IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 14, 2021, accepted January 31, 2021, date of publication February 8, 2021, date of current version February 17, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3057587

A Real-Time, Distributed, Directional TDMA MAC Protocol for QoS-Aware Communication in Multi-Hop Wireless Networks

SHIVAM GARG[1], (Student Member, IEEE), VENU SRI SUSHMA KUCHIPUDI[2], ELIZABETH SERENA BENTLEY[3], (Member, IEEE), AND SUNIL KUMAR[1,2], (Senior Member, IEEE)

[1]Computational Science Research Center, San Diego State University, San Diego, CA 92182, USA
[2]Department of Electrical and Computer Engineering, San Diego State University, San Diego, CA 92182, USA
[3]Air Force Research Laboratory, Rome, NY 13441, USA

Corresponding author: Sunil Kumar (skumar@sdsu.edu)

This work was supported by the U.S. Air Force Research Laboratory, under agreements FA8750-14-1-0075 and FA8750-20-1-1005. Distribution A. Approved for public release: Distribution unlimited: AFRL-2020-0539 on 17 Dec 2020.

ABSTRACT Time division multiple access (TDMA) based medium access control (MAC) schemes are widely used for communication among directional nodes since they can provide a conflict-free transmission schedule. However, the existing directional TDMA schemes introduce significant overhead and delay, and cannot adapt in real-time to topology changes in a directional multi-hop network. These schemes also incur considerable overhead and delay in order to support the QoS (quality of service) traffic. In this paper, a novel, real-time, distributed, directional TDMA scheme is presented for directional multi-hop wireless networks. This scheme adapts to the topology changes and/or flow requirements in *real-time*, and facilitates *QoS-aware communication* with no *notification overhead*. In the proposed scheme, the 1-hop neighborhood of every node is divided into *fully connected 1-hop neighborhoods*, which allows the node to intelligently serve multiple routes without requiring a globally converged scheduling solution. This feature allows the use of a low-complexity rank-based mechanism to obtain a distributed, real-time transmission schedule for a directional multi-hop network. The following new features are also added in the proposed scheme: (*i*) REQ period which reduces slot wastage, (*ii*) *throughput scaling* which ensures fairness and helps in congestion management, and (*iii*) piggyback reservation period which increases the spatial reuse and adapts to the dynamic requirements of multiple flows in real-time. The control-period overhead in our scheme is low and linearly changes with the number of nodes in a fully connected 1-hop neighborhood, instead of the total number of nodes in the entire network. Simulation results and comparisons with other recent, distributed TDMA-based schemes show that our scheme provides a higher throughput with very low control overhead for both static and mobile network topologies.

INDEX TERMS Directional communication, distributed medium access control (MAC), multi-hop network, mobile network, quality of service (QoS), time-division multiple access (TDMA).

I. INTRODUCTION

The use of directional antennas in wireless communication reduces the co-channel interference and extends the coverage range, thereby increasing the spatial reuse and network capacity. However, it also introduces several challenges, such as node deafness, the hidden terminal problem, the head-of-the-line (HOL) blocking problem, and the MAC (medium access control) layer capture problem [1]–[3]. The inability of a beam-formed node to receive incoming signals from other directions is called node deafness, which can lead a hidden node to cause interference to ongoing communications in its neighborhood.

Single-beam directional antennas (SBA) are widely-used in disaster response networks, flying ad-hoc networks, military networks and sensor networks [1]–[4]. Since the coverage area of a directional node is limited by its beam-width (θ^0), it must beam-form in the direction of the transmitter to successfully receive the packets. Since a receiver node in CSMA (carrier sense multiple access)-based MAC scheme

The associate editor coordinating the review of this manuscript and approving it for publication was Cunhua Pan.

remains unaware of when its transmitter (or neighbor) node will start the communication, it fails to beam-form in the direction of transmission and therefore misses the packet(s) which aggravates the deafness problem [1]–[3], [5]–[11]. To address this issue, some CSMA-based schemes [9], [11], [12] use two antenna system, where omnidirectional antenna overhears transmissions of neighbor nodes and SBA is used for data packet transmission. However, the use of an omnidirectional antenna reduces the benefits of spatial reuse and leads to the well-known gain-asymmetry problem [1], [13], [14]. In some other schemes [5], [15], [16], a node steers its antenna at least $(360^0/\theta^0)$ times to scan and notify its entire neighborhood, which introduces significant sweeping delay [17]. Moreover, *none of the above-mentioned CSMA-based schemes completely resolve the issues of deafness, hidden terminal, and capture effect* [6]–[8], [10], [13], [18]. *Therefore, they are not suitable for directional communication*.

The *TDMA (time division multiple access) schemes are widely used for directional communication since they can provide a conflict-free transmission schedule and avoid deafness and capture effect* [6]–[8], [10]–[14], [18], [19]. TDMA-based schemes also offer a better QoS (quality of service) support than random access-based schemes since they reserve a guaranteed period of time for each node to access the channel [19], [20]. *However, TDMA-based schemes can introduce large overhead and delay and are unable to adapt to topology changes in a multi-hop network in real-time [21], as discussed below*.

A. REVIEW OF DIRECTIONAL TDMA SCHEMES

Most directional TDMA-based MAC schemes (such as [13], [14], [18], [22]) divide a frame into three phases: neighbor discovery, reservation, and data traffic. During the neighbor discovery phase, nodes search for their neighbors and agree upon a reservation slot in which they negotiate data traffic slots. The neighbor discovery is performed only at the start of each frame, whereas the other two phases are repeated until the end of the frame. In a multi-hop network, a node may be required to serve flows on multiple routes. Combining the slot selection for the reservation period with neighbor discovery limits the node's ability to serve these flows on multiple routes in the same frame. Further, since the scheduling of traffic slots depends on the order of reservation slots, nodes with higher reservation-slot indices may not get enough traffic slots to transmit their packets, which can degrade their throughput and impact the fairness of the network [6], [22].

A master-slave like approach is used in [7], [8] in which nodes transmit control packets in the contention-period (random access phase) to compete for the conflict-free data traffic slots. The successful nodes become master nodes which then control the communication of the unsuccessful slave nodes in the conflict-free period. As mentioned before, the random access approaches suffer from the deafness and collision problems [19], which make these schemes unsuitable for multihop topologies where the number of hidden nodes (hence, deafness problem) increase significantly. Moreover, the above-mentioned TDMA schemes (i.e., [7], [8], [13], [14], [18], [22]) do not provide QoS support.

To obtain the QoS-aware conflict-free schedules, many TDMA-based schemes (both centralized [23], [24] and distributed [6], [20], [21], [25], [26]) use the *graph coloring* techniques, which require each node to transmit its data traffic demand and each neighbor node to receive (or overhear) that packet. In centralized schemes, the information gathered at the central node (which can also act as a single point of failure) easily becomes obsolete when the topology changes, whereas a large *notification overhead*[1] and delay are incurred in distributed schemes since each node periodically retransmits its local schedule until a globally converged (or a feasible) schedule is obtained [19]. Therefore, none of them provide a *real-time* solution (i.e., compute a conflict-free schedule instantly or with very low delay), when the network topology and data rate, routing table, flow priority, etc., change dynamically [19].

To address the *notification overhead* problem, schemes in [27], [28] calculate the *rank matrix* at each node in the single-hop network, by using the node ids in the hash function. Based on this rank matrix, a conflict-free schedule for the reservation period is calculated at each node in real-time. However, [27], [28] are not suitable for a multi-hop topology because the neighboring nodes can generate contradicting rank matrices, when their 1-hop neighbors are different [19]. To resolve this issue and avoid conflict, each node in [29] informs its 1-hop neighbors about its selected reservation slots, which introduces significant overhead and leads to wastage of slots. Further, the length of reservation as well as other control periods of an n-node network in these rank-based MAC schemes is of the order of $O(n)$, which results in a large control overhead [19].

The above-mentioned directional TDMA schemes cannot adapt in real-time to the variations in link rates and topology changes as new flows (or nodes) are admitted or revoked. They also introduce a large overhead and delay, which increase with network size, making them unsuitable for multihop topology. Moreover, these schemes incur a large overhead for supporting the QoS requirements.

B. CONTRIBUTIONS OF OUR PROPOSED REAL-TIME, DISTRIBUTED, DIRECTIONAL TDMA SCHEME

To the best of our knowledge, no real-time, distributed, directional TDMA scheme exists in literature, which can provide a conflict-free schedule for a dynamic, multi-hop network.

In this paper, *we propose a novel, distributed, pure directional TDMA MAC scheme for multi-hop networks, which adapts to the topology changes and/or flow requirements in real-time, and facilitates QoS-aware communication with no notification overhead*. Here, pure directional means that no omnidirectional antenna is used at the nodes. Like many

[1] Each directional node needs to transmit its data traffic slot schedule to its 1-hop neighbor nodes so that they can detect conflict in their schedules with this node [17]. This process is repeated until the conflict at each node is resolved [19], which introduces a large notification overhead and delay.

other TDMA schemes (such as [7], [10], [25], [30]–[32]), our proposed TDMA MAC scheme assumes the knowledge of 2-hop neighborhood which can be obtained using a gossip-based neighbor discovery scheme, such as [33], [34]. Note that, neighbor discovery is an essential part of directional MAC schemes since the node cannot transmit its packets omnidirectionally [33], [34]. Further discussion on neighbor discovery is out-of-scope of this paper.

The **main contributions** of our scheme are as follows:

1. A Low-Complexity Rank-based Scheme for Multi-hop Topology: As discussed in Section I-A, the rank-based scheduling schemes are not suitable for a multi-hop network as the contradicting ranks can be generated for nodes. In our scheme, the 1-hop neighborhood of each node is divided into *fully connected 1-hop neighborhoods*, where every node is in the 1-hop transmission range of all other nodes. Each node then independently generates a rank matrix for each of its *fully connected 1-hop neighborhood*. It refers to one of these rank matrices in each slot of the Hello period (which is partially analogous to the reservation period mentioned above), and chooses an action (transmits or listens to a neighbor).

2. Real-time and Fully-Distributed Scheduling Scheme: Since every node independently generates its rank matrix for each of its *fully connected 1-hop neighborhood*, our proposed rank-based scheme is fully-distributed. These rank matrices are non-contradicting (see Section II for details); therefore, the nodes are not required to notify their rank matrices to their neighbors to resolve the scheduling conflict. As a result, a scheduling solution is obtained in real-time as the *notification overhead* and the resulting delay are completely eliminated.[2] Each node in our scheme can easily detect changes in the network topology, by using its updated 2-hop neighborhood information (obtained via neighbor discovery mechanism) and recalculate its *fully connected 1-hop neighborhood(s)* in real-time, which makes our scheme suitable for a dynamic network topology.

3. Low Control Overhead: Unlike traditional rank-based schemes, (e.g., [27], [28]), where the length of reservation and other control periods of an n-node network is of the order of $O(n)$ [19], our proposed scheme depends only on the number of nodes in a *fully connected 1-hop neighborhood*. Therefore, the rank matrix computed in our scheme is smaller and requires fewer reservation and other control slots, which significantly reduces the control overhead and delay.

4. Real-time Adaptation to Dynamic QoS Requirements: With the flexibility to choose from multiple rank matrices, a node can select the link (i.e., next-hop node) it wants to serve in the current frame based on the QoS-metric value of the packets stored in its buffer. This allows a node to easily *adapt to the dynamic QoS requirements in real-time*.

[2]In our scheme, nodes broadcast their 1-hop neighborhood information during neighbor discovery so that fully connected 1-hop neighborhoods can be constructed at each node. Note that every directional TDMA scheme available in literature employs the neighbor discovery for multi-hop and/or mobile topology. Since nodes, in our scheme, do not broadcast their rank matrices and schedule, the notification overhead is completely prevented.

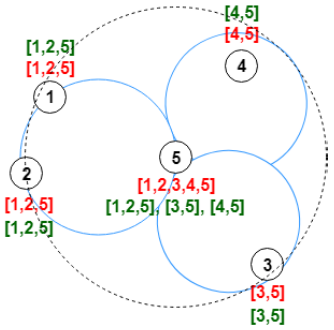


FIGURE 1. A 2-hop network topology with randomly placed nodes 1 to 5. The dotted black circle represents the communication range of node 5, which includes all its 1-hop neighbors. The blue circles represent three fully connected 1-hop neighborhoods of node 5. The rank matrix obtained at each node with the traditional rank-based scheduling scheme, such as [27], [28], is shown in red color, whereas the rank matrices in green color are obtained using our proposed scheme.

5. Improved Slot Utilization: A new (but optional) REQ (*requisition*) period is added to notify the intended receiver about the updated slot requirement. This period reduces slot wastage and increases channel utilization, and thereby, network throughput at hotspot[3] nodes.

6. A Fair Slot Allocation Mechanism: A *throughput scaling* mechanism is added which increases fairness by accommodating all traffic requests regardless of the order of their arrival, and therefore, helps in congestion management. Note that this is a major drawback in many distributed schemes, such as [13], [14], [18].

7. Improved Spatial Reuse: An optional *piggyback reservation* mechanism is added which further increases the spatial reuse of our scheme by enabling an intermediate transmitter (or receiver) node to accommodate multiple flows in a frame.

