Computer Networks 185 (2021) 107659

Contents lists available at ScienceDirect

Computer Networks

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

Intra- and inter-cluster link scheduling in CUPS-based ad hoc networks

M. Levent Eksert [a,*], Hamdullah Yücel [b], Ertan Onur [a,c]

[a] Department of Computer Engineering, METU, 06800 Ankara, Turkey
[b] Institute of Applied Mathematics, METU, 06800 Ankara, Turkey
[c] Department of Computer Science, Stony Brook University, Stony Brook, NY 11794, United States of America

ARTICLE INFO

Keywords:
Clustered ad hoc networks
Intra- and inter-cluster link scheduling
Control-user plane separation

ABSTRACT

While control and user (data) plane separation (CUPS) through clustering improves the scalability of ad hoc networks in comparison to flat topologies, it introduces additional challenges for resource scheduling when contention-free medium access is employed. This paper addresses intra- and inter-cluster link scheduling problem in multi-channel ad hoc networks employing CUPS architecture. We first describe the CUPS architecture. Then, we present a novel intra- and inter-cluster link scheduling mechanism over the CUPS architecture. We propose a link scheduling strategy that is responsive to instant communication demands and available resources as a non-linear optimization problem, which is then reduced to a linear form by eliminating non-linearities in order to simplify the problem definition and enhance optimizer performance. The results of the optimizer show that the run time cost of the optimization function drastically increases by the parameter size growth. Therefore, we propose an iterative solution to decrease the running time. The adaptation of the iterative history-based approach makes the solution feasible and ensures near-optimal satisfaction and efficiency.

1. Introduction

Wireless ad hoc networks are basically a group of self-organized communicating entities independent from static infrastructure assistance. Therefore, contrary to an infrastructure-based networks with base stations or access points, ad hoc networks can be deployed to any location in which static terminals may not be available. However, management tasks such as resource sharing become more challenging in the absence of centralized controllers.

In contrast to fully-centralized structures that possess centralized managers and flat structure that has no hierarchical entity, clustered structure contains independent local coordinators in charge of control tasks such as resource scheduling and routing [1]. Control and user (data) plane separation (CUPS) in wireless ad hoc networks through clustering introduces a hierarchical model that shows a distributed behavior on a global scale and mimics centralized behavior in a local scale. In CUPS-based networks, control information may flow over a different sub-network than the data plane packets. Furthermore, nodes may have different roles in control and data planes. While the control plane functions such as routing or link scheduling can be run by some designated nodes (we will call them as cluster heads later on), data plane functions such as forwarding the packet on an allocated channel at a specific slot can be accomplished by a different set of nodes. In this paper, we concentrate on the link scheduling task in CUPS-based ad hoc networks.

Since random access protocols are prone to collisions that hamper channel efficiency in wireless ad hoc networks, contention-free channel access protocols, for instance time-division multiple access (TDMA), are generally preferred when quality of service (QoS) is desired. In TDMA, each pair of transmitting and receiving nodes in the network has to follow a (predetermined) link schedule to access the medium for reliable and concurrent data transfer. If the medium consists of more than one channel, then this resource allocation problem is called multi-channel link scheduling that is NP-Hard [2].

Some of the significant application domains that require a high level of QoS are emergency communication systems, public safety networks, or military systems where data transmission may be critical imposing some stringent QoS requirements. For instance, in military ad hoc networking applications, a bound on delay for voice communication as well as a minimum throughput for video streaming may be needed in multi-hop communication among military units. TDMA link scheduling can then be used for handling sound and video streaming tasks for guaranteeing a certain level of QoS. However, due to the complexity of multi-channel link scheduling, performance degradation is possible if TDMA is directly applied on a highly-populated network. In this work, we employ a divide-and-conquer approach to reduce the search space by clustering nodes into small groups, each of which are implicitly scheduled for intra-cluster communication. Inter-cluster communication, which has the problem size of two groups combined, is maintained

* Corresponding author.
E-mail addresses: levent.eksert@metu.edu.tr (M.L. Eksert), yucelh@metu.edu.tr (H. Yücel), ertan.onur@stonybrook.edu (E. Onur).

https://doi.org/10.1016/j.comnet.2020.107659
Received 29 January 2020; Received in revised form 12 October 2020; Accepted 27 October 2020
Available online 30 October 2020

by the proposed inter-cluster TDMA link scheduling mechanism for QoS support, as well. The idea of partitioning the network elements is motivated by the civilian and military human distributions in rural areas. For instance, the standard size of a mountaineers' party or a military squad is around 10 people, which can be grouped as a cluster in an ad hoc network structure [3,4].

In Fig. 1, a simple CUPS-based ad hoc network is shown. In this example, we assume nine nodes are grouped into two clusters. Left cluster has nodes 1, 2, 3, 4, and 5. Right cluster has nodes 4, 6, 7, 8, and 9. Node 4 acts as the gateway between two clusters. Nodes 2 and 7 are the cluster heads of left and right clusters, respectively. Let us consider that left and right clusters are assigned to a different set of channels by spatial multiplexing. Assume node 1 wants to send 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., does routing) and then forwards the packet in the data plane over gateways and cluster heads (e.g., over 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 packet) to its cluster head. While nodes have different roles (cluster head, gateway or ordinary node) in the control plane, all nodes are equal in the data plane. The cluster head (node 2 in this example) runs the routing function and determines the route to the destination, i.e., path 1-3-6-9. Concurrent to the routing task, the cluster heads together with the gateways run the link scheduling function to assign channel–slot pairs (resource blocks) to hops in the data plane. Then, node 1 transmits the packet to node 3 in the allocated time slot over the allocated channel. Node 3 transmits the packet to node 6 which is in another cluster in the allocated resource block. Finally, node 6 delivers the packet to node 9 in the right cluster.

As network division through clustering has the advantage of reducing the problem size, CUPS-based networks introduce inter-cluster coordination issues into the TDMA scheduling problem. For instance, peer contact between the cluster heads as well as mutual agreement on occupied resources for both communicating clusters are required. In this study, we present a solution to this novel coordination problem by the proposed inter-cluster link scheduling mechanism implemented through gateways.

There are many challenges in CUPS-based ad hoc networks such as clustering and assignment of roles to nodes [5], routing [6], or neighbor discovery across clusters. In this paper, we concentrate on link scheduling in CUPS based ad hoc networks. We propose a scheme where intra-cluster link scheduling such as node 1 to node 3 link in Fig. 1 and inter-cluster link scheduling such as node 3 to node 6 link in the same example are considered jointly where neighboring clusters employ different radio channels to minimize interference by spatial multiplexing. The problem we define in this paper differs from the related work in three major perspectives. Firstly, majority of the approaches in the literature concentrate on a single collision domain. In this paper, we focus on CUPS-based clustered ad hoc networks that employ spatial multiplexing among clusters. Consequently, clusters generate distinct collision domains. To the best of our knowledge, we are the first to define inter-cluster link scheduling problem in such a scheme. Secondly, a comprehensive multi-channel time-division link scheduling method for CUPS-based networks responsive to link requirements is yet to be proposed for end-to-end forwarding of packets considering intra- and inter-cluster link scheduling jointly. Thirdly, we consider not only unicast packets but also multicast and broadcast packets which makes the multichannel link scheduling problem complicated.

The main idea behind the proposal in this paper is to define resource allocation history as a constraint to the optimization problem that allows us to schedule the links in an iterative fashion. Assigning new links over the existing schedule, the scheduler is able to reduce its time cost in a high traffic network, to handle recent link requests with the presence of previously allocated resource set of the current schedule, and to arrange intra- and inter-cluster link scheduling. The study also proposes an inter-cluster link scheduling mechanism so that two neighboring nodes that belong to two distinct clusters can communicate over a link within a multi-channel TDMA scheme.

The main contributions of this work and the outline of the paper are as follow.

- We define a multi-channel link scheduling problem that extends to the intra- and inter-cluster communication in Section 2 that is briefly assigning link requests into available resource blocks conforming to requirements imposed by radio hardware features and time singularity of an active network element. The arrangement of the links is expected to prompt a node once at a time as with maximum utilization of the whole network elements. We assume nodes may require messages destined to multiple nodes. In other words, the problem definition covers multicast and broadcast communication requirements that are not extensively addressed in the literature.
- We introduce an efficient plan for connecting two heterogeneous structures in terms of communication: neighboring clusters, by identifying inter-cluster link scheduling steps within CUPS-based architecture in Section 2.3.2 where spatial multiplexing is employed.
- We present a two-phase scheduling optimization that is capable of performing control link allocation and repetitive link reservation: the history-based approach in Section 2.5. We consider not only unicast link demands but also multicast and broadcast link demands in order to schedule one-to-many node communication.
- Our scheduling method is extensively experimented by Monte Carlo simulations and the results are presented in Section 3.
- We evaluate basic link scheduler excluding history-based approach in Section 3.1 and propose an iterative history-based solution in Section 3.2 that is able to accelerate the link scheduling task by applying an optimization process into partitioned input data over a previously accumulated optimized scheduling. The results show that iteratively rescheduling the links over a predetermined history link set reduces the time cost of the scheduler as the number of assigned links almost remains unchanged.
- We elaborate on the contributions of this paper by making a thorough inspection of the related work in Section 4. Moreover, at the end of Section 4, we compare the attributes of the proposed method and the related work in Table 2 suggesting our study has the most comprehensive and interoperable method by means of channel cardinality, link type, and traffic flow.

2. Intra- and inter-cluster multi channel link scheduling

We formulate the Intra- and Inter-Cluster Multi-Channel Link Scheduling (ICLS) problem in this paper. We introduce the overall system model of the applied method with clustered network assumptions and CUPS-based system design details and assumptions imposed by clustered network link scheduling. Next, constraints imposed by clustered link scheduling and TDMA are presented. Eventually, we go into detail of ICLS problem definitions and run time analysis.

2.1. Network model

Ad hoc networks are defined as a decentralized wireless network composed of almost identical devices by means of communication skills and energy capacity. These homogeneous entities, namely nodes, are able to directly communicate each other without an intermediate gateway hardware or central access point and do not rely on a separate centralized controller. That means, the network is entirely self-configured and stand-alone. Nodes can move across the territory and routes of network nodes do not have to fit in a mobility pattern or model.

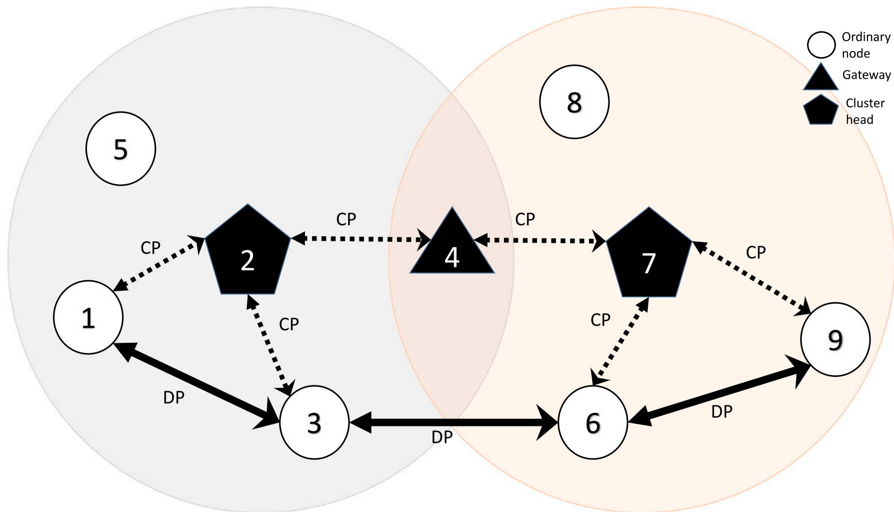


Fig. 1. An illustration of a CUPS-based ad hoc network. Nine nodes are grouped into two clusters. Left cluster has nodes 1, 2, 3, 4, and 5. Right cluster has nodes 4, 6, 7, 8, and 9. Node 4 (designated with a triangle) acts as the gateway between two clusters. Nodes 2 and 7 are the cluster heads (designated with pentagons) of left and right clusters. Ordinary nodes are designated with circles. Left and right clusters are assigned a different set of channels by spatial multiplexing.

