An End-to-End QoS Aware Greedy Distributed Scheduling Framework for WiMAX Mesh Networks

Ankit Kapoor†, Vinay J. Ribeiro*
Department of Computer Science and Engineering
Indian Institute of Technology, Delhi
ankitkap.86@gmail.com†, vinay@cse.iitd.ac.in*

***Abstract*—In this paper, we propose a greedy framework for distributed scheduling in the IEEE 802.16 MeSH mode, which uses a novel "End-to-end QoS aware bandwidth Reservation Protocol" (EQRP) to provide end-to-end QoS guarantee to intra-mesh flows. The proposed framework provides an efficient and integrated solution to QoS aware routing and call admission control in distributed WiMAX mesh networks. The framework does not rely on any special node for resource management which makes it more scalable and robust to node failures. To save expensive control overheads, EQRP learns from previous bandwidth reservation failures to maintain a rank list of next hops for every destination and uses the *"Greedy Forwarding"* algorithm to do bandwidth reservation using slot information from only two hop neighbor routers'.**

We compare EQRP with Race Free Protocol (RFP) and evaluate its performance with extensive simulations in ns2. Simulations show that for static WiMAX mesh network, the "Greedy Forwarding" algorithm used by EQRP admits approximately 10% more VOIP calls. For a random topology of 25 mesh routers, the aggregate signaling overhead generated by EQRP at a high call arrival rate of 1/2000 (calls/milli-seconds) is 76% less than that generated by RFP. In comparison with RFP, EQRP took 200 milli-seconds less average call setup time.

I. Introduction

The IEEE 802.16 standard (WiMAX) is capable of providing high speed Internet access and better QoS compared to WiFi. The IEEE 802.16 standard defines a TDMA MAC, which provides better QoS differentiation to flows compared with the CSMA/CA MAC of the WiFi devices. Moreover WiMAX devices can work over much longer links (between four to eight miles) than can WiFi devices (few hundred feet).

The IEEE 802.16 specifies two different modes of operations named PMP mode (Point to Multipoint) and MeSH mode. Both the modes have their own advantages and disadvantages. In PMP mode, all subscriber stations (SS) (Mesh Routers) lie in range of a base station (BS) i.e. all subscriber stations are within one hop from the base station. The Base station works as the Internet gateway for a network. A number of clients can be attached to each subscriber station. In this mode of operation, there is only one logical link between each subscriber station and the base station. The failure of this link will prevent clients attached to the corresponding subscriber station from accessing various services.

The MeSH mode is more robust to link/node failures as for each subscriber station there are a large number of multihop paths to reach the base station and other subscriber stations.For MeSH mode a special MAC is defined in the IEEE 802.16, which provides two different types of scheduling mechanisms. In centralized scheduling, all requests are forwarded to the base station, which is responsible for the resource management (bandwidth reservation etc.) in a network. The centralized scheduling is suitable for scheduling of inter-mesh flows and it provides end-to-end QoS. A different scheduling framework supported by the WiMAX MeSH mode MAC is distributed scheduling framework, which provides facility of scheduling intra-mesh traffic between subscriber stations in coordinated or uncoordinated fashion. The distributed scheduling framework does not provide end-to-end QoS to flows but makes mesh networks suitable to be used for adhoc setup in battlefields where a robust and scalable network is required.

The IEEE 802.16 MeSH mode MAC is based on Time Division Multiple Access (TDMA). Time is partitioned into fixed length time frames. The Control subframe and Data subframe together form a complete frame. As the name implies, the Control subframe is used for transmitting control information where as the Data subframe is reserved for actual data traffic. Each frame is made up of OFDM symbols and each control subframe is made up of seven consecutive OFDM symbols (modulated using QPSK 1/2 encoding). The number of control opportunities per frame can be controlled by the MSH-CTRL-LEN parameter. The remaining OFDM symbols in a frame are divided equally among a maximum of 256 mini-slots (slots) that are used by subscriber stations for data traffic.