Paper Organization: This paper is organized in six sections. We discuss our distributed rank-based scheduling scheme for multi-hop network topology in Section II. Different control periods and mechanisms used in our scheme are discussed in Section III. The working principle of our proposed scheme is explained in Section IV. Simulation results and comparison analysis are discussed in Section V, followed by conclusions in Section VI.

II. DESIGN OF A DISTRIBUTED RANK-BASED SCHEDULING SCHEME FOR MULTI-HOP TOPOLOGY

In rank-based schemes, every node independently calculates the hash value (i.e., rank) for each of its 1-hop neighbor node [19]. The reservation slots are then allocated to the nodes in the descending order of their ranks. These schemes provide a solution for 1-hop network topologies where every node has a direct link to all other nodes. However, *they fail in a multi-hop topology, where some nodes have different 1-hop neighbor nodes because a node can have a different rank at each of its 1-hop neighbor nodes* [19]. For example, Fig. 1 shows a 5-node, 2-hop network where node 5 has all the four nodes (nodes 1, 2, 3, and 4) in its 1-hop neighborhood, whereas nodes 3 and 4 are not in the 1-hop neighborhood of nodes 1

[3]A hotspot node serves more than one flow.

Algorithm 1 Pseudocode for construction of fully connected 1-hop neighborhoods

Input: node x and its 1-hop neighborhood $N^1(x)$
Initialize a global variable FC1HN which stores fully connected 1-hop neighborhoods of node x
stackList = []
CalculateNeighborhood($N^1(x)$, x, stackList) //*This function calculates fully connected neighborhoods*
Remove subsets from FC1HN //*For example, if FC1HN contains [1,2,3,4] and [1,2,3], remove [1,2,3]*
Output: FC1HN contains length(FC1HN) unique fully connected 1-hop neighborhood(s)

CalculateNeighborhood(CurrList, CurrNode, Elements)

```
while CurrList do
    if length(CurrList) == 1 then
        Append([CurrList, CurrNode, Elements])
        break //Come out of while loop
    else
        if CurrNode is not in Elements then
            tempE = [Elements, CurrNode]
        else
            tempE = Elements
        tempCurrNode = CurrList.pop(0) //First element
        tempCurrList=CurrList ∩ N^1(tempCurrNode)
        DiffCurrList = CurrList - tempCurrList
        if length(DiffCurrList) == 0 then
            if length(tempCurrList) == 1 then
                Append([tempCurrList, tempCurrNode, tempE])
            else
                CalculateNeighborhood(tempCurrList, tempCurrNode, tempE)
                break //Come out of while loop
        else
            for each v in DiffCurrList do
                Append([v, tempE])
            if length(tempCurrList) == 0 then
                Append([tempCurrNode, tempE])
            else
                CalculateNeighborhood(tempCurrList, tempCurrNode, tempE)
//End of CalculateNeighborhood function
```

Append(newNeighborhood)

```
newNeighborhood = sort(elements of newNeighborhood in the increasing order of node id)
if newNeighborhood is not in FC1HN then
    FC1HN.append(newNeighborhood)
//End of Append function
```

and 2. Here, nodes 3 and 4 have only node 5 in their respective 1-hop neighborhood. The rank matrix at each node can be obtained (shown in red color in Fig. 1) by using any rank-based scheduling scheme available in literature (e.g., [27], [28]). Here, node id is used as the hash value for simplicity, where a smaller value represents a higher rank. Every node transmits its packet in a unique time slot based on its rank. For example, as per the rank matrix at node 2 ([1,2,5]), the nodes 1, 2, and 5 transmit in slots 1^{st}, 2^{nd}, and 3^{rd}, in that order. Here, node 2 receives a packet from node 5 in its 3^{rd} slot. However, according to the rank matrix at node 5 ([1,2,3,4,5]), it transmits to node 2 in its 5^{th} slot. As a result, nodes 2 and 5 do not steer their beams towards each other simultaneously and therefore cannot communicate with each other.

For such random and/or multi-hop topology, *the existing rank-based schemes (e.g.,* [27], [28]*) require local convergence to obtain conflict-free rank matrices at neighboring nodes*. For example, after local convergence, the rank matrices at nodes 1 to 5 in Fig. 1 would be [1,2,3,4,5], which allow node 2 to receive a packet from node 5 in the 5^{th} slot. However, the local convergence in a multi-hop and mobile network topology would introduce an overhead and delay as each node must repeatedly notify its 1-hop neighbors about its conflicting slots until the conflict is resolved. Furthermore, the number of slots used during the reservation period are of the order of $O(n)$ in a network of n nodes, which results in a larger control period [19].

Our distributed rank-based scheduling scheme for multi-hop topology is discussed below.

A. CONSTRUCTION OF FULLY CONNECTED 1-HOP NEIGHBORHOODS

To address this issue, the 1-hop neighborhood at each node is divided into *fully connected 1-hop neighborhoods* in our scheme. Using the available 2-hop neighbor information, each node identifies the nodes in each of its *fully connected 1-hop neighborhood*, where every neighbor node forms a direct link with all other nodes. For example, since node 5 in Fig. 1 has 1-hop neighbor information of nodes 1, 2, 3, and 4, it forms three *fully connected 1-hop neighborhoods* of rank matrices ([1,2,5], [3,5], and [4,5]) to cover its entire 1-hop neighborhood (shown in green color in Fig. 1). If node 5 wants to transmit to node 2, it knows from its 1^{st} rank matrix ([1,2,5]) that node 2 will steer its beam toward node 5 in the 3^{rd} slot, and hence, resolves the conflict between nodes 2 and 5. Moreover, using its rank matrices, node 5 chooses between nodes 1, 3, and 4 in the 1^{st} slot based on the available route information and/or flow or link priorities (see Section III-A2 for slot selection process). Similarly, it decides whether to listen to node 2 or transmit to node 3 or 4 in the 2^{nd} slot. *The pseudocode to obtain fully connected 1-hop neighborhoods is given in Algorithm 1.*

B. COMPUTING NODE's RANK MATRIX

For each of its fully connected 1-hop neighborhoods, a node constructs its rank matrix as follows:

Step 1: Calculate the hash value for node i of the j^{th} fully connected 1-hop neighborhood (FC1HN) of node x as,

$$Rank(i) = MD5(node\ id(i)),\ where,\\ i \in FC1HN_j(x)\ and\ \cup_j FC1HN_j = N^1(x). \quad (1)$$

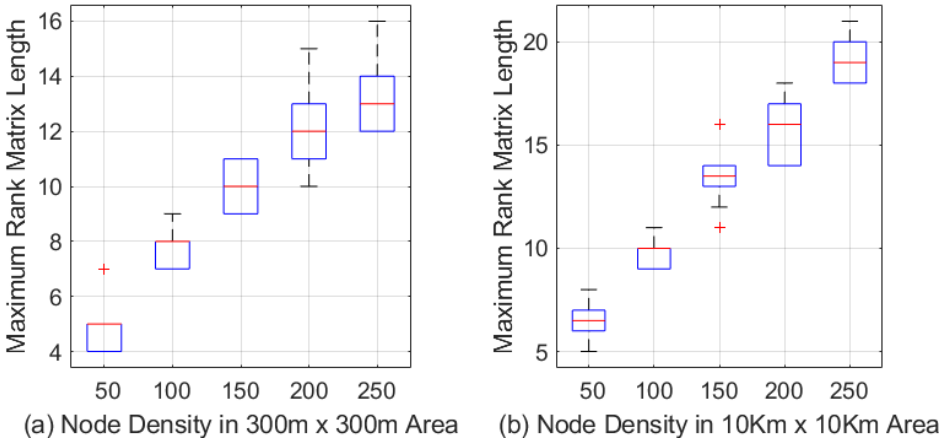


FIGURE 2. Maximum rank matrix length (*k*) in our proposed scheme at varying network sizes in (*i*) 300 m × 300 m and (*ii*) 10 km × 10 km network area. Note that each control period in our scheme has *k* slots.

Here, $N^1(x)$ is the set of 1-hop neighborhoods of node x.

Step 2: Arrange the nodes of a fully connected 1-hop neighborhood in their decreasing order of ranks, Rank(i).

Note that the schemes in [27], [28] include the timestamp as a random seed in their hash function so that the transmitter node with a higher reservation-slot index (i.e., low rank) can get a fair chance to schedule its data traffic slots. Instead, each node in our scheme uses the *throughput scaling* mechanism (discussed in Section III-B), which allows a receiver node to fairly distribute its data traffic slots among all of its transmitter nodes regardless of their ranks (i.e., reservation-slot indices).

The rank matrix in traditional rank-based schemes depends on the network size (n) (i.e., order of $O(n)$ [19]), because each transmitter node needs unique slots for the reservation and other control periods. However, the size of rank matrix in our proposed scheme depends only on the number of nodes in a *fully connected 1-hop neighborhood*. Therefore, *the rank matrix computed in our scheme is much smaller (see an example below) and requires fewer reservation and other control period slots, which significantly reduces the protocol overhead and delay for a multi-hop network.*

Example: The rank matrix lengths in our scheme for varying network sizes and two different network topologies are shown in Fig. 2. Here, nodes are randomly placed and each experiment is repeated 10 times. Note that our scheme requires only 13 and 19 slots for a 250-node network in Fig. 2a and 2b, respectively, which is significantly less than $O(250)$ slots required in traditional rank-based schemes [19].

III. DIFFERENT CONTROL PERIODS AND MECHANISMS IN OUR PROPOSED TDMA SCHEME

In our proposed TDMA scheme (see Fig. 3), a frame is divided into three to five periods Hello, REQ (Requisition), reservation, piggyback reservation (PR), and data traffic. The first four periods use control packets to schedule conflict-free traffic slots, whereas data is transmitted during the data traffic period. Note that only two control periods (i.e., Hello and Reservation) are mandatory in our scheme, similar to the existing TDMA schemes. The remaining two control periods (i.e., REQ and PR) are optional. A new REQ period is introduced in our scheme to reduce the slot wastage and increase link throughput. Using the PR period increases spatial reuse at a node. Using a *throughput scaling* mechanism after the Hello and REQ periods increases fairness and introduces the adaptive slot scheduling capability, which helps in congestion management. These periods along with their packet structure are described below.

A. HELLO PERIOD

The Hello period reserves a conflict-free handshake slot in which the transmitter and receiver pair can negotiate and schedule data traffic slots. During this period, only the transmitter node sends Hello packet which includes the number of its data packets (called *desired throughput*) to be sent in the current frame. The Hello packet structure (shown in Fig. 4) includes the node id (i.e., MAC address) of the transmitter and receiver nodes, beam id of transmitter node, timestamp to store this packet's origination time, slot index in which this Hello packet is transmitted, and whether the transmitter node wants to request the use of the PR period in the next frame to negotiate data traffic slots. Note that the use of *Timestamp* and *Hello slot index* fields help in node synchronization.

During the Hello period, each node decides its action (whether to transmit or listen) in each Hello slot, based on the available route information and QoS-metric value, as discussed below. In a multi-hop topology, several complex situations can arise, which are described below with the help of an example topology shown in Fig. 1.

1) COMPUTING THE QoS METRIC

Similar to the cross-layer schemes presented in [24], [30]–[32], our scheme also assumes that each node has the route information, such as previous and next hops of the route, remaining hop count and flow priority for the routes passing

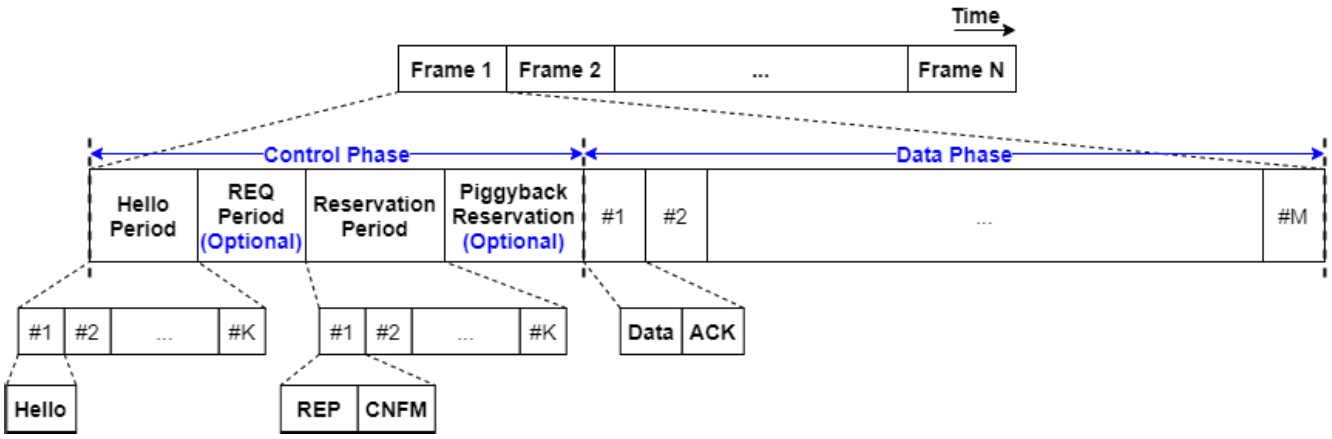


FIGURE 3. The frame structure used in our proposed TDMA scheme. Note: Hello, REQ, and both Reservation periods use *K* slots each, where *K* is the maximum number of nodes present in a *fully connected 1-hop neighborhood*. Here, the REQ and PR periods are optional as discussed in Section III.

through it, which can be provided by both proactive and reactive routing schemes.