In this work, we assume the network employs a clustered ad hoc structure. That is, nodes are grouped in proximity of some controller nodes, namely **cluster heads**, in terms of some criteria such as neighborhood, energy, and mobility. Disjoint node groups, each of which are managed by a distinct cluster head, are called **clusters** composed of a cluster head node and ordinary nodes, namely **cluster members**. Formally,

$$Cl_i = \left\{ v_1, v_2, \dots, v_{N_i} \right\},$$
$$V = \bigcup_{i=1}^{m} Cl_i,$$
$$Cl_i \cap Cl_j = \emptyset \qquad i \neq j,$$

where v_i denotes a node in Cl_i cluster that contains N_i number of nodes, and V is the set of all nodes in the network. The whole network denoted by V consists of disjoint clusters, Cl_is. In other words, each node in the network belongs to exactly one cluster Cl_i, $i = 1, 2, \dots, N_i$ in the data plane. Gateways are associated with multiple clusters in the control plane. On the other hand, some special cluster members in the vicinity of neighboring cluster heads are determined as **gateway** nodes in order to connect both cluster heads. Gateway nodes are involved in the control mechanism providing the inter-cluster communication from a cluster to a node that belongs its neighboring cluster as well.

2.2. The CUPS architecture

Ad hoc networks are essentially network organizations that accommodate spontaneously communicating nodes. Unlike some ad hoc network models [7,8], the proposed model in this paper does not possess some designated sink nodes to which the traffic of all or a part of nodes is directed. Hence, communication demands do not exhibit a pattern and the network handles the management of irregular demands autonomously.

Each cluster head is the local controller of its own cluster. Cluster heads and gateway nodes constitute a backbone that manages network traffic and resource sharing of the entire network. Cluster heads and gateways are the appointed roles in the **control plane** which is responsible for routing and scheduling tasks among others. To accomplish those tasks, control messages, which contain routing and scheduling related decisions, are transmitted over the control plane. In contrast, data messages can visit any node in the network. Therefore, **data plane**, which is the group of nodes transmitting or receiving data, refers to all the nodes in the network.

In Fig. 1, transmission of a data packet through CUPS architecture is illustrated. All the links from sender to receiver node are maintained by the corresponding cluster heads and gateway nodes in control plane. As a result, data packet can be transmitted through the shortest route without engaging busy controller nodes, i.e., cluster heads or gateways. Transferring data packet on the shortest path does not only decrease transmission time but also improves load balancing of the data traffic. Furthermore, the control plane can be extended beyond the coverage area of the ad hoc network by employing additional low cost radio in the control plane. The separation of planes enhances the programmability and the flexibility of the network significantly.

2.3. Communication in clustered structure

There are two main challenges for clustered networks from the resource scheduling perspective: intra- and inter-cluster scheduling. The former can be provided by simply scheduling all link demands with respect to the proposed constraints on the cluster head node as a whole, whereas the latter is more complicated since it requires mutual agreement between scheduling of two communicating clusters that may employ different channel resources by employing spatial multiplexing.

2.3.1. Intra-cluster scheduling

TDMA divides shared communication resource into the same length time durations named **slots**. Each slot is assigned to zero or one node to transmit a fixed-length message. In multi-channel TDMA structure, which is employed by the proposed interference model, more than one transmission can be assigned to a slot within separate channels. Each link, which is defined by a sender node and one or more destination nodes, can be reserved to a **resource block**, a channel–slot pair. Slots are divided into the same number of resource blocks and consecutive slots are grouped into the same-length **frames** as shown in Fig. 2.

The links can be defined as $v_s \rightarrow \mathcal{D}_s$, where $v_s \in \mathcal{V}$ is the source node, $\mathcal{V} = \{1, 2, \dots, N\}$, $\mathcal{D}_s \subset \mathcal{N}$, $v_s \notin \mathcal{D}_s$, and N is the total number of nodes in the cluster. Each link is assigned to a resource block (i, j), where i is the slot and j is the channel number, respectively. An example network traffic following the schedule depicted in Fig. 2 is illustrated in Fig. 3. In the example, there are 4 channels and 4 slots in a frame and $N = 5$ is the number of nodes in the cluster. Node 2 is the cluster head and the other nodes are the cluster members.

In Fig. 3, node 1 transmits to node 4 at slot 1 on channel 1, denoted as $1 \rightarrow \{4\}$. Similarly, node 2 sends to node 3 at slot 1 on channel 2 as $2 \rightarrow \{3\}$. At slot 3, occupied by link $5 \rightarrow \{2, 4\}$, node 5 performs

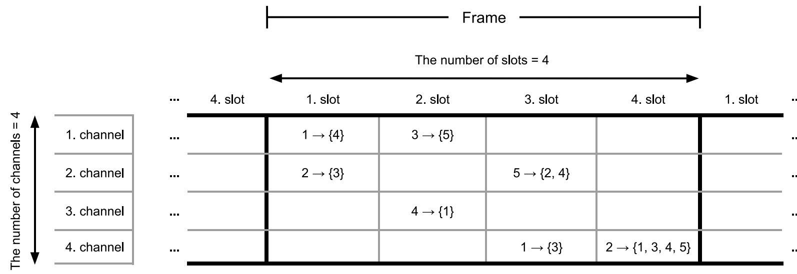
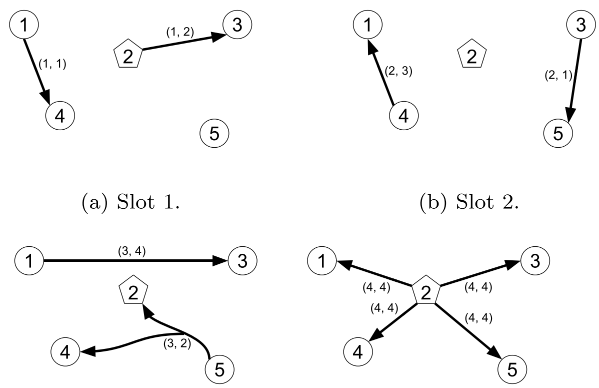


Fig. 2. A proper schedule example of a cluster.



(a) Slot 1. (b) Slot 2. (c) Slot 3. (d) Slot 4.

Fig. 3. A regular transmission scenario with a schedule given in Fig. 2.

multicasting by transmitting to node 2 and node 4 on the same channel (channel number 2) as node 2 broadcasts to all other nodes at slot 4 on channel 4 by link $2 \rightarrow \{1, 3, 4, 5\}$.

A cluster is a part of the network managed in a centralized manner. Similarly, the scheduling mechanism is applied for the whole elements of the cluster and determined by the cluster head.

Data links are created on-demand by collecting link requests of member and gateway nodes into cluster head and by assigning a proper link in the schedule of that cluster. The scheduler organizes links for repetitive and consecutive frames that consist of a fixed number of slots. Prior to the beginning of each frame, each node sends its link demands to its local cluster head. The schedule of that cluster for the following frame is formed and broadcast to every cluster node in a beacon message before the frame starts. Therefore, scheduling of the next frame has to be decided in an instant time period which is at least smaller than a frame duration. That means, run time performance of the scheduling process has a significant role in the overall mechanism. The same procedure is applied in each frame during the lifetime of the network.

The links may be scheduled for one or more frames depending on the communication requirement of the transmitting and receiving nodes. For instance, bursty packets require instant and minimal messaging where sound or video calls are composed of stream packets that need to reside in one or more particular slots during consecutive frames. Besides, the control messages which are vital for the sustainability of the network should be granted higher priority in the schedule for rapid response to control action. In this study, a set of repetitive link reservations containing data messages spreading over multiple frames and urgent control messages is defined as the **history** of the schedule. Before scheduling current link demands, the history link distribution is moved to the current schedule by reserving the links inherited from the previous frame on the exact resource blocks and is updated by adding recent history links. After that, the link demands collected for the current frame are scheduled on the reserved history links.

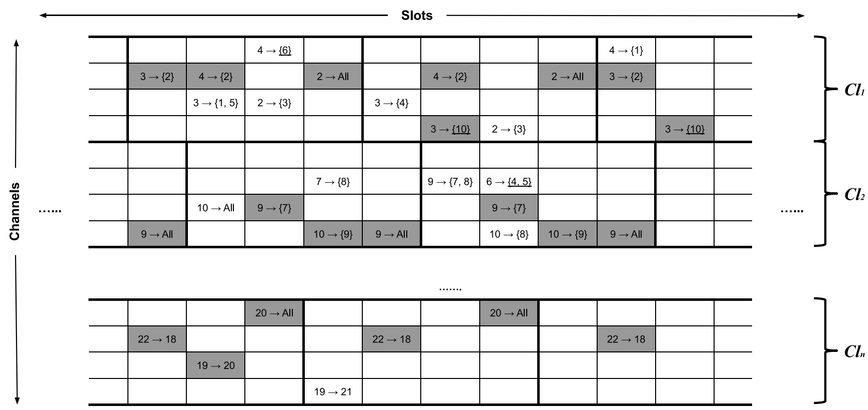
The steps followed by a cluster head for intra-cluster scheduling process is given as the following:

1. Collect all the link demands from cluster nodes over some static or dynamic uplinks.
2. Compose a history for the schedule so as to reserve prior blocks for fixed and primary links.
3. Run an optimization algorithm with the history input in order to provide as many links as possible for collected and internal link demands determined by route discovery.
4. Send the resulting schedule in the first slot of the corresponding frame as a beacon and repeat the steps for the next frame.

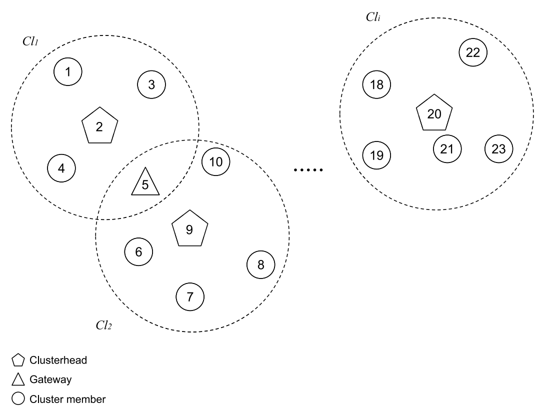
2.3.2. *Inter-cluster scheduling*

Since intra-cluster communication is scheduled in the cluster head nodes locally, simultaneous use of a common channel may occur within different clusters, resulting in interference in message transfer between two nodes in the same cluster. For reliable intra-cluster communication, it is essential that a cluster does not share the same channel set with the clusters in its communication range. Our technique employs a version of Spatial Division Multiple Access (SDMA) [9] adapted to ad hoc clustered network structure and relies on spatial multiplexing. More specifically, the channel spectrum of communication resources is divided into subsequent and disjoint channel ranges. Each cluster Cl_i utilizes a single range containing one or more channels as the intra-cluster resource. That means, each cluster communicates through a multi-channel TDMA scheme within a predefined channel spectrum range for internal messaging of cluster nodes. Dividing the whole channel resources into disjoint segments and distributing these resource blocks among the clusters reduce problem space. Compared to a fully centralized scheduler containing all the variables, operating simultaneous schedulers on each cluster head that manages its cluster nodes and utilizes its dedicated channel range is more practical since each distributed scheduler sustains less number of channels and nodes. On the other hand, in a large network with a significant amount of clusters, assigning each channel to a distinct cluster is not feasible due to the limited channel spectrum. Channels can be shared in intra-cluster communication of non-neighboring clusters, providing **spatial reuse** without causing interference with outer intra-cluster messages. Channel assignment can be managed by cluster heads with a distributed resource sharing methods such as [10] and [11].

An overall intra- and inter-cluster scheduling on multiple clusters Cl_i, $i = 1, 2, \ldots, m$, is illustrated in Fig. 4(a). The corresponding network topology of the schedules given in Fig. 4(a) is illustrated in Fig. 4(b) as well. **All** in Fig. 4(a) refers to all nodes in the enclosing cluster. Shaded blocks are the reserved links as history and the rest are the optimized links and vacant resource blocks. In this example, a history may contain schedule broadcast and uplinks to the cluster head for the transmission of link demands. Repetitive links for data transmission may exist in the histories as well. For instance, voice and video calls require permanent links during the call in order to establish a continuous data transmission



(a) Schedules using disjoint channel ranges simultaneously.



(b) Network topology of the schedules given in Figure 4(a).

Fig. 4. Inter-cluster link scheduling example.

and high quality of service. It should be noted that the frames of the clusters are not aligned on the slot axis. Frame alignment among clusters is not mandatory due to the self-organized property of intra-cluster scheduling although we assume time synchronization.

In Fig. 4(a), some reserved resource blocks have underlined destined nodes, which belong to neighboring clusters. For instance, $4 \rightarrow \underline{\{6\}}$ at the first frame of Cl_1 denotes to a link that transmits from node 4 to node 6, which is a cluster member of Cl_2. Links with underlined destined nodes are inter-cluster links in the CUPS architecture, which are elaborated in the following sections.

The distributed management concept of the proposed clustered network leads to heterogeneity among neighboring cluster schedules. Consequently, constructing an inter-cluster link between two neighboring clusters is considered as a tedious task in terms of compatibility and time complexity. We propose an efficient and feasible mechanism that provides consensus for both sides of the communication. To achieve this, the assistance of gateway nodes is used in the scheduling process.

