A Topology-Independent Scheduling Scheme for Wireless Mesh Networks

V.Loscri'
D.E.I.S. Department, University of Calabria
87036 Rende, CS, Italy
e-mail: vloscri @deis.unical.it

Abstract—**To meet the needs of wireless broadband access, the IEEE 802.16 protocol for wireless metropolitan networks has been recently standardized. The medium access control (MAC) layer of the IEEE 802.16 has point-to-multipoint (PMP) mode and mesh mode. Previous works on the IEEE 802.16 have focused on the PMP mode. In the mesh mode, all nodes are organized in an ad hoc fashion and use a pseudo-random function to calculate their transmission time. In this paper, we implemented the Coordinated Distributed Scheme (CDS) of the mesh mode of the IEEE Std. 802.16 in a well-known simulation tool, ns2. Through extensive simulations we tracked some characteristics of the CDS and after that we developed a different scheduling scheme, the Randomized-MAC (R-MAC). R-MAC is a totally distributed scheduling scheme and it tries to overcome some intrinsic "limit" of the CDS. We compared CDS and R-MAC and through extensive simulations, we observed that our R-MAC protocol works very well to cope with variations in the network. Our protocol outperforms CDS mechanism both in throughput and average end-to-end data packet delay. These results are related with the different mechanism to compute the next transmission time implemented in R-MAC.**

Index Terms— **Wireless, 802.16, MAC, scheduling scheme, mesh networks, TDMA.**

I. Introduction

THe IEEE 802.16 Working Group created a new standard, commonly known as WiMax [1, 2, 3], for broadband wireless access at high speed, at low cost, which is easy to deploy. The standard IEEE 802.16 [14] defines two modes of operation, Point-to-Multipoint (PMP) and Mesh mode. In the PMP mode traffic is directed from the Base Station (BS) to Subscriber Station (SSs, i.e. a common user), or vice-versa. Different to that within the Mesh mode, traffic can occur directly among SSs, without being routed through the BS (Mesh BS). As relatively new standard, IEEE 802.16 has been studied much less than access technologies as IEEE 802.11.

Eklund et al. presented an system level overview of 802.16 standards family in [15]. Redana and Lott modeled and compared the control message overhead between centralized and distributed scheduling mechanisms in [16]. From a different angle, Cao et al. proposed a theoretic model to compute the schedule interval of 802.16 coordinated distributed scheduling in [17]. With the algorithm to grant data requests left open in the standard, the schedule interval is an important common performance metric that reflects the scheduling latency of coordinated distributed scheduling. Other important works about mesh networks include QoS support in mesh mode [19] and cross-layer optimization of routing based on MAC layer scheduling behaviors [18, 20]. The IEEE 802.16 has three mechanisms to schedule the data transmission in mesh mode – centralized scheduling, coordinated distributed scheduling and uncoordinated distributed scheduling. In centralized scheduling, the BS works like a cluster head and determines how the SS's should share the channel in different time slots. Because all the control and data packets need to go through the BS, the scheduling procedure is simple, however the connection setup delay is long. Hence, the centralized scheduling is not suitable for occasional traffic needs [4]. In distributed scheduling, every node competes for channel access using a pseudo-random election algorithm based on the scheduling information of the two-hop neighbors. The distributed channel access control is more complex because every node computes its transmission time without global information about the rest of the network. In this work, we implemented the Coordinated Distributed Scheduling scheme CDS of the mesh mode of the Std IEEE 802.16 in a well known simulation tool, ns-2 [5]. After we tracked some considerations about characteristics of the CDS, we designed another scheduling scheme, Randomized-MAC (R-MAC) that works in a totally distributed fashion and is based only on local information. The main difference between R-MAC and CDS is in the computation of the next transmission time. In fact, in CDS scheme we have to set two different parameters as we will detail in section III in order to calculate a time interval in which the node is not eligible to compete. We tried to release our scheme from this characteristic. In fact, our feeling is that the need to set some parameters that influences the number of competing nodes makes a scheme not sufficiently robust in different network conditions. Another characteristic of the CDS is that control slots (*Opportunities to Transmit*) are managed in a way that a certain amount of slots is unassigned even if the density of the network is high and each node has a high number of neighbors (1 and 2 hop neighbors). The Randomized-MAC (R-MAC) is totally distributed and requires only local information to compute the schedules. Specifically, it addresses schedule updates in the face of network change. R-MAC protocol differs from the IEEE 802.16 standard in that it increases the

WICON 2007, October 22-24, Austin, USA
Copyright © 2007 978-963-9799-12-7
DOI 10.4108/wicon.2007.2106

speed with which the schedules are calculated, provided a reasonable degree of bandwidth efficiency is achieved and a reasonable degree of fairness is kept. The remainder of this paper is organized as follows. In Section II we summarize the network model that we have used. In Section III we present details about Coordinated Distributed Scheduling scheme of the IEEE 802.16. Section IV presents the proposal of the new scheduling scheme mechanism called Randomized-MAC (R-MAC). Section V, presents the throughput and delay results of both schemes, CDS and R-MAC. Section VI concludes the paper summarizing the main ideas presented.

II. NETWORK MODEL

In this paper we consider a Wireless Mesh Network as composed of three distinct network elements (Fig. 1):

- Network Gateway: one (or more gateway, Wired Internet Backbone) can be deployed to allow access to a different IP sub-network.
- Access Points (Mesh Routers (MRs) or Subscriber Stations (SSs)): the access points form a wireless backbone, providing connectivity in places otherwise difficult to access through traditional wired infrastructure. We assume the access points use IEEE 802.16 technology.
- Mobile Nodes (Mesh Clients or Wireless Clients): we consider as terminal user any device that can access the network gateway through direct or multi-hop communication (using the access points as relays). PDAs, laptops etc. can be considered available devices.