If a node has packets for multiple receiver nodes in its buffer, it selects the receiver node based on a QoS-metric which is explained below. In our proposed scheme, the node calculates the QoS value for a packet i in its queue as:

$$(QoS_{value})_i = \frac{(Priority)_i * (Remaining\ Hop\ Count)_i}{(TTE)_i} \quad (2)$$

Here, the *Priority* represents the priority value of a flow that the packet belongs to. The *Remaining Hop Count* represents the number of remaining hops the packet needs to traverse on a given route in order to reach the destination. *TTE* is the time to expiry which represents the remaining packet lifetime based on its time-to-live (TTL). The QoS metric allows a node to prioritize the packets, which belong to a high priority flow, are relatively far from the destination, and have a small TTE, so that they can reach the destination before their expiry.

All the packets in the node queue are arranged in the decreasing order of their QoS value. Once a packet with the highest QoS value is selected, the node checks its receiver node id, and transmits the Hello packet to that receiver node during the corresponding Hello slot.

2) POSSIBLE SITUATIONS IN EACH HELLO SLOT

The following three situations are possible:

(a) When a node X forwards data packets to multiple receiver nodes in different fully connected 1-hop neighborhoods: Consider the rank matrices of node 5 in Fig. 1 ([1,2,5], [3,5], [4,5]), where node 5 has data packets for nodes 3 and 4 which are in two different *fully connected 1-hop neighborhoods*. Since node 5 has the same rank in both of these rank matrices, it can transmit a Hello packet to either node 3 or node 4 in the 2^{nd} Hello slot in a frame. Here, node 5 uses the QoS metric to select between nodes 3 and 4 and sends a Hello packet accordingly. On the other hand, when a node has different ranks in its *fully connected 1-hop neighborhoods*, it uses each of the Hello slot (corresponding to its rank) to transmit the Hello packet towards the corresponding receiver node. For example, if node 5 has data packets for nodes 1 and 3, it transmits Hello packets towards nodes 3 and 1 in the 2^{nd} and 3^{rd} Hello slots, respectively, based on its rank matrices.

(b) When a node X forwards data packets to multiple receiver nodes in the same fully connected 1-hop neighborhood: Consider the rank matrix [1,2,5] of node 5 in Fig. 1. Here, nodes 1 and 2 are in the same *fully connected 1-hop neighborhood* of node 5. Hence, if node 5 has data packets for both nodes 1 and 2, it can transmit a Hello packet to either of these nodes in the 3^{rd} Hello slot.

(c) When a node X does not have data packets to forward and chooses to listen: If node X is not a part of any route, it stays idle. However, if it belongs to multiple routes, it listens to that node for which it is the next hop. If there are more than one such nodes, node X listens to the node it did not listen to in previous frame(s).

B. THROUGHPUT SCALING

During the Hello period, a node can receive/transmit multiple Hello packets, each with varying *desired throughput* requirement. The node cannot serve all the requests if the sum of all *desired throughput* requests exceeds the number of available data traffic slots. Traditional TDMA-based schemes schedule the slots in the first-come first-served (FCFS) order which can not only result in increased queuing delay but also starve one or more nodes. To address this, each node, in our scheme, scales down the individual data requirement as:

$$(Scaled\ Desired\ Throughput)_i = \left\lfloor \frac{(Desired\ Throughput)_i * (Total\ Data\ Slots)}{\sum_{j=1}^{k}(Desired\ Throughput)_j} \right\rfloor \quad (3)$$

Here, k is the total number of Hello packets transmitted and received by a node in the Hello period, and i corresponds to the i^{th} Hello packet, such that $i \in [1, k]$. The denominator is the sum of data traffic slots requested in the individual (i.e., j^{th}) Hello packet.

In Fig. 1, node 1 (rank matrix [1,2,5]) transmits a Hello packet to node 2 in the 1^{st} Hello slot with a request of S_1 data

Tx node MAC address (48 bits)	Tx node beam Id (3 bits)	Rx node MAC address (48 bits)	Time stamp (32 bits)	Hello slot index (5 bits)	Desired throughput (7 bits)	Enable redundant slot? (1 bit)

FIGURE 4. 18 byte Hello packet structure. Note: the REQ packet is similar to the Hello packet except that it does not contain the *Enable Redundant Slot?* field.

Tx node MAC address (48 bits)	Tx node beam Id (3 bits)	Rx node MAC address (48 bits)	Time stamp (32 bits)	Scaled desired throughput (7 bits)	Traffic slots (100 bits)	Enable redundant slot? (1 bit)	Redundant slot request accepted? (1 bit)

FIGURE 5. 30 byte REP packet structure. The CNFM packet has the same structure, except the *Enable Redundant Slot?* field.

traffic slots and also receives a Hello packet from node 5 in the 3^{rd} Hello slot with a request of S_2 data traffic slots. At the end of the Hello period, node 1 has the total desired throughput of $S_1 + S_2$ slots. If node 1 has d_1 data slots available and if $S_1 + S_2 > d_1$, then the accepted data slots (at node 1) for nodes 2 and 5 are $S_1 * d_1/(S_1 + S_2)$ and $S_2 * d_1/(S_1 + S_2)$, respectively. Hence, *throughput scaling* allows a node to serve multiple links (and, hence, routes) within a frame by fairly distributing the data traffic slots among all requests, and helps in congestion management by reducing queue overflow.

C. REQ PERIOD (Optional)

After *throughput scaling*, the *desired throughput* of a transmitter node can decrease. If the transmitter node fails to tell its updated requirement to its receiver, the receiver node would waste its data traffic slots. Hence, a separate REQ period is used after the Hello period which has the same number of slots as the Hello period. A node transmits an REQ packet towards its receiver only when it has an updated *desired throughput*. Note that the REQ and Hello slot indices are the same. In the example given in Section III-B, node 1 transmits REQ packet to node 2 in its 1^{st} REQ slot with $S_{1,updated}$ *desired throughput*. The structure of REQ packet is identical to the Hello packet except that it does not have the *Enabled Redundant Slot?* field (see Fig. 4).

D. RESERVATION PERIOD

The reservation period is used to schedule conflict-free data traffic slots in the data traffic period. Here, each slot is divided into two sub-slots, namely REP (reply) and CNFM (confirm). The receiver nodes reply to the received Hello packet, in the first sub-slot, with an REP packet (shown in Fig. 5), which includes the transmitter and receiver node id (i.e., MAC address), beam id of the transmitter node, packet generation time, scaled throughput, and available data traffic slots at the receiver. Since the number of data traffic slots assumed in our scheme is 100, the *Traffic Slots* field in the REP packet is of 100 bits where each bit corresponds to a data traffic slot. The other two fields *Enable Redundant Slot?* and *Redundant Slot Request Accepted?* are used to enable the PR period as discussed in Section III-E.

In the second sub-slot, the transmitter node replies with a CNFM packet which has a packet structure similar to REP, except that it does not contain the *Enable Redundant Slot?* field. The CNFM packet includes the accepted data traffic slots (i.e., corresponding bit in the *Traffic Slots* field is set to 1). Transmitting the REP and CNFM packets reduce interference at the receiver node by preventing the receiver node's 1-hop neighbors, which are exposed to the communication of this transmitter-receiver pair, from scheduling the same data traffic slots. If a transmitter node does not receive a response (i.e., REP packet) from its receiver during the reservation period, it can retransmit its Hello packet in the next frame.

E. PIGGYBACK RESERVATION PERIOD (Optional)

Unlike the reservation period where a receiver node responds to the transmitter node, the piggyback reservation (PR) period represents the receiver- or transmitter-initiated communication. As discussed in Section III-A2(a), when a node X has the same rank in its two or more rank matrices, it can select only one of them in a frame. If this node is unable to schedule all its data traffic slots with the selected node, its unscheduled data traffic slots can be scheduled for communication with another node on a different link by using the piggyback reservation. Similar to the reservation period, each slot in the PR period is divided into REP and CNFM. During the PR period, a node *X* uses the REP packet to notify its unscheduled data traffic slots to the nodes on its other link(s).

However, since nodes use a directional antenna, the intended receiver of this REP packet must know when to steer in the direction of node *X* in order to receive the REP packet and schedule the data traffic slots. This is enabled by using the *Enabled Redundant Slot?* field in the Hello packet where the transmitter node requests the receiver node to use the PR period in the following frame so that it can transmit its Hello packet on a different link in the next frame, and thereby, accommodate both flows. Note that the receiver node can also request to use the PR period by setting the *Enabled Redundant Slot?* to 1 in its REP packet while responding to the transmitter in the reservation period of the current frame. In both cases, the other node must agree on using the PR period in the next frame by setting its *Redundant Slot Request Accepted?* field to 1 in the current frame. Thus, the receiver and transmitter nodes give their consent on using the PR period via their REP or CNFM packet, respectively.

For using the PR period, the transmitter and receiver nodes store the reservation slot number of the current frame. In the next frame, the receiver node transmits the REP packet with

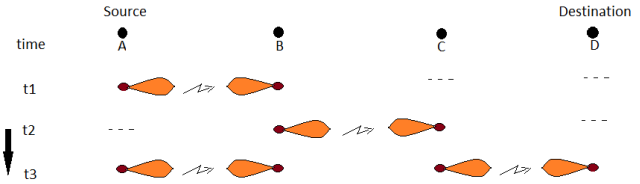


FIGURE 6. Working of a directional TDMA scheme for a 3-hop topology, where source node A transmits packets to destination node D over the 3-hop route A-B-C-D. In time slot t1, node A forwards its packet to intermediate node B. In t2 time slot, node B steers its beam towards node C and forwards the packet. In time slot t3, spatial reuse is achieved as links A-B and C-D communicate simultaneously.

its unscheduled data traffic slots and 0 value for the *Scaled Desired Throughput* towards the intended transmitter, in the corresponding slot of the PR period. The transmitter node then selects the common available data traffic slots, updates corresponding bits in the *Traffic Slots* field, and transmits the CNFM packet. By this way, the transmitter and/or receiver node serves more than one rank matrices in which it has the same rank.

The use of PR period is optional since it is required only in the following three cases: (*i*) At an intermediate node of a multi-hop route, (*ii*) When a node has data packets for its multiple 1-hop receiver nodes in the same or different rank matrices, and (*iii*) When a node receives data packets from its multiple 1-hop transmitter nodes of different rank matrices. Further, it is useful for light and moderate traffics, where a node can schedule its unused slots to non-conflicting flows. The working of PR period is explained in Appendix.

F. DATA TRAFFIC PERIOD

Each slot in this period is divided in two sub-slots - data and ACK (acknowledgment). Transmitter node transmits data packet in the first sub-slot as per the negotiated schedule, and receiver node responds with ACK packet in the second sub-slot. The frame finishes with the end of the data traffic period and all nodes recalculate their *fully connected 1-hop neighborhoods* for the next frame.

IV. WORKING PRINCIPLE OF OUR PROPOSED DISTRIBUTED, DIRECTIONAL TDMA SCHEME

The working of a directional TDMA-based MAC scheme for a 3-hop route is shown in Fig. 6. *In our proposed TDMA scheme, a frame (see Fig. 3) is divided into three periods (Hello, reservation period and data traffic period).* Here, the first two periods represent the control period of the protocol. Our scheme also adds two new optional and conditional REQ and PR control periods as discussed in Section III. The number of control time slots depends on the maximum number of nodes in a *fully connected 1-hop neighborhood*.

At the beginning of a frame, every node calculates the rank matrix for each of its *fully connected 1-hop neighborhood*. In each **Hello Period slot**, a node selects a node id from its rank matrices, towards which it steers its beam to either transmit or receive the Hello packet. This allows the node to reserve a conflict-free slot for the transmitter and receiver pair to negotiate the scheduling of data traffic slots. Note that this order of the selected node id (obtained during the Hello period) also remains the same in the REQ and both reservation periods. If a node decides to transmit the Hello packet in a slot, it updates the *desired throughput* field with the count of data packets it has in its queue for that receiver (see Section III-A for more details).

At the end of the Hello Period, if a transmitter can no longer utilize the total number of reserved traffic slots, it sends an **REQ packet** to its receiver node(s) with the updated *desired throughput*. Using an REQ packet, therefore, prevents the wastage of unused data traffic slots at a node. If the total requested data traffic slots exceed its available data traffic slots, the node uses *throughput scaling* to suitably scale down the requested *desired throughputs* as discussed in Section III-B. The throughput scaling mechanism improves the network *fairness* and helps in congestion management.

Upon receiving a Hello packet, the receiver node records the transmitter node id and the *desired throughput*, sends a REP packet during the **Reservation Period** with a list of its own available data traffic slots, and waits for the CNFM packet from transmitter during the same reservation slot. The transmitter node accepts the common slots and sends a CNFM packet to the receiver node. Since a collision occurs when a receiver node receives more than one signal at the same time, transmitting the REP and CNFM packets ensure that the nodes, exposed to the communication of this transmitter-receiver pair, are prevented from scheduling the same data traffic slots. Finally, data packets are transmitted in the **Data Traffic Period**.