Intra-cluster communication is held by local scheduling on cluster head nodes in a dedicated collection of channels. However, any two nodes that belong to two separate clusters, and possibly use different channel ranges for intra-cluster communication, should also communicate with each other for the integrity of the network. An example case for an inter-cluster link between node 3 and 6 is shown in Fig. 1. To accomplish that, inter-cluster links should be maintained between nodes that are the members of two neighboring clusters. Since two neighboring clusters transmit packets through different channel sets, a common channel should be provided. Control plane entities determine inter-cluster links with respect to routing requirements and inform communicating data plane elements. In this paper, we assume that physical resources (channels) of the sender's cluster are employed for inter-cluster communication.

We follow a set of procedures that establish an inter-cluster link on control plane. First, the proposed inter-cluster scheduling is initiated by cluster head in the sender side with a message requesting an inter-cluster link resource from the gateway node communicating to the destination cluster. Then, gateway node determines the proper resource block for the inter-cluster link and sends its decision to both sender and destination cluster heads. Note that the gateway node is scheduled to listen to schedules of both clusters before the beginning of both cluster frames so that the inter-cluster resource block can be specified as appropriate blocks of each cluster schedule. Therefore, any conflicts between the previously reserved resource blocks of both schedules can be avoided. After, the inter-cluster resource block is received by both cluster heads, it is applied to the next schedules of clusters and the inter-cluster message is transmitted and received at the corresponding reserved resource blocks of both sender and receiver cluster schedules. The steps of the inter-cluster scheduling process are displayed in Fig. 5 as close-section of the inter-cluster link between node 3 and node 6 shown in Fig. 1. This process can be handled together with the network layer routing task in a cross-layer fashion to further enhance the performance.

2.4. Constraints

Organizing all the link demands in the frame introduces some rules, i.e., resolving some conflicts leading to interference in communicating nodes, as the scheduler should serve as many link requests as possible. The restrictions are basically to assign a link demand to at most one

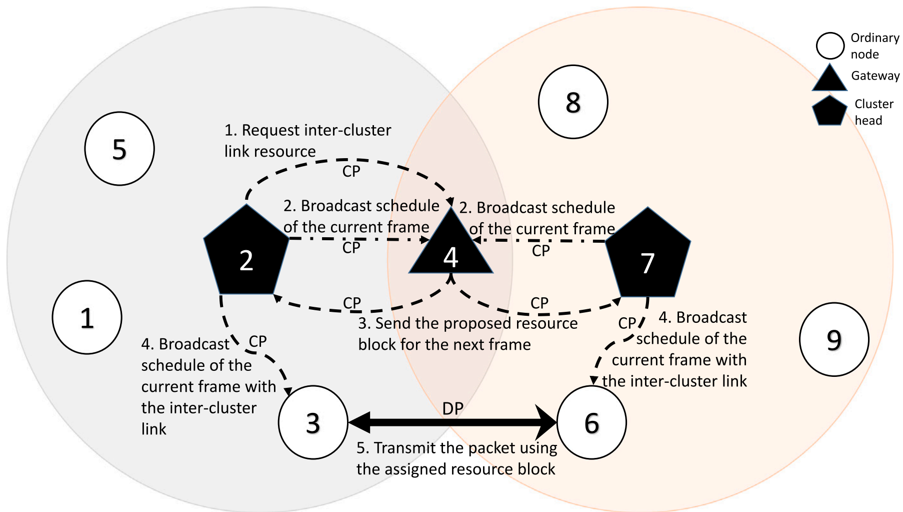


Fig. 5. The inter-cluster link scheduling mechanism. Node 3 is the sender and node 6 is the receiver node in a neighboring cluster.

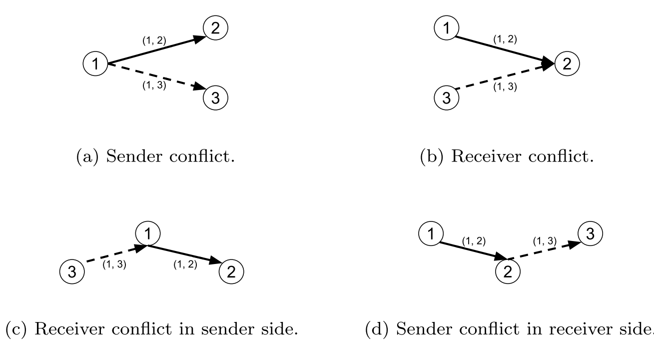


Fig. 6. Visualizations of sender-receiver conflict types.

slot-channel block and vice versa as well as to prevent a node from either transmit or receive only one message at a time.

In Fig. 6, a set of scheduling conflicts are illustrated. In all cases, node 1 attempts to send a message to node 2 in the resource block $(1, 2)$ and a conflicted transmission from node 1 to node 3 occurs at the same time in resource block $(1, 3)$. Four distinct conflicts are depicted separately in Fig. 6 in separate subfigures. In Fig. 6(a), $1 \rightarrow 2$ transmission is scheduled in the same slot and causes a conflict in the sender node since a node cannot transmit two or more messages simultaneously. This conflict type is called **sender conflict**.

In Fig. 6(b), node 1 and node 3 plan to send a message to node 2 at the same slot. Since a node cannot listen to two distinct channels at a time, at least one of the messages cannot be received by node 2. This conflict type is known as **receiver conflict**.

A node either sends or receives a message at a time due to the half-duplex characteristics of most radio hardware in devices. Therefore, if a sender node is scheduled to receive a message from a third node at the same slot, a receiver conflict occurs in the sender side as shown in Fig. 6(c). Similarly, scheduling a node to send a message which is already scheduled to receive from another node (similar to node 2 in Fig. 6(d)) ends up with conflict in that node.

The conflicts for inter-cluster communication and the constraints for inter-cluster link scheduling are not thoroughly addressed in the literature; we fulfill this gap in this paper. An inter-cluster link should ensure the compatibility of two intra-cluster schedules of both sender and receiver clusters. That means, the transmission link should not interfere with an intra- or inter-cluster link and both sender and receiver nodes should not be assigned to another message transmission at that time. Inter-cluster link should conform to the following constraints:

- The reserved resource block should not be used by another intra-cluster link.
- Inter-cluster link should not cause sender and receiver conflicts in sender and receiver intra-cluster schedule, respectively.
- Receiver conflict in sender side and sender conflict in receiver side should also be avoided in sender and receiver intra-cluster schedules, respectively.
- All inter-cluster links of a cluster should satisfy the requirements imposed for intra-cluster scheduling.

In Fig. 4(a), $4 \rightarrow \underline{\{6\}}$ is a valid inter-cluster link instance occupying one resource block in the first frame of Cl_1. Neither node 4 nor node 6 has an active role in intra-clustering schedules of their own clusters at the slot that intra-cluster link resides. Besides, there are no extra inter-cluster links conflicting with $4 \rightarrow \underline{\{6\}}$ link.

It can be derived from the listed constraints that applying intra-cluster constraints to the combination of sender and receiver channel resources is sufficient for generating inter-cluster link scheduling. Therefore, our approach is to create the inter-cluster link in the gateway node with respect to constraints generated from both histories of intra-cluster schedules. After the inter-cluster link is generated, it is added to the histories of both intra-cluster schedules in the next frame. The inter-cluster link appears in the intra-cluster schedules of the next frames of sender and receiver clusters, thereby being served in the schedules of both the sender and receiver sides.

2.5. Intra- and Inter-Cluster Multi-Channel Link Scheduling (ICLS) problem

The scheduling scheme is aimed to organize intra- and inter-cluster links with maximum utility and to resolve the constraints concurrently. Restrictions originated from any node affect all other cluster nodes due to densely distributed nodes within the cluster. Therefore, proportional to the amount of node population and packet traffic, the scheduler may be overloaded by a large list of rules, resulting in complex computations and long run time durations. The designed schedule is adaptive and responsive to the communication needs of nodes as well so that any size of instant or repetitive packet demand of cluster nodes can be satisfied by means of available resources.

The mathematical formulation of the problem extends to multicast and broadcast links, takes into consideration previous scheduling which contains permanently reserved blocks, and introduces extra restrictions for the current scheduling process.

Table 1 shows the parameters used in the problem definition. The matrices are given with their sizes in parentheses. **D** is the demand matrix, which denotes the message demand of each node given with

Table 1
Nomenclature.

Parameter/Matrix	Definition
T	Total number of time slots in a frame
C	Total number of channels
N	Total number of nodes
M	Maximum number of link request of a node in a frame
$\mathbf{D}$	Demand matrix $(N \times M \times N)$
$\mathbf{S}$	Intermediate resulting matrix $(N \times N \times M \times T \times C)$
$\mathbf{H}$	History matrix $(N \times N \times T \times C)$
$\mathbf{X}$	Resulting matrix $(N \times M \times T \times C)$

the destination node(s). $\mathbf{H}$ matrix shows the preserved resource blocks from the previous frame, whereas $\mathbf{X}$ matrix simply shows the result of assignments, i.e., whether the transmission is assigned to a resource block or not. $\mathbf{S}$ is an intermediate matrix used for resolving sender and receiver conflicts. It is basically a detailed version of $\mathbf{X}$ matrix in which destination nodes of message demand are expanded on an additional dimension. $\mathbf{D}$ and $\mathbf{H}$ are the input matrices when $\mathbf{X}$ is defined as the decision variable and $\mathbf{S}$ is categorized as an output matrix.

$$\mathbf{D}_{imj} = \begin{cases} 1, & \text{if node } i \text{ is allocated to a reasonable block} \\ & \text{for its } m\text{th demand to transmit node } j, \\ 0, & \text{otherwise.} \end{cases}$$

$$\mathbf{S}_{ijmtc} = \begin{cases} 1, & \text{if node } i \text{ is allocated to } (t,c)\text{th reasonable block} \\ & \text{to transmit node } j \text{ for its } m\text{th demand,} \\ 0, & \text{otherwise.} \end{cases}$$

$$\mathbf{H}_{ijtc} = \begin{cases} 1, & \text{if node } i \text{ is allocated to } (t,c)\text{th reasonable block} \\ & \text{to transmit node } j, \\ 0, & \text{otherwise.} \end{cases}$$

$$\mathbf{X}_{imtc} = \begin{cases} 1, & \text{if node } i \text{ is allocated to } (t,c)\text{th reasonable block} \\ & \text{for its } m\text{th demand,} \\ 0, & \text{otherwise.} \end{cases}$$

Here, i and j are used as subscripts in matrices that range from 1 to N and represent the source and destination node indices, respectively. Similarly, $t = 1, 2 \dots, T$, $c = 1, 2 \dots, C$, and $m = 1, 2 \dots, M$ refer to slot, channel, and link demand indices, respectively. Using the parameters and the matrices in Table 1, we define the following non-linear ICLS problem:

$$\text{maximize} \qquad U_{NL} = \sum_{i=1}^{N} \sum_{m=1}^{M} \sum_{t=1}^{T} \sum_{c=1}^{C} \mathbf{X}_{imtc} \qquad (1)$$

subject to

$$\sum_{i=1}^{N} \sum_{m=1}^{M} \mathbf{X}_{imtc} \leq 1, \qquad (2a)$$

$$\sum_{t=1}^{T} \sum_{c=1}^{C} \mathbf{X}_{imtc} \leq 1, \qquad (2b)$$

$$\sum_{j=1}^{N} \mathbf{D}_{imj} - \sum_{t=1}^{T} \sum_{c=1}^{C} \mathbf{X}_{imtc} \geq 0, \qquad (2c)$$

$$\sum_{t=1}^{T} \sum_{c=1}^{C} \left(\mathbf{X}_{imtc} \mathbf{D}_{imj} - \mathbf{S}_{imjtc} \right) = 0, \qquad (2d)$$

$$\left(\sum_{i=1}^{N} \sum_{m=1}^{M} \mathbf{X}_{imtc} \right) \left(\sum_{i=1}^{N} \sum_{j=1}^{N} \mathbf{H}_{ijtc} \right) = 0, \qquad (2e)$$

$$\left(\sum_{c=1}^{C} \mathbf{X}_{im_1tc} \right) \left(\sum_{j=1}^{N} \sum_{c=1}^{C} \left(\mathbf{S}_{im_1jtc} - \left(\mathbf{H}_{ijtc} \sum_{m_2=1}^{M} \mathbf{S}_{im_2jtc} \right) \right) \right) = 0, \qquad (2f)$$

$$\sum_{c=1}^{C} \left(\mathbf{X}_{i_1m_1tc} + \sum_{i_2=1}^{N} \left(\mathbf{H}_{i_2i_1tc} + \sum_{m_2=1}^{M} \mathbf{S}_{i_2ni_1pr} \right) \right) \leq 1, \qquad (2g)$$

$$\sum_{c=1}^{C} \left(\mathbf{S}_{im_1j_1tc} + \sum_{j_2=1}^{N} \left(\mathbf{H}_{j_1j_2tc} + \sum_{m_2=1}^{M} \mathbf{S}_{j_1m_2j_2tc} \right) \right) \leq 1, \qquad (2h)$$

$$\sum_{i=1}^{N} \sum_{c=1}^{C} \left(\mathbf{H}_{ijtc} + \sum_{m=1}^{M} \mathbf{S}_{imjtc} \right) \leq 1. \qquad (2i)$$

