# Progressive Decentralized TDMA based MAC: Joint Optimization of Slot Allocation and Frame Lengths

Muhammad Hafeez Chaudhary, and Bart Scheers
Royal Military Academy Belgium
{mh.chaudhary, bart.scheers}@rma.ac.be

***Abstract*—The TDMA based MAC scheduling is considered appropriate for many applications in which deterministic medium access schedule plays a crucial role. The existing plentiful literature on the TDMA based scheduling focus on the conflict-free slot allocation where the scheduling problem is broken into two disjoint problems: first find conflict free slot allocation, minimizing the number of slots used, and subsequently select frame sizes in which to use the assigned slots. In this paper, we show that such a sequential approach could lead to suboptimal performance when analyzed from the perspective of slot or channel reuse, which is the prime objective of the spatial reuse TDMA schemes. To his end, we formulate the channel scheduling as an optimization problem that aims to maximize the slot reuse factor whereby we jointly optimize the slots assignment and the frame lengths. The problem being inherently NP-hard, to solve it we propose a greedy heuristic based algorithm by which nodes complete slot assignment in a progressive decentralized way. We show that under the proposed algorithm, all nodes are guaranteed to find a conflict-free transmission schedule. Besides, we provide upper bound on the convergence time of the algorithm for a single node, and for the whole network. Finally, with simulation examples, we show that the proposed algorithm when compared with other TDMA scheduling schemes could give better performance in terms of slot reuse factor.**

## I. INTRODUCTION

Last decade has seen an explosive growth in the applications of wireless communication and networking technologies bringing ubiquitous mobile service into the everyday realm. Besides, the demand for more and faster data transfer is increasing than ever before. The phenomenal increase in the wirelessly connected devices and the applications running on them have put an extremely high premium on the communications spectrum, and thus placing great demand on designing spectrum efficient communication and networking protocols to meet the requirements of the current and emerging applications in wireless networking. Currently, spatial reuse based MAC protocol design is a key area of research, which is driven by the requirement of having a technology that enables efficient utilization of spectrum resources.

Given the number of wirelessly connected devices is on the rise, the demand for efficient use of the spectrum resources is increasing. To meet the spectrum demand, it is necessary to allow the channel usage by node which are sufficiently far apart. TDMA based MAC scheduling is considered to be an appropriate choice for applications requiring predictable quality of service (e.g., in emergency response services and military tactical networks) [1]. There are numerous algorithms dealing with the TDMA based MAC scheduling in ad hoc networks, whereby nodes can find conflict-free slot assignment. In [2], a TDMA slot scheduling protocol named GinMAC is presented; the protocol requires the network be arranged in a tree-like structure. Building on the GinMAC, in [3] the BurstProbe slot assignment protocol is proposed. These slot scheduling protocols require global topology information which may not be available in ad hoc networks that are inherently bereft of coordinating and controlling infrastructure. In [4] a slot assignment protocol called USAP is proposed which allows nodes to get conflict-free slot assignment in a distributed way using local topology information. Kanzaki and his colleagues proposed a protocol named ASAP in [5] which can be viewed as an extension of the USAP, adding details on dynamic frame-size selection and more detailed procedures about the nodes joining/leaving the network. Another related protocol named DRAND is proposed in [6], and later on extended to Z-MAC [7] which combines artifacts of both TDMA and CSMA medium access schemes. The main focus of these protocols (and others like in [8], [9]) is to find conflict-free slot assignment to nodes in a distributed way. The maximization of the channel utilization efficiency (i.e., the spatial reuse of the slots) is not explicitly considered. Thus from channel utilization viewpoint, these protocols may give suboptimal performance. Recently, in [10] a TDMA based MAC scheduling protocol named HUDSAP is proposed which gives better channel utilization efficiency than the DRAND.

Most of the existing algorithms that deal with the TDMA based MAC scheduling problem, break the problem into two disjoint subproblems which are solved in a sequential way: first find a conflict free slot assignment to nodes, subsequently find the frame lengths. In this paper we show that this disjoint and sequential treatment of the two problems could lead to sub-optimal solution when our objective is to maximize the reuse of the slots within the network. In order to maximize the slot reuse factor, here we propose a decentralized TDMA based MAC protocol, which we call progressive decentralized MAC (PD-MAC), for applications in ad hoc networks. Under the proposed MAC scheduling protocol, each node assigns time slot(s), and selects frame length in which to use the slot(s). The slot assignment and frame size selection are jointly optimized using a greedy heuristics based approach. Each node performs these functions in a decentralized way using information only from its one-hop neighbors. For a node

---

The research leading to these results has received funding from the European Communitys Seventh Framework Programme (FP7/2007-2013) under grant agreement number 285417 .

to join the network, the protocol assumes at least one of its neighbors already has joined the network. For a newly deployed network this requirement means we need to seed the network. Concretely that means, one node (could be any node) is selected as the seed node which directly assigns slot(s) and selects frame length, and starts transmitting messages to announce ownership of the assigned slots. Then the neighbors of the this node assign slots and select frame lengths. The process is repeated progressively until all nodes complete slot assignment. The proposed protocol ensures conflict free channel access schedule among all nodes. The convergence time of the PD-MAC for a new node $i$ is upper bounded by $O(\lambda_i)$, where $\lambda_i$ is a parameter related to the node degree. Overall convergence time for the whole network is upper bounded by $O(d\lambda)$, where $\lambda$ is the maximum node degree encountered in the network, and $d$ diameter of the network connectivity graph.