To make distributed scheduling suitable for providing the required end-to-end QoS to intra-mesh flows, we designed a greedy framework for distributed scheduling in WiMAX mesh networks. The proposed framework uses End-to-end QoS aware bandwidth Reservation Protocol (EQRP) that provides distributed call admission control to mesh network.

The Rest of the paper is organized as follows. In next section, we discuss about QoS routing and bandwidth reservation in TDMA based wireless multihop networks. Section III discusses the proposed framework for distributed scheduling in mesh networks. This section also provides details of the "Greedy Forwarding" algorithm used by EQRP. In Section IV, we discuss the experimental setup and analyze the performance of EQRP in different scenarios. We conclude this paper and discuss future work in Section V.

II. Related Work

A lot of research has been done in areas like distributed/centralized scheduling algorithms, QoS in centralized scheduling and distributed scheduler performance analysis for WiMAX mesh mode. In addition, the problem of providing

978-1-4244-5489-1/10/$26.00

QoS guaranteed end-to-end bandwidth reservation has been studied extensively for TDMA MAC based MANETS. We thus categorize previous work depending on whether the work was carried out for static mesh networks or for mobile adhoc networks.

A. QoS Routing and Bandwidth Reservation in TDMA based MANETS

Chen and Nahrstedt [1] suggested a ticket based bandwidth reservation protocol for mobile adhoc networks. The suggested proactive protocol makes the assumption that bandwidth of a node can be calculated independent of its neighbors. This assumption might require the network to use multiple directional antennas. The protocols suggested in [2][3] considered CDMA over TDMA MAC, thus these protocols cannot be used with the WiMAX MeSH mode which is a simple TDMA based MAC.

Liao et al. [4] proposed a distributed end-to-end bandwidth reservation protocol for TDMA MAC-based mobile adhoc networks. The suggested protocol floods a QoS request in whole network. When the request reaches destination, it selects a path to do bandwidth reservation over. The reserved path is used for data traffic until it is freed by a special cancel packet. The authors solved the well known hidden terminal and exposed terminal problems by making every node share its data slots' states (*Free/Reserved*) with its two hop neighbor nodes via synchronous update packets. A slot's state is changed from the *Free* state to the *Reserved* state only if a reply indicates the slot to be used either for data reception or data transmission; similarly the slot's state is changed back from the *Reserved* state to the *Free* state on the reception of a corresponding cancel packet.

Jawhar et al. in [5] discussed that maintaining only two states *(Free/Reserved)* for each slot may result in race reservations or parallel reservations, especially in dense networks when the request rate is very high. The Race Reservation Problem occurs when a router reserves the same data slot for two different flows whereas Parallel Reservation occurs when data slots reserved by two neighbor routers violates the slot selection criteria (see Section III-E3) resulting in the hidden terminal problem. For a more detailed explanation the reader can refer the [5].

To solve above two problems, in [5] the authors proposed the Race Free Bandwidth Reservation Protocol (RFP). Instead of having a slot only in the *Free* or the *Reserved* state, a new *Allocated* state was proposed, which indicates that the slot has been temporarily allocated for transmission/reception but may or may not be used for final data transmission/reception depending on whether the reply from the request's destination comes over this path or not. When a node forwards a request, it changes the states of selected slots from the *Free* state to the *Allocated* state and updates its neighbor nodes about this state change, so that neighbor nodes consider this slot temporarily reserved. If the reply does not come over the temporarily reserved path, all temporarily allocated slot states automatically change to the *Free* state.

RFP solves the race and the parallel reservation problems; but flooding of requests can result in very large signaling overhead and may unnecessarily result in drop of requests because of temporary allocation of slots.

B. Bandwidth Reservation in WiMAX Mesh Networks

The WiMAX MeSH mode's distributed scheduling framework [6][7] defines a 3-way handshake mechanism for bandwidth reservation on one hop links between neighbor nodes, but it does not guarantee any end-to-end quality of service.