The objective of the formula is to maximize the utilization of available resource blocks U_{NL} as defined in (1). The constraint (2a) specifies that a resource block should be occupied by at most one message demand. Similarly, (2b) states that a message demand should be served in at least one resource block. (2c) ensures that mth demand of node i should exist in the $\mathbf{D}$ matrix. (2d) states that if a message demand in $\mathbf{D}$ is served (the corresponding entry in $\mathbf{X}$ is set to 1) then it is also presented in $\mathbf{S}$ by setting the related indices to 1. (2e) denotes that a message demand should not be placed to a resource block which a history message belongs to. (2f) provides that a message demand does not have a common sender node with another current demand and history at the same slot if the message is served in a resource block. In other words, it tries to resolve sender conflict depicted in Fig. 6(a). Similarly, (2i), which is also a definition of receiver conflict in Fig. 6(b), constrains that none of the target nodes of a served message demand receives a message from another node at the same slot. (2g) ensures that a history message or a current demand does not transmit to the node which sends a message to another node as a served message demand at the same slot (resolving receiver conflict in sender side shown in Fig. 6(c)). Finally, (2h) ensures that a served message demand does not transmit to a node which is also transmitting as a history message or a current demand at the same slot (resolving sender conflict in receiver side shown in Fig. 6(d)).

The resolution of the non-linear optimization problem in (1)–(2) contains a big number of constraints with matrix dimensions up to 5. Besides, the nonlinear structure of some of the constraints, e.g., (2e) and (2f), avoid to offer a feasible solution. Therefore, the optimization problem should be reformulated by eliminating nonlinearities. Redesigning the problem structure, the conflicts are combined into more compact constraints and nonlinearities are eliminated. The linear and simplified formulation, linear ICSL, is given as follows:

$$\text{maximize} \qquad U_L = \sum_{i=1}^{N} \sum_{m=1}^{M} \sum_{t=1}^{T} \sum_{c=1}^{C} (N-i)(M-m)\mathbf{X}_{imtc} \qquad (3)$$

subject to

$$\begin{aligned} &\sum_{t=1}^{T} \sum_{c_1=1}^{C} \Bigg(\mathbf{X}_{i_1mtc_1} + 2 \sum_{j_1=1}^{N} \Bigg(\mathbf{X}_{i_1mtc_1} \Bigg(1 - \mathbf{D}_{i_1mj_1} \\ &\quad + \mathbf{D}_{i_1mj_1} \sum_{c_2=1}^{C} \sum_{i_2=1}^{N} \sum_{j_2=1}^{N} \left(\mathbf{H}_{i_1j_1tc_2} + \mathbf{H}_{i_2j_1tc_2} + \mathbf{H}_{j_1j_2tc_2} + \mathbf{H}_{i_2i_1tc_2} \right) \Bigg) \Bigg) \Bigg) \\ &\quad \times \leq 1, \end{aligned} \qquad (4a)$$

$$\sum_{m_1=1}^{M} \sum_{c_1=1}^{C} \mathbf{X}_{i_1m_1tc_1} + \sum_{i_2=1}^{N} \sum_{m_2=1}^{M} \sum_{c_2=1}^{C} \mathbf{X}_{i_2m_2tc_2} \mathbf{D}_{i_2m_2i_1} \leq 1, \qquad (4b)$$

$$\sum_{i_1=1}^{N} \sum_{m=1}^{M} N\mathbf{X}_{i_1mtc} + \sum_{i_2=1}^{N} \sum_{j=1}^{N} \mathbf{H}_{i_2jtc} \leq N. \qquad (4c)$$

The objective U_L in (3)–(4) is to maximize the total number of resource blocks used similar to the previous formulation except with a multiplier extension for preventing race conditions among the identical resource blocks. To overcome that, the resource blocks with less time and channel indices are favored in order to lead the scheduler to reserve these highlighted indices in prior. (4a) states that each demand can be assigned to at most one (t,c) pair and ensures that zero entries in **D** matrix are not reserved to a resource block. It also resolves the sender and receiver conflicts related to history assignments. (4b) is related to the schedule assignment of the recent resource demands and simply imposes that a demand either sends or receives in a slot. In (4c), served demands are constrained to reside in exactly one resource block provided that it is not utilized by a history demand.

2.6. Problem complexity

In the given objective function (3), $(N\times M)+(N\times T)+(T\times C)$ number of constraints are introduced to optimize output matrix $\mathbf{X}$ attaining $N\times M\times T\times C$ number of 0-1 cells. Therefore, the objective function requires $(N\times M+N\times T+T\times C)\times 2^{N\times M\times T\times C}$ number of variable substitutions for the constraint check, denoting the run time complexity of the problem if the constraints are considered to be initially constructed.

Although the number of candidate solutions grows exponentially, eliminating some redundant variable substitutions is possible with the aid of some techniques such as branch and bound [12]. Besides, the values of the parameters are restricted by requirements of a couple of neighboring clusters at most. The problem definition is feasible with the ability of serving multicast and broadcast links by taking into account link demands and previous link reservations.

3. Results

The scheduling problem introduced in Section 2.5 is interpreted as an Integer Linear Optimization problem and solved in MATLAB Optimization Toolbox [13] on a computer with Intel® Xeon® Silver 2.20 GHz with 40 CPU and 62 GB RAM.[1] The results of the MATLAB implementation are shown and discussed in this section.

Unicast, multicast, and broadcast messages are randomly created in a link demand matrix following discrete uniform distribution. Various combinations of N, M, T, and C parameters are given as an input ranging from 4 to 10. Each value in the figures are obtained by taking the average of at most 100 simulation runs with a 95% confidence interval. The objective function parameters are modified so that resource blocks are prioritized inversely proportional to slot and channel numbers in order to improve *run time* metric of the optimizer as formulated in (3) and (4).

Two metrics are introduced to assess results: *Satisfaction* is the ratio of the served link demands to the total number of input link demands. *Efficiency* is a performance measure indicating the utility of resource blocks, which is the metric that is optimized by ICSL, and it is calculated by dividing the number of reserved link demands to the total number of resource blocks, $R = T\times C$.

Moreover, the normalized demand load l

$$l = \frac{\sum_{i=1}^{N}\sum_{m=1}^{M}\sum_{j=1}^{N}\mathbf{D}_{imj}}{N\times M\times(N-1)}, \tag{5}$$

represents packet density of a case where a source node can target at most $N-1$ number of nodes within a link demand as a broadcast message and each node can have at most M number of link demand in a frame. The normalized load, l is expressed as the total amount of destined nodes introduced by all link demands divided by the maximum number of target nodes that can be generated for that frame, which is $N\times M\times(N-1)$. $l = 0$ if none of the nodes requests a link request and $l = 1$ if all nodes demand M number of broadcast links in a frame. Note that l is simply a link demand parameter that corresponds to link demand density in a frame duration and does not refer to neither ratio of occupied slots in a frame nor data rate of nodes.

We propose two different approaches to solve the scheduling problem: *holistic approach* simply attempts to devise an instant solution to the given demand on entire resource blocks and *iterative approach*, which exploits our history based heuristics by gathering link demands into groups of equal size, G, and solves scheduling problem of each partitioned link demands on available resource blocks some of which have reserved for previous link demand groups, D_is.

We discuss the Iterative ICLS use about possible improvement of the algorithm performance in particular use cases. In order to evaluate Iterative ICLS under practical use, Iterative ICLS with single iteration is performed by introducing additional terms, i.e., d, a, and l_f. d is the number of new link requests collected in a frame duration and l_f is the normalized frame load l_f

$$l_f = \frac{a}{N\times T}, \tag{6}$$

representing the capacity of a frame, where a is the total number of active (sender or receiver) nodes in all slots. Note that a frame is fully utilized if N number of nodes is in either sender or receiver role in each slot due to the constraints mentioned in Section 2.4. Similar to l parameter, l_f is basically the total number of active nodes in each slot divided by its maximum value, $N\times T$. $l_f = 0$ if the frame is empty and $l = 1$ if the frame is totally reserved and unable to assign a new link request.

3.1. The holistic approach

In this part, the algorithm is tested where all the link demands are given as input at once altogether. Initially, the history matrix is considered as an empty matrix. In other words, there is no resource block occupied in the previous frames and all link demands are to be placed into empty resource blocks in one run.

Fig. 7 shows experimental results of holistic approach with a different number of nodes and packet densities. Two cases representing moderate and dense link demands are tested on fixed slot size $T = 16$ and 3 different channel sizes $C = 3, 4,$ and 5.[2] M is fixed to 6 for each holistic approach simulation and N has various values ranging from 4 to 10.

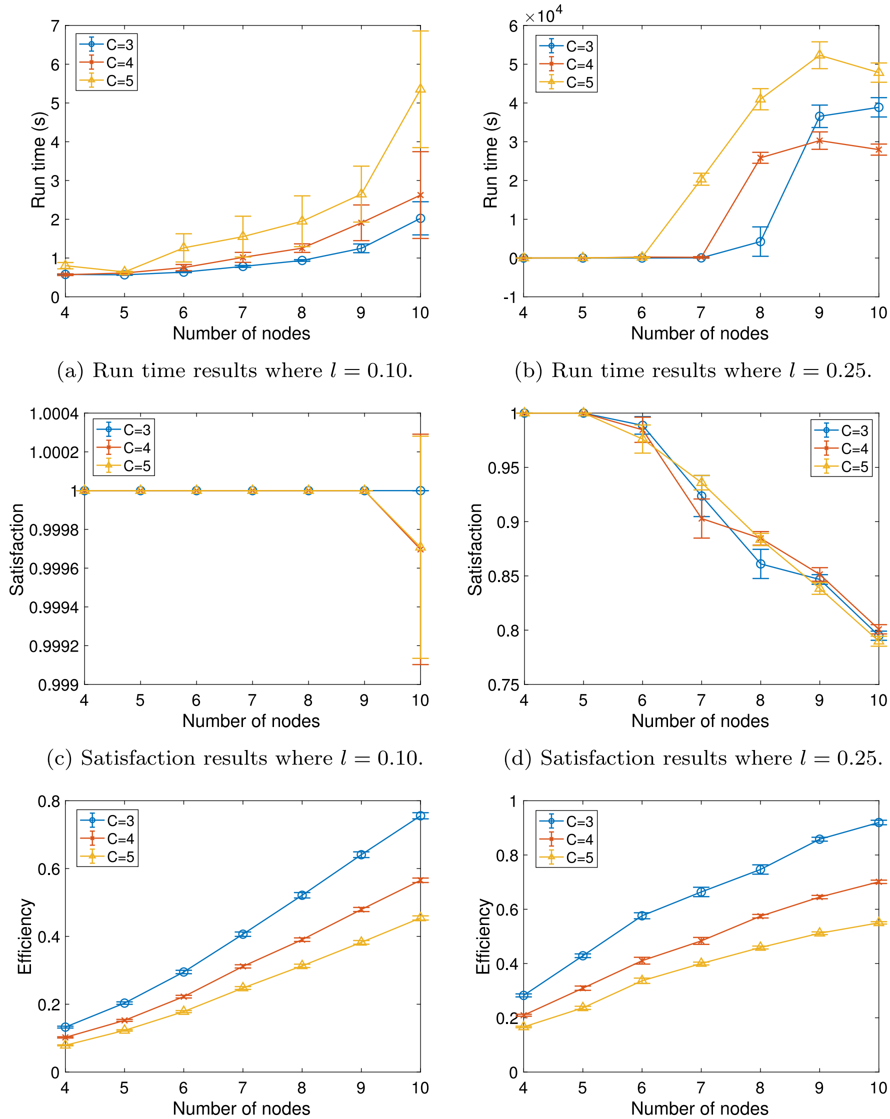
100 parallel simulation runs are performed for each normalized $l\times N$ case combination with different C values. At most 20 simulations are run simultaneously. The simulations which last more than twice of average run times are terminated and ignored in the calculation of performance results as outliers.