The remainder of the paper is organized as follows. Section II presents preliminaries and the problem formulation. Section III outlines solution to the scheduling problem. Towards the end, Section IV evaluates the performance of the proposed protocol with some simulation examples and implementation on the USRP based test platform. And, finally, Section V provides concluding remarks.

## II. PRELIMINARIES AND PROBLEM FORMULATION

In the proposed TDMA based MAC protocol, each node allocates time slot(s) and selects its frame length within which to use the slot(s). The slot allocation and the frame size selection works in a joint way at each node. The objective of the considered MAC scheduling scheme is to maximize the reuse of the slots with the constraint that no two nodes within a two-hop neighborhood assign the same slot. We consider a network in which network connectivity is given by an un-directed graph $G(\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ and $\mathcal{E}$ be the set of nodes and the set of edges among the nodes, respectively. The cardinality of $\mathcal{N}$, i.e., $|\mathcal{N}|$ denotes the number of nodes in the network and let it be denoted by $N$. An edge exists between nodes $i$ and $j$ if and only if they are reachable from each of them, i.e., $(i, j) \in \mathcal{E}$ and $(j, i) \in \mathcal{E}$. The set of one-hop neighbors of a node $i$ is defined as $\mathcal{O}(i) = \{j : (i, j) \in \mathcal{E}, \forall j \in \mathcal{N}\}$ and the set of two-hop neighbors of the node $i$ is defined as $\mathcal{T}(i) = \left\{\bigcup_{j \in \mathcal{O}(i)} \mathcal{O}(j) \setminus \{i\} \cup \mathcal{O}(i)\right\}$. The set of all nodes in the contention area of the node $i$ is defined as $\mathcal{C}(i) = \{i\} \cup \mathcal{O}(i) \cup \mathcal{T}(i)$.

We wish to maximize the communication channel utilization among the nodes. In most of the existing studies, the problem of TDMA based MAC design is broken into two subproblems:

1) Minimize the number of slots used in the networks, and
2) Select frame lengths in which to use the assigned slot(s) to each node.

The two problems are sequentially solved. Where, often the former problem is casted as a graph coloring problem. Given the graph $G(\mathcal{N}, \mathcal{E})$, coloring of nodes can be viewed as a mapping $f : \mathcal{N} \mapsto \mathcal{S}$, where $\mathcal{S}$ is the set of colors (which corresponds to slots), usually represented by a small set of positive integers. In this setting, the slot allocation problem is to find the solution to the following problem:

$$\begin{aligned} &\text{minimize } |\mathcal{S}|, \\ &\text{subject to } s_i \neq s_j, \text{ for } s_i, s_j \in \mathcal{S},\ i, j \in \mathcal{C}(i),\ \forall i, j \in \mathcal{N}. \quad (1) \end{aligned}$$

However, finding an optimal solution of this problem is NP-hard [11], [12]. To this end, in the literature, heuristic-based suboptimal solutions are proposed that vary in the level of suboptimality, the convergence time, and the message overhead. For instance, in [13], three *greedy* heuristic-based slot assignment procedures are proposed: namely, the RAND (random), the MNF (minimum neighbor first), and the PMNF (progressive minimum neighbor first). The basic principle underlying each of these schemes is essentially the following: first, give a unique label to each node, and then assign slots to nodes in decreasing order of their labels. In RAND, the nodes are labeled in a random way; in MNF, the node with minimum number of neighbors is labeled first; and in PMNF, the nodes are labeled as in MNF with a difference that after labeling a node, the node and its edges are removed. Effectively that means, at each step, among the nodes that have not been assigned slot yet, the RAND takes a node at random and allots time slot to it; the MNF takes the node with maximum number of neighbors and allots slot to it; and the PMNF first removes the nodes and the associated edges that have already been assigned slots, then within the updated network assign slot to the node with maximum number of neighbors. The problem with these schemes is that they require knowledge of the global network topology; that is, they are centralized schemes. For ad hoc networks, distributed or decentralized schemes are sought because such networks are devoid of coordinating and controlling infrastructure and the global topology knowledge is hard to come by at individual nodes. For ad hoc networks, distributed versions of RAND (named DRAND) and MNF (named HUDSAP) are proposed in [6] and [10], respectively.