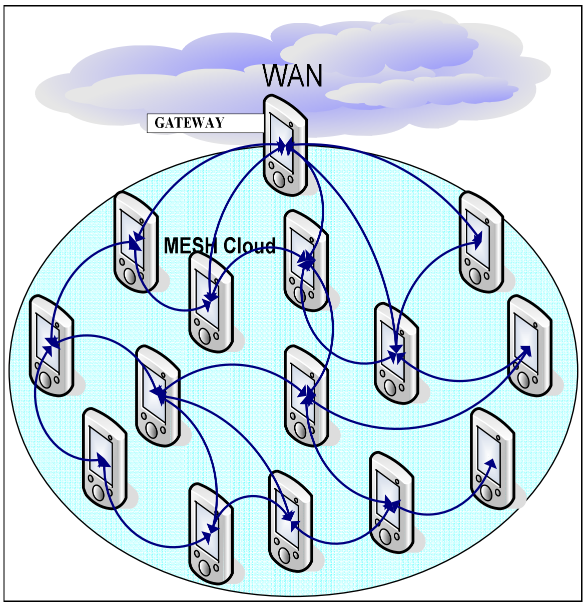

Fig. 1 Wireless Mesh Network. The Gateway node provides Wide Area Network (WAN) connectivity to all other Mesh nodes.

We assume that Access Points are fixed and static in the network. Whereas each mobile node can possibly communicate with any other mobile node in the network, most expected traffic streams will occur between the mobile nodes and the network gateway [11].

A. *Node States description*

In this paper we consider two different scheduling schemes, the CDS and R-MAC. Both schemes are used in a Time Division Multiple Access protocol at MAC (Medium Access Control) layer. With constraints required by conflict-free TDMA transmissions, the activity of a node n_i in a slot s can be classified into the following states:

- TX: Transmits to a set of neighbors R: (*state(s) = Transmit, target(s) = R*)
- RX: Receive from a neighbor n_j: (*state(s)=Recv, target(s) =* n_j)

If a node is not transmitting or receiving in this slot, it is in one of the following (passive) states:

- Blocked from transmitting because at least one of its neighbors receives from another node, and none of its neighbors transmits: *(state(s) Block_TX)*,
- Blocked from receiving because at least one neighbor is transmitting to another node, and none of its neighbors receives: (*state(s) = Block_RX*),
- Both Blocked from transmitting because at least one neighbor is receiving, and blocked from receiving because at least another neighbor is transmitting: (state(s) = *Block_TX_RX*),
- Experiencing a collision when it is supposed to receive from a neighbor (*state(s) = Collision*),
- Idle, when none of its neighbors transmits or receives in this slot: (*state(s) = Idle*).

Naturally, these states are mutually exclusive.

III. IEEE 802.16 MESH MODE

A. *General Description of IEEE 802.16*

Almost all the existing works about the IEEE 802.16 are on the PMP mode [6, 7, 8]. The main difference between PMP and Mesh mode is that in the PMP mode, traffic only occurs between the BS (Base Station) and a SS, while in the Mesh mode, traffic can be relayed via other SSs, and also can occur directly between SSs (or MRs). Comparing with the tree-based multi-hop network topology of 802.16 tree-based mobile multi-hop relay (MMR) [9, 10], 802.16 mesh mode places more challenges on the link scheduling algorithms. In order to achieves efficient collision-free multi-hop data transmission, the Mesh mode defines three scheduling schemes, i.e., centralized, coordinated distributed and uncoordinated distributed scheduling, to resolve wireless interference occurred in the 2-hop neighborhood of a node. Each frame in the IEEE 802.16 standard is divided into two parts: i) a control sub-frame consisting of MSH_CTRL_LEN (0-15) transmission opportunities (*XmtOps* in the standard) and ii) a TDM data sub-frame consisting of up to 256 minislots. All the *XmtOps* are in fixed length of 7 OFDM symbols. There are two types of control sub-frame, i.e., schedule control sub-frame and network control sub-frame. Schedule control sub-frame manages bandwidth resources and permits control slots to be assigned and schedules to be created. Network control sub-frame manages configuration of the network, i.e. a new node enters the network, etc.

B. Coordinated Distributed Scheduling (CDS)

Coordinated distributed scheduling (CDS) is designed to achieve collision-free periodical transmissions for two types of control messages, i.e., MSH-NCFG and MSH-DSCH, respectively. The first one, MSH-NCFG is used to manage the configuration of the network (i.e., a new node enters the network, either a node switches on or switches off) and the second, MSH-DSCH is used in the distributed scheduling scheme. Since the exact same algorithm is used independently for these two types of messages in separated *XmtOps* , we can simply analyze the behaviors of one, and the result is applicable to the other. In this part, a general introduction about the IEEE 802.16 distributed scheduling behavior is given. In the 802.16 scheduling algorithm, the control message and data packet are allocated in different time slots in a frame. The allocation of the data time slots is performed through the control message exchange so that there is no contention in the data time slots. In the distributed scheduling, a node selects its next transmission time in the current one. Because other nodes may also transmit in the selected time slot so that the node uses an election algorithm to compute whether it can win or not. The general concept of CDS is to let nodes running the scheduling algorithm independently derive pseudo-random but predictable behaviors by exchanging 2-hop (or 3-hop) neighborhood schedule information with each other. Both randomness and predictability are achieved by dynamically constructing random generator seeds for each node according to a common rule. The seed for a node is based on its unique node ID and the index (or timestamp) of a candidate *XmtOp.* Given the neighborhood information, the random number generated locally will be the same with the corresponding one generated at a neighboring node. In distributed scheduling, the scheduling information for each station is described by two parameters:

$$\begin{cases} NextXmtXm :: 5bits \\ XmtHoldoffExponent :: 3bits \end{cases} \tag{1}$$

Given these two parameters of a specific neighbor, a node can determine a bounded interval for *NextXmtTime* as well as the following:

$$NextXmtXm * 2^{XmtHoldoffExponent} \prec NextXmtTime \leq (NextXmtXm + 1) * 2^{XmtHoldoffExponent} \tag{2}$$

$$EarliestSubsequentXmtTime = NextXmtTime + 2^{XmtHoldoffExponent+4} \tag{3}$$

$$XmtHoldoff\ Time = 2^{(XmtHoldoff\ Exponent\ +4)} \tag{4}$$

XmtHoldoffTime is the number of MSH-NCFG/MSH-DSCH transmit opportunities after *NextXmtTime* (there are *16 MSH-CTRL-1*) that this station is not eligible to transmit MSH-NCFG/MSH-DSCH packets. For example, if *NextXmtMx* = 2 and *XmtHodoffExponent* = 4 the station would be eligible between the 33 and 48 transmissions opportunities.