In Fig. 7(a) it can be observed that average run time of the ICSL optimizer increases linearly up to around 5 s as number of nodes increases to 10 when $l = 0.10$ and $C = 5$. Fig. 7(b) illustrates that in a dense link demand case where $l = 0.25$, as the number of nodes increases, average run time shows an exponential growth when the number of nodes is 9 and below. The run time of ICSL optimizer slightly increments or decrements when the number of nodes is 10, since after some point, available resource blocks cannot respond to increasing input link demand which eventually restricts the solution domain proportional to the number of nodes. However, optimizing link demands of 9 nodes when $l = 0.25$ and $C = 5$ takes about 25000 s, approximately 14 h, in average which is extremely infeasible for scheduling of a single frame. In Figs. 7(a) and 7(b), run time increases with C as the problem size grows with channel size except the demand traffic is dense and $N \geq 9$ case in which shortage of frame resources ($T\times C$ value) is prone to produce excessive number of intermediate solutions in ICSL optimizer, thereby expands solution tree and prolongs the run time.

Fig. 7(c) exhibits that almost all link demands are served by the ICSL scheduler in the moderate load case. In contrast, in Fig. 7(d) satisfaction

[1] At most half of the computing resources are utilized and each simulation is run concurrently.

[2] Each slot can acquire $N/2$ number of links at most due to the fact that a link occupied in a resource block eliminates at least two nodes from being active in the same slot. Therefore, T is chosen relatively greater than C in every simulation.



(a) Run time results where $l = 0.10$.
(b) Run time results where $l = 0.25$.
(c) Satisfaction results where $l = 0.10$.
(d) Satisfaction results where $l = 0.25$.
(e) Efficiency results where $l = 0.10$.
(f) Efficiency results where $l = 0.25$.

Fig. 7. Holistic approach performances with $N = (4, 5, \ldots, 10)$, $M = 6$, $T = 16$, and $C = 3, 4,$ and 5.

slightly decreases as the number of nodes increases to 10 for all C values.

The efficiency exhibits a linear growth as the number of nodes increases for each case as illustrated in Figs. 7(e) and 7(f). Average efficiency increases as the link demand gets denser and channel size shrinks, as well.

3.2. The iterative approach

The ICSL optimizer with holistic approach is not able to support the scenarios in which the input demand matrix and available resources are relatively larger. Therefore, heuristics are required for optimizing the cases where the problem size is relatively large. Extending history approach to the optimization process is considered to resolve this issue. More specifically, dividing the link demands into separate groups and optimizing these sets of link demands by keeping the previous optimization results as history at each step aid to overcome the parameter size limitation. In Fig. 8, the iterative approach steps are illustrated.

In Fig. 9, a similar experimental setup in Fig. 7 is constructed for iterative approach with the identical parameter set. Input link demands are partitioned into link demand groups with group sizes, $G = 1, 2, 5, 8,$ and 10. 100 separate link demands are generated for each $l \times N$ case combination and each link demand is ideally divided into the same-length link demand groups, D_is. Formally, $|D_i| = G$, where $i = 1, 2, \ldots, n-1$ and $|D_n| \le G$. At each iteration, D_is are optimized and assigned to available resource blocks and previous allocations are given to optimizer as a history input.

In Figs. 9(a) and 9(b), the run time performance of iterative approach is illustrated with 0.10 and 0.25 l values, respectively. In both Figs. 9(a) and 9(b), link demand groups with $G = 1$ has the worst run time performances since assigning link demands one by one takes a relatively higher amount of time compared to handling less number of link demand groups with greater sizes. In contrast, Fig. 9(b), the link demand group with $G = 10$ has the second highest run time as greater link demand groups combined with high l values lead to increase in the complexity of optimization problem. For every l case, run time increases with l value and cases in which $G = 5$ outperform the cases with the other link demand sizes.

Satisfaction results depicted in Fig. 9(c) show that almost all link demands are fully served in moderate demand load cases. In Fig. 9(d), satisfaction rates drop linearly as number of nodes increases. All satisfaction values are nearly the same and do not differ with respect to G value.

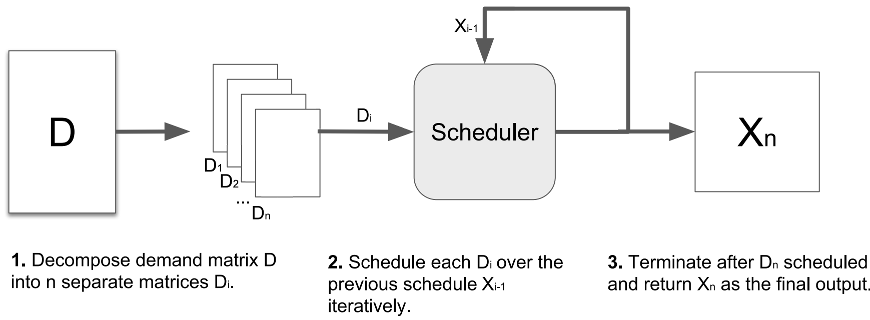
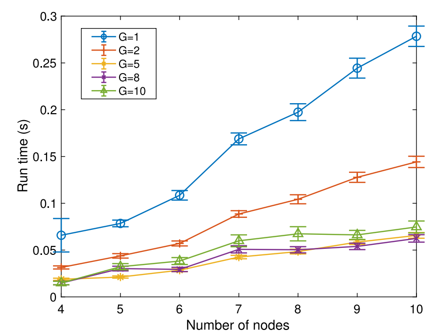
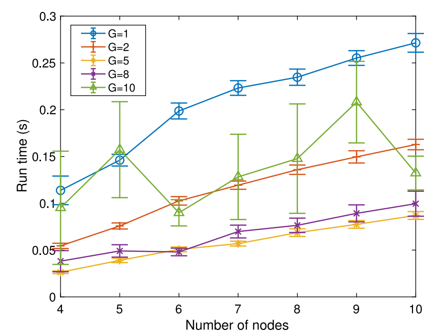


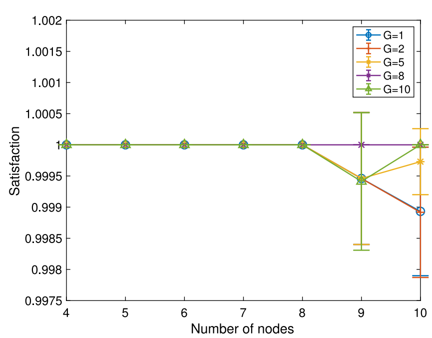
Fig. 8. The steps of the Iterative approach.



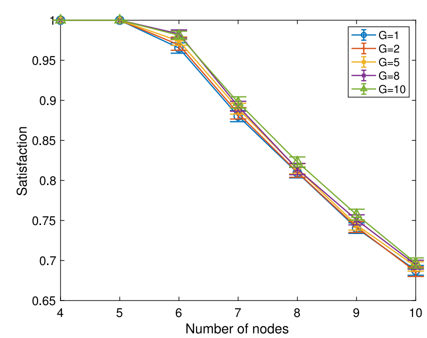
(a) Run time results where $l = 0.10$.



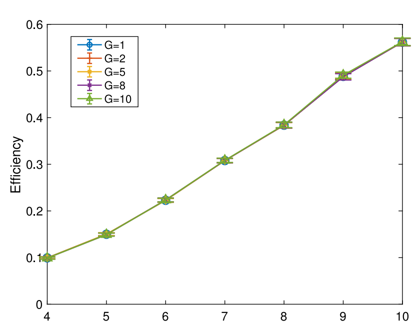
(b) Run time results where $l = 0.25$.



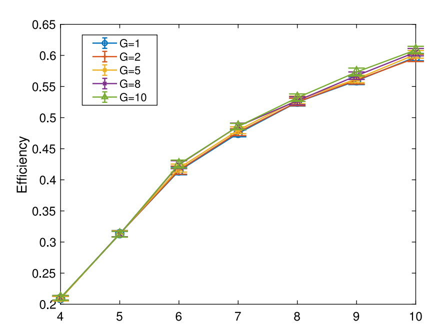
(c) Satisfaction results where $l = 0.10$.



(d) Satisfaction results where $l = 0.25$.



(e) Efficiency results where $l = 0.10$.



(f) Efficiency results where $l = 0.25$.

Fig. 9. Iterative approach performances with $N = (4, 5, \dots, 10)$, $M = 6$, $T = 16$, and $C = 4$.

In Fig. 9(e), efficiency values increase proportional to number of nodes. In contrast, the slope in Fig. 9(f) slightly degrades for the cases in which number of nodes values are greater than 5 as the satisfaction decreases in Fig. 9(d). In general, the efficiency increases with l and N values. Similar to the satisfaction results, efficiency results are not affected by the group size, G.

Fig. 10 provides a further inspection of the performance of the iterative approach on scheduling over-populated clusters. Fig. 10(a) shows that the iterative approach carries out scheduling process in a short period of time with particular G values, e.g., below 0.2 s where G is equal to 2 or 5. (Further analysis of G parameter is given in Section 3.4.) In Fig. 10(b), decrease in satisfaction performance stems from the concurrence of stable resource amount and the growth of the

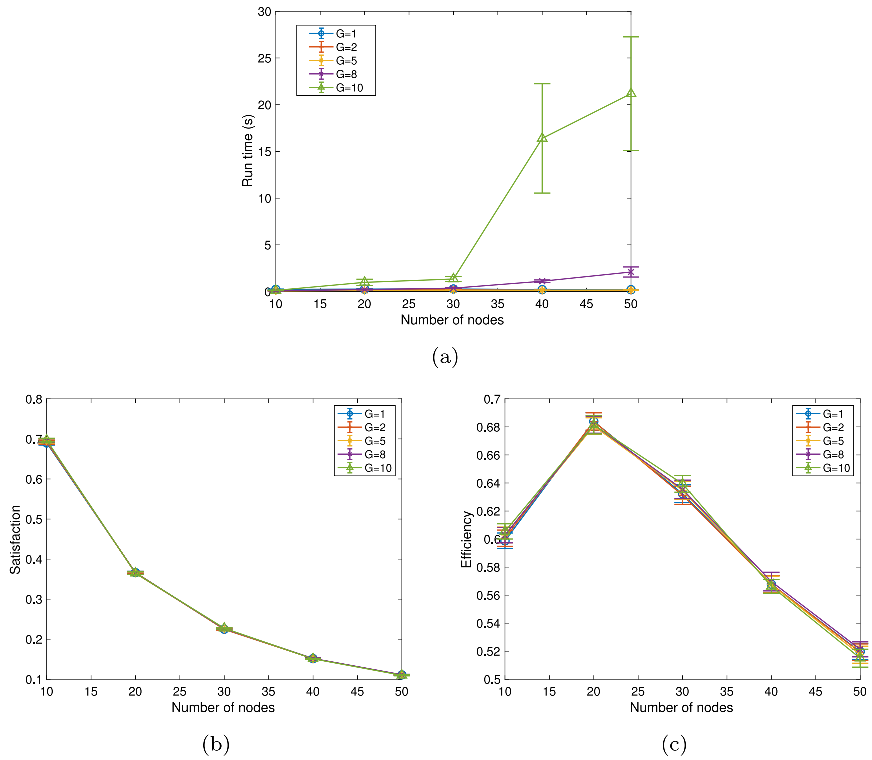


Fig. 10. (a) Runtime, (b) satisfaction, and (c) efficiency results of the iterative approach with $l = 0.25$, $N = (10, 20, \dots, 50)$, $M = 6$, $T = 16$, and $C = 4$.

number of nodes. However, in Fig. 10(c) the efficiency increases when the number of nodes is 20 then decreases since fixed T number fails to meet resource requirements of increasing number of destined nodes of each multicast links as N rises. Similar to the results in Fig. 9, the amount of resource blocks occupied does not rely on G value.

3.3. Comparison of performance assessments

In order to further investigate holistic and iterative approach results, a combined quality measure, which is efficiency divided by run time, has been applied to each simulation run in Figs. 7 and 9 and results are compared in Fig. 11 with their corresponding efficiency values. Fig. 11 shows that the holistic approach has the least and the iterative approach using link demand groups with $G = 5$ has the best quality measures for both moderate and dense link demand cases. It can be observed that the orders of quality assessment are similar among all link demand groups except the rank of link demand group with $G = 8$ and $G = 10$ in Fig. 11(a) drops compared to dense link demand case illustrated in Fig. 11(c). Figs. 11(b) and 11(d) show that all Iterative ICLS results are as optimal as that of Holistic ICLS with a slight exception in Fig. 11(d) when number of nodes is greater than 7. Compared to holistic approach results in Figs. 7 and 11 the iterative approach shows nearly the same efficiency performance in substantially less amount of time with identical experimental setups.