Once all nodes in $\mathcal{C}(i)$ have assigned a slot, the node $i$ selects frame length $L_i$ in which to use that slot as a function of $s_{\max}^{(i)} = \max\{s_j : \forall j \in \mathcal{C}(i)\}$, the *maximum slot number* used within the contention area. Concretely, $L_i = 2^{a_i}$, where $a_i$ satisfies [1]

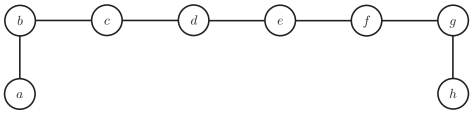
$$2^{a_i - 1} \leq s_{\max}^{(i)} \leq 2^{a_i}, \quad a_i > 0. \qquad (2)$$

This splitting of the TDMA based channel scheduling problem into two disjoint problems, although, ensures conflict-free transmission schedule. However, from the objective of maximizing the channel reuse, the approach could lead to quite suboptimal schedule, as shall be seen in the ensuing discussion.

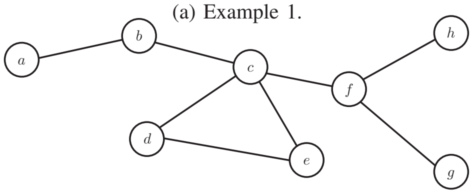
For a network comprising $N$ nodes, conflict-free slot allocation can be ensured within a frame length of $N$ slots, assigning each node a slot without slot reuse. We call such a scheduling scheme as "star-like", due to its equivalence to the star topology. Based on this premise, we define channel/slot reuse factor for slot allocation protocol $A$ as follows:

---

[1] Restricting frame length to the power of two simplifies the scheduling problem.



(a) Example 1.



(b) Example 2.

Fig. 1: Example networks.

TABLE I: Comparison of the slot reuse factor of different protocols for network in Fig. 1a.

| | Node ID | $a$ | $b$ | $c$ | $d$ | $e$ | $f$ | $g$ | $h$ |
|---|---|---|---|---|---|---|---|---|---|
| DRAND | Slot Nr. | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 |
| | $L$ | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| | $\eta$ | 2 | | | | | | | |
| HUDSAP | Slot Nr. | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 |
| | $L$ | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| | $\eta$ | 2 | | | | | | | |
| PD-MAC | Slot Nr. | 4 | 1 | 2 | 4 | 1 | 2 | 4 | 1 |
| | $L$ | 4 | 2 | 8 | 4 | 2 | 8 | 4 | 2 |
| | $\eta$ | 2.5 | | | | | | | |

***Definition 1:*** We define *slot reuse factor* for a TDMA based channel scheduling protocol $A$ as follows:

$$\eta_A = \frac{\xi_A}{N}, \tag{3}$$

where $\xi_A$ denotes the total number of conflict-free transmissions of scheme $A$ within a frame length of $N$ slots.

According to the definition, for the star-like scheme, $\eta_S$ is exactly one, as each slot is used once within the frame length of $N$ slots. Thus, for instance $\eta_A = 2.5$ would imply 2.5 transmissions per time-slot, on average in the network. Table I and Table II gives a comparison of the slot reuse factors for three schemes, DRAND, HUDSAP, and PD-MAC; where PD-MAC is the TDMA based protocol proposed in this paper. The slot allocation and frame length selection in DRAND and HUDSAP are based on the two-step procedure outlined earlier. From the tables, we can observe that the proposed scheme could give substantially higher slot reuse factor. As shall be explained in the ensuing discussion, the PD-MAC gives this higher slot reuse factor by jointly optimizing the slot allocation and frame size selection.

## III. JOINT OPTIMIZATION OF SLOT ALLOCATION AND FRAME LENGTHS

In this work our objective is to maximize the slot reuse factor $\eta$, which is a function of assigned slots and frame lengths, such that no two nodes in a contention area transmits in the same slot. Concretely, we consider the following optimization problem

TABLE II: Comparison of the slot reuse factor of different protocols for network in Fig. 1b.

| | Node ID | $a$ | $b$ | $c$ | $d$ | $e$ | $f$ | $g$ | $h$ |
|---|---|---|---|---|---|---|---|---|---|
| DRAND | Slot Nr. | 1 | 2 | 3 | 1 | 4 | 5 | 1 | 2 |
| | $L$ | 4 | 8 | 8 | 8 | 8 | 8 | 8 | 8 |
| | $\eta$ | 1.1250 | | | | | | | |
| HUDSAP | Slot Nr. | 2 | 3 | 1 | 4 | 5 | 2 | 3 | 4 |
| | $L$ | 4 | 8 | 8 | 8 | 8 | 8 | 4 | 4 |
| | $\eta$ | 1.3750 | | | | | | | |
| PD-MAC | Slot Nr. | 1 | 4 | 2 | 1 | 6 | 14 | 1 | 4 |
| | $L$ | 2 | 4 | 8 | 2 | 16 | 32 | 2 | 4 |
| | $\eta$ | 2.2188 | | | | | | | |

$$\begin{aligned} \underset{s_i, L_i, \forall i \in \mathcal{N}}{\text{maximize}} \quad & \eta(s_i, L_i) \\ \text{subject to} \quad & s_i \neq s_j \quad \text{for } i, j \in \mathcal{C}(i),\ \forall i, j \in \mathcal{N}, \\ & s_i, \log(L_i) \in \mathbb{N}_+,\ \forall i \in \mathcal{N}, \end{aligned} \tag{4}$$

where $log$ denotes the logarithm to the base 2, and the set $\mathbb{N}_+$ contains all positive integers. Finding the optimal solution to this scheduling problem is NP-hard. That means, to solve this problem we have to rely on heuristics based approaches where we have to balance trade-off between optimality and computational complexity. Besides, it is desirable that the solution could be implemented in a distributed or decentralized way. In what follows, we propose a greedy optimization approach that works in a decentralized way.