In [8], authors extended this basic 3-way handshake mechanism to provide end-to-end bandwidth reservation in WiMAX mesh networks. The suggested End-to-end Bandwidth Reservation Protocol (EBRP) provides call admission control to network but relies on some layer three routing protocol to find a suitable path between source and destination routers. Unlike EBRP, EQRP provides an integrated solution to the QoS routing and the call admission control in mesh network.

The distributed scheduling algorithms of [9][10] provide fairness to intra-mesh flows but do not guarantee end-to-end QoS. Moreover the scheduling algorithms [9][10] do not provide call admission control in network.

III. Proposed QoS Aware Greedy Distributed Scheduling Framework

To save expensive control bandwidth, EQRP employs a "Greedy Forwarding" algorithm which uses only two hop neighbor routers' information to reserve resources for QoS requests. EQRP is an on-demand, source-based protocol which uses topology information of mesh network to find a QoS-satisfying path from source to destination routers.

The proposed framework and EQRP are based on the Race Free Protocol and uses the solution suggested in [5] to solve race reservation and parallel reservation problems.

In the next section, we discuss various components of the proposed scheduling framework. Section III-B discusses important data structures maintained by mesh routers. Section III-C gives overview of EQRP's functionality. In Section III-D, we discuss about various Information Elements (IEs) that are exchanged between routers in distributed scheduling (DSCH) packets. Finally, Section III-E discusses the "Greedy Forwarding" algorithm used by EQRP.

A. Framework Components

Figure 1 shows the Information Elements (IEs) flow and interaction among various components within a mesh router. Basic functionalities of the components (modules) are given below.

The *Classifier* distinguishes different IEs in DSCH packets received from upper layers or from neighbor mesh routers and push them to appropriate queues or modules for further processing. The *Resource Allocation Module (RAM)* is the most important component of the scheduling framework. This module is responsible for scheduling of flow requests. It implements the "Greedy Forwarding" algorithm which finds appropriate free data slots and next hop routers to schedule data traffic and to forward requests on paths capable of providing the required QoS. Two important data structures maintained by this module are the *Slot-Map* and the *Probe-Map*.

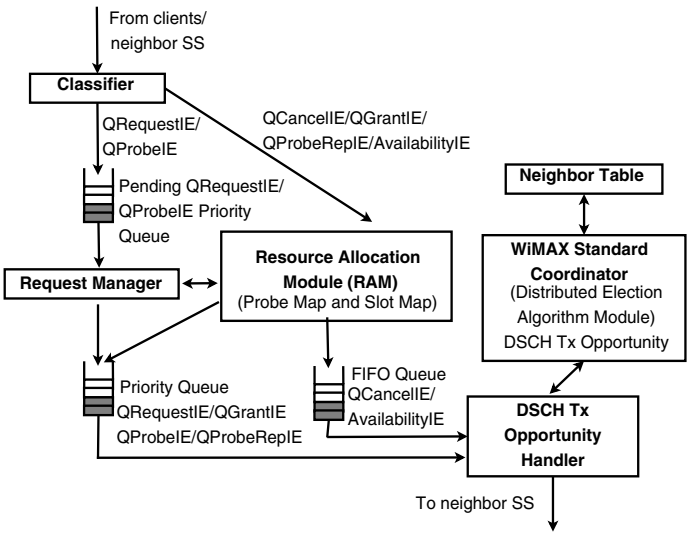

Fig. 1. End-to-end QoS Aware Distributed Scheduling Architecture

The *Request Manager* interacts with this module to temporarily allocate resources for pending requests and probes. In a DSCH transmission opportunity, the Request Manager scans pending priority queues and interacts with RAM to allocate resources for requests and probes before pushing them to outgoing queues. The requests and probes are serviced in order of priority. To have less call setup delay, a greedy approach is used by giving more priority to the request or probe that has least call validity time left. A request or probe for which RAM fails in finding a next hop remains in the pending queue till its validity expires.