Fig. 12 shows the performance assessments and efficiency values of another experiment that is devised by applying the same simulation run procedures with that of the experiments in Fig. 11. In Fig. 12(a), increasing the normalized load, l, leads to a decrease in performance where $N > 5$. However, holistic approach and iterative approach using link demand groups with $G = 5$ yield to the worst and the best performance, respectively. In contrast, Fig. 12(b) shows that Iterative ICLS has identical efficiency values for all G parameters. In addition, efficiency values for both Holistic and Iterative ICLS are nearly the same for $l \leq 0.2$ and close for $l > 0.2$ with a 0.07 (i.e., 7 percentage points in efficiency) difference at most. Therefore, it can be concluded that compared to Holistic ICLS, Iterative ICLS reaches near optimal results with less time costs.

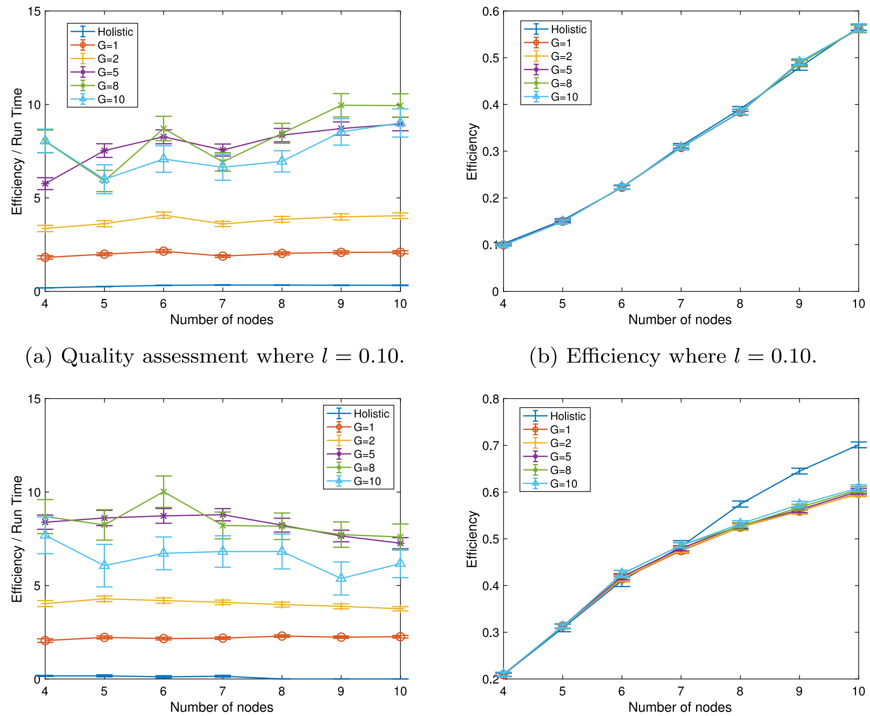
3.4. Group size (G) analysis

From previous simulations, it can simply be inferred that the performance results strictly depend on the parameter configuration. Apart from the input parameters such as N, M, T, and C, which are static and adopted from the communication medium requirements before the beginning of the scheduling process, an implicit and adaptable parameter, the group size, G, has a direct influence on the performance of Iterative ICLS algorithm as addressed in Fig. 9 prominently. Besides, an improper choice of G parameter may inflate the run time of Iterative ICSL, similar to the case in which run time is 20 s in average when $G = 10$ and $N = 50$ in Fig. 10(a). Therefore, it is essential to estimate G parameter properly prior to the execution of algorithm for the optimal scheduling performance.

In Fig. 13, each parameter in the fixed parameter set $l = 0.25$, $N = 8$, $M = 6$, and $R = 64$ ($T = 16$ and $C = 4$) is set to 3 different values and tested with various G parameters in order to illustrate the impact of G on performance measures. Fig. 13(b) shows that peak performances are correlated to the M value, as optimal G value is around $M = 6$ in Figs. 13(a), 13(c), and 13(d). It can be inferred that M is the most effective parameter on optimal G and choosing G close to M value leads iterative ICLS to work efficiently. The optimization process takes less amount of time when D_is have common nodes among the links, forcing the links to assign different slots and eliminating some of the cases that the algorithm tests. Selecting G close to M value leads to overlap D_is with D rows, causing links with the same source nodes reside in the same D_is, thereby increasing the run time performance.

3.5. Discussion on iterative ICLS use

Iterative ICLS does not only significantly contribute to performance but also has some advantages in terms of flexibility as the algorithm can be adapted to various scenarios for boosting the scheduling process and solve some specific problems. For instance, when the frame size is large, D_is contain relatively small amount of links, and the frame is empty, CSMA (Carrier Sense Multiple Access) can be conducted or the target



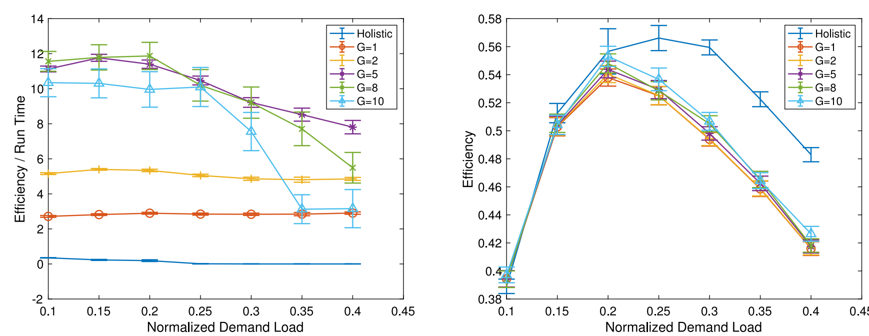
(a) Quality assessment where $l = 0.10$.

(b) Efficiency where $l = 0.10$.

(c) Quality assessment where $l = 0.25$.

(d) Efficiency where $l = 0.25$.

Fig. 11. Quality assessment and efficiency of holistic and iterative approaches with $N = (4, 5, \dots, 10)$, $M = 6$, $T = 16$, and $C = 4$.



(a) Quality assessment.

(b) Efficiency.

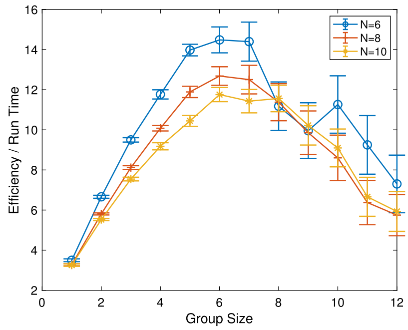
Fig. 12. Quality assessment and efficiency of holistic and iterative approaches with $l = (0.10, 0.25, \dots, 0.40)$, $N = 8$, $M = 6$, $T = 16$, and $C = 4$.

frame can be restricted for narrowing down problem space. Another enhancement can be achieved by adjusting frequency of scheduling process rather than running the scheduler for each frame. Alternatively, some links or nodes can be reserved to the same resource blocks in consecutive frames as a hybrid solution. The variations of Iterative ICLS executions can be applied according to the needs of target clustered ad hoc networks and out of the scope of the study.

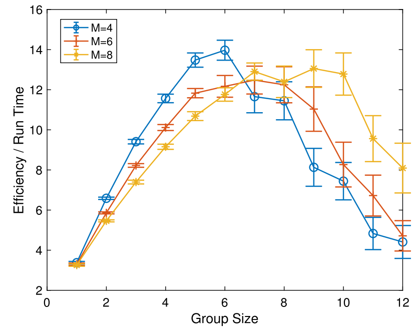
The inter-cluster scheduling conducted on gateway nodes, which is described as the schedule sent to neighboring cluster heads in the third step in Fig. 5, can be simply conducted by Iterative ICLS by the following strategy: The inter-cluster link is scheduled in one Iterative ICLS iteration over combination of both neighboring intra-cluster frames. Single iteration of a single Iterative ICLS leads to an immediate inter-cluster schedule by alleviating the effect of large domain size of combined resource space on run time performance.

An iteration of Iterative ICLS is not useful only in inter-cluster links, but it is also capable of maintaining intra-cluster link scheduling during the network lifetime. D input with large l values that can occupy nearly all slots of an empty frame is not common, and is possible in a few particular cases only such as starting of the network. Total number of link demands newly arrived in a cluster head in a frame duration is expected to be less than that of new link demands within a frame in previous simulations, in which worst case l values are tested.

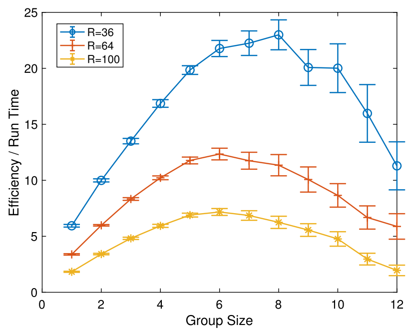
In Fig. 14, run time performances of Iterative ICLS with a single iteration is tested with a few link request, demonstrating the expected input behavior in maintenance phase of Iterative ICLS scheduler. Given previously scheduled frames with 3 different l_f values 0.10, 0.25, and 0.40, recent d link demands are assigned to available resource blocks in the frame. Fig. 14 shows that the Iterative ICLS responses in a reasonable period of time, around 10 ms, which is practical for handling new request for a fully operational network. In Fig. 14, for instance, given slot size of 4096 bits and slot time of 2 ms, providing up to 2 Mbps data rate and generating maximum 30 link demands per second, the scheduling lasts less than 1 frame duration, $2 \times T = 100$ ms, for all d and N values. This slot time configuration also allows the scheduling of the 187.5 link requests per second in 2 frame times for the test case with $N = 10$ and $G = 8$ in Fig. 9(a), where recent link requests and all previously scheduled active links are scheduled simultaneously on an empty frame unlike the maintenance phase.



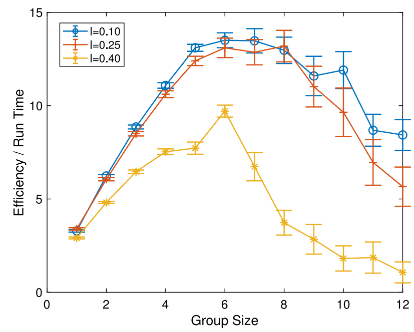
(a) Quality assessment with different N values.



(b) Quality assessment with different M values.

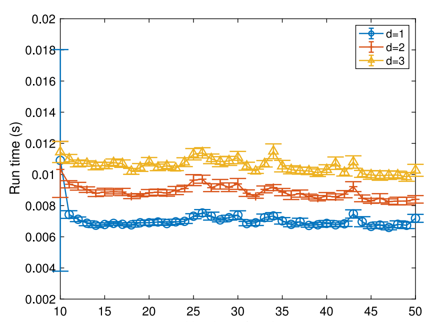


(c) Quality assessment with different R values.

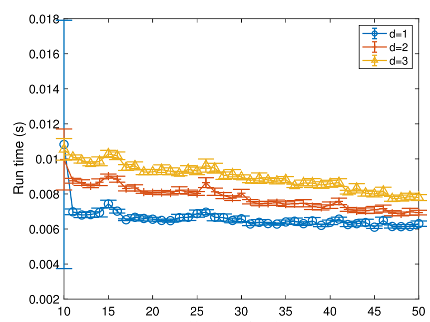


(d) Quality assessment with different l values.

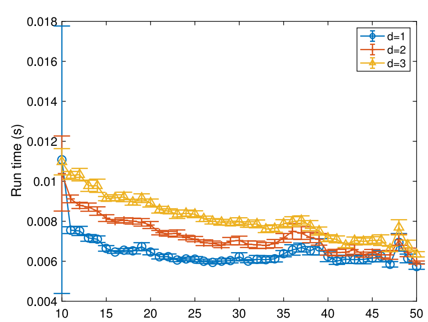
Fig. 13. The impact of group size on performance measures with different parameters.



(a)



(b)



(c)

Fig. 14. Run time performances of maintenance phase with (a) $l_f = 0.10$, (b) $l_f = 0.25$, and (c) $l_f = 0.40$ where $N = (10, 11, \dots, 50)$, $T = 50$, and $C = 5$.

The ICLS is tested up to 50 nodes with $T = 50$ slots. Results are ranging from 6 ms to 12 ms which is responsive to the scheduler needs and stable as N increases. The scheduler performance can further be increased if more lightweight optimizer alternatives are used.

4. Related work

This section provides an extensive investigation of scheduler algorithms in the literature, considering not only similar settings with the proposed study but also diverse perceptions and conditions such as node scheduling, single-channel MAC, WSNs etc. Inspected features of the studies are mostly uncorrelated. Therefore, in the following part of this section, algorithms are summarized in Table 2 displaying their aspects rather than categorizing with respect to a single property.