### *A. Algorithm Description*

The proposed algorithm works in rounds. Let $\mathcal{P}_\kappa$ be the set of nodes that has completed slot assignment at the end of round $\kappa$. The proposed slot assignment works in a progressive decentralized way from a seed node (which could be any node in the network). The seed node assign slot one within frame length of two [2]. In the algorithm, each node decides on the slot allocation and frame length selection jointly.

***Definition 2:*** All nodes which are one-hop neighbors of the nodes that has completed slot assignment are called the *frontier nodes*. The set of frontier nodes at the beginning of round $\kappa$ can be defined as

$$\mathcal{F}_\kappa = \bigcup_{l \in \mathcal{P}_{\kappa-1}} \mathcal{O}(l) \setminus \mathcal{P}_{\kappa-1}. \tag{5}$$

At any given time, only the frontier nodes attempt slot assignment, i.e., the nodes in the set $\mathcal{F}_\kappa$. For working of the proposed algorithm, we make following assumptions.

*Assumption 1:* Each node in the network has a unique ID.

*Assumption 2:* Each packet of node $i$ can be successfully delivered to all nodes in $\mathcal{O}(i)$ provided no other node from $\mathcal{C}(i) \setminus \{i\}$ transmits in this slot.

---

[2] In the optimization problem, it is relatively straightforward to incorporate prioritization mechanism in the assignment of number of slots and frame length to the nodes. For instance, the frame length could be lower bounded to exclude the possibility of a node taking too much channel bandwidth.

**Algorithm 1** Construct SAV

```
Let $i$ be a node belonging to $\mathcal{P}_\kappa \cup \mathcal{F}_\kappa$.
Listen to the ongoing slotted transmissions.
if $i \in \mathcal{F}_\kappa$ and successfully received a valid packet then
    Set the size of $\mathbf{a}_i$ to the frame length in the received packet, slot index $\iota = 1$, and $\mathbf{a}_i(\iota) = 1$.
end if
if $i \in \mathcal{P}_\kappa$ then
    Set size of $\mathbf{a}_i$ to $L_i$, $\iota = s_i$, and set $\mathbf{a}_i(\iota) = 1$.
end if
while $i \in \mathcal{P}_\kappa \cup \mathcal{F}_\kappa$ do
    Increment the slot index $\iota$
    if Successfully received a valid packet then
        if Received frame length $>$ length of $\mathbf{a}_i$ then
            Reset size of $\mathbf{a}_i$ to the frame length in the received packet.
        end if
        Set $\iota = \iota \bmod \text{length}(\mathbf{a}_i)$ and $\mathbf{a}_i(\iota) = \mathbf{a}_i(\iota) + 1$.
    else
        Set $\iota = \iota \bmod \text{length}(\mathbf{a}_i)$.
    end if
end while
```

*Assumption 3:* Each node $i \in \mathcal{P}_\kappa$, always transmits a packet in its assigned slot according to its frame length. The packet contains frame length information.

*Remark 1:* The first assumption ensures that all nodes in the network are uniquely distinguishable. The second assumption guarantees the quality of underlying physical layer communication. The last assumption serves dual purpose: firstly, it ensures ownership of the slot and secondly, the new node joining the network can use the slot boundary to do time synchronization. In essence, a node joining the network, can simultaneously do time synchronization and slot allocation. Any node that hears the time-slotted transmissions on the given channel can synchronize its clock using slot boundary as a reference. The time synchronization can be maintained in a perpetual way as long as the node hears valid transmissions on the channel [14].

Each node $i$ that either has completed slot assignment or is a frontier node maintains information: on the maximum frame length used within its one-hop neighborhood and the slot activity within that frame length. All this information is put in a vector which we call *slot activity vector* (SAV), denoted by $\mathbf{a}_i$. Being limited to one-hop neighborhood, the slot activity is passively built by the node by listening to the ongoing transmissions. Besides, each node *asynchronously* builds its SAV inasmuch as the node needs not to know the beginning of the frames of its one-hop neighbors. While building SAV, node $i$ sets its length to the maximum frame length within its active one-hop neighbors, i.e., $\mathcal{P}_\kappa \cap \mathcal{O}(i)$. The routine to build SAV is outlined under Algorithm 1.

To assign slot and select frame length, the nodes in $\mathcal{F}_\kappa$ runs the PD-MAC algorithm which works in rounds, where each round is divided into two subrounds. The algorithm is independently executed on each node.