The *WiMAX Standard Coordinator* runs standard WiMAX mesh election algorithm [6][7][11], to find DSCH control packet transmission opportunities in a conflict free, fair and distributed manner. In the control slot in which a mesh router wins a DSCH transmission opportunity, the *DSCH Tx Opportunity Handler* fills the outgoing DSCH packet with the IEs stored in outgoing queues.

B. Data Structures maintained at Routers

Each mesh router maintains Slot-Map and Probe-Map tables. These data structures are consulted and updated by RAM while processing different DSCH IEs.

1) Slot-Map Table: Similar to RFP, this table stores the current state of the data slots of a router and its extended neighborhood (two hop neighbors). Depending on current state of a slot, RAM decides whether the slot can be reserved or not. A slot can be in one of the following states:

- *Free*: This state indicates that the slot is currently free and can be scheduled for data transmission or data reception.
- *Allocated States*: The *Tx-Allocated* and *Rx-Allocated* states indicate that the slot has been temporarily allocated for data transmission and data reception respectively but may or may not be used by actual data traffic. These states are associated with a validity timer. If the *Tx-Allocated* and the *Rx-Allocated* states so not transition to the *Tx-Reserved* and the *Rx-Reserved* states respectively with in validity timeout period, the states are automatically changed back to the *Free* state.
- *Reserved States*: The *Tx-Reserved* and *Rx-Reserved* states indicate that the slot is confirmed to be used for data transmission and data reception respectively. These states are associated with a connection timeout timer. If the mesh router does not notice data traffic in a reserved slot for a timeout period, the slot's state is changed back to the *Free* state.

2) Probe-Map Table: To save expensive control bandwidth, unlike RFP, EQRP maintains the Probe-Map table. During initialization, each mesh router updates this table with possible next hop routers for each destination and stores the next hop's Node-Id, original rank, current rank and number of hops (num-hops) to reach the destination router. It ranks next hops on the basis of hops required to reach the destination. The "Greedy Forwarding" algorithm uses the current rank list to find an appropriate next hop router for a request. For a request, the next hop router capable of finding a least hop QoS path to request's destination, is given more priority than other routers. Initially, both current and original ranks of next hop routers are the same. Later, on receiving a QoS grant from a destination, RAM updates num-hop field of the next hop router (via which the request was forwarded towards the destination router) with the number of hops taken by the request to reach the destination.

Each entry in this table is associated with a timer. When RAM forwards a QRequest IE to a next hop router, it sets a timer associated with the corresponding Probe-Map entry's with a timeout value equal to the request validity period. If RAM does not receive the corresponding QoS grant, the timer fires and makes this entry invalid by setting its num-hop field to infinity. For a particular destination, an infinite hops value associated with a next hop router represents that this next hop was previously not able to find a QoS satisfying path and therefore needs to be probed (see Section III-E) before further forwarding any QoS request meant for this destination.

C. EQRP Overview

Figure 2 shows the request and the probe forwarding mechanism used by EQRP to control signaling overhead. The initial Probe-Map of the router S is shown in Table I-O. Figure 2-A shows forwarding of a new QoS request which originated at the router S for the destination D. The next hop router C (whose initial current rank (CR) is one) requires the least hops to reach the destination. Therefore router S forwards the request to it but as the links ($\{$E,F$\}$ and $\{$E,D$\}$) have no free bandwidth (shown by dotted lines), the request get dropped at the router E. On the request failure, source router S updates its Probe-Map table to indicate that router C is not capable of finding a required QoS path towards destination D and sets the corresponding NHops (num-hops) entry in its Probe-Map table to infinity (see Table I-A).