The pseudocode of our proposed scheme is given in Algorithm 2 and an example to explain its working is given in Appendix. Although the introduction of the optional REQ and PR periods can slightly increase the frame length, the channel utilization, spatial reuse, and link fairness increase considerably. Moreover, it allows nodes to receive and forward packets in the same frame which reduces the end-to-end delay. The control overhead and running time metrics of our scheme are compared with other recent, fast, distributed TDMA schemes in Section V-B.

V. SIMULATION RESULTS AND PERFORMANCE COMPARISON

The performance of our proposed rank-based TDMA scheme is evaluated in Section V-A for real-time traffic flows over multi-hop routes by using different data rates, TTL values, and QoS metric. Then the control overhead of our scheme is compared with recent, fast, distributed TDMA schemes proposed in [30]–[32] in Section V-B, followed by the analysis of their performance comparison for different static and mobile scenarios in Section V-C.

A. PERFORMANCE OF THE PROPOSED TDMA PROTOCOL

The simulation setup is discussed below, followed by a discussion on the maximum achievable flow throughput and then the performance of our scheme is evaluated for different experiments.

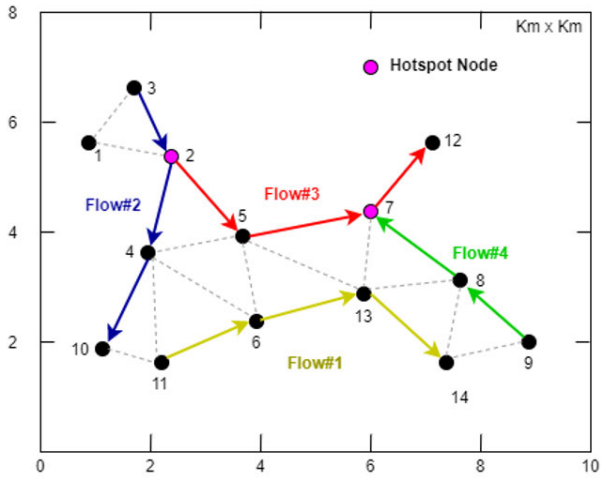


FIGURE 7. Network topology where each node is equipped with a single-beam directional antenna. Tested routes are (*i*) 11 to 14 (Flow#1) (*ii*) 3 to 10 (Flow#2), (*iii*) 2 to 12 (Flow#3), and (*iv*) 9 to 7 (Flow#4). Nodes 2 and 7 are hotspot nodes, as they are serving more than one routes.

TABLE 1. Nodes of fig. 7 and their rank matrices in their 1-hop fully connected neighborhoods.

Node	Rank matrices for 1-hop fully connected neighborhoods at each node
1	[3,2,1]
2	[3,2,1], [4,5,2]
3	[3,2,1]
4	[4,5,2], [4,5,6], [4,6,11], [4,11,10]
5	[4,5,2], [4,5,6], [5,7,6,13]
6	[4,5,6], [4,6,11], [5,7,6,13]
7	[5,7,6,13], [7,8,13], [7,12]
8	[7,8,13], [8,14,9], [8,13,14]
9	[8,14,9]
10	[4,11,10]
11	[4,6,11], [4,11,10]
12	[7,12]
13	[5,7,6,13], [7,8,13], [8,13,14]
14	[8,13,14], [8,14,9]

TABLE 2. Duration of different periods of a frame and its sub-frames.

Control/Data Period	Length (in ms)
Hello period (of 10 slots)	0.403
REQ period (of 10 slots)	0.403
Reservation period (of 10 slots)	1.097
PR period (of 10 slots)	1.097
Data Traffic period (of 100 slots)	86.760
Frame length	89.760

1) SIMULATION SETUP

The simulations are run in MATLAB version R2017b for the network topology consisting of 14 nodes shown in Fig. 7. The rank matrix for each node is shown in Table 1. Each node is equipped with a directional antenna with a beam-width of 45^0 and 2 km transmission range. We assume an ideal beam with no side or back lobes. The network size is 10 km $\times$ 10 km, where nodes are randomly placed. The channel capacity is 10 Mbps. The number of data traffic slots in each frame is 100 and the data packet size is 1000 Bytes. The buffer size at each node is infinite. We assume that each node knows about its neighborhood and the route(s) passing through it.

We assume that a *fully connected 1-hop neighborhood* can have up to 10 nodes. Therefore, the number of slots in Hello, REQ, and both reservation periods is 10. These periods are calculated in (4) and (5), as shown at the bottom of the page, and their lengths are given in Table 2. The MAC control period overhead (which includes Hello, REQ, and both reservation periods) is just 3.34% in our scheme. For a unit decrease in the number of fully connected 1-hop neighbor nodes, the control period decreases linearly by 0.3 ms. The data traffic period length is calculated in (6), as shown at the bottom of the page, where the *Num_Hello_Slots* and *Num_Data_Slots* represent the number of Hello and data traffic slots in a frame, respectively. The *Hello_pkt*, *REP_pkt*, *CNFM_pkt*, *Data_pkt*, *ACK_pkt*, PLCP_Header, and *MAC_Header* represent the size of respective packets and headers. The slot length in the control and data traffic periods include maximum propagation delay (*Max_Prop_Delay*), which allows a packet to be received in the same slot by the nodes located at the coverage boundary of transmitter node. A node waits for a short inter-frame space (SIFS) duration of 10 μs before responding to the received packet.

We study the performance of our proposed MAC scheme for the network topology in Fig. 7, where the source nodes 11, 3, 2, and 9 generate packets for the destination nodes 14, 10, 12, and 7, respectively. The routes corresponding to these four source-destination

$$Hello\ Period\ Length = Num\_Hello\_Slots * \left[\frac{(Hello\_pkt + PLCP\_Header)}{Channel\ Rate} + Max\_Prop\_Delay\right] \tag{4}$$

$$Reservation\ Period\ Length = Num\_Hello\_Slots * \left[\frac{REP\_pkt + CNFM\_pkt + 2 * PLCP\_Header}{Channel\ Rate} + 2 * Max\_Prop\_Delay + SIFS\right] \tag{5}$$

$$Data\ Traffic\ Period\ Length = Num\_Data\_Slots * \left[\frac{Data\_pkt + MAC\_Header + PLCP\_Header + Ack\_pkt}{Channel\ Rate} + 2 * Max\_Prop\_Delay + SIFS\right] \tag{6}$$

Algorithm 2 Pseudocode of the proposed TDMA scheme for a frame *f* at node *i*

Global variables: Maximum number of nodes in a *fully connected 1-hop neighborhood* (K), total data slots D in a frame, a Kx1 vector *curr_PR_status* to store node id(s) with which node i agreed (in frame f-1) to communicate in PR period of current frame (i.e., frame f). If k^{th} PR slot of node i is free, its *curr_PR_status*[k] = 0.
//*Frame starts*
Step 1: Construction of rank matrices
At the beginning of the frame, node obtains its 2-hop neighborhood information via neighbor discovery
Create rank-matrix using Algorithm 1 and Section II-B
Step 2: Functionality in Hello period
Initialize Kx1 size vectors for *order, desired_throughput*, and *next_PR_status*
for each slot k in Hello period (i.e., $k \in$[1,K]) **do**
    Select a node id (n) of rank k from its rank matrices (as discussed in Section III-A2)
    order[k] = n //*use this order in subsequent subperiods*
    if node i is transmitter for link i-n **then**
        desired_throughput[k] = min(D, packets for node n stored in the buffer of node i)
        if node i wants to use PR period in next frame & *next_PR_status*[k] is 0 **then** //*see details in Section III-E*
            Set *Enable Redundant Slot?* field to 1
            next_PR_status[k] = n
        Transmit Hello packet
    else
        Receive Hello packet and store requested desired throughput in *desired_throughput*[k]. If textitEnabled Redundant Slot? field is 1, set *next_PR_status*[k] = n.
Step 3: Perform *throughput scaling* at **transmitter** nodes
if total requested *desired throughput* at node $i > D$ **then**
    perform *throughput scaling* (see Section III-B)
    is_REQ_required = True
Step 4: Functionality in REQ period
for each slot k in REQ period (i.e., $k \in$[1,K]) **do**
    if *is_REQ_required* & node i is transmitter for link i-*order*[k] **then**
        Transmit REQ packet towards receiver node *order*[k] with the updated *desired_throughput*[k]
    else if node i is receiver for link i-*order*[k] & it receives REQ packet **then**
        Update its *desired_throughput*[k]
Step 5: Perform *throughput scaling* at **receiver** nodes
Step 6: Functionality in Reservation period
for each slot k in reservation period (i.e., $k \in$[1,K]) **do**
    if node i received Hello packet in k^{th} Hello slot **then**
        Set bits corresponding to its available data slots to 0 in *Traffic Slots* field, and update *Scaled Desired Throughput*
        if *next_PR_status*[k] is not 0 **then**
            Set *Redundant Slot Request Accepted?* = 1
        else if wants to use PR period in next frame **then**
            Set *Enabled Redundant Slot?* field to 1
            next_PR_status = *order*[k]
        Transmit REP packet; receive CNFM packet
        Update its reserved data traffic slots
        if *Redundant Slot Request Accepted?* is 0 **then**
            next_PR_status[k] = 0
    else if node i sent Hello packet in k^{th} Hello slot **then**
        Wait for REP packet
        if no REP received **then**
            Retransmit Hello packet in next frame
        else
            Select up to *Scaled Desired Throughput* common available data slots and update *Traffic Slots* field
            if *Enabled Redundant Slot?* field is 1 & its *next_PR_status*[k] is 0 **then**
                Set *Redundant Slot Request Accepted?* = 1
            Transmit CNFM packet
Step 7: Functionality in PR period
for each slot k in PR period (i.e., $k \in$[1,K]) **do**
    if *curr_PR_status*[k] is not 0 **then**
        if node i is receiver on link i-*order*[k] **then**
            Set bits for available data slots to 0 in *Traffic Slots* field and *Scaled Desired Throughput* field to 0
            Transmit REP packet towards node *order*[k]
            Wait for CNFM packet
            Upon receiving CNFM packet, update the reserved data traffic slots
        else
            Wait for REP packet
            Upon receiving REP packet, select common available data slots, and update *Traffic Slots* and *Scaled Desired Throughput* fields
            Transmit CNFM packet
Step 8: Communicate during Data Traffic period
Step 9: *curr_PR_status=next_PR_status //for next frame*
//*Frame completes*

node pairs are 11-6-13-14 (a 3-hop route for Flow#1), 3-2-4-10 (a 3-hop route for Flow#2), 2-5-7-12 (a 3-hop route for Flow#3), and 9-8-7 (a 2-hop route for Flow#4). Here, we call nodes 2 and 7 as the *hotspot nodes* because they forward the data packets of more than one flows and therefore can experience congestion.

Performance Metrics: The following performance metrics are used in our simulation:

- Per node CUR is the fraction of time that a node either transmits or receives data packets over the total simulation time [13].
- PDR for a flow is the ratio of total packets received by the destination node over total packets generated at the source. Since PDR represents a normalized throughput,

the flow throughput can be computed as *PDR* x *Data Rate*.

- The end-to-end delay plot is shown only for Section V-A5 which includes a scenario when packet TTL is not used. Since all other experiments have a TTL value, their end-to-end delay plots are omitted to save space in the manuscript.

2) MAXIMUM ACHIEVABLE FLOW THROUGHPUT IN THE DIRECTIONAL TDMA SCHEME

The nodes use directional antenna and can share their data traffic slots among one or more active links, where an active link is a part of a route on which data is transmitted in each frame. For example, node 6 in Fig. 7 has two active links, 11-6 and 6-13. The throughput at a node decreases as the number of its active links increases. Therefore, the link throughput cannot exceed 50% of channel capacity in a multi-hop route because the node receives and forwards the data packets. In fact, after considering the control packet overhead, the maximum achievable link throughput for a flow on a multi-hop route would be less than 50% of the channel capacity. In our scheme, each frame (as shown in Table 2) consists of a control period and a data traffic period where the size of data packet payload, MAC header, ACK packet, and PLCP are 1000, 34, 14, and 24 bytes, respectively. Thus, the size of a 1000-byte data packet increases to 1058 bytes after adding the MAC and physical layer protocol headers (the higher layer protocol headers are ignored here). The receiver acknowledges a successful reception of data packet by sending a 14-byte ACK packet to the transmitter in the same data traffic slot.

As shown in Table 2, the frame consisting of 100 data traffic slots has a length of 89.76 ms. Since each data traffic slot can forward a packet carrying 1000-byte data payload, the maximum flow data rate of 8.9 Mbps can be supported on a 10 Mbps channel when the source and destination are 1-hop away. For a latency-constrained streaming application over a $\geq$ 2-hop path, the maximum flow data rate (when using directional communication shown in Fig. 6) would decrease to half of 8.9 Mbps (4.45 Mbps). Note that a higher flow data rate over a multi-hop path would further decrease the maximum achievable data rate due to congestion and TTL-based packet expiry.

3) CHANNEL UTILIZATION FOR DIFFERENT DATA RATES

The maximum per node CUR is 0.97 for the simulation setup considered in Section V-A1. The hotspot node 2 is an intermediate node of Flow#2 and the source of Flow#3 (see Fig. 7). Since it uses all its data slots at 3 to 5 Mbps data rates to accommodate both flows, its CUR is maximum in Fig. 8. Note that it allocates the same number of data slots to its downstream nodes 4 (for Flow#2) and 5 (for Flow#3), when QoS metric is not used. Therefore, intermediate nodes 4 and 5 also have a constant CUR in Fig. 8. Since the number of data slots required by a node increases with traffic density, CUR values of the remaining nodes in Fig. 8 increase with the data rate.