Every node calculates its *NextXmtTime* during the current transmission according the distributed election algorithm [2]. In this algorithm one node sets the first transmission slot just after the *XmtHoldoffTime* as the temporary next transmission opportunity. In this instant this node shall compete with all the competing nodes in the two-hop neighborhood (this node is called Node A). There are different types of competing nodes (Fig. 2):

- *NextXmtTime* includes the temporary transmission slot (Node B).
- *EarliestSubsequenceXmtTime* (equal to *NextXmtTime* + *XmtHoldoffTime*) is ≤ the temporary transmission slot (Node C)
- The *NextXmtTime* is not known (Node D).

This algorithm is a pseudo-random function which uses the slot number and the Node's ID as the inputs, this algorithm is executed in each node. It generates pseudo-random values depending in the input. The node wins when its result is the largest mixing value (Fig. 3).

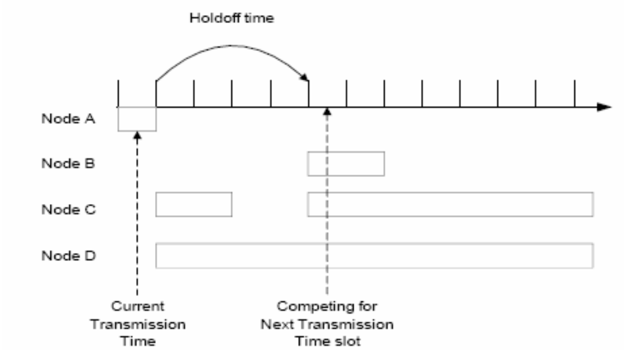
Holdoff time
Node A
Node B
Node C
Node D
Current Transmission Time
Competing for Next Transmission Time slot

Fig. 2 Competing Nodes for Next Transmission time slot.

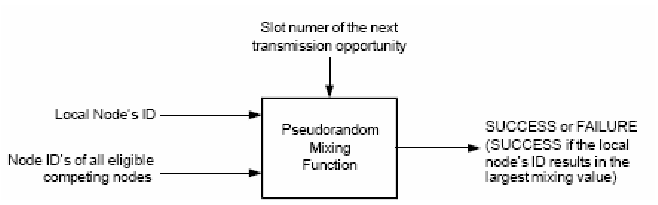
Slot numer of the next transmission opportunity
Local Node's ID
Node ID's of all eligible competing nodes
Pseudorandom Mixing Function
SUCCESS or FAILURE (SUCCESS if the local node's ID results in the largest mixing value)

Fig. 3 Pseudorandom Mixing Function.

When any node wins, it sets the temporary transmission opportunity as its next transmission time and logically it shall communicate this information to all the neighbors by sending the corresponding packet. In the case a node has not won, it chooses the next transmission opportunity and repeat the algorithm as many times as it needs to win. Since the exact scheduled *XmtOp* of the neighborhood is unknown, as implementation issue, one may define *NextXmtTime* to be the last *XmtOp* within the interval when calculating *EarliestSubsequentXmtTime*. The holdoff exponent value decides the channel contention time of node so it is an important parameter that can affect the system performance.

IV. A NEW DISTRIBUTED SCHEDULING SCHEME: RANDOMIZED-MAC

Based on the description of the Coordinated Distributed Scheduling scheme CDS of the Std. IEEE 802.16 (CDS), every node competes for the channel access and tries to broadcast its scheduling information periodically (a node can transmit its schedule and can try to reserve new data slots in the current *XmtOp*). The channel contention result is correlated with the total nodes number, exponent value and network topology. A key factor in a similar scenario is represented by the capacity to use control slots in an effective fashion and some parameters as *XmtHoldofExponent* and *NextXmtXm* have to be opportunistically set in order performance to be improved. We will show, through simulation results , that the usage of control slots of the CDS is not optimal, that is we will measure the number of unassigned control slots in each frame. This parameter permits to understand the dependence of the topology of the network with the scheduling scheme.

A. Details of the proposed Randomized-MAC (R-MAC)

The proposed scheme differs from the IEEE 802.16 Coordinated Distributed Scheduling scheme in the selection of the new *XmtOp*. In fact, a node transmitting in the current *XmtOp* runs a Random Function to select the next *XmtOp* instead to consider the Hash Function (MeshElection) of the 802.16 standard. The frame structure is shown in Fig. 4. The scheduling mechanism for a Local Node (LN) is described in Fig. 5. In Fig. 5 Random Function is a function that randomly picks-up an available slot. A slot is available if the state in the next frame of the Local Node is IDLE. The Redistribution Function is applied if and only if a node did not find an available *XmtOp* slot in the previous frame or it lost it for some reason as shown in Fig. 4. Redistribution Function is a Random Function in which a node, that needs an *XmtOp* for the current frame, analyzes a set of available slots (the state of the Local Node is IDLE in this slot) and randomly selects one of the available slots. In this way all the neighbors that does not have a valid *XmtOp* slot in the current frame will apply the same random function and it can happen, above all when density network increases, that two 1 or 2 hop neighbors select the same control slot. In this case there will be a collision which will be resolved in the next frame, in which each node will compute a new control slot. The main difference between R-MAC and CDS is in the assignment mechanism of control slots. In CDS based on the setting of specific parameters, a node has to wait for a certain amount of time before to compete anew. R-MAC generates schedules not necessarily conflict-free, but the assignment of control slots is realized in a more "aggressive" fashion that permits to use almost all control slots in each frame as we will see in the Results Section. Changing network topology implies that the parameters as *XmtHoldoffExponent* and *NextXmtXm* have to be set anew in CDS. To the contrary we have to set no parameters in R-MAC. Each node considers a number of *XmtOp (opportunities)* that is 16 as in the IEEE 802.16 standard and this is the number of opportunities to transmit (control slots used to update neighbors and make new data slots requests).

