.1109/ACCESS.2020.3011651
25. Guest, T. (2018). Tactical communications—Increasing use of mobile ad hoc networks. *European Security and Defense*, *2*, 74–76.
26. Hertiana, S. N., Hendrawan, & Kurniawan, A. (2016). Performance analysis of flow-based routing in software-defined networking. In *22nd Asia-Pacific conference on communications (APCC)* (pp. 579–585). https://doi.org/10.1109/APCC.2016.7581501.
27. Hyytiä, E., & Virtamo, J. (2007). Random waypoint mobility model in cellular networks. *Wireless Networks, 13*(2), 177–188.
28. Ismail, Z., & Hassan, R. (2011). Effects of packet size on AODV routing protocol implementation in homogeneous and heterogeneous MANET. In *Proceedings of the third international conference on computational intelligence, modelling simulation* (pp. 351–356).
29. Jiang, M., Li, J., & Tay, Y. C. (1999). *Cluster based routing protocol (CBRP) functional specification*. Internet-Draft draft-ietf-manet-cbrp-spec-01, Internet Engineering Task Force. https://datatracker.ietf.org/doc/html/draft-ietf-manet-cbrp-spec-01. Work in Progress.
30. Joa-Ng, M., & Lu, I. T. (1999). A peer-to-peer zone-based two-level link state routing for mobile ad hoc networks. *IEEE Journal on Selected Areas in Communications, 17*(8), 1415–1425.
31. Kamble, S., & Dhope, T. (2016). Reliable routing data aggregation using efficient clustering in WSN. In *Proceedings of the international conference on advanced communication control and computing technologies (ICACCCT)* (pp. 246–250).