As shown in Figure 2-B, for a new request generated at the router S, router B becomes the best next hop to forward the request to the destination D. While forwarding the request, source router S also probes the destination via router C, whose

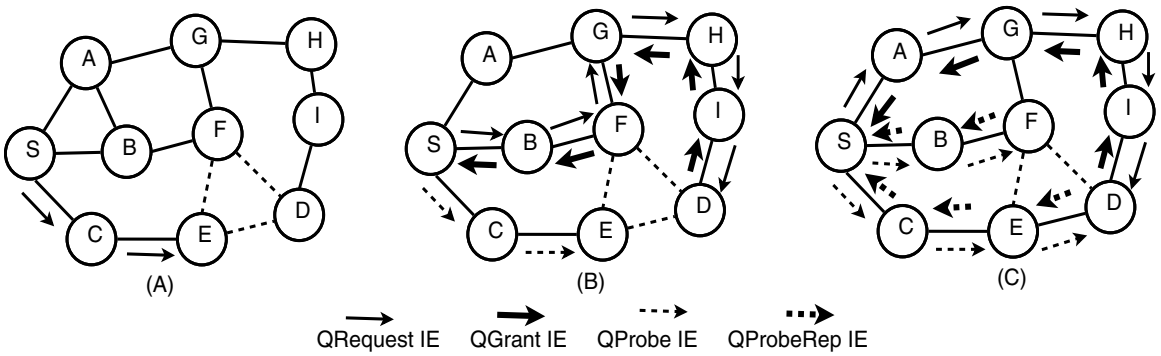

Fig. 2. EQRP in sample mesh network

TABLE I
PROBE-MAP (O,A,B,C)

NH	OR	CR	NHops
C	1	1	2
B	2	2	2
A	3	2	3

NH	OR	CR	NHops
C	1	3	∞
B	2	1	2
A	3	2	3

NH	OR	CR	NHops
C	1	3	∞
B	2	2	5
A	3	1	3

NH	OR	CR	NHops
C	1	1	2
B	2	3	5
A	3	2	4

original rank (OR) is less than that of the router B. When the request reaches router F, the Greedy Forwarding algorithm (see Section III-E) finds that link {F,D} (shown by dotted line) does not have required free bandwidth and thus forwards the request to the router G. The QGrant IE generated by the destination D is used to get number of hops (5) on the reserved path and this value is updated in the router S's Probe-Map table (see Table I-B). For subsequent requests towards destination D, router A becomes the best next hop candidate to reach the destination via four hops. While forwarding the requests via router A, the source router S also sent probes to router B and router C to check if they are now capable of finding a QoS path to the destination D. Router F replies to a QProbe IE towards destination D with a QProbeRep IE with its min-hops field set to three, as the current shortest QoS satisfying path from it to the destination is {F,G,H,I,D}. The router S receives this ProbeRep IE via router B and therefore sets the corresponding num-hops field in its Probe-Map to five.

We assume that the link E,D is now capable of providing the required QoS. For the Probe IE generated by source router via router C, Figure 2-C shows generation of the corresponding QProbeRep IE by the destination D. When router S receives this ProbeRep IE, it updates its Probe-Map table to indicate that next hop router C is now capable of finding a two hop QoS path {S,C,E,D} to the destination D. **Note:** Figure 2-C shows that router G forwards the QRequest IE to the router H instead of forwarding it to the router F. Here we assumed that router G has already learned that router F is not best candidate to reserve paths towards destination D.

D. Information Elements

The messages that are used to reserve and free QoS paths form source to destination routers are conveyed as Information Elements (IEs) in distributed scheduling (DSCH) control packets. Except QProbe and QProbeRep IE, all IEs used by EQRP are based on the messages used by RFP [5] with small modifications. Every router in the mesh is uniquely identified by a 2 byte Node-Id. The different IEs used by EQRP are as follows.