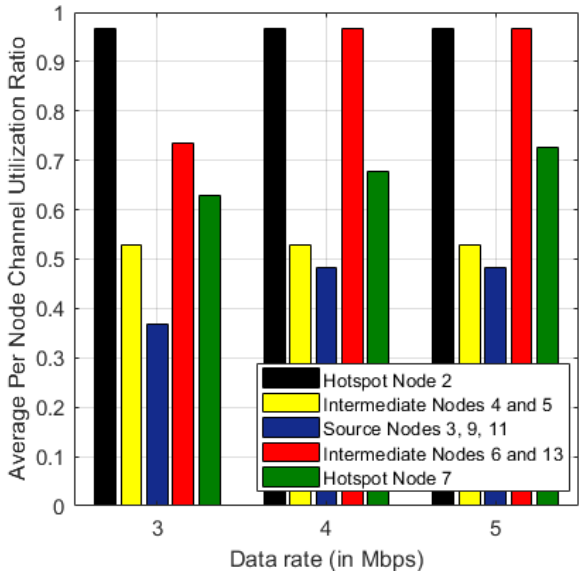


FIGURE 8. Average per node channel utilization ratio in the proposed TDMA scheme at different data rates, when the QoS metric is not used.

Source nodes 3, 9 and 11 in Fig. 7 only forward their packets to the next-hop nodes, whereas intermediate nodes 2, 4 to 8 and 13 receive and forward the packets. Therefore, the CUR value is the lowest at the source nodes 3, 9 and 11 for all traffic densities in Fig. 8. Note that the congestion at hotspot node 2 increases with data rate. As a result, hotspot node 7 receives and forwards fewer packets of Flow#3. Since it also receives packets of Flow#4 (see Fig. 7), its CUR is higher than that of intermediate nodes 4 and 5 but lower than the CUR of intermediate nodes 6 and 13, in Fig. 8.

4) PERFORMANCE FOR DIFFERENT DATA RATES, AT TTL = 0.5 s

Here, the performance of our proposed TDMA scheme is evaluated for the four real-time flows (for network in Fig. 7) at a constant data rates of 2 to 5 Mbps per flow which represent a low, moderate, and heavy traffic load, at a data packet TTL value of 0.5 s. Since the QoS metric is not used, the packets of each flow are arranged and forwarded in the FIFO (first_in_first_out) order from the queue.

- Performance of Flow#1: Flow#1 uses an independent route with no hotspot node. The PDR of Flow#1 (see yellow bars in Fig. 9) is 100% for up to 4 Mbps data rates, which are less than the maximum achievable flow throughput of 4.45 Mbps (as discussed in Section V-A2). At 5 Mbps data rate, congestion builds at the source node 11, which leads to packet expiration due to an increased queuing delay, and the PDR of Flow#1 degrades to 79%.
- Performance of Flow#2 and Flow#3: These flows attain 100% PDR at the low data rate of 2 Mbps in Fig. 9 because their intermediate nodes do not experience congestion. Note that the flows serviced by a hotspot node experience a higher queuing delay, and therefore, have a higher packet drop due to TTL expiry. In addition, the

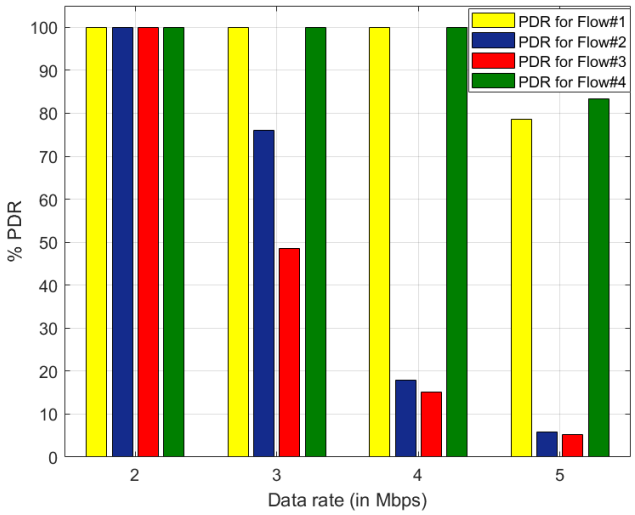


FIGURE 9. PDR performance of the proposed TDMA scheme at different data rates, when packet TTL is 0.5 s.

congestion experienced at a hotspot node increases with the data rate. Therefore, PDR is the highest in Flow#1, then in Flow#2, followed by Flow#3 at 3 to 5 Mbps data rates in Fig. 9 because they have zero, one and two hotspot nodes, respectively (see Fig. 7).

Since hotspot node 2 attains the maximum CUR at 3 Mbps data rate (see Fig. 8), an increase in the data rate further aggravates its congestion. Therefore, the PDR values of Flow#2 and Flow#3 considerably degrade when data rate increases from 3 Mbps to 5 Mbps in Fig. 9.

- Performance of Flow#4: Due to the high congestion-induced queuing delay experienced at hotspot node 2 at 3 to 5 Mbps data rates, fewer packets of Flow#3 reach intermediate node 7. As a result, intermediate node 8 of Flow#4 can reserve sufficient slots with the destination node 7 in each frame, and attains a 100% PDR for Flow#4.

  At 5 Mbps data rate, congestion builds up at the source node 9, which leads to packet drop due to TTL expiry, and thereby, reduces PDR of Flow#4 to 83%. Since Flow#1 is one hop longer than Flow#4, its end-to-end delay and, thereby, packets dropped due to TTL expiry are higher than that of Flow#4. Therefore, the PDR of Flow#1 is lower than PDR of Flow#4 at 5 Mbps data rate in Fig. 9.

5) PERFORMANCE FOR DIFFERENT TTL VALUES AT A 5 Mbps DATA RATE

Here, we evaluate the performance of our proposed TDMA scheme in the presence of traffic congestion for different flow latency, and show that a higher throughput is achieved when the flow can tolerate a higher latency.

We study the performance of our proposed TDMA scheme at 5 Mbps data rate (where all four flows experience congestion) for TTL values of 0.5 s, 1 s, 3 s, and when TTL is not used. Fig. 10(a) shows that PDRs of all the four flows increase with an increase in the TTL value because less packets expire due to congestion-induced queuing delay. As explained in the previous section, PDRs of Flow#2 and Flow#3 are lower than the PDRs of the other two flows due to congestion induced packet drops. Although the PDR of Flow#4 at node 7 is 100% for TTL $\geq$ 1 s and no TTL, the end-to-end delay is higher when TTL is not used (see green bars in Fig. 10(b)) because packets of Flow#3 are not dropped at nodes 2 and 5, due to which, node 7 receives more Flow#3 packets for forwarding them to node 12. As a result, node 7 schedules fewer data traffic slots with node 8, which increases the queuing delay of packets of Flow#4. Note that the end-to-end delay increases with simulation time when the data rate is high, causing the congestion. Since packets do not expire in the queues of source and/or intermediate nodes when TTL value is not used, all four flows achieve 100% PDR.

6) PERFORMANCE OF QoS-AWARE TDMA SCHEME

We evaluate the performance of our proposed TDMA scheme with the QoS metric at different data rates. While Flow#3 has a lower priority (Priority = 1), the remaining three flows have a higher priority (Priority = 2). The value of TTL for a packet is 0.5 s.

Since Flow#1 uses an independent route (i.e., it does not have any hotspot node), its PDR and end-to-end delay are the same in the QoS-aware and without-QoS MAC schemes (see yellow bars in Fig. 11 and 9, respectively). As Flow#2 has a higher priority (2x) than Flow#3, the hotspot node 2 forwards more packets of Flow#2, which increases the queuing delay of packets of Flow#3. As a result, PDR of Flow#2 (see blue bars in Fig. 11) in QoS-aware MAC scheme increases at the cost of Flow#3 (see red bars in Fig. 11). Although a higher queuing delay experienced by Flow#3 packets at node 2 decreases their TTE value, along with their higher value of remaining hop count (which is 3), the QoS metric for Flow#3 is still lower than that of Flow#2 (due to a higher priority of Flow#2). Here, the PDR of Flow#3 is low because its packets expire in the queues at the intermediate nodes 5 and 7. Since fewer packets of Flow#3 arrive at node 7, congestion does not build up at node 8. As a result, the PDR of Flow#4 remains the same at all data rates in the QoS-aware and without-QoS MAC schemes (see green bars in Fig. 11 and 9).

> ***Observation 1 (QoS Metric):*** *In the QoS-aware MAC scheme, PDR of higher QoS flow(s) increases at the expense of lower QoS flows. However, the use of the QoS metric does not impact the PDR of independent flows.*

7) ADVANTAGE OF PIGGYBACK RESERVATION PERIOD IN THE PROPOSED TDMA SCHEME

In this section, we study the impact of using the piggyback reservation on the performance of our proposed MAC scheme

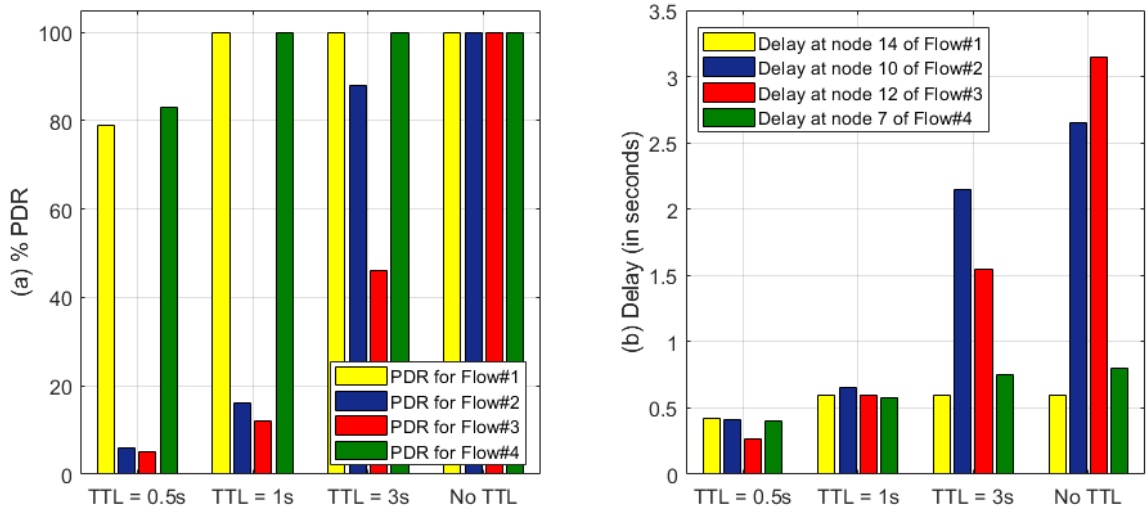


FIGURE 10. Impact of packet TTL value on (a) PDR and (b) end-to-end delay performances of the proposed TDMA scheme for heavy traffic (5 Mbps per flow).

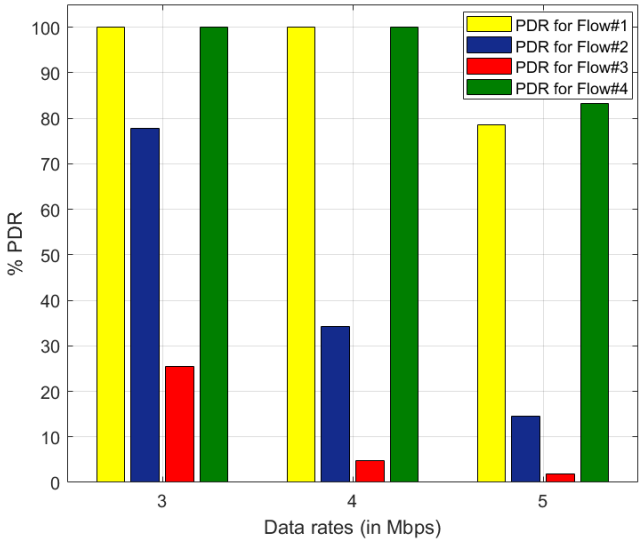


FIGURE 11. PDR at destination nodes for the QoS-aware proposed TDMA scheme at different data rates.

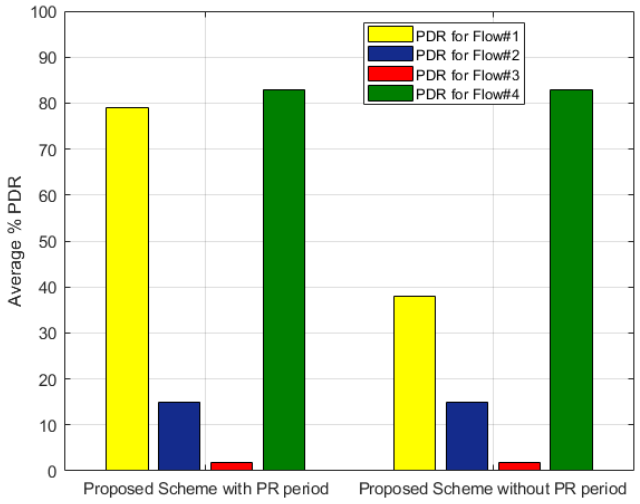


FIGURE 12. PDR performance of the proposed TDMA scheme with and without using Piggyback Reservation period, when data rate is 5 Mbps and TTL = 0.5 s.

for the network topology shown in Fig. 7. Here, the flow data rate is 5 Mbps and TTL is 0.5 s.