32. Kaur, H., & Singh, J. (2017). Analysis of hybrid routing protocols ZRP, HCR and ANTHOCNET: A review. *International Journal of Advanced Research in Computer Science and Software Engineering, 7*, 642–647.
33. Kaur, K., Kaur, S., & Gupta, V. (2016). Software defined networking based routing firewall. In *International conference on computational techniques in information and communication technologies (ICCTICT)* (pp. 267–269). https://doi.org/10.1109/ICCTICT.2016.7514590.
34. Kim, B. S., Kim, K. I., Roh, B., & Choi, H. (2018). Hierarchical routing for unmanned aerial vehicle relayed tactical ad hoc networks. In *IEEE 15th international conference on mobile ad hoc and sensor systems (MASS)* (pp. 153–154). https://doi.org/10.1109/MASS.2018.00034.
35. Kreutz, D., Ramos, F. M. V., Veríssimo, P. E., Rothenberg, C. E., Azodolmolky, S., & Uhlig, S. (2015). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE, 103*(1), 14–76. https://doi.org/10.1109/JPROC.2014.2371999
36. Labraoui, M., Boc, M. M., & Fladenmuller, A. (2016). Software defined networking-assisted routing in wireless mesh networks. In *International wireless communications and mobile computing conference (IWCMC)* (pp. 377–382). https://doi.org/10.1109/IWCMC.2016.7577087.
37. Lin, C. R., & Gerla, M. (1997). Adaptive clustering for mobile wireless networks. *IEEE Journal on Selected Areas in Communications, 15*(7), 1265–1275.
38. Microchip: Microchip RN1810 (2018).
39. Ming, L., Zhao, G., Xie, G., & Kuang, X. (2007). HOLSR: A novel routing scheme of ad hoc wireless networks for pervasive computing. In *Proceedings of the 2nd international conference on pervasive computing and applications* (pp. 661–666).
40. Mohamed, A., Onireti, O., Imran, M. A., Imran, A., & Tafazolli, R. (2016). Control-data separation architecture for cellular radio access networks: A survey and outlook. *IEEE Communications Surveys Tutorials, 18*(1), 446–465.
41. Nguyen, V. G., Do, T. X., & Kim, Y. (2016). SDN and virtualization-based LTE mobile network architectures: A comprehensive survey. *Wireless Personal Communications, 86*(3), 1401–1438. https://doi.org/10.1007/s11277-015-2997-7
42. Niu, X., Tao, Z., Wu, G., Huang, C., & Cui, L. (2006). Hybrid cluster routing: An efficient routing protocol for mobile ad hoc networks. In *Proceedings of the IEEE international conference on communications* (vol. 8, pp. 3554–3559).
43. Parvathi, P. (2012). Comparative analysis of CBRP, AODV, DSDV routing protocols in mobile Ad-hoc networks. In *International conference on computing, communication and applications* (pp. 1–4).
44. Pei, G., Gerla, M., & Chen, T. W. (2000). Fisheye state routing: A routing scheme for ad hoc wireless networks. In *IEEE international conference on communications (ICC)* (vol. 1, pp. 70–74).
45. Pham, T. M., Nguyen, T. T., & Kim, D. S. (2017). Geographical awareness hybrid routing protocol in mobile ad hoc networks. *Wireless Networks, 23*(1), 1–13. https://doi.org/10.1007/s11276-015-1119-5
46. Poularakis, K., Iosifidis, G., & Tassiulas, L. (2018). SDN-enabled tactical ad hoc networks: Extending programmable control to the edge. *IEEE Communications Magazine, 56*(7), 132–138. https://doi.org/10.1109/MCOM.2018.1700387
47. Poularakis, K., Qin, Q., Nahum, E. M., Rio, M., & Tassiulas, L. (2019). Flexible SDN control in tactical ad hoc networks. *Ad Hoc Networks, 85*, 71–80. https://doi.org/10.1016/j.adhoc.2018.10.012
48. Regragui, Y., & Moussa, N. (2018). Dynamics of network connectivity in tactical MANETs. In *IEEE international conference on advanced communication technologies and networking (CommNet)*. https://doi.org/10.1109/COMMNET.2018.8360278.
49. Riasudheen, H., Selvamani, K., Mukherjee, S., & Divyasree, I. (2020). An efficient energy-aware routing scheme for cloud-assisted MANETs in 5G. *Ad Hoc Networks, 97*, 102021. https://doi.org/10.1016/j.adhoc.2019.102021
50. Rossi, G. V., Fan, Z., Chin, W. H., & Leung, K. K. (2017). Stable clustering for ad-hoc vehicle networking. In *Proceedings of the IEEE WCNC* (pp. 1–6).
51. Shirmarz, A., & Ghaffari, A. (2020). Performance issues and solutions in SDN-based data center: A survey. *The Journal of Supercomputing, 76*, 1–49.
52. SparkLAN: SparkLAN WSDB-102GN (2018).
53. Sreevatsan, A. P., & Thomas, D. (2016). An optimal weighted cluster based routing protocol for MANET. In *Proceedings of the international conference on data mining and advanced computing (SAPIENCE)* (pp. 310–316).
54. Sun, Z., Du, P., Nakao, A., Zhong, L., & Onishi, R. (2019). Building dynamic mapping with CUPS for next generation automotive edge computing. In *IEEE 8th international conference on cloud networking (CloudNet)* (pp. 1–6). https://doi.org/10.1109/CloudNet47604.2019.9064135.
55. Tomovic, S., Lekic, N., Radusinovic, I., & Gardasevic, G. (2016). A new approach to dynamic routing in SDN networks. In *18th Mediterranean electrotechnical conference (MELECON)* (pp. 1–6). https://doi.org/10.1109/MELCON.2016.7495433.
56. Tomovic, S., Pejanovic-Djurisic, M., & Radusinovic, I. (2014). SDN based mobile networks: Concepts and benefits. *Wireless Personal Communications*. https://doi.org/10.1007/s11277-014-1909-6
57. Wang, F., Chen, Z., Zhang, J., Zhou, C., & Yue, W. (2019). Greedy forwarding and limited flooding based routing protocol for UAV flying ad-hoc networks. In *IEEE 9th international conference on electronics information and emergency communication (ICEIEC)* (pp. 1–4). https://doi.org/10.1109/ICEIEC.2019.8784505.
58. Wang, J. N., Van Hook, J., & Deutsch, P. (2015). Inter-domain routing for military mobile networks. In *IEEE military communications conference (MILCOM)* (pp. 407–412). https://doi.org/10.1109/MILCOM.2015.7357477.
59. Wang, Y., Dong, L., Liang, T., Yang, X., & Zhang, D. (2009). Cluster based location-aided routing protocol for large scale mobile ad hoc networks. *IEICE Transactions on Information and Systems, E92.D*(5), 1103–1124.
60. Yang, B., Yang, X., Ge, X., & Li, Q. (2018). Coverage and handover analysis of ultra-dense millimeter-wave networks with control and user plane separation architecture. *IEEE Access, 6*, 54739–54750. https://doi.org/10.1109/ACCESS.2018.2871363
61. Yang, Y., & Zhang, G. (2017). Clustering routing protocol based on segmentation for vehicular ad hoc networks. In *Proceedings of the 3rd IEEE international conference on computer and communications (ICCC)* (pp. 188–192).
62. Yassein, M. B., & Hijazi, N. (2010). Improvement on cluster based routing protocol by using vice cluster head. In *Fourth international conference on next generation mobile applications, services and technologies* (pp. 137–141).
63. Yu, J. Y., Chong, P. H. J., & Zhang, M. (2008). Performance of efficient CBRP in mobile ad hoc networks (MANETS). In *IEEE 68th vehicular technology conference* (pp. 1–7).
64. Zhang, H., & Yan, J. (2015). Performance of SDN routing in comparison with legacy routing protocols. In *2015 International conference on cyber-enabled distributed computing and knowledge discovery* (pp. 491–494). https://doi.org/10.1109/CyberC.2015.30.
65. Zhou, B., Cao, Z., & Gerla, M. (2009). Cluster-based inter-domain routing (CIDR) protocol for MANETs. In *Proceedings of the sixth international conference on WONS* (pp. 19–26). IEEE.

66. Zhu, J., & Du, Q. (2016). Group adaptive hybrid routing algorithm based on group mobility in Tactical MANET. In *IEEE information technology, networking, electronic and automation control conference* (pp. 6–10). https://doi.org/10.1109/ITNEC.2016.7560308.

Doğanalp Ergenç is a Ph.D. candidate in the IT-Security and Security Management group at the Universitaet Hamburg since 2019. He has received his M.Sc. degree in computer engineering from Middle East Technical University, Turkey in 2018. He has worked as system and software engineer in different companies and involved various research projects including the areas wireless networking, programmable networks, and network security. He is currently working on the design of resilient mission-critical networks and time-sensitive communication.

Ertan Onur is a professor of computer engineering at Middle East Technical University, Ankara Turkey, and the founding director of the Wireless Systems, Networks and Cybersecurity Laboratory (WINS Lab). He received the B.Sc. degree in computer engineering from Ege University, Izmir, Turkey in 1997, and the M.Sc. and Ph.D. degrees in computer engineering from Bogazici University, Istanbul, Turkey in 2001 and 2007, respectively. During the M.Sc. and Ph.D. studies, he worked as a project leader at Global Bilgi, Istanbul, and as an R&D project manager at Argela Technologies, Istanbul. After obtaining his Ph.D. degree, he worked as an assistant professor at Delft University of Technology, Netherlands. From 2014 on, he is with Middle East Technical University, Turkey. Dr. Onur's research interests are in the area of computer networks, wireless networks, and network security. He was a visiting professor at Stony Brook University in 2020 on a Fulbright award. He is a member of IEEE.