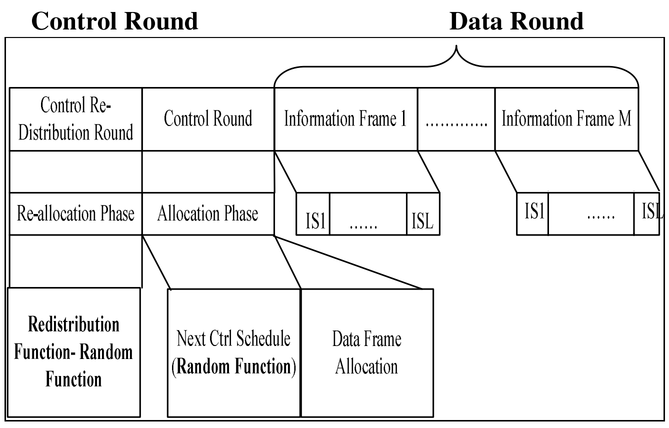

Fig. 4 Frame Structure of the Randomized-MAC (R-MAC). At the beginning of each control frame, Re-Distribution Round is applied, it will be applied only for the nodes that did not maintain a valid *XmtOp* for the current frame. The Redistribution Function considered is the same Random Function used in the Allocation Phase. In the Control Round each node computes the next *XmtOp* applying a Random Function and sends updated information in the current *XmtOp*. Data slots allocation takes place in a different portion of the frame.

The channel is partitioned in two sub-portions: a Control Round where the schedules are updated and Data Round where user data transmission takes place (Fig. 4). The state (one of the states considered in Section II.A) of the nodes in each slot is updated during the control round when each node in the current *XmtOp* computes the next *XmtOp* and sends the updated information to the neighbors. The info-schedule (or data schedule) will be updated at the beginning of each new frame (Data Frame Allocation); in this way different transmission requirements nodes can be accommodated. Two different information are considered in the Control Round: 1) current information (or current schedule) and 2) next information (or next schedule). The current schedule is the actual schedule used by node to transmit and to compute another *NextXmtTime* (in accordance with the IEEE 802.16 notation, that is the next *XmtOp*). After the Random Function is called and the *NextXmtTime* is computed a node updates its schedule and sends this information (the updated schedule) to its neighborhood (1 and 2-hop away nodes) through the message (MSH-DSCH) where the request of a certain bandwidth (data slots requests) is contained.

Let us assume that during a frame (F_{curr}) a node tries to acquire a new slot (*XmtOp*). Once a node receives information from the neighbors it updates neighbors information and it can happen that the slot it tried to reserve has already been reserved from another node. In these conditions the Local

Node (LN) sets its state in this slots as *Block_RX* (see Fig. 6).

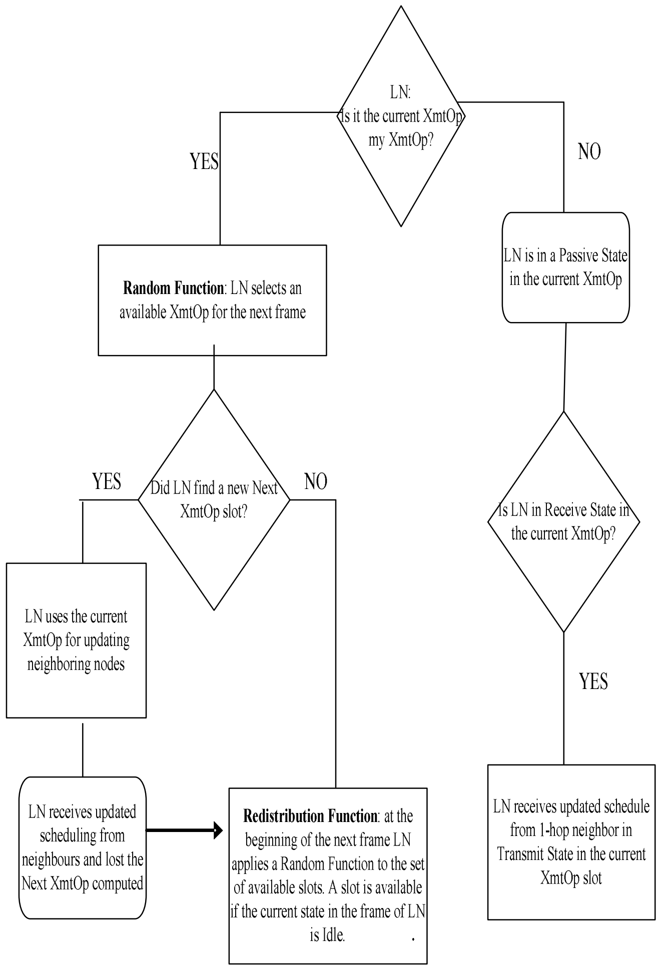
LN: Is it the current XmtOp my XmtOp?
YES
NO
Random Function: LN selects an available XmtOp for the next frame
LN is in a Passive State in the current XmtOp
Did LN find a new Next XmtOp slot?
YES
NO
Is LN in Receive State in the current XmtOp?
LN uses the current XmtOp for updating neighboring nodes
YES
LN receives updated scheduling from neighbours and lost the Next XmtOp computed
Redistribution Function: at the beginning of the next frame LN applies a Random Function to the set of available slots. A slot is available if the current state in the frame of LN is Idle.
LN receives updated schedule from 1-hop neighbor in Transmit State in the current XmtOp slot

Fig. 5 Flow Chart of Local Node (LN) scheduling mechanism of R-MAC.

Once the situation as described happens LN remains without an opportunity to transmit in the Frame ($F_{next\text{--}curr}$) even if there are some unassigned slots.

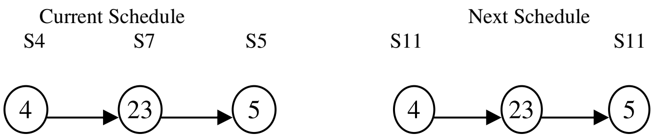

Fig. 6 Current Schedule: node 4 is transmitting in the slot S4, node 23 is transmitting in the slot S7 and node 5 is transmitting in the slot S5. Node 4 reserves the slot S11, in the current frame for the next frame, as *XmtOp*. This reservation takes place in the slot S4. Node 4 transmits the updated schedule to its neighborhood. Node 23 is notified with the updated schedule. In the next slot, S5, node 5 reserves the same slot S11 (selected randomly). Node 5 will be notified that this slot has already been reserved by another neighbor node (node 4) in the slot S7 by node 23. Node 5 will set its state in the slots S11 as state(S11) = Block_RX.