Node 6 in Fig. 7 is an intermediate node of Flow#1 and has rank matrices of [4,6,11] and [5,7,6,13] (see Table 2). To schedule data traffic slots with both previous and next hops of Flow#1 (i.e., nodes 11 and 13, respectively), it requires two unique conflict-free reservation slots. In the absence of *PR period*, it can reserve only one reservation slot in a frame by either transmitting its Hello packet to node 13 or receiving the Hello packet of node 11, in the 3^{rd} Hello slot. As a result, node 6 schedules data traffic slots either with node 11 or node 13, in a frame, and wastes its remaining unutilized data traffic slots. This reduces spatial reuse at node 6 and increases the queuing delay of packets of Flow#1, which leads to packet drop. Hence, PDR of Flow#1 degrades from 79% (of the proposed MAC scheme which uses PR period) to 38% (proposed MAC scheme *without PR period*) in Fig. 12.

Hotspot nodes 2 and 7 experience the same situation (as node 6) for links 2-4 and 2-5, and 5-7 and 7-12, respectively. However, unlike node 6, they experience high congestion. Hence, they schedule all of their data traffic slots in the Reservation period, which leaves zero data traffic slot for the PR period. For this reason, removing PR period does not degrade the PDRs of Flow#2, Flow#3, and Flow#4.

> ***Observation 2 (Piggyback Reservation):*** *The PR period allows an intermediate node to receive and forward packets in the same frame, which reduces the queuing delay and improves the PDR.*

B. CONTROL OVERHEAD COMPARISON WITH OTHER TDMA SCHEMES

The use of two additional (optional) control periods can slightly increase the control overhead and frame length of our

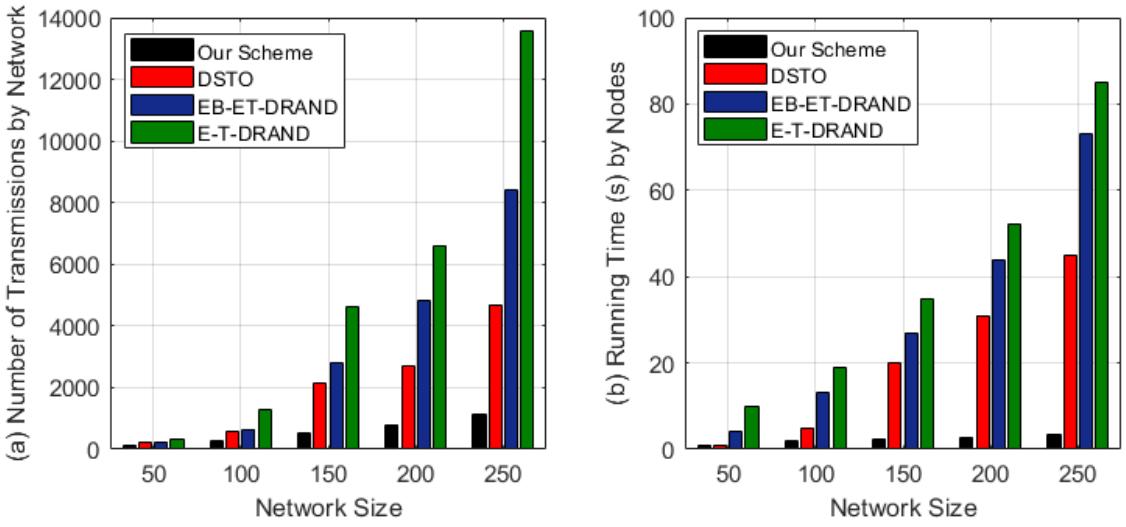


FIGURE 13. Comparison of the (a) total message transmissions and (b) average running time to obtain a conflict-free reservation slot at each node for different network sizes.

scheme. In this section, we compare the control overhead of our scheme with the recently published distributed TDMA schemes[4] (i.e., DSTO (distributed scheduling using topological ordering) [30], EB-ET-DRAND (distributed TDMA scheduling algorithm based on the exponential backoff rule and energy-topology factor) [31], and E-T-DRAND (distributed TDMA slot scheduling algorithm based on the energy-topology factor) [32]). These schemes reduce the control overhead and running time required to obtain a conflict-free reservation slot for each node in a dynamic, multi-hop network. Note that these schemes do not consider the data traffic period. As a result, a frame corresponds to the Hello and Reservation control periods in our scheme. The following two metrics are used for the comparison:

1) Number of transmissions is the total control messages transmitted by all nodes to obtain a conflict-free schedule for the reservation period. A lower transmission count reduces the control overhead [30].
2) Average running time is the time taken for all nodes to acquire a conflict-free reservation slot. A lower running time is desired for dynamic topology [30].

We have used the simulation setup used by [30]–[32], where the network size is varied from 50 to 250 nodes in a 300 m $\times$ 300 m simulation area, with a signal transmission range of 40 m. Each node randomly selects its node pair (i.e., receiver node) from its 1-hop neighborhood and each receiver node knows its transmitter node(s). Each experiment is run 10 times.

As shown in Fig. 13(a), the **average number of transmissions** required in our proposed scheme is significantly lower than the distributed TDMA schemes in [30]–[32] for different network sizes. Note that each node repeatedly broadcasts its updated schedule to resolve a conflict with its 1-hop neighbors in [30]–[32]. As shown in Fig. 13(b), our scheme has a significantly lower **average running time** as compared to the schemes in [30]–[32], due to its lower notification overhead. Note that the number of slots required in the control period in our scheme at a network density is determined from Fig. 2(a).

C. PERFORMANCE COMPARISON WITH OTHER TDMA SCHEMES

To the best of our knowledge, no other directional, distributed TDMA scheme exists which can provide a conflict-free schedule for a dynamic, multi-hop topology in real-time with no notification overhead and delay. Therefore, we have compared our scheme with a recently published omnidirectional, distributed TDMA scheme [30], which provides a conflict-free reservation slot for each node in a dynamic, multi-hop network, while minimizing the running time and control overhead. We refer to the scheme in [30] as typical distributed TDMA scheme from here onward.

In this section, the performance is compared, in terms of PDR and end-to-end delay, for varying traffic densities in static and mobile network topologies. Note that the control overhead and running time were compared in the previous section. For a fair comparison, we consider that the typical distributed TDMA scheme [30] uses an omnidirectional antenna to obtain reservation slot schedule (i.e., sweeping overhead and delay are not considered) and an SBA in data traffic period.

The simulation setup is discussed below, followed by the comparison analysis for random flows and mobile nodes.

1) SIMULATION SETUP

The simulations are run in MATLAB version R2017b for the network topology consisting of 50 nodes. Each experiment

[4]Schemes in [30]–[32] use an omnidirectional antenna and require each node to broadcast its schedule in its 1-hop neighborhood. Note that the directional variants of these schemes incur a huge sweeping delay and overhead, which we have ignored in this comparison, for simplicity.

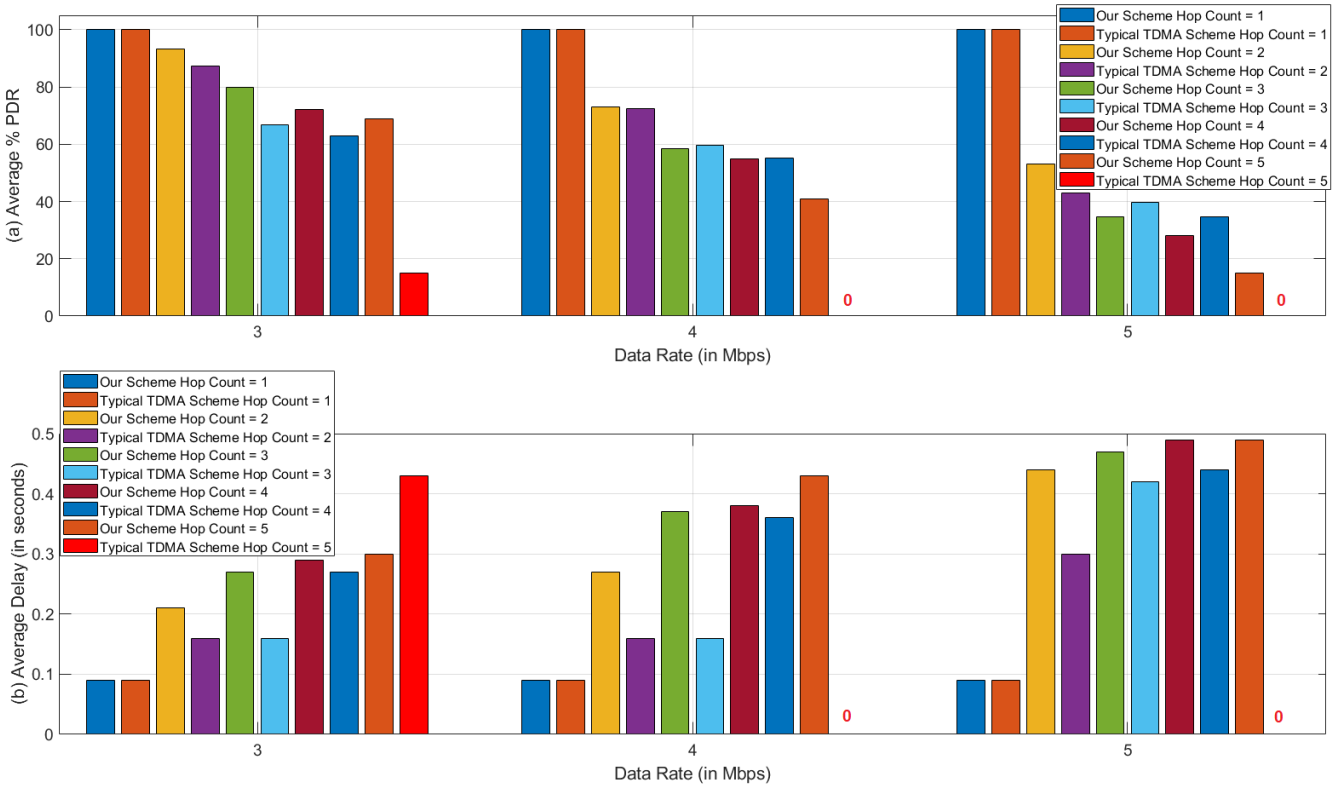


FIGURE 14. Comparison of the average (a) PDR and (b) end-to-end delay for a random static topology of 50 nodes at different traffic densities when the packet TTL is 0.5s.

is run for 100 s and repeated 10 times. Here, the channel capacity is 10 Mbps, TTL is 0.5 s, packet size is 1000 Byte, and queue size is infinite. We consider a frame with 100 data slots and a slot length of 8.7 μs which allows the reception of ACK packet in the same data slot. The length of rank matrix and each control period in our scheme are determined using Fig. 2 and Table 2, respectively.

Note that our scheme can recompute reservation slots frequently because it has a very low reservation slot allocation overhead (see Table 2). On the other hand, the reservation slot allocation overhead is large for typical distributed TDMA scheme (e.g., the average running time of DSTO scheme for a 50-node network is 1 s in Fig. 13(b)). Therefore, we consider a 5 s frame for the typical distributed TDMA scheme, which includes a reservation slot allocation period of 1 s and a 4 s data traffic period.[5] The reservation slot allocation period is used only at the start of the simulation in typical distributed TDMA scheme for the static scenarios where links do not break with time.

2) PERFORMANCE COMPARISON FOR RANDOM FLOWS

In this experiment, 10 source-destination pairs are randomly selected for a static network of 50 nodes randomly placed in a 300 m $\times$ 300 m area. Here, the flow hop-count varies from 1 to 5.

The average PDR and end-to-end delay for random flows with different hop counts and traffic densities are shown in Fig. 14(a) and 14(b), respectively. At 2 Mbps data rate, nodes do not experience congestion. Therefore, PDR of each flow is 100%. The end-to-end delay increases with the flow hop-count, which results in an increase in the number of packets dropped due to TTL expiry. Therefore, the flow PDR for each scheme degrades with the flow hop-count at all traffic densities.

Since the typical distributed TDMA scheme schedules data traffic slots on the FCFS basis, its reservation slot schedule remains the same for the simulation duration, which degrades the average network PDR. In addition, an intermediate node in the typical distributed TDMA scheme may not be able to forward the received packets when it schedules the majority of its data slots with its upstream node (e.g., PDR for the flow with 5 hop-count in typical distributed TDMA scheme is 0 at 4 and 5 Mbps data rates in Fig. 14(a)). Whereas, each node in our proposed scheme fairly distributes its data traffic slots among all the *desired throughput* requests by using the *throughput scaling* mechanism. Therefore, the total average network PDR for all flows is higher in our scheme as compared to the typical distributed TDMA scheme.