- In first subround, node $i \in \mathcal{F}_\kappa$ builds its SAV $\mathbf{a}_i$ using Algorithm 1. To do that, the node listens to the ongoing transmissions for a certain duration selected at random. For all slots in which it detects valid transmissions are marked as occupied whereas all others are taken as free/inactive slots. In building the SAV, its size is set to the maximum frame length currently observed by the node.
- After building its SAV, in the second subround, the node tentatively selects frame length and one of the inactive slot to be used within that frame length. The constraints in making this selection are the following:
    - the frame length must be an integer power of two, and
    - at least one slot must remain inactive within the selected frame length, should the selected slot be activated.

  Next, the frontier node attempts to assign the selected slot and frame length. For this it sends *slot assignment request* (SAR) packet to all its active neighbors in the selected slot. After sending the request, the node waits for the response from its neighbors. If the requested slot is not used by any of the nodes in $\mathcal{T}(i) \cap \mathcal{P}_\kappa$, then all nodes in $\mathcal{O}(i) \cap \mathcal{P}_\kappa$ will receive this packet and will respond. However, if this slot is used by any node in $\mathcal{T}(i) \cap \mathcal{P}_\kappa$, then some of the nodes in $\mathcal{O}(i) \cap \mathcal{P}_\kappa$ will not listen this packet and thus would not respond. When a node $j \in \mathcal{O}(i) \cap \mathcal{P}_\kappa$ receives a SAR packet, the node will prepare its response. The procedure to do so is outlined hereunder.
    - The node checks if granting this request would leave at least one slot free within its SAV. If results of this check is affirmative yes, then the node checks if it can suggest to use this slot in a frame length of less than what is proposed by the requesting node. To do this, The node computes the maximum period of inactive slots within its SAV, starting from and ending at the requested slot. Let the period be $p_i$. The node suggested frame length $\hat{L}_i^j$ must statisfy

      $$\hat{L}_i^j = 2^{b_i} \geq p_i, \quad \text{where } b_i \geq 1, \tag{6}$$

      that is, the periodicity $p_i$ gives a lower bound on the frame length that the node can allow. In this case, the node response will be in the form of a packet, called *slot assignment grant* (SAG) packet containing following parameters: slot index, proposed frame length. In this case, the proposed frame length is either the same as requested or less by a factor which shall be an integer power of 2.
    - If granting the requested slot does not leave at least one slot free in its SAV, the node calculates frame length (increasing in multiple of 2) within which allowing that slot would leave at least one slot free and send SAG packet to the node. In this case the proposed frame length will be greater than what is requested.
    - After sending SAG packet, the responding node

will hold this slot until further notification from the requesting node or for a particular time out period, a system design parameter. During this time, should the node receives SAR packet from any other node, it will not grant this slot.

If the node $i$ receives SAG response from all of its active one-hop neighbors, that is, from all nodes in $\mathcal{O}(i) \cap \mathcal{P}_\kappa$, it can assign the requested slot. To decide on its frame length, the node $i$ collates suggested frame lengths as follows:

$$L_i = \max\{\hat{L}_i^j : \forall j \in \mathcal{O}(i) \cap \mathcal{P}_\kappa\}. \tag{7}$$

Following this step, the node $i$ sends out a *slot assignment confirmation* (SAC) packet in the selected slot, notifying its neighbors of its assignment of the slot and the frame length. However, if the node $i$ does not receive SAG from all of its active neighbors during a randomly chosen duration, the node sends out a *slot assignment failure* (SAF) packet, notifying its neighbors to release this particular slot for assignment to other nodes. In this case, the node, sequentially tries to assign the remaining inactive slots. If it fails to assign any inactive slot, the node $i$ would conclude that all of the inactive slots are used by its active two-hop neighbors, i.e., nodes in $\mathcal{T}(i) \cap \mathcal{P}_\kappa$. This scenario will arise when introduction of node $i$ produces a *cycle* within its contention area (i.e., in $\mathcal{C}(i)$). This could happen when node $i$ has two or more active one-hop neighbors. In this case, to assign slot, node $i$ will do the following:

- For the slot for which the node $i$ received response from maximum number of its active one-hop neighbors, the node $i$ sends out a *slot release request* (SRR) to its one-hop neighbors. Upon receiving the SRR packet, the nodes in $\mathcal{O}(i) \cap \mathcal{P}_\kappa$ forwards this request to nodes in $\mathcal{T}(i) \cap \mathcal{P}_\kappa$.
- Upon receiving the SRR, the node in the set $\mathcal{T}(i) \cap \mathcal{P}_\kappa$ which is using this slot, will release this slot by doubling its frame length, and using this slot only in the second half of its new frame length. After completion of this step, the node $i$ would receive response to its SAR message from all of its active one-hope neighbors, and thus would succeed in assigning a slot.

*Remark 2:* The PD-MAC at each node only uses information from its one-hop neighbors. Only the frame length information are exchanged among one-hop neighbors through explicit messages. The information on the assigned slots are passively learned by listening to the ongoing transmissions. Besides, for joint slot and frame length assignment all but one control messages are exchanged with one-hop neighbors. The only control message that need to be routed to the two-hop neighbors is the SRR packet.

### *B. Algorithm Analysis*

In this section, we first show that the algorithm is guaranteed to produce a conflict-free slot assignment among nodes under the given constraints. Afterwards, we give bounds on the expected convergence time in which a single node as well as the whole network would complete slot assignment.