Based on these considerations we developed a scheduling scheme in which a Re-Distribution Control sub-phase is implemented. In this way we try to use a higher number of control slots. We would outline that our approach does not introduce any additional different overhead packet with respect to the standard approach (CDS). In fact, the scheme developed is based only on a different computation of the *XmtOp*. By the way, it can happen that some more control MAC packets is sent using R-MAC due to an higher slots assignment. We will show an estimation of this potential incremental overhead in the next Section.

V. Performance Evaluation

In order to evaluate the effectiveness of our algorithm we implemented it in a well-known simulation tool, ns2 [5]. Moreover, we implemented CDS of the Std IEEE 802.16 in ns2. In fact, in ns2 there exists a MAC module comprising the Std. IEEE 802.11, but there not exist yet a CDS module. We created different networks scenarios. Specifically, we focused on varying the number of nodes in the network in order to create different network densities. We believe that this kind of simulation campaigns is effective to show in different conditions how the two mechanisms work. The CDS MAC module we implemented consists of the scheduling controller that handles the signalling channel contention and in the current transmission slot contends the next transmission time using the election mechanism defined in the standard based on the collected neighbors' information. Another fundamental component of this module is the data channel that receives and transmits data packets in the allocated time slots. We used different *XmtHoldoffExponent* and *NextXmtXm* for each scenario, selecting a couple of values that permits better performance in terms of throughput and delay to be obtained. Specifically, different simulation campaigns have been conducted with the same traffic load and the same topology and different *XmtHoldoffExponent* and *NextXmtXm* values and the better values, in terms of delay and throughput, have been selected. Each node of the network is assigned the same couple of *NextXmtXm* and *XmtHoldoffExponent* during a simulation run. The new scheduling scheme introduced, the R-MAC, does not present this characteristic and presents a higher robustness in this sense. In fact, in R-MAC, we are released from the constraint that a parameter based on network density considerations or network size has to be set. Parameters evaluated in the simulation campaigns in this work are:

- *Throughput*: is obtained by dividing the number of data packets delivered at each destination with the data packets originated by the sources.
- *Delay*: is the average end-to-end delay and includes all delays caused by buffering during route discovery phase, queuing delay at the interface, retransmission delay at the MAC, propagation and transfer times;
- *Unassigned Control Slots*: is obtained as the ratio of the number of control slots that are not assigned and the total number of control slots in the simulation time.

In order to evaluate these parameters we considered AODV routing protocol for all the simulation scenarios. In this paper, we evaluated performance of wireless mesh networks in tree-based architectures. The node 1 works as a gateway and the others nodes work as Access Points (Subscriber Stations SSs, or Mesh Routers in Fig. 1). We simulated the traffic of terminal users by changing traffic loads at the Access Points. In all the figures, R-MAC represents the new randomized distributed scheduling scheme developed in this work (R-MAC), 802.16 represents Coordinated Distributed Scheduling (CDS) scheme of the Std 802.16.

A. Results

In the experiments conducted in this work we varied the total number of nodes in the network in order to consider a different nodes density and the impact of varying the density in the network maintaining the same traffic load. Moreover, we also varied the traffic load in the network and we evaluated the impact of an heavier traffic load on the performance of the network. In Fig. 7 we show the message format of the MSH-DSCH. This is useful in order to roughly evaluate the additional overhead introduced by our scheme. Remember that we do not add any new control packet but there could be some additional overhead because more control slots are assigned to nodes with our scheme. As we have already seen, we considered two different data traffic load. Fig. 8 shows throughput when light traffic load is considered. In fact, 20 sources generate traffic at different rates, from 4 pkts/sec to 400 pkts/sec in a 1000x1000 sq meters grid in which nodes are evenly distributed for a total of 200000 data packets in 500 sec of simulation. We can observe that R-MAC outperforms CDS in terms of throughput. This is due to the fact that R-MAC permits a better usage of control slots to be realized. The lower is node density in the network, the lower is the positive impact of R-MAC against CDS. When density in the network increases each node has an higher number of neighbors (1 and 2 hop neighbors) and the number of competing nodes increases too. The R.MAC throughput is higher than CDS throughput, above all, at 50 and 60 nodes. When the number of competing nodes in the network increases the positive impact of the Re-Distribution phase in R-MAC is more evident in the network. In fact, at 50 and 60 nodes performance in terms of throughput is better than 40 nodes. This is due to the fact that when the number of competing nodes is much more lesser than the number of *Opportunity to Transmit* (equal to 16, in both of schemes), the probability for a node to acquire a control slot, that is an *Opportunity to Transmit* is higher than when the number of competing nodes is higher than the number of *Opportunities to Transmit*. The R-MAC Re-Distribution phase permits to have some additional chance to use unassigned *Opportuniiesy to Transmit*. The difference between CDS and R-MAC is also related with the assignment phase of the control slots. In fact, in R-MAC each node competes in each round through a Random Function. Randomness of the function permits to say that the R-MAC mechanism is fair, and we do not introduce any priority and nodes are not constrained to wait any specific order.