A hotspot node in our scheme selects the flow(s) to serve based on the QoS-metric value of the packets stored in its queue, which increases the queuing delay of the packets of

[5]The control overhead of the typical distributed TDMA scheme (i.e., DSTO [30]) decreases when a longer data traffic period is used. We observed the best performance for the typical distributed TDMA scheme at 5 s frame length.

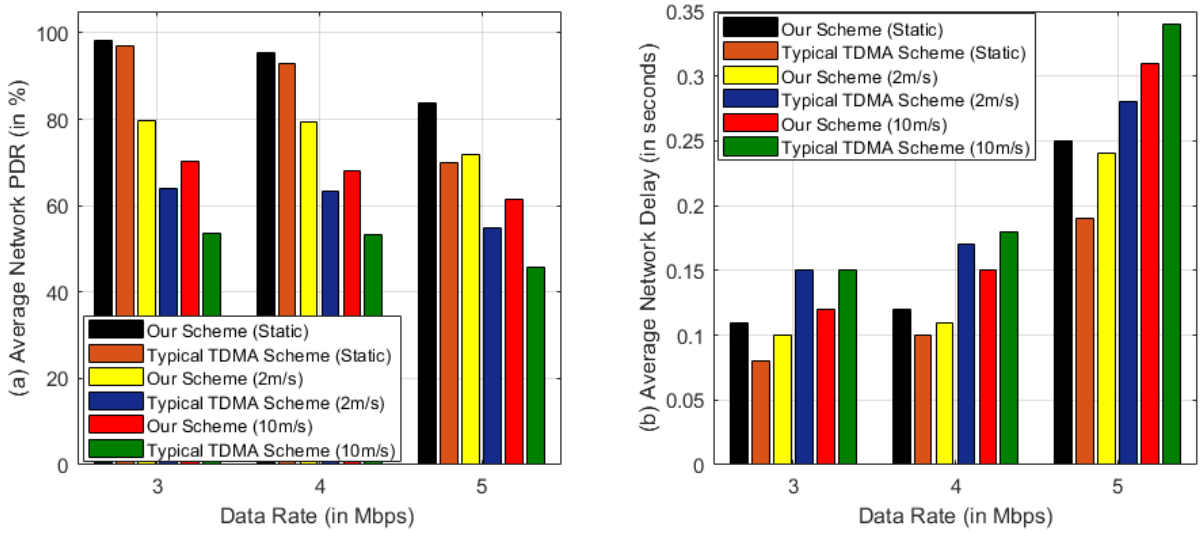


FIGURE 15. Comparison of average (a) PDR and (b) end-to-end delay of our scheme with the typical TDMA-based scheme (e.g., DSTO [30]) at different traffic densities and node speeds for a 50-node network topology, where the packet TTL is 0.5s.

other flows, whereas a hotspot node always prefers the same flow in typical distributed TDMA scheme which leads to a lower delay for the packets of the selected flow. Therefore, the average end-to-end delay is generally higher in our scheme as compared to the typical distributed TDMA scheme at all traffic densities in Fig. 14(b).

3) PERFORMANCE COMPARISON FOR STATIC AND MOBILE TOPOLOGIES

In this section, the performances of the schemes are compared for (*a*) 420 m × 420 m static grid topology and (*b*) 1 km × 1 km mobile topology, where nodes move at (*i*) 2 m/s and (*ii*) 10 m/s, under the random-waypoint mobility model with zero pause time. Here, 25 flows are randomly selected for each scenario where each source node randomly selects a receiver node from its 1-hop neighborhood.

With a **static topology**, both schemes do not experience congestion at 2 Mbps data rate, and therefore, have 100% PDR. Since the typical distributed TDMA scheme reserves a slot on the FCFS basis and cannot accommodate multiple flows, its PDR is lower than our scheme in Fig. 15(a). However, the end-to-end delay of our scheme in Fig. 15(b) is slightly higher than the typical distributed TDMA scheme.

For a **mobile topology**, frequent link breaks lead to a high queuing delay, which increases the end-to-end delay and, thereby, the total packets dropped due to expiry of packet TTL. Therefore, the PDR is lower in Fig. 15(a) and the end-to-end delay is higher in Fig. 15(b) for mobile scenarios as compared to the static scenario. The PDR of both schemes further degrade as the node speed increases from 2 m/s to 10 m/s due to an increase in the link breaks. Since the typical distributed TDMA scheme takes a long time to adapt to the topology changes and recompute the reservation slot schedule, its PDR is lower than our scheme at all traffic densities for mobile scenarios in Fig. 15(a).

Congestion in the network increases with the traffic density, which leads to a high delay and, therefore, lower PDR values for both schemes in static and mobile scenarios.

VI. CONCLUSION

A novel, real-time, distributed, and directional TDMA MAC scheme was presented for multi-hop wireless networks. This scheme adapts to the topology changes and/or flow requirements in *real-time*, and facilitates QoS-aware communication with no *notification overhead*. In the proposed scheme, the 1-hop neighborhood of every node is divided into *fully connected 1-hop neighborhoods*, which allows the node to intelligently serve multiple routes without requiring a globally converged scheduling solution. This feature allows the use of a rank-based mechanism to obtain a real-time transmission schedule for a random multi-hop network.

The following new features were also added in the proposed scheme: (*i*) REQ period which reduces slot wastage, (*ii*) *throughput scaling* which ensures fairness, (*iii*) PR period which increases the spatial reuse and adapts to the dynamic requirements of multiple flows in real-time. The use of these features is optional, which allows a node to customize its frame based on the flow requirements and traffic conditions. Table 3 shows the usefulness of these features for different conditions, such as light, moderate, and high traffic loads, and **i**ndependent **r**outes (IR) with no hotspot node, as well as **r**outes **w**ith at least one **h**otspot node (RwH).

The control-period overhead and running time in our scheme are significantly low as compared to recent, distributed TDMA schemes, and linearly change with the number of nodes in a *fully connected 1-hop neighborhood*. Simulation results showed that our scheme achieved a high PDR and per node channel utilization ratio for real-time traffic, and has a superior performance over recent, distributed TDMA

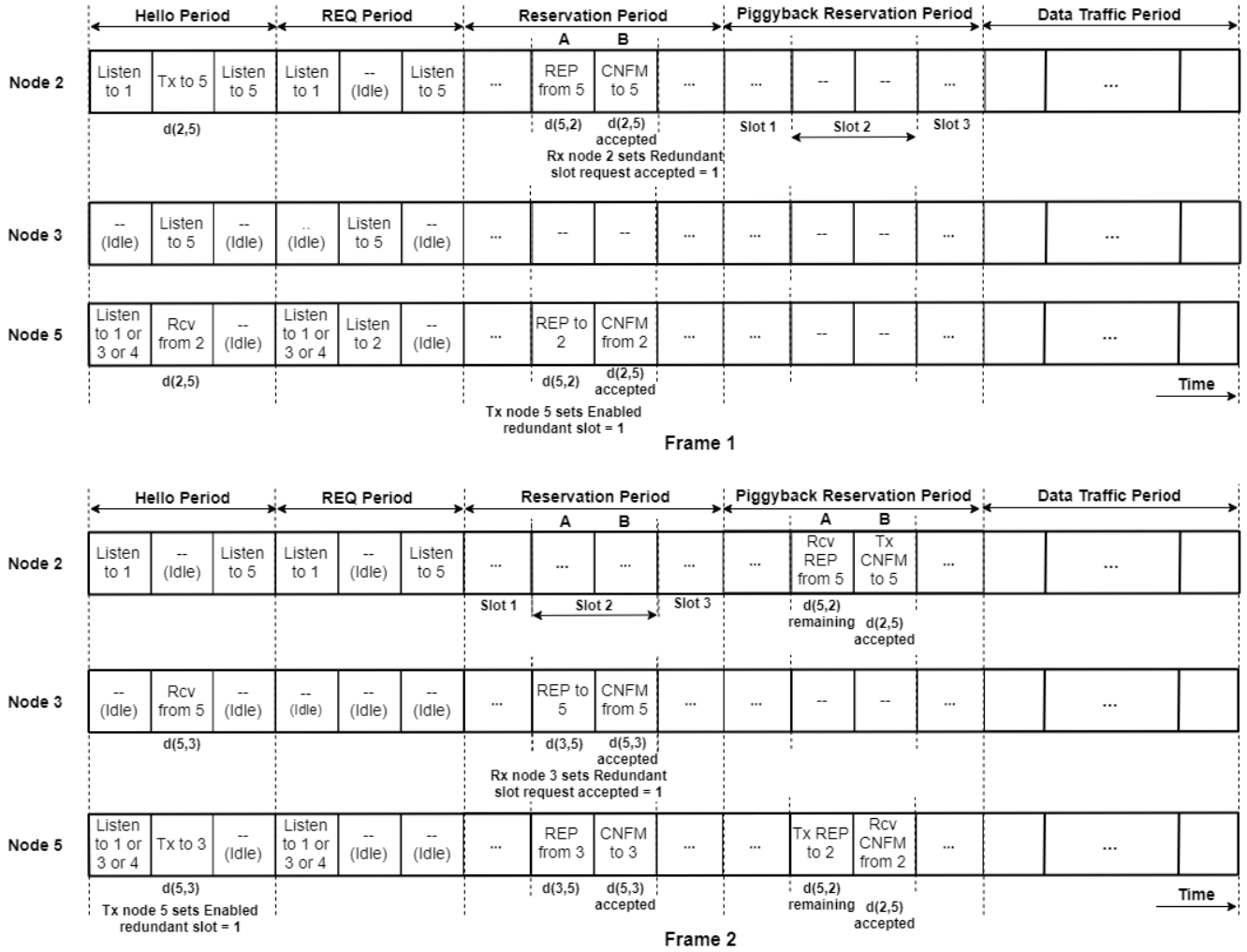


FIGURE 16. Timing diagram of our proposed MAC scheme where node 5 (in Fig. 1) is an intermediate node which receives packets from source node 2 and forwards them to destination node 3. Here, we show the use of PR period which allows node 5 to talk to both nodes 2 and 3 in the same frame (from Frame 2 onwards).

TABLE 3. Usefulness of different periods/mechanisms for different flow types and traffic conditions.

Period/ Mechanism	Light Traffic		Moderate Traffic		Heavy Traffic	
	IR	RwH	IR	RwH	IR	RwH
Throughput Scaling	×	✓	×	✓	✓	✓
REQ Period	×	✓	×	✓	×	✓
Piggyback Reservation	✓	✓	✓	✓	×	×

schemes at different traffic densities for static and mobile network topologies.

APPENDIX
AN EXAMPLE FOR THE WORKING PRINCIPLE OF OUR SCHEME

The example given below explains our proposed scheme, when an intermediate node receives and forwards data packets to/from nodes in the same or different rank matrices. Here, we use the network topology shown in Fig. 1 where intermediate node 5 receives packets from source node 2 and forwards them to destination node 3. As per the rank matrices of node 5 (i.e., [1,2,5], [3,5], [4,5]), it can either receive a Hello packet from node 2 or transmit its own Hello packet to node 3 in the 2^{nd} Hello slot. As a result, it cannot communicate with both nodes in a given frame. Hence, it uses PR period as discussed below and shown via a timing diagram in Fig. 16. Here, we assume that a source node generates 40 packets per frame, the total data traffic slots are 100, and one data packet is transmitted per data traffic slot.

Frame 1:

Step 1: Frame 1 starts. Node 5 decides to listen to node 2 in its 2^{nd} Hello slot (HP S2) and receives a Hello packet from it, with the *desired throughput* value of 40 data packets.

Step 2: Node 5 does not receive any REQ packet from node 2 during 2^{nd} REQ slot (i.e., REQ S2).

Step 3: Node 5 transmits an REP packet to node 2 in Reservation Period slot 2A (RP S2A) with the *Enable Redundant Slot?* field set to 1. This allows node 5 to ask node 2 whether it agrees for using PR Period as it needs to send a Hello packet

to node 3 in the next frame (i.e., Frame 2) and can have unused Data slots which can be reserved for node 2.

Step 4: Node 2 receives an REP packet, selects 40 common available data traffic slots, and transmits a CNFM packet towards node 5 in Reservation Period slot 2B (RP S2B), with the *Redundant Slot Request Accepted?* field set to 1.

Step 5: Node 5 receives a CNFM packet, stores the selected data traffic slots, and notes that node 2 has agreed to use PR period in the next frame (i.e., Frame 2). *Note:* No packet is transmitted or received by either node 2 or 5 in the PR Period of the current frame (i.e., Frame 1).

Step 6: Node pair 2 and 5 exchange data packets in the 40 reserved data traffic slots.

Step 7: Frame 1 is completed. Frame 2 starts.

Frame 2:

Step 8: Node 5 transmits a Hello packet to node 3 in Hello period slot 2 (i.e., HP S2) with *desired throughput* request of 40 and *Enabled Redundant Slot?* field set to 1. No REQ packet is sent in this case.

Step 9: Node 3 receives this Hello packet of node 5 in its HP s2 slot.

Step 10: Node 3 transmits an REP packet in Reservation Period slot 2A (i.e., RP S2A) with its 100 available data traffic slots and sets the *Redundant Slot Request Accepted?* field to 1.

Step 11: Node 5 receives an REP packet, selects 40 common available slots, and then transmits the CNFM packet towards node 3 in Reservation Period slot 2B (i.e., RP S2B).