***Proposition 1:*** The PD-MAC is guaranteed to find a conflict free slot assignment for each node.

*Proof:* Once a node $i$ has assigned a slot and has selected a frame length, its assigned slot would not be used by any node in its contention area, i.e., $\tilde{\mathcal{C}}(i) = \mathcal{C}(i) \setminus \{i\}$. According to the PD-MAC, in round $\kappa$, node $i$ will assign the slot and the frame length only if it does not cause conflict with nodes in $\tilde{\mathcal{C}}(i) \cap \mathcal{P}_\kappa$. Once it has assigned the slot, in the subsequent rounds, its inactive one- and two-hop nodes, i.e., $\tilde{\mathcal{C}}(i) \setminus \mathcal{P}_\kappa$ will succeed in assigning slots only if that do not cause conflict with the assignment of node $i$. All in all, PD-MAC ensures that no two nodes in $\mathcal{C}(i)$ will use same slots. $\square$

***Proposition 2:*** For a new node $i$, the convergence time of the PD-MAC, in terms of number of rounds, is upper bounded by $O(\lambda_i)$, where $\lambda_i = |\mathcal{C}(i) \setminus \mathcal{P}_\kappa|$, the number of one and two-hop neighbors excluding active neighbors, when it first time becomes the frontier node.

*Proof:* The node $i$ starts executing PD-MAC in a round $\kappa$ when it becomes a frontier node. During each subsequent round at least one node from $\mathcal{C}(i) \setminus \mathcal{P}_\kappa$ would assign a slot, on average. So the time for the node $i$ to assign a slot after becoming the frontier node is upper bounded by $\lambda_i$, where $\lambda_i < |\mathcal{C}(i)|$. $\square$

***Proposition 3:*** The convergence time, in terms of number of rounds, of the PD-MAC for a random but connected planar network graph $G(\mathcal{N}, \mathcal{E})$ is upper bounded by $O(\lambda d)$, where $\lambda$ and $d$, respectively, denotes the maximum node degree in the network, and diameter of the graph.

*Proof:* According to Prop. 2, the convergence time of the PD-MAC for a single node can be upper bounded by $O(\lambda_i)$. Building on that argument, it is relatively straight forward to show that the total number of rounds required to find slot assignment for all nodes in $G$ is upper bounded by $\lambda d(o) \leq \lambda d$. Where $d(o)$ is a measure of the eccentricity of the seed node, denoted by $o$. The eccentricity is defined as follows $d(o) = \max_{\forall v \in \mathcal{N}} \epsilon(o, v)$ in which $\epsilon(o, v)$ measures the shortest path, in number of hops, between nodes $o$ and $v$. The term $d$ denotes the diameter of the graph which is given by $d = \max_{\forall o \in \mathcal{N}} d(o)$; and $\lambda = \max\{|\mathcal{C}(i)| : \forall i \in \mathcal{N}\} \geq \max\{\lambda_i : \forall i \in \mathcal{N}\}$. $\square$

The requirement of the seed node is essential for the algorithm to work on a network where no node has yet assigned a slot. This requirement is not limiting the usability of the algorithm; in most of the practical networks, for instance, in ad hoc sensors networks there is usually a fusion node or gateway node through which the user of the network interacts with the network. This fusion node can act as the seed node. In other networks, for example in military tactical ad hoc networks the group commander node can act as the seed node. In a network where some nodes has already assigned slots, in that network, there is no need of a seed node. The new nodes can simply join the network using the procedure outlined in the preceding section. So in essence, the need of a seed node would not be seen as a limiting factor in majority of the applications. Rather the requirement of the seed node provides the flexibility to control how the slot assignment is done among the nodes. For example, the seed node and the nodes around it would be the first nodes to assign slots, so we can give them freedom to assign as many slots as they need. In the context of the sensors networks, where data flows from the sensors towards the fusion center, there nodes near the fusion center need more slots (or shorter frame lengths) to forward data coming from the other sensor nodes. There the control provided by the seed node could be beneficial. Apart from that, the availability of the seed node, can be used to do time synchronization, where all nodes try to synchronize their clocks with the clock of the seed node using it as the root node in a fashion described in the TPSN protocol by exchanging time stamped messages [15].

## IV. PERFORMANCE EVALUATION

In this section, with some numerical experiments, we evaluate slot reuse factor of the proposed PD-MAC protocol and compare its performance with the centralized schemes RAND and MNF of [13]. Note that HUDSAP is proven to achieve the same channel utilization efficiency as MNF [10] and the DRAND as RAND [6]; that is why comparison of PD-MAC with HUDSAP and DRAND is not performed here. We generate a randomly connected network deployed in a planar region. To generate the network, we deploy nodes in a 200-by-200 planar region and to generate a connected network we use the spatial network model from [16]. For performance evaluation, we consider a network comprising $N$ nodes, where for convenience $N$ is restricted to be a multiple of two. Each point in the numerical results is obtained by averaging over $100$ random deployments. For the PD-MAC, we vary the seed node from $1$ to $N$ and calculate the maximum, minimum, and average slot reuse factor over the seed nodes. In MNF and RAND, the two-step scheduling procedure is used for the slot assignment and frame length selection. Besides, in RAND, the nodes are always ordered as $1$ to $N$.