For this reason we can affirm that the mechanism of R-MAC does not introduce nodes starvation. Once the number of competing nodes increases the latency introduced by CDS scheme increases too. Worth to note that the values of *XmtHoldoffExponent* and *NextXmtXm* would be different for different traffic needs of a Mesh Router (MR). In fact, if a node needs to transmit more data traffic than other nodes the "waiting" time before to compete again would be smaller. Unfortunately, CDS does not introduce a dynamic mechanism to manage these slots and we believe that this is the main reason that this mechanism does not work so well when the density in the network increases, above all in terms of average end-to-end data packet delay. Delay results drawn in Fig. 9 confirm as we have observed in terms of throughput. In fact, at 50 and 60 nodes, where we obtained good performance in terms of R-MAC throughput, the delay is smaller in respect of the other network densities and in respect of the CDS. This is due to the fact that the latency introduced with our scheme is smaller than the latency introduced by CDS. In our mechanism each node can compete in each round and the fairness is ensured thanks to the randomness of the function used to set the next control slot. In this way we do not introduce any "waiting" time. Moreover, our mechanism introduces a Re-Distribution phase that permits a better use of control slots to be realized and a more compact frame usage to be obtained. As we have already observed, the schedules generated with the R-MAC are not necessarily conflict-free. This undesired condition is due to the Re-Distribution phase, in which each node does not have sufficient information about all the network. In the Re-Distribution phase each node knows the own condition and the control slots that are potentially available to be set as Transmission Time slot for the current frame. Unfortunately, two neighbor nodes could be in the same situation, that is without an available control slot for the current frame and could try to randomly catch the same control slot, in this case, there will be a collision in the current frame. It is worth to note that this collision will be resolved in the next frame where each node computes a new control slot (*Opportunity to Transmit*). In order to evaluate the effectiveness of our scheme we introduced a heavier data traffic load (data packets introduced in the network in the second scenario is doubled in respect of the first scenario analyzed) as shown in Figs. 10 and 11 where throughput and delay have been evaluated considering 40 sources. We can observe as performance in terms of throughput and delay are worst in this case in respect of the previous case for both schemes. In fact, Fig. 8 shows that R-MAC throughput varies from 60 to 72 % and CDS throughput varies from 54 to 67 % while the same schemes present a throughput varying from 43 and 53% and 38 and 47% for R-MAC and CDS respectively.. The same consideration is available for the delay. In fact, delay degenerates when more data sources are introduced. Figs. 10 and 11 show as R-MAC outperforms CDS mechanism even if data traffic load increases. When higher traffic load is considered the better usage of control slots realized trough the R-MAC in respect of the CDS mechanism is more evident. In order to confirm that a better usage of control slots is realized through the R-MAC, we evaluated the parameter called Unassigned Control Slots (Fig. 12). This parameter reveals the differences between the management of the control slots of the two scheduling mechanisms. Remember that control slots are used to reserve new data slots and to update neighbors schedules. In fact, when a node acquires a new *Opportunity to Transmit (XmtOp)* it will send a Control Message (MSH-DSCH) in order to update its neighborhood about its schedule and it will try to reserve new data slots if it needs. In Fig. 12 we show the percentage of unassigned control slots. Of course, when the number of nodes is smaller in the network the number of competing nodes is smaller too. In this case the number of *Opportunities to Transmit* (*XmtOp*) is sufficient to "cover" the requests of the nodes in the network. Let us to consider, for example, the situation of the network when the number of nodes is 30. We can compute the average number of neighbors in two different ways: 1) Analytical way and 2) Simulated way.

When using the first approach the number of neighbors is evaluated as:

$$N^{'} = \rho \pi r^{2} \qquad (5)$$

where N' is the number of 1-hop neighbors

r is the transmission range of each station fixed to 250 m

ρ is the density of the network.

we write ρ as

$$\rho = \frac{N}{\pi R^{2}} \text{ and } N^{'} = N(\frac{r}{R})^{2} \qquad (6)$$

where N is the number of nodes in the network

R is the grid dimension fixed to 1000x1000 sq meter

In this case we obtain an average number of neighbors equal to 1,875 when we consider 30 nodes. Considering simulated scenarios we have an higher number of neighbors equal to 3,5. We chosen to use this value. As we considered a number of control slots equal to 16, each node will be assigned an *Opportunity to Transmit* when the number of nodes is small (30 and 40 nodes). When the number of nodes increases in the network, the mean number of neighbors and consequently the mean number of competing node for each node increases too.

This means that the number of control slots is not sufficient each node to be assigned a control slot. When observing the network behavior at 30 and 40 nodes in Fig. 12 we can conclude that the mechanism used by R-MAC to assign slots uses different control slots instead of re-assign in a conflict free fashion control slots already assigned. In fact, the number of control slots unassigned at 30 and 40 nodes as far as R-MAC is concerned is almost zero and this means that almost all control slots in each frame are assigned. This behavior can be associated with the randomness of the R-MAC. When the number of nodes increases the evaluation of the curve has to be made from a different point of view. In fact, when the number of competing nodes is approaching the number of control slots (16) (at 50 nodes the mean number of neighbors for each node in the simulated scenarios is 17), both scheduling mechanisms will try to assign all control slots. In fact, CDS curve slope of Fig. 12 decreases rapidly.

For the same reason is logical that the mean number of unassigned slots increases when using R-MAC mechanism. In fact, when the number of competing nodes increases each node will try to be assigned a control slot, but the competition is higher and some slots can be unassigned or there could be a collision slot assigned during the Re-Distribution phase. Also when changing the network density we can observe as the distribution of control slots is better managed through our approach. In Figs. 8 and 10 we can observe that throughput increases when considering 80 nodes instead of 70 nodes. This apparently strange behavior is due to the mean number of hops of paths. In fact, we checked the mean number of hops of the paths with 70 and 80 nodes in the simulated scenarios, computed through a shortest path approach (Dijkstra mechanism) and we discovered that in the simulation scenarios created through ns2, the mean number of hops when 70 nodes are considered was higher than the mean number of nodes for 80 nodes. This is confirmed when observing the average end-to-end delay in Figs. 9 and 11. In fact, the delay does not increase when we evaluate the network at 70 and 80 nodes in Fig. 11 and it decreases when we pass from 70 to 80 nodes in Fig. 9. It is worth to notice that some more additional overhead could be introduced in terms of control MAC packets through our mechanism. In the worst case the additional number of control packets that has to be sent is 16% more MSD-DSCH (at 30 nodes in Fig. 12).

As far as additional control overhead, at MAC layer, introduced by the R-MAC, is concerned, we roughly evaluated it. In practice, we observed in the Fig. 12 that the worst case, regarding the control overhead, is represented by the maximum distance between two correspondent points of the two curves. In our case this value is obtained in correspondence of 30 nodes. In fact, in this case R-MAC could send more information than CDS (notice that in this case we are evaluating the overhead considering that the average number of control slots CDS scheme assigns during a simulation is 84 % of the total available control slots. This estimation is not so accurate because some slots could be assigned to different nodes that are not neighbors to each other. In practice, some control slots could be re-used in a conflict-free fashion in CDS and R-MAC could assign different slots even if nodes are far away to be assigned the same control slot). Let us to compute how much more information R-MAC sends in the worst case. In [4] the MSH-DSCH message format is indicated. The constant number of bits that a node sends in an *XmtOp* is given by the sum of the different fields in the MSH-DSCH message (see Fig. 7). The variable field MSH-DSCH-Scheduling_IE() varies in dependence with the number of neighbors. As we already seen we can use two different approaches to evaluate the mean number of neighbors for each node. We choose to consider the simulated approach in which the mean number of neighbors with 30 nodes in the networks has been estimated to be 3,5.