1) *Availability IE* is used by every mesh router to update its extended neighborhood about modifications to states of slots. An Availability IE is composed of Source Id, Sequence Number, Type, NumSlots, Slot List, TTL and Timeout fields. The *Type* field indicates new states of the slots whose ids are present in the *Slot List*. The *Timeout* value is valid only when the updated slot's state is either *Tx-Allocated* or *Rx-Allocated*.
2) *QRequest IE* is used in the *Path Discovery Phase* of EQRP to temporarily allocate required bandwidth on a path from source to destination router. A QRequest IE is composed of Source Id, Destination Id, Sequence Number, Bandwidth, Num Previous Hops, Previous Hop List, Next Hop, Slot List and Validity fields. The pair {Source Id, Sequence number} uniquely identifies each request. The field Bandwidth represents QoS requirement in terms of number of data slots required by a flow. For a QoS request, the Previous Hop List indicates Node-Ids of the intermediate mesh routers that forwarded this request and did temporary slot allocations for it. When a router selects a suitable neighbor to forward a request towards destination, it sets the Next Hop field of the QRequest IE to the Node-Id of the selected neighbor and updates the IE's Slot List with Ids of the data slots temporarily allocated by it. The Validity represents the frame number after which the request becomes invalid.
3) *QGrant IE* is used in the *Path Reservation Phase* of EQRP to mark temporarily allocated slots as reserved. Except Next Hop and Slot List field, this IE is composed of fields similar to that of a Request IE. A QGrant IE

is generated when a request (QRequest IE) reaches its destination.

4) *QCancel IE* is used in *Path Cancellation Phase* of EQRP to free a previously reserved QoS path. Except for the Validity field, all fields of a QCancel IE are similar to that of a QGrant IE.
5) *QProbe IE* is used by the probing mechanism described in Section III-E to update ranks of next hops in the Probe-Map table. Unlike QRequest IE, a QProbe IE can be forwarded to more than one neighbor router. Therefore a QProbe IE maintains a list of {Next Hop, Slot List}. The rest of the fields of a QProbe IE are similar to that of QRequest IE.
6) *QProbeRep IE* is generated when a probe (QProbe IE) reaches its destination or an intermediate router knows a valid next hop for the QProbe IE's destination. It is composed of a Min Hops field along with fields similar to that of a QGrant IE. The Min Hop field is used to update intermediate and source mesh routers about the minimum number of hops required to the reach QProbe IE's destination via the QProbeRep IE's source. Unlike QGrant IE, on reception of a QProbeRep IE, a mesh router does not change temporarily allocated (by its corresponding QProbe IE) states of the data slots to their respective reserved states.

The *Path Discovery Phase* of EQRP is used to temporarily reserve a QoS-satisfying path from a source to destination router. For a QRequest IE, on receiving a service request of the Request Manager, the Resource Allocation Module of a mesh router (R) uses the "Greedy Forwarding Algorithm" to select a next hop (NH) and data slots (DS) to temporarily schedule data transmission from router R to router NH. RAM consults the Probe-Map table to find next hop routers whose original ranks are less than the original rank of selected next hop and probes the destination via them. RAM updates the QRequest IE's Previous Hop List with the router's unique Node-Id and updates Slot List with the selected slots' Ids. Before forwarding the updated QRequest IE, RAM changes states of the selected slots from the *Free* state to the *Tx-Allocated* state. On reception of this QRequest IE, router NH updates its Slot-Map to indicate temporary allocation of the selected slots (DS) for data reception from the router R by changing the slots' states from the Free state to the Rx-Allocated state. The changes in the Slot-Map by router R and router NH are accompanied with state change notifications to their respective extended neighborhood via appropriate Availability IEs. This phase ends when the QRequest IE reaches its destination router.

The *Path Reservation Phase* of EQRP starts when a QRequest IE reaches its destination. The destination router creates a QGrant IE and sets its Previous Hop List to that of the received QRequest IE. The Previous Hop List stores Node-Ids of the intermediate mesh routers that temporarily allocated slots for this QoS request. This information is used to forward the QGrant IE to the request's source router. Before forwarding the QGrant IE, the Resource Allocation Module of a mesh router changes states of the slots temporarily allocated by corresponding QoS request (during Path Discovery Phase), from the *Allocated* state to the *Reserved* state and updates its extended neighborhood via an appropriate Availability IE. This phase ends when the request's source router receives a grant (QGrant IE) corresponding to its request.