Obtained results are plotted in Fig. 2. From the figure, we can observe that the proposed slot scheduling scheme could achieve substantially higher slot reuse factor compared to the given MNF and RAND based schemes. For instance, when $N = 256$, the PD-MAC could allow approximately 10 more transmissions per slot than the MNF and RAND. This higher channel reuse is realized by jointly optimizing the slot allocation and the frame length selection. As a result, we can conclude that the sequential two-step MAC scheduling methods—first assigning slots to nodes and then selecting frame lengths—are inefficient when we aim to maximize the channel reuse factor. The slot reuse could be improved by joint optimization of the slot allocation and frame length selection. From Fig. 2 we can also see that the seed node do have an impact on the achieved slot reuse factor. However, under the PD-MAC, the minimum achieved slot reuse factor, over all possible seed nodes, is still noticeably better than the other schemes. This observation underlines the robustness of the protocol to the seed node selection.

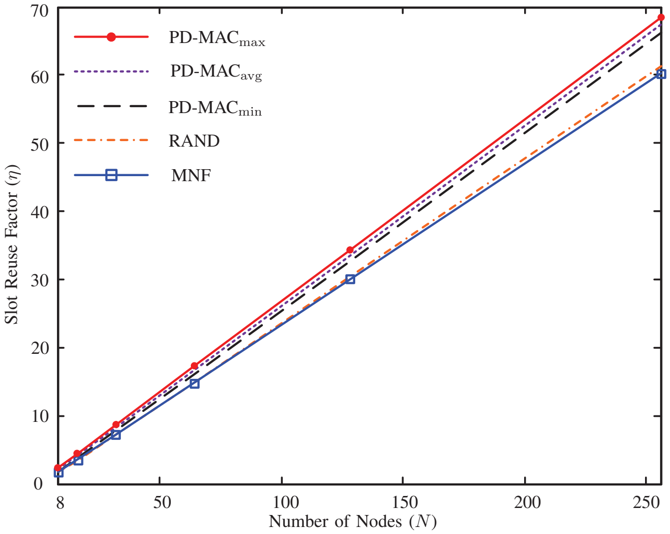


Fig. 2: Slot reuse factor comparison for different schemes.

We have implemented the PD-MAC protocol on a USRP-based test platform. The MAC protocol, underlying physical and application layers are programmed in C++ language in Qt-creator development environment. Fig. 3 shows screen shots captured from a real-time spectrum analyzer for a three-node network. The first screen shot shows when there is only one node in the network which occupies the first slot in a frame length of two. Subsequently, when the second node joins the network, it occupies the second slot in a frame length of four. When the third node arrives, it occupies the forth slot in a frame length of eight. For time synchronization, the nodes use slot boundaries of the ongoing transmissions on the channel as a reference as mentioned under Remark 1 and further discussed in [14], where we show that the nodes can stay synchronized in a perpetual way insomuch as there are valid transmissions on the channel. As a part of our planned work, currently, we are working on extending the implementation to a multi-hop network topology.

## V. CONCLUDING REMARKS

The TDMA based MAC scheduling is considered to be appropriate for many applications in which deterministic medium access scheduling is crucial. The plentiful literature on the TDMA based scheduling focuses on the conflict-free slot allocation. In those works, the problem of TDMA scheduling is broken into two disjoint problems which are sequentially solved: first find conflict free slot allocation, minimizing the number of slots used, and subsequently select frame sizes in which to use the assigned slots.

In this work, we showed that such an approach could lead to suboptimal performance when analyzed from the perspective of slot or channel reuse factor, which is the prime objective of the spatial reuse TDMA schemes. The paper formulates the optimization problem in which we aim to maximize the slot reuse factor whereby we jointly optimize the slots assigned to nodes and the frame length in which to use the assigned slots. The problem being inherently NP-hard, to solve it we proposed a greedy heuristic based solution according to which nodes complete slot assignment in a progressive decentralized

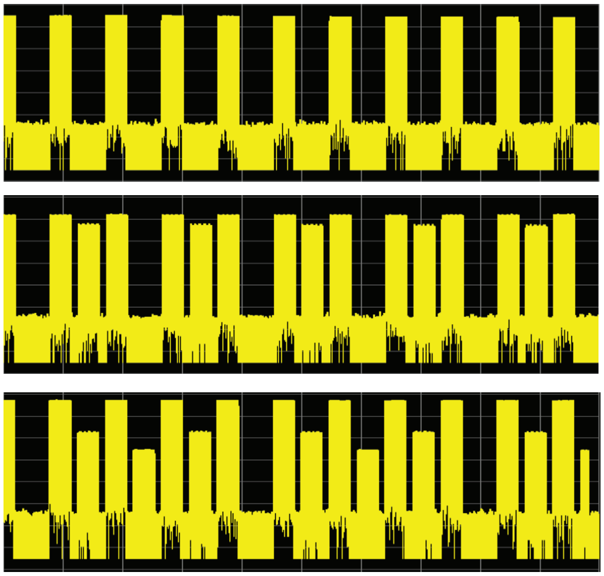


Fig. 3: Screen shots from a real time spectrum analyzer showing working of the PD-MAC protocol for a three-node network implemented on a USRP based test platform.