The variable field of the MSH-DSCH message format we are considering is the MSH-DSCH-Scheduling-IE(). It depends of the number of neighbors and is

16 bits + 24 * #Neigh (No_SchedEntries)

So in the worst case we have an additional overhead information to send, in terms of bits of

Total Information = 220 * 2,5 = 550 bits

Management Message Type = 41	8 bits
Coordination Flag	1 bit
Grant/Request Flag	1 bit
Sequence Counter	6 bits
No. Requests	4 bits
No. Availabilities	4 bits
No. Grants	6 bits
Reserved	2 bits
MSH-DSCH-Scheduling_IE()	*variable*
MSH-DSCH-Request_IE()	16 bits
MSH-DSCH-Availability_IE()	32 bits
MSH-DSCH_Grant_IE()	40 bits

Fig. 7 MSH-DSCH message format.

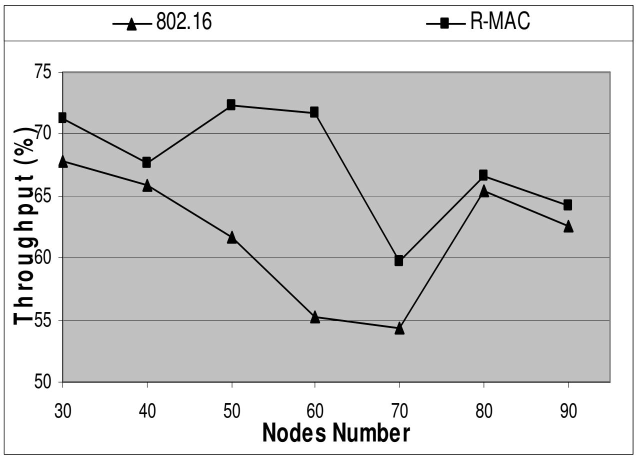

Fig. 8 Throughput: percentage of packet delivered to a destination vs the number of nodes (20 sources).

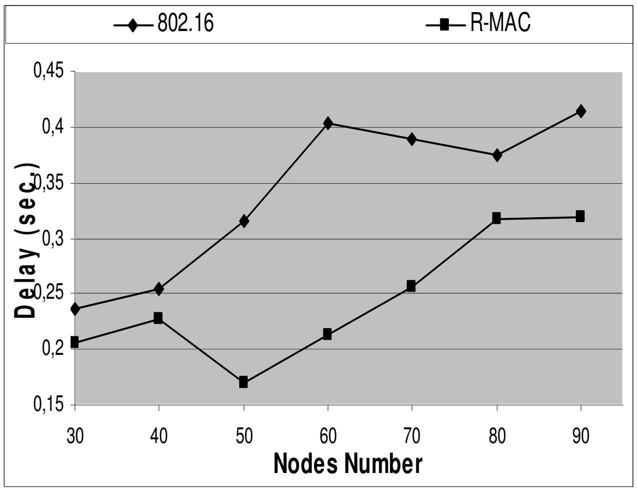

Fig. 9 Average end-to-end data packet delay (20 sources).

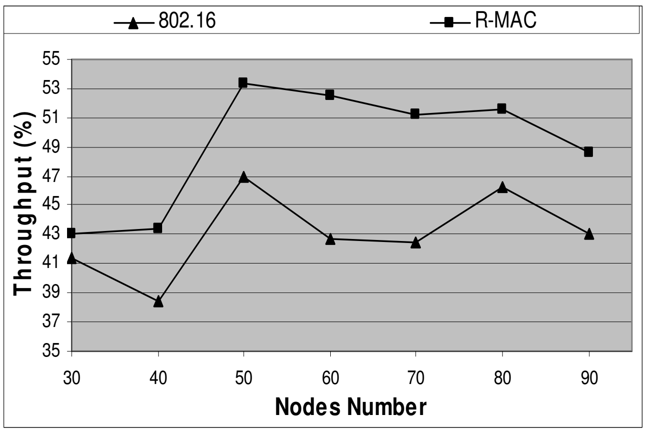

Fig. 10 Throughput: percentage of packet delivered to a destination vs the number of nodes (40 sources).

The rough estimation of the additional overhead introduced permits to conclude that there is no significant additional overhead with R-MAC in respect of the CDS scheme, above all considering better performance in terms of throughput and delay that we have obtained. Another important consideration is that we developed our scheme releasing it from the need to set parameters based on density network considerations.

VI. CONCLUSIONS

In this paper we proposed a new totally distributed scheduling scheme, Randomized-MAC (R-MAC), that permits conflict-free schedules to be built with high probability. The novelty in the scheme is represented by a Re-Distribution phase in which control slots that are unassigned can be assigned to some node that needs it. This Re-Distribution phase is realized in a totally random way and this randomness permits fairness among the nodes to be ensured. Unfortunately, two different nodes, neighbors (1 or 2 ho neighbors) are not able to exchange any information in this portion of the frame and for this reason if they catch the same slot a collision will be determined in the current frame. Moreover, we implemented the Coordinated Distributed Scheduling scheme (CDS) of the Std. IEEE 802.16 in ns2. We studied performance in terms of throughput and delay of both schemes, the R-MAC and CDS. Simulation results are very interesting. Even if CDS permits conflict-free schedules to be built, R-MAC outperforms it in terms of throughput and delay. This is due of the different policy applied to compute schedules that permit a better usage of control slot to be obtained in R-MAC than CDS. Furthermore, R-MAC is more robust than CDS in different network scenarios, because in CDS we need to set two parameters in order to determine the competition window of each node. The new scheme we developed has been implemented without this characteristic. In fact, we do not need to set any specific parameter based on density and traffic consideration as well as in CDS scheme. As future work we intend to study the possibility to implement a new totally distributed scheduling scheme that permits conflict-free schedules to be built without the need to set any parameters as in CDS.