The *Path Cancellation Phase* starts when for a reserved path, a cancel request is generated by a client. The Cancel packet indicates the client's wish to free a previously reserved QoS path. The source mesh router starts this phase by forwarding a QCancel IE. The Previous Hop List of this QCancel IE is set to the Node-Ids of the mesh routers that are on the corresponding reserved QoS path towards the destination. This information is used to forward the QCancel IE to the destination router. Before forwarding the QCancel IE, the Resource allocation Module of the router changes states of the slots reserved by the corresponding QoS request (during Path Reservation Phase), from the *Reserved* state to the *Free* state and updates its extended neighborhood via appropriate Availability IEs. This phase ends when the destination router receives the QCancel IE.

E. Greedy Request and Probe Forwarding Algorithm

1) Request Forwarding: In this section we explain the "Greedy Forwarding Algorithm" in detail. The algorithm is greedy because it forwards every request via the next hop candidate that is capable of reserving a least hop QoS satisfying path towards the request's destination. Algorithm 1 explains processing of a QRequest IE (*ie*) {*src, dst, seq, b, nph, phl, nh, sl, v*} by RAM on a request of the Request Manager. The Node-Id of the mesh router processing *ie* is represented by *ni*. The network topology is denoted by *topo*.

For a valid request, if the router is not the source of *ie*, RAM updates states of the slots whose ids are present in Slot List field, from the *Free* state to the *Rx-Allocated* state. If this router is the destination of received IE, RAM updates above slots' state from the *Rx-Allocated* state to the *Rx-Reserved* state and generates a QGrant IE destined for source of the request. If the request is not destined for this router, RAM calls *removeNodes* method to create a temporary topology in which all routers that have already forwarded this IE are disconnected. This is to prevent forwarding of an IE to a router which has already allocated resources for it. RAM then uses the Probe-Map table to compute a list (*next_hops*) of valid next hops for the request's destination (this list contains next hops whose num-hops fields is not equal to infinity). *findNextHop* method is used to find a next hop to forward the request provided required QoS criteria gets fulfilled. *findNextHop* method scans *next_hops* list in order of current ranking of neighbor routers in the Probe-Map table and uses the Slot-Map table and slot selection criteria (given in Section III-E3) to check if a next hop is capable of scheduling the request. If a suitable next hop router capable of providing required bandwidth is found, RAM forwards updated QRequest IE to it. The invalid next hop routers having original rank less than that of the selected next hop router are pushed to the *p_hops* list. All next hop routers in *p_hops* are probed to check their current path finding capabilities. If RAM does not find a suitable next hop for

the request then all invalid next hop routers for the request's destination are probed.

Algorithm 1 Greedy Request Forwarding Algorithm

```
1: if v < current_frame then
2:    delete ie from pending queue
3:    return
4: end if
5: if src ! = ni then
6:    update states of the slots in sl from Free to Rx-Allocated
7: end if
8: if dst == ni then
9:    update states of the slots in sl from Rx-Allocated to
      Rx-Reserved
10:   generate QGrant IE {dst, src, seq, b, nph, phl, v}
11: else if  then
12:   new_topo = removeNodes(topo,phl)
13:   next_hops = getValidNextHops(new_topo, ni, dst)
14:   p_hops = getDisconnectedHops(new_topo, dst)
15:   {n_hop, n_slots} = findNextHop(next_hops, b)
16:   if n_slots ! = φ then
17:      p_hops = getHopsToProbe(p_hops, n_hop, dst)
18:      new_phl = phl|ni, nh = n_hop, sl = n_slots
19:      delete ie from pending queue
20:      push a QRequest IE {src, dst, seq, b, nph+1, new_phl,
         nh, sl} to outgoing queue
21:      update the states of slots in sl from Free to Tx-
         Allocated
22:      push an Availability IE {ni, Tx-Allocated, b, sl}
23:   end if
24:   if probe_list ! = φ then
25:      push a Probe IE to all neighbors in p_hops
26:   end if
27: end if
```