way. We showed that under the proposed algorithm, all nodes are guaranteed to find a conflict-free slot assignment. Besides, we showed that for a single node the convergence time of the algorithm is upper bounded by the node degree. While the convergence time for the whole network is upper bounded by the maximum node degree and the diameter of the network connectivity graph. Finally, with simulation examples, we showed that the proposed algorithm when compared with other TDMA scheduling schemes could give better performance in terms of channel or slot reuse factor.

## REFERENCES

[1] P. Suriyachai, U. Roedig, and A. Scott, "A survey of MAC protocols for mission-critical applications in wireless sensor networks," *IEEE Commun. Surveys and Tutorials*, vol. 14, no. 2, pp. 240–264, Mar. 2011.

[2] P. Suriyachai, J. Brown, and U. Roedig, "Time-critical data delivery in wireless sensor networks," in *Proc. of 6th IEEE Int. Conf. on Distributed Computing in Sensor Systems (DOCSS'10)*, Santa Barbara, CA (USA), June 2010, pp. 216–229.

[3] J. Brown, B. McCarthy, U. Roedig, T. Voigt, and C. J. Sreenan, "Burstprobe: debugging time-critical data delivery in wireless sensor networks," in *Proc. of European Conference on Wireless Sensor Networks (EWSN'11)*, vol. 2, Bonn (Germany), Feb. 2011, pp. 195–210.

[4] C. D. Young, "USAP multiple access: Dynamic resource allocation for mobile multihop multichannel wireless networking," in *Proc. of IEEE Military Commun. Conf. (MILCOM'99)*, vol. 1, Atlantic City, NJ (USA), Nov. 1999, pp. 271–275.

[5] A. Kanzaki, T. Hara, and S. Nishio, "An efficient TDMA slot assignment protocol in mobile ad hoc networks," in *Proc. of ACM Symposium on Applied Computing (SAC'07)*, Mar. 2007, pp. 891–895.

[6] I. Rhee, A. Warrier, J. Min, and L. Xu, "DRAND: Distributed randomized tdma scheduling for wireless ad hoc networks," *IEEE Trans. on Mobile Computing*, vol. 8, no. 10, pp. 1384–1396, Oct. 2009.

[7] I. Rhee, A. Warrier, M. Aia, , J. Min, and M. L. Sichitiu, "Z-MAC: A hybrid mac for wireless sensor networks," *IEEE/ACM Trans. on Networking*, vol. 16, no. 3, pp. 511–524, Jun. 2008.

[8] L. Bao and J. J. Garcia-Luna-Aceves, "A new approach to channel access scheduling for ad hoc networks," in *Proc. of 7th ACM Int. Conf. on Mobile Computing and Networking (SIGMOBILE'01)*, Rome (Italy), Jul. 2001, pp. 210–221.

[9] A. Rao and I. Stoica, "An overlay MAC layer for 802.11 networks," in *Proc. of ACM Int. Conf. on Mobile Systems, Applications, and Services (MobiSys'05)*, Seattle, Washington (USA), Jun. 2005, pp. 135–148.

[10] M. H. Chaudhary and B. Scheers, "High spatial-reuse distributed slot assignment protocol for wireless ad hoc networks," in *Proc. of Military Commun. and Inform. Systems Conf. (MCC'12)*, Gdańsk (Poland), Oct. 2012, pp. 1–8.

[11] E. L. Lloyd and S. Ramanathan, "On the complexity of link scheduling in multi-hop radio networks," in *Proc. of 26th Conf. on Inform. Science and System (ISS'92)*, 1992.

[12] S. Even, O. Goldreich, S. Moran, and P. Tong, "On the NP-completeness of certain network testing problems," *Jr. on Networks*, vol. 14, no. 1, pp. 1–24, 1984.

[13] S. Ramanathan, "A unified framework and algorithm for (T/F/C)DMA channel assignment in wireless networks," in *Proc. of 16th IEEE Int. Conf. on Computer Commun. (INFOCOM'97)*, vol. 2, Kobe (Japan), Apr. 1997, pp. 900–907.

[14] M. H. Chaudhary and B. Scheers, "Joint time synchronization and slot allocation in a TDMA based MAC," in *submitted in Military Commun. and Inform. Systems Conf. (MCC'13)*, France, Oct. 2013.

[15] S. Ganeriwal, R.Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proc. of 1st Int. Conf. on Embedded Networked Sensor Systems (SenSys)'03*, Los Angeles, CA (USA), Nov. 2003, pp. 138–149.

[16] M. T. Gastner and M. E. J. Newman, "Shape and efficiency in spatial distribution networks," *Jr. Stat. Mechanics (2006)*, pp. 1–15, Jan. 2006.