Step 12: Based on its exchange with node 2 in Frame 1, Node 2 steers its beam towards node 5 in the PR Period slot 2A (i.e., PRP S2A). Node 5 transmits an REP packet with its 60 available data traffic slots towards node 2 in PRP S2A. The *Scaled desired throughput* field is set to 0 in this REP packet. (*Note:* node 5 has scheduled its 40 out of 100 slots with node 3 in Step 11. Hence, it sends its remaining 60 unreserved data traffic slots to node 2).

Step 13: Node 2 receives the REP packet and selects 40 common available data traffic slots, updates the *Traffic slots* and *Scaled desired throughput* fields in the CNFM packet, and then transmits it in PRP S2B to node 5.

Note: Since node 5 had 60 available slots, source node 2 was able to forward all 40 packets to node 5. However, if the data rate increases (e.g., 60 packets per frame are generated at source node 2), node 5 can offer only 40 unreserved slots to node 2, which would leave the remaining 20 packets in the buffer at node 2. This would lead to congestion at node 2 in the subsequent frames and increase the queuing delay. If during this PR period in Frame 2, node pair 2 and 5 agrees on using PR period in Frame 3, packets can expire due to higher queuing delay. Therefore, node pair 2 and 5 must talk in the Hello period of the next frame (i.e., Frame 3) to ensure fairness for link 2-5. Therefore, the REP and CNFM packets in the PR Period do not set the *Enable Redundant Slot?* and *Redundant Slot Request Accepted?* fields to 1.

Step 14: Node 5 receives a CNFM packet of node 2 in PRP S2B.

Step 15: Node 5 communicates over links 2-5 and 5-3 in the data traffic period. Frame 2 ends and Frame 3 starts.

Frame 3:

Step 16: Node 2 transmits a Hello packet to node 5 in Hello period slot 2 (i.e., HP S2), where the *desired throughput* is set to 40. Node 5 receives the Hello packet in its HP S2 slot.

Step 17: Node 5 transmits an REP packet in Reservation period slot 2 mini slot A (i.e., RP S2A) with its 100 available data traffic slots and the *Enable Redundant Slot?* field set to 1.

Step 18: Node 2 receives an REP packet, selects 40 common available slots, sets *Redundant Slot Request Accepted?* field to 1, and transmits the CNFM packet to node 5 in RP S2B.

Step 19: In PRP S2A (i.e., PR period slot 2A), node 3 transmits an REP packet towards node 5 with its available data traffic slots.

Step 20: Node 5 replies with a CNFM packet where it selects 40 common data traffic slots in PRP S2B.

Step 21: Node 5 communicates over links 2-5 and 5-3 in the same frame. Frame 3 ends and Frame 4 starts.

*Step 22: Go to Step 8 (*note: *Frame 4 is same as Frame 2).*

DISCLAIMER

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory or the U.S. government.

REFERENCES

[1] O. Bazan and M. Jaseemuddin, "A survey on MAC protocols for wireless adhoc networks with beamforming antennas," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 2, pp. 216–239, 2nd Quart., 2012.

[2] R. Sharma, G. Kadambi, Y. A. Vershinin, and K. N. Mukundan, "A survey of MAC layer protocols to avoid deafness in wireless networks using directional antennas," in *Mobile Computing and Wireless Networks, Concepts, Methodologies, Tools, and Applications*. Hershey, PA, USA: IGI Global, 2016, pp. 1758–1796.

[3] D. Wong, Q. Chen, and F. Chin, "Directional medium access control (MAC) protocols in wireless ad hoc and sensor networks: A survey," *J. Sensor Actuator Netw.*, vol. 4, no. 2, pp. 67–153, Jun. 2015.

[4] G. Kuperman, R. Margolies, N. M. Jones, B. Proulx, and A. Narula-Tam, "Uncoordinated MAC for adaptive multi-beam directional networks: Analysis and evaluation," in *Proc. 25th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Aug. 2016, pp. 5–10.

[5] P. Li, H. Zhai, and Y. Fang, "SDMAC: Selectively directional MAC protocol for wireless mobile ad hoc networks," *Wireless Netw.*, vol. 15, no. 6, pp. 805–820, Aug. 2009.

[6] J. Wang, Y. Zhang, and L. Jiang, "A novel time-slot allocation scheme for ad hoc networks with single-beam directional antennas," in *Proc. IEEE Int. Conf. Commun. Softw. Netw. (ICCSN)*, Jun. 2015, pp. 227–231.

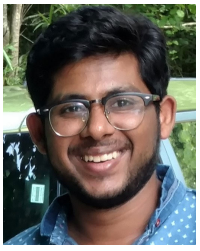
[7] A. El Masri, L. Khoukhi, and D. Gaiti, "A TDMA-based MAC protocol for wireless mesh networks using directional antennas," in *Proc. Int. Conf. Commun. Theory, Rel., Qual. Serive (CRTQ)*, 2011, pp. 95–98.

[8] D. Hao and D. P. Liu, "A distributed TDMA-based MAC protocol for ad hoc networks with directional antennas," in *Proc. Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2012, pp. 1–6.

[9] L. Shaohua and D.-H. Cho, "Directional antenna based time division scheduling in wireless ad hoc networks," in *Proc. MILCOM IEEE Mil. Commun. Conf.*, Nov. 2008, pp. 1–7.

[10] Y.-X. Tian and K.-J. Wu, "A novel dynamic TDMA protocol for ad hoc networks using directional antennas," in *Proc. Int. Conf. Electric Inf. Control Eng.*, Apr. 2011, pp. 65–69.

[11] J. Chang, W. Liao, and J. Lai, "On reservation-based MAC protocol for IEEE 80.11 wireless ad hoc networks wit directional antenna," *IEEE Trans. Veh. Tech.*, vol. 60, no. 6, pp. 2669–2679, May 2011.
[12] J. Wang, H. Zhai, P. Li, Y. Fang, and D. Wu, "Directional medium access control for ad hoc networks," *Wireless Netw.*, vol. 15, no. 8, pp. 1059–1073, 2009.
[13] G. Jakllari, W. Luo, and S. V. Krishnamurthy, "An integrated neighbor discovery and MAC protocol for ad hoc networks using directional antennas," *IEEE Trans. Wireless Commun.*, vol. 6, no. 3, pp. 1114–1124, Mar. 2007.
[14] Z. Zhang, "Pure directional transmission and reception algorithms in wireless ad hoc networks with directional antennas," in *Proc. IEEE Int. Conf. Commun. ICC*, 2005, pp. 3386–3390.
[15] E. Shihab, L. Cai, and J. Pan, "A distributed directional-to-directional MAC protocol for asynchronous ad hoc networks," in *Proc. IEEE GLOBECOM Global Telecommun. Conf.*, 2008, pp. 1–5.
[16] M. Takata, M. Bandai, and T. Watanabe, "A directional MAC protocol with deafness avoidance in ad hoc networks," *IEICE Trans. Commun.*, vol. E90-B, no. 4, pp. 866–875, Apr. 2007.
[17] H. Gossain, T. Joshi, C. D. Cordeiro, and D. P. Agarwal, "DRP: An efficient directional routing protocol for mobile ad hoc networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 12, pp. 1438–1451, Oct. 2006.
[18] Y. Tu, Y. Zhang, and H. Zhang, "A novel MAC protocol for wireless ad hoc networks with directional antennas," in *Proc. 15th IEEE Int. Conf. Commun. Technol.*, Nov. 2013, pp. 494–499.
[19] A. Bhatia and R. C. Hansdah, "RD-TDMA: A randomized distributed TDMA scheduling for correlated contention in WSNs," in *Proc. 28th Int. Conf. Adv. Inf. Netw. Appl. Workshops*, May 2014, pp. 378–384.
[20] P. Djukic and S. Valaee, "Delay aware link scheduling for multi-hop TDMA wireless networks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 3, pp. 870–883, Jun. 2009.
[21] A. Bhatia and R. C. Hansdah, "A classification framework for TDMA scheduling techniques in WSNs," 2020, *arXiv:2002.00458*. [Online]. Available: http://arxiv.org/abs/2002.00458
[22] Z. Bai, B. Li, Z. Yan, M. Yang, X. Jiang, and H. Zhang, "A classified slot re-allocation algorithm for synchronous directional ad hoc networks," in *Proc. Int. Conf. Heterogeneous Netw. Qual., Rel., Secur. Robustness*, 2017, pp. 194–204.
[23] R. Gunasekaran, S. Siddharth, P. Krishnaraj, M. Kalaiarasan, and V. R. Uthariaraj, "Efficient algorithms to solve broadcast scheduling problem in WiMAX mesh networks," *Comput. Commun.*, vol. 33, no. 11, pp. 1325–1333, Jul. 2010.
[24] S. Kumar V and V. Sharma, "Joint routing, scheduling and power control providing QoS for wireless multihop networks," in *Proc. 21st Nat. Conf. Commun. (NCC)*, Feb. 2015, pp. 1–6.
[25] D. J. Vergados, N. Amelina, Y. Jiang, K. Kralevska, and O. Granichin, "Towards optimal distributed node scheduling in a multihop wireless network through local voting," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 400–414, Oct. 2018.
[26] C.-T. Chiang, H.-C. Chen, W.-H. Liao, and K.-P. Shih, "A decentralized minislot scheduling protocol (DMSP) in TDMA-based wireless mesh networks," *J. Netw. Comput. Appl.*, vol. 37, pp. 206–215, Jan. 2014.
[27] F. Hu, X. Li, E. Bentley, L. Hu, K. Bao, and S. Kumar, "Intelligent multi-beam transmission for mission-oriented airborne networks," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 55, no. 2, pp. 619–630, Apr. 2019.
[28] X. Li, F. Hu, J. Qi, and S. Kumar, "Systematic medium access control in hierarchical airborne networks with multibeam and single-beam antennas," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 55, no. 2, pp. 706–717, Apr. 2019.
[29] T. Wang, X. Wang, X. Tian, and X. Gan, "HD-MAC: A throughput enhancing TDMA-based protocol in high dynamic environments," in *Proc. 12th Int. Conf. Mobile Ad-Hoc Sensor Netw. (MSN)*, Dec. 2016, pp. 105–109.
[30] T.-T. Nguyen, T. Kim, and T. Kim, "A distributed TDMA scheduling algorithm using topological ordering for wireless sensor networks," *IEEE Access*, vol. 8, pp. 145316–145331, 2020.
[31] Y. Li, X. Zhang, T. Qiu, J. Zeng, and P. Hu, "A distributed TDMA scheduling algorithm based on exponential backoff rule and energy-topology factor in Internet of Things," *IEEE Access*, vol. 5, pp. 20866–20879, 2017.
[32] Y. Li, X. Zhang, J. Zeng, Y. Wan, and F. Ma, "A distributed TDMA scheduling algorithm based on energy-topology factor in Internet of Things," *IEEE Access*, vol. 5, pp. 10757–10768, 2017.
[33] S. Vasudevan, J. Kurose, and D. Towsley, "On neighbor discovery in wireless networks with directional antennas," in *Proc. IEEE 24th Annu. Joint Conf. IEEE Comput. Commun. Societies*, Mar. 2005, pp. 2502–2512.
[34] F. N. Nur, S. Sharmin, M. A. Habib, M. A. Razzaque, M. S. Islam, A. Almogren, M. M. Hassan, and A. Alamri, "Collaborative neighbor discovery in directional wireless sensor networks: Algorithm and analysis," *EURASIP J. Wireless Commun. Netw.*, vol. 2017, no. 1, pp. 119–134, Dec. 2017.



SHIVAM GARG (Student Member, IEEE) received the M.S. degree in electrical engineering from San Diego State University (SDSU), San Diego, USA, in 2017. He is currently pursuing the Ph.D. degree in computational science jointly with SDSU and University of California, Irvine. His research interests include real-time and multiple access communication in highly dynamic wireless networks using cross-layer protocols.

VENU SRI SUSHMA KUCHIPUDI received the M.S. degree in electrical engineering from San Diego State University, CA, in 2018. Her research interests are in directional MAC and network layer protocols for wireless networks. She is currently working as a Software Engineer with Aruba Networks (HPE), Santa Clara, CA, USA.

ELIZABETH SERENA BENTLEY (Member, IEEE) received the B.S. degree in electrical engineering from Cornell University, Ithaca, NY, in 1999, the M.S. degree in electrical engineering from Lehigh University, Bethlehem, PA, in 2001, and the Ph.D. degree in electrical engineering from the State University of New York at Buffalo, in 2007. Since 2008, she has been employed at the Air Force Research Laboratory, Information Directorate in Rome, NY. Her research interests are in cross-layer optimization, directional networking, wireless video transmission, and network modeling and simulation.

SUNIL KUMAR (Senior Member, IEEE) received the M.S. and Ph.D. degrees in electrical and electronics engineering from the Birla Institute of Technology and Science, Pilani, India, in 1993 and 1997, respectively. He is currently a Professor and Thomas G. Pine Faculty Fellow with the Electrical and Computer Engineering Department, San Diego State University, San Diego, California, USA. He has published over 160 research articles in refereed journals and conferences, including four books. His research has been supported by grants from the National Science Foundation, Air Force Research Laboratory, Department of Energy, and industry. His research areas include wireless networks, cross-layer and QoS-aware wireless protocols, and error-resilient video compression.