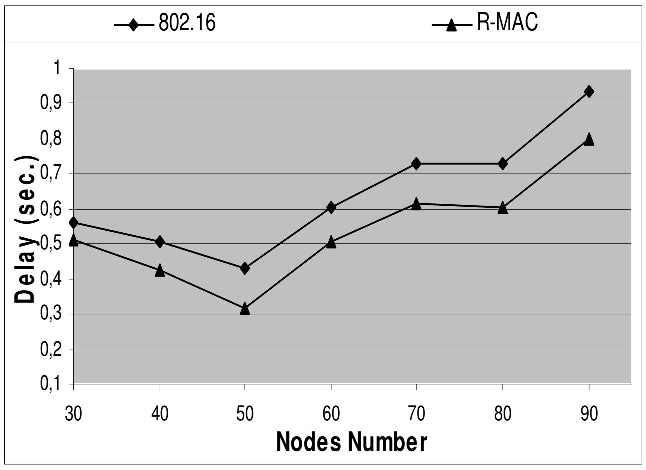

Fig. 11 Average end-to-end delay (40 sources).

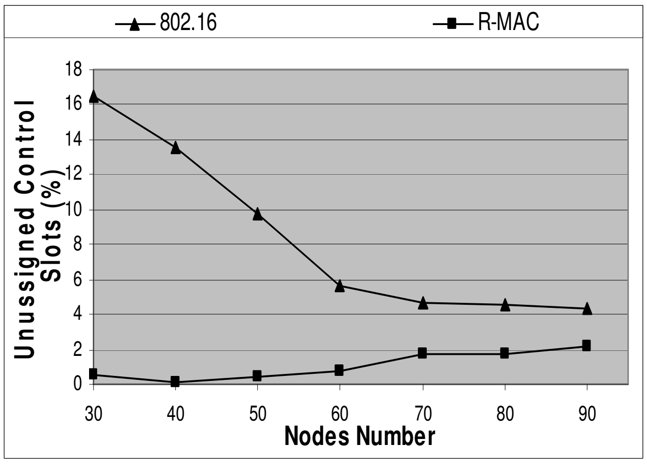

Fig. 12 Percentage of unassigned slots over the total number of slots in a simulation vs the number of nodes.

REFERENCES

[1] I.F. Akyildiz, X. Wang, and W. Wang. Wireless Mesh Networks: A survey. *Computer Networks Journal (Elsevier)*, March 2005.

[2] IEEE Std 802.16a-2003, "IEEE Standard for Local and metropolitan area networks—Part 16: Air Interface for Fixed Broadband Wireless Access Systems—Amendment 2: Medium Access Control Modifications and Additional Physical Layer Specifications for 2-11 GHz," 2003

[3] IEEE Std 802.16-2004 (Revision of IEEE Std 802.16-2001), "IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Fixed Broadband Wireless Access Systems," 2004.

[4] URL: http://wirelessman.org/tga/contrib/C802.16a-02_30r1.pdf.

[5] Ns (network simulator), 1995. URL: http://www-mash.cs.berkeley.edu/ns/.

[6] Christian Hoymann, Markus Puttner, and Igo Forkel. The HIPERMAN standard – a performance analysis. IST SUMMIT 2003.

[7] Guosong Chu, Deng Wang, and Shunliang Mei. "A QoS architecture for the MAC protocol of IEEE 802.16 BWA System". IEEE International Conference on Communications Circuits & System and West Sino Expositions, vol.1, pp. 435-439, China 2002.

[8] Kitti Wongthavarawat and Aura Ganz.. " IEEE 802.16 based last mile broadband wireless military networks with quality of service support." IEEE Milcom 2003, vol.2, pp. 779-784.

[9] M. Nohara, "Ad Hoc Meeting Report: Mobile Multihop Relay Networking in IEEE 802.16", document IEEE 802.16-05/51, 21 July 2005.

[10] M. Asa, D.T. Chen, and N. Natarajan, "Concepts for 802.16-based Mobile Multihop Relay Networking", document IEEE C802.16-05/15, 19 July 2005.

[11] Ashish Raniwala and Tzi cker Chiueh. Architecture and algorithms for an ieee 802.11-based multi-channel wireless mesh network. *In Twenty-Fourth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005)*, March 2005.

[12] S. Gobriel, R. Melhem, and D. Mosse. A unified interference/collision analysis for power-aware ad hoc networks. *In Twenty-Third Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2004)*, March 2004.

[13] H. Takagi and L. Kleinrock. Optimal Transmission ranges for randomly distributed packet radio terminals. *IEEE Transactions on Communications*, 32:246-257, March 1984.

[14] IEEE Std 802.16-2004/Corl, Corrigendum to IEEE Standard for Local and Metropolitan Area Networks – Part 16: Air Interface for Fixed Broadband Wireless Access Systems, Draft5, 2005-09-12.

[15] C. Eklund, R. Marks, K.L. Stanwood, and S. Wang, "IEEE Standard 802.16: A Technical Overview of the WirelessMAN^{TM} Air Interface for Broadband Wireless Access", *IEEE Communications Magazine*, June 2002, pp. 98-107.

[16] S. Redana and M. Lott, "Performance Analysis of IEEE 802.16a in Mesh Operation Mode", Proc. Of the 13th IST SUMMIT, Lyon, France, June, 2004.

[17] M. Cao, W. Ma, Q. Zhang, X. Wang, and W. Zhu, "Modeling and performance analysis of the distributed scheduler in IEEE 802.16 mesh mode," in Proceedings of the 6th ACM Int'l Symp. On Mobile Ad Hoc Networking and computing (Mobihoc'05), Urbana-Champaign, IL, 2005, pp.78-89.

[18] H.-Y. Wei, S. Ganguly, R. Izmailov, and Z.J. Haas, "Interference-aware IEEE 802.16 WiMax mesh networks," 61st IEEE Vehicular Technology Conference (VTC 2005-Spring), Vol.5, 2005, pp. 3102-3106.

[19] F. Liu, Z. Zeng, J. Tao, Q. Li, and Z. Lin, "Achieving QoS for IEEE 802.16 in Mesh Mode", 8th Int'l Conference on Computer Science and Informatics, Salt Lake City, Utah, July, 2005.

[20] H. Shetiya and V. Sharma, "Algorithms for routing and centralized scheduling to provide QoS in IEEE 802.16 mesh networks," in Proceedings of the 1st ACM workshop on Wireless multimedia networking and performance modelling, 2005, pp. 140-149.