Several scheduling algorithms have been proposed for wireless communication [14,15]. One approach is to schedule the nodes so that each node transmission may be received by all of its neighboring nodes without interference, namely node scheduling [14]. Some of the centralized node scheduling algorithms, in which the scheduling process is carried out by a single entity, optimize resource utilization with some auxiliary methods, e.g., neural networks [16], proposing a novel hysteretic noisy chaotic neural network (HNCNN), and genetic algorithms [17]. In contrast, [18] is a study presenting a distributed node scheduling method, in which scheduling is performed in each node in the network. TDMA scheduling in a central entity is not feasible since the scheduler node requires a high level of centrality. In fact, node scheduling algorithms, even distributed, are yet incapable of resolving the conflicts in receiver types depicted in Fig. 6(b).

Coloring- and Coding-based Multi-Channel TDMA (CC-TDMA) [19] is a multichannel and graph-based TDMA link scheduling algorithm based on edge coloring and algebraic coding theory. Although the algorithm prevents collisions to some extent, it fails to avoid collisions in some cases. Zhang et al. [20] propose a graph-based TDMA link scheduling algorithm, namely Coloring- and Probability-Based TDMA (CP-TDMA), using distributed edge coloring as well. The algorithm performs well in terms of throughput. However, it has a limited slot utilization and conflict avoidance. Apart from the strategies taken into account of topology and neighborhood, some algorithms, namely cross-layer TDMA scheduling algorithms, attempt to optimize different network metrics such as route selection, load balancing, and power consumption. In Fairness Adaptive TDMA Scheduling algorithm (FATS) [21], overall fairness of sensor networks can be obtained adaptively according to link quality. However, extensive use of Acknowledgment (ACK) messages introduces overhead to network communication. Load-Balanced-Fair Flow Vector Scheduling Algorithm (LB-FFVSA) [22] aims to determine an optimized and load-balanced slot assignment in terms of overall performance and fairness per flow. The algorithm increases frame length in high network traffic conditions, resulting in increase in end-to-end delay. Hybrid MAC Protocol for Emergency Response Wireless Sensor Networks (ER-MAC) [23] and Flexible-schedule-based TDMA Protocol (FlexiTP) [24] are hybrid algorithms, which use Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA) in setup phase and TDMA during network lifetime. ER-MAC is applicable in emergency scenarios and energy-efficient yet it is not scalable for high-density networks. Although FlexiTP is fault-tolerant and energy-efficient since nodes can modify their number of scheduled slots based on local information, the network has tree structure consisting of improper parent–child pairs leading to fragile schedules. In [25] Advertisement-based TDMA protocol (ATMA), in which a frame is divided into contention-based synchronization and advertisement period followed by a contention-free data period, is introduced. On ATMA, links are reserved in the data period according to successful transmission to advertisement slots. Therefore, link demands are randomly served without an optimization process.

The algorithms focusing on MAC protocols also introduce various multichannel scheduling strategies in wireless sensor networks (WSNs) which employs mostly data gathering tree or forest network structure. Le et al. [26] propose to arrange nodes that communicate frequently into the same channel and separate nodes that do not communicate much into different channels since frequency synthesizer brings extra overhead to stabilize. The protocol is designed for a sink tree structure and is not tested for other network traffic cases. This method may also cause some packets sent to transmitting nodes drop in case of frequent frequency switch. Game Based Channel Assignment algorithm (GBCA) [27] models a channel assignment game to solve the problem with a suboptimal result by negotiating channel usage among each Parent-Children Set (PCS) in a static and of tree/forest network topology. However, the solution is not effective as the method takes network topology into account rather than the packet traffic. In Multi-Channel Lightweight Medium Access Control (MC-LMAC) [28], time is slotted and several consecutive slots are organized as a frame within a multi-channel MAC protocol and a fully-distributed scheduling mechanism in which each node shares its bitmap of the slot and channel usage with its neighboring nodes in order to prevent interference. The algorithm incurs extra overhead due to control messaging transmission. In [29], a heterogeneous frame structure, which involves slots of different lengths, is proposed in order to specialize in serving multiple urgent and small messages simultaneously. Several data transmissions from a source industrial wireless sensor node (IWSN) to sink node, named as flows, are scheduled on divisible slots. The operation of scheduling simultaneous flows requires centralized management. Therefore, the method is not applicable for clustered structure and poses problems of centralized ad hoc networks.

Linear programming may generate several numbers of columns that are infeasible to handle and ineffective to reach the optimal solution for some cases. Column generation method which basically decomposes the problem into master and subproblems is an approach applied to integer programming in order to avoid redundant calculations. This method is used for optimizing link scheduling by a number of proposed methods in order to come up with a relatively more scalable scheme. For instance, [30] conducts resource optimization with both node and link scheduling, yet applied to single-channel spatial TDMA (or STDMA). Besides, as the least input load case, the optimizer assigns 26 links from 10 nodes into 17 resource blocks in 6 s, which is a long time compared to the Iterative ICLS optimizer performance in 10(a). [31] presents a cross-layer resource allocation with a nonlinear column generation technique. Therein, nodes are assumed to communicate with single-channel links and time performance of the proposed technique is not discussed in the study. [32] compares two approaches of scheduling: TC (Transmission Scheduling)- approach and tree-based approach (Round-Optimal Schedule (ROS)) for multi-channel TDMA protocol on WSNs and focuses on optimizing sensor coverage of targets on a network with a limited (many-to-one) communication pattern.

A randomly distributed TDMA scheduling algorithm, Distributed Randomized TDMA Scheduling (DRAND), is presented in [33] operating a state diagram with 4 states: idle, request, grant, and release and the nodes transit among these states in order to be assigned to a slot. The algorithm randomly schedules a single channel TDMA protocol with high messaging complexity rather than adaptive to the communication needs of the nodes. In [34], two centralized TDMA scheduling algorithms consisting of a classical node-based and a novel level-based approach are introduced together with a distributed node scheduling with graph coloring. Similar to the aforementioned WSN schedules, the scheduling algorithm is designed for restricted network traffic, which forms a tree and all packets are destined to a sink node.

Some studies also conducted on scheduling in clustered wireless networks. In [35], centralized network-wide optimized TDMA schedules are aimed to provide high power efficiency, zero conflict, and reduced end-to-end delay. The objective of the scheduling algorithm is to achieve minimum TDMA frame length together via maintaining the slot reuse concept. However, the proposed method is limited to resolve single-channel TDMA scheduling and based upon WSN network traffic pattern. [36] proposes a scheduling method targeting an efficient time slot assignment while reducing power transmission in a clustered network structure. The algorithm adjusts the transmission power level with respect to the node distance. However, inter-cluster communication relies on backbone and slots are statically assigned to cluster members, causing battery drainage and redundant use of resources on certain nodes. Lee et al. [37] propose multi-channel TDMA scheduling algorithms for intra-cluster and inter-cluster communications. The proposed algorithms distribute available non-overlapping channels to

Table 2
Overall comparison of scheduling algorithms.

Algorithm	Scheduler type	Scheduler deployment	Channel cardinality	Supported link types	Optimization	Target network	Traffic flow
ICLS	Link	Clustered	Multi-channel	Unicast Multicast Broadcast	✓	Ad Hoc Network	Distributed
HNCNN [16]	Node	Centralized	Single-channel	Broadcast	✓	Ad Hoc Network	Distributed
[17]	Node	Centralized	Single-channel	Broadcast	✓	WiMAX	Distributed
[18]	Node	Distributed	Single-channel	Unicast Multicast Broadcast	✓	Ad Hoc Network	Distributed
CC-TDMA [19]	Link	Distributed	Multi-channel	Unicast		Ad Hoc Network	Distributed
CP-TDMA [20]	Link	Distributed	Single-channel	Unicast		Ad Hoc Network	Distributed
FATS [21]	Link	Centralized	Single-channel	Unicast		WSN	Tree
LB-FFVSA [22]	Link	Centralized	Single-channel	Broadcast		Wireless Multi-hop Network	Distributed
ER-MAC [23]	Link	Distributed	Single-channel	Unicast Broadcast		WSN	Tree
FlexiTP [24]	Link	Distributed	Single-channel	Unicast Broadcast		WSN	Tree
ATMA [25]	Link	Distributed	Single-channel	Unicast	✓	WSN	Distributed
[26]	Link	Distributed	Multi-channel	Unicast	✓	WSN	Tree
GBCA [27]	Link	Distributed	Multi-channel	Unicast	✓	WSN	Tree
MC-LMAC [28]	Node	Distributed	Multi-channel	Unicast Broadcast		WSN	Tree
[29]	Link	Centralized	Multi-channel	Unicast		IWSN	Tree
[30]	Node Link	Centralized	Single-channel	Unicast Broadcast	✓	Ad Hoc Network	Distributed
[31]	Link	Centralized	Single-channel	Unicast	✓	Ad Hoc Network	Distributed
TC and ROS [32]	Link	Centralized	Single-channel	Unicast	✓	WSN	Tree
DRAND [33]	Node	Distributed	Single-channel	Broadcast		Ad Hoc Network	Distributed
[34]	Node	Distributed	Single-channel	Broadcast		WSN	Tree
[35]	Link	Centralized	Single-channel	Unicast	✓	WSN	Tree
[36]	Node	Clustered	Multi-channel	Unicast	✓	Ad Hoc Network	Clustered
[37]	Link	Clustered	Multi-channel	Unicast Broadcast	✓	Ad Hoc Network	Clustered
[38]	Link	Centralized	Single-channel	Unicast		Tactical Data Link	Distributed
[39]	Node	Distributed	Single-channel	Unicast Broadcast		VANET	Tree
[40]	Link	Centralized	Single-channel	Unicast Broadcast	✓	Visible Light Communication Network	Distributed
SD-MUCS [41]	Link	Clustered	Multi-channel	Unicast	✓	5G	Distributed

proper clusters and reduce the optimal number of TDMA slots by scheduling them over the allocated channels. Although the algorithm manages to shrink frame size, the scheduling problem is addressed slightly since the instructions of handling constraints are not available in the study. In case of conflicts, extra channels (if available) for the clusters with dense packet traffic, which leads to long frame size, are reserved as the solution which is not effective and comprehensive for scheduling clustered wireless networks.

Further, scheduling techniques are presented for various types of networks. A recent work [38] is an extension to TDMA structure in Tactical Data Link [42] and reserves extra interrupt time slots on fixed time slot scheduling, aiming to decrease communication delay. The proposed method is not adaptive and very simple although the performance results show improvements. [39] proposes a vehicular ad hoc network scheduler providing better deadline miss ratios and faster response time. However, the method schedules safety and non-safety data traffic only between mobile vehicles and road side units, thereby limited to 1-to-n communication. In [40] a spatial TDMA is introduced for visible light communication networks with a weighted mixed integer linear programming model. Similar to the study in [39], the network structure is different than our proposed multi-channel TDMA ad hoc networks in which a transmission interferes with all other links in the same channel rather than a network model which can be represented as an undirected graph. Semi-Distributed Multi-Channel Scheduling algorithm, called SD-MUCS, [41] is another link scheduler using network graphs. A multi-channel TDMA schedule is proposed in favor of resource reuse and connected nodes are grouped into clusters so that link scheduling is handled within clusters and inter-cluster links are scheduled in a centralized manner. The method attempts to minimize the number of required channels ignoring singular communication needs and separates controller elements from the communicating nodes as a feature of 5G architecture which is different than our target structure, ad hoc network.

The scheduling algorithms are summarized in Table 2 with their properties. Scheduler type implies whether the strategy of the algorithm is to schedule the nodes or the links. Scheduler deployment states the position of the scheduler entity in the network. The scheduling operation is carried out in a single entity if its scheduler type is centralized, whereas it is distributed in the case which every node is involved in scheduling process. If clustered, only cluster heads operate scheduling. Channel cardinality is the count of available channels as supported link types implies the amount of the destination node(s) within an individual link recognizable by the scheduler. Table 2 presents if any metric is optimized on the scheduling process as well as its use of target network. Last, traffic flow shows the destination characteristics of the links. Distributed traffic flow displays no specific pattern as links with tree flow destined to a significant entity, e.g., sink, base station etc., with a single- or multi-hop. In contrast, clustered flow links are destined to the cluster head of the cluster head and inter-cluster communication takes place among cluster heads, building a backbone in the network topology. Table 2 acknowledges that among other methods, ICLS is the only multi-channel link scheduler that supports any link type, performs optimization, and operates on distributed links which do not follow a certain flow pattern. Furthermore, ICLS benefits clustered deployment in which benefits of centralized and distributed structure coalesce for more supervised and responsive schedule.

5. Conclusion

Link scheduling in clustered ad hoc networks is a tedious task in terms of scalability and compatibility. In this paper, the problem formulation of intra- and inter-cluster link scheduling on CUPS architecture is introduced and conflicted cases are presented for different scheduling scenarios. Link scheduling is formulated as an integer nonlinear optimization problem and reduced to its linear form for simplifying the formulation and discarding redundant computation. The performance analysis for different parameter combinations is discussed. The results show that the time complexity of the optimization problem exponentially grows by the node size. In addition, the optimization requires a vast amount of calculation even in small parameter sizes. However, dividing link demands into separate link groups with size M then optimizing iteratively by using the previous optimization results as history is a basic yet effective strategy and yields to rapid and efficient results.

In the future, some refinements will be applied on the algorithm for reaching more scalable solution, e.g., testing with other optimizers that are faster and compatible to the scheduling task. In order to obtain performance results more relevant to real life conditions, the algorithm will run in a network simulator on a complete clustered network setup with a clustering method, routing algorithm, and an optimizer adapted to the simulator environment.

CRediT authorship contribution statement

M. Levent Eksert: Conceptualization, Methodology, Software, Validation, Writing - original draft. **Hamdullah Yücel:** Supervision, Writing - review & editing. **Ertan Onur:** Conceptualization, Supervision, Methodology, Writing - review & editing, Funding acquisition, Project administration.

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 is partially supported by ASELSAN project HBT-TE-2017-012. We would like to thank Doğanalp Ergenç for his contributions to our study. Dr. Onur is partially funded by the Fulbright Visiting Scholar Program.

References

[1] P. Ruiz, P. Bouvry, Survey on broadcast algorithms for mobile ad hoc networks, ACM Comput. Surv. 48 (1) (2015) 1–35.

[2] M. Adler, R.K. Sitaraman, A.L. Rosenberg, W. Unger, Scheduling time-constrained communication in linear networks, in: Proceedings of the Tenth Annual ACM Symposium on Parallel Algorithms and Architectures, ACM, 1998, pp. 269–278.

[3] How to: determine party size for mountaineer trips, 2020, https://www.mountaineers.org/blog/how-to-determine-party-size-for-mountaineers-trips, Accessed: 2020-07-20.

[4] Modern military force structures, 2020, https://www.cfr.org/backgrounder/modern-military-force-structures, Accessed: 2020-07-20.

[5] D. Ergenç, L. Eksert, E. Onur, Dependability-based clustering in mobile ad-hoc networks, Ad Hoc Netw. 93 (2019) 101926.

[6] D. Ergenç, E. Onur, Cupsman: Control user plane separation based routing in ad-hoc networks, 2018, ArXiv, abs/1807.10747.

[7] W.R. Heinzelman, A. Chandrakasan, H. Balakrishnan, Energy-efficient communication protocol for wireless microsensor networks, in: Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, IEEE, 2000, pp. 10–pp.

[8] O. Younis, S. Fahmy, HEED: a hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks, IEEE Trans. Mob. Comput. 3 (4) (2004) 366–379.

[9] S.V. Bana, P. Varaiya, Space division multiple access (SDMA) for robust ad hoc vehicle communication networks, in: ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 01TH8585), IEEE, 2001, pp. 962–967.

[10] A. Goldberg, S. Plotkin, G. Shannon, Parallel symmetry-breaking in sparse graphs, in: Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing, ACM, 1987, pp. 315–324.

[11] F. Kuhn, R. Wattenhofer, On the complexity of distributed graph coloring, in: Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing, ACM, 2006, pp. 7–15.

[12] M. Conforti, G. Cornuéjols, G. Zambelli, Integer Programming, Vol. 271, Springer, 2014.

[13] MATLAB optimization toolbox, 2020, https://www.mathworks.com/products/optimization.html, Accessed: 2019-07-20.

[14] A. Sgora, D.J. Vergados, D.D. Vergados, A survey of TDMA scheduling schemes in wireless multihop networks, ACM Comput. Surv. 47 (3) (2015) 53.

[15] P. Huang, L. Xiao, S. Soltani, M.W. Mutka, N. Xi, The evolution of MAC protocols in wireless sensor networks: A survey, IEEE Commun. Surv. Tutor. 15 (1) (2013) 101–120.

[16] M. Sun, L. Zhao, W. Cao, Y. Xu, X. Dai, X. Wang, Novel hysteretic noisy chaotic neural network for broadcast scheduling problems in packet radio networks, IEEE Trans. Neural Netw. 21 (9) (2010) 1422–1433.

[17] R. Gunasekaran, S. Siddharth, P. Krishnaraj, M. Kalaiarasan, V.R. Uthariaraj, Efficient algorithms to solve broadcast scheduling problem in WiMAX mesh networks, Comput. Commun. 33 (11) (2010) 1325–1333.

[18] Y. Liu, V.O. Li, K.-C. Leung, L. Zhang, Topology-transparent distributed multicast and broadcast scheduling in mobile ad hoc networks, in: Vehicular Technology Conference (VTC Spring), 2012 IEEE 75th, IEEE, 2012, pp. 1–5.

[19] X. Zhang, J. Hong, L. Zhang, X. Shan, V.O. Li, CC-TDMA: Coloring-and coding-based multi-channel TDMA scheduling for wireless ad hoc networks, in: Wireless Communications and Networking Conference, 2007. WCNC 2007. IEEE, IEEE, 2007, pp. 133–137.

[20] X. Zhang, J. Hong, L. Zhang, X. Shan, V.O. Li, CP-TDMA: Coloring-and probability-based TDMA scheduling for wireless ad hoc networks, IEICE Trans. Commun. 91 (1) (2008) 322–326.

[21] Z. Wang, F. Yu, J. Tian, Z. Zhang, A fairness adaptive TDMA scheduling algorithm for wireless sensor networks with unreliable links, Int. J. Commun. Syst. 27 (10) (2014) 1535–1552.

[22] D.J. Vergados, A. Sgora, D.D. Vergados, D. Vouyioukas, I. Anagnostopoulos, Fair TDMA scheduling in wireless multihop networks, Telecommun. Syst. 50 (3) (2012) 181–198.

[23] L. Sitanayah, C.J. Sreenan, K.N. Brown, ER-MAC: A hybrid MAC protocol for emergency response wireless sensor networks, in: Sensor Technologies and Applications (SENSORCOMM), 2010 Fourth International Conference on, IEEE, 2010, pp. 244–249.

[24] W.L. Lee, A. Datta, R. Cardell-Oliver, FlexiTP: a flexible-schedule-based TDMA protocol for fault-tolerant and energy-efficient wireless sensor networks, IEEE Trans. Parallel Distrib. Syst. 19 (6) (2008) 851–864.

[25] S. Ray, I. Demirkol, W. Heinzelman, Supporting bursty traffic in wireless sensor networks through a distributed advertisement-based tdma protocol (ATMA), Ad Hoc Netw. 11 (3) (2013) 959–974.

[26] H.K. Le, D. Henriksson, T. Abdelzaher, A practical multi-channel media access control protocol for wireless sensor networks, in: Proceedings of the 7th International Conference on Information Processing in Sensor Networks, IEEE Computer Society, 2008, pp. 70–81.

[27] Q. Yu, J. Chen, Y. Fan, X. Shen, Y. Sun, Multi-channel assignment in wireless sensor networks: A game theoretic approach, in: INFOCOM, 2010 Proceedings IEEE, IEEE, 2010, pp. 1–9.

[28] O.D. Incel, L. Van Hoesel, P. Jansen, P. Havinga, MC-LMAC: A multi-channel MAC protocol for wireless sensor networks, Ad Hoc Netw. 9 (1) (2011) 73–94.

[29] C. Xia, X. Jin, L. Kong, C. Xu, P. Zeng, Heterogeneous slot scheduling for real-time industrial wireless sensor networks, Comput. Netw. 157 (2019) 68–77.

[30] P. Björklund, P. Värbrand, D. Yuan, A column generation method for spatial TDMA scheduling in ad hoc networks, Ad Hoc Netw. 2 (4) (2004) 405–418.

[31] M. Johansson, L. Xiao, Cross-layer optimization of wireless networks using nonlinear column generation, IEEE Trans. Wireless Commun. 5 (2) (2006) 435–445.

[32] M. Bakshi, B. Jaumard, M. Kaddour, L. Narayanan, On TDMA scheduling in wireless sensor networks, in: Electrical and Computer Engineering (CCECE), 2016 IEEE Canadian Conference on, IEEE, 2016, pp. 1–6.

[33] I. Rhee, A. Warrier, J. Min, L. Xu, DRAND: Distributed randomized TDMA scheduling for wireless ad hoc networks, IEEE Trans. Mob. Comput. 8 (10) (2009) 1384–1396.

[34] S.C. Ergen, P. Varaiya, TDMA scheduling algorithms for wireless sensor networks, Wirel. Netw. 16 (4) (2010) 985–997.

[35] L. Shi, A.O. Fapojuwo, TDMA scheduling with optimized energy efficiency and minimum delay in clustered wireless sensor networks, IEEE Trans. Mob. Comput. 9 (7) (2010) 927–940.

[36] S.-C. Kim, An energy efficient scheduling algorithm for cluster-based mobile wireless networks, in: Ubiquitous and Future Networks (ICUFN), 2012 Fourth International Conference on, IEEE, 2012, pp. 357–362.

[37] H. Lee, C.-C. Lim, J. Choi, Cluster-based multi-channel scheduling algorithms for ad hoc networks, in: Wireless and Optical Communications Networks, 2007. WOCN'07. IFIP International Conference on, IEEE, 2007, pp. 1–5.

[38] T. Liu, X. Lu, Z. Dou, An optimal design of time division multiple access protocol for data link network, in: 2016 IEEE International Conference on Electronic Information and Communication Technology (ICEICT), 2016, pp. 158–161.

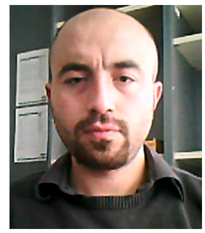
[39] K. Bok, J. Lim, S. Hong, J. Yoo, A multiple RSU collaborative scheduling scheme for data services in vehicular ad hoc networks, Cluster Comput. 20 (2) (2017) 1167–1178.

[40] P. Adasme, F. Seguel, I. Soto, E.S. Juan, Spatial time division multiple access for visible light communication networks, in: 2017 First South American Colloquium on Visible Light Communications (SACVLC), 2017, pp. 1–6.

[41] Z. Naghsh, S. Valaee, Semi-distributed conflict-free multichannel TDMA link scheduling for 5G, in: 2017 51st Asilomar Conference on Signals, Systems, and Computers, 2017, pp. 1407–1411.

[42] W. Wenzheng, Z. Jinglun, L. Pengcheng, Review of tactical data link technology, Electron. Opt. Control 15 (11) (2008) 41–46.

Mustafa Levent Eksert received a B.S. degree and M.S. degree in Computer Engineering Department from Middle East Technical University (METU). He is currently a Ph.D. Candidate in Computer Engineering Department in METU and his research areas are ad hoc and wireless networks, clustering and scheduling in wireless networks. He is working as a senior software design engineer in ASELSAN.



Hamdullah Yücel received his B.S. degree in Mathematics from Middle East Technical University (METU) in 2007. The same year he enrolled integrated doctoral education program in Institute of Applied Mathematics, METU and he completed his Ph.D. thesis titled "Adaptive Discontinuous Galerkin Methods For Convection Dominated Optimal Control Problems" in 2012. Throughout the same period he worked as a research assistant in the Department of Mathematics in METU. During his Ph.D. studies, he has been in Technische Universität Darmstadt in Germany and Rice University in the USA as a visiting research assistant. After his Ph.D. he worked as a postdoctoral researcher at the Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg in Germany. Since 2017, he is a faculty member of Institute of Applied Mathematics, METU.

Dr. Hamdullah Yücel's research topics include partial differential equation (PDE) constrained optimization (optimal control), discontinuous Galerkin methods, adaptive finite element methods, and uncertainty quantification.

Ertan Onur 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. After the B.Sc. degree, he worked for LMS Durability Technologies GmbH, Kaiserslautern, Germany. During the M.Sc. and Ph.D. degrees, he worked as a project leader at Global Bilgi, Istanbul and as an R&D project manager at Argela Technologies, İstanbul. He developed and managed many commercial telecommunications applications, and has a patent. After obtaining his Ph.D. degree, he worked as an assistant professor at Delft University of Technology, the Netherlands. From 2014 on, he is with Middle East Technical University, Turkey. He was the editor/convenor of the Personal Networks Group of Ecma International Standardization Body. Dr. Onur's research interests are in the area of computer and wireless networks and their security. He is the founder of the Wireless systems, Networks and Cybersecurity Laboratory (WINS Lab). He is a member of IEEE.