2) Probe Forwarding: On receiving probe request for a destination, the Resource Allocation Module of a router checks its Probe-Map to find a valid next hop router having least num-hops value to reach the destination. If it finds such a next hop router, it replies back with a QProbeRep IE to the probes' source. This QProbeRep IE is used by the source and the intermediate routers (which forwarded the probe) to update their Probe-Map table with minimum number of hops required to reach the probe's destination. If the router does not know any valid next hop, it then forwards the QProbe IE to all invalid next hops, after temporarily allocating slots for it. If required free slots are not found to fulfill QoS demand, the QProbe IE remains in its pending queue and this procedure is repeated until the QProbe IE expires. Similar to the "Greedy Request Forwarding" algorithm, all next hop computations are done using a temporary topology in which all routers that have already forwarded this IE are disconnected. Similar to a QRequest IE, forwarding and reception of a QProbe IE is accompanied with a change of selected slots from the Free state to the Tx-Allocated state and the Rx-Allocated state respectively.

3) Slot Selection Criteria: Suppose one hop neighbors of router R1 are denoted by N1 and one hop neighbors of router R2 are denoted by N2. A slot S can be used for transmission from router R1 to router R2, if following conditions are satisfied.

- $\forall router \in N1$, data slot S is not in *Rx-Reserved* or *Rx-Allocated* state.
- $\forall router \in N2$, data slot S is not in *Tx-Reserved* or *Tx-Allocated* state.
- slot S is in *Free* state on both router R1 and router R2.

IV. Simulations and Results

We now compare the performance of EQRP with RFP [5]. We chose RFP because EQRP is similar to RFP in many ways. Unlike EQRP, RFP was designed for mobile adhoc networks. Therefore RFP does not use network topology information while finding a QoS-satisfying path for a QoS request. RFP floods the network with requests resulting in very high signaling overhead and unnecessary blocking of requests at routers. EQRP searches for the best next hop to forward a request towards the destination using the "Greedy Forwarding" algorithm (see Algorithm 1). We found through simulations that this approach actually reduces signaling overhead, average call setup delay and call blocking probability. The call blocking probability is defined as the ratio of number of failed QoS requests to the number of QoS request made by mesh routers. We implemented RFP and EQRP in ns2 [12] patched with ns2mesh80216 [13]. The following simplifying assumptions have been made for simulation setup.

- The wireless channel is assumed to be error free and no ARQ mechanism is available. To construct the network topology at each router, link state packets are broadcasted by each mesh router at a very slow rate. The link profiles (modulation, retransmission rate etc.) broadcasted in link state packets can be used to prevent forwarding of requests over links experiencing huge error rates.
- We also assumed a single channel for both control and data frames.
- All pending and outgoing queues maintained by routers are assumed to be of infinite size.

A. Experimental Setup

We simulate both EQRP and RFP using two different topologies for 1800 seconds. Topology 1 represents an adhoc placement of 25 mesh routers simulating a battle or disaster field scenario whereas Topology 2 represents a planned placement of 24 mesh routers in a 6X4 grid topology. In adhoc topology, we place 25 mesh routers in a square field of 80 Kms X 80 Kms. The transmission range of each router is set to 10 Kms. VOIP calls of 120 seconds (exponential average) duration demanding QoS of 19.2 Kbps bandwidth, are generated by the Request Generator at each mesh router at some specified exponentially distributed call arrival rate. The destination for each call is selected randomly. The simulation parameters and their values are given in Table II. The mesh election algorithm [6] [7] [11] uses the *Holdoff exponent* parameter to compute control packet transmission opportunities of a mesh router.

The *DSCH data slot* parameter indicates maximum number of data slots of a mesh router that can be scheduled by the distributed scheduling framework. The simulation results for

TABLE II
SIMULATION PARAMETERS

Simulation Parameter	Value
Bandwidth	10MHz
Frame length	10ms
OFDM symbol duration	25μs
Holdoff exponent	0
DSCH data slots	150
Data slot modulation	BPSK-1/2
Call validity time	100 Frames (1 Second)
Call bandwidth request	1 Data Slot (19.2 kbps)
Average call duration	120 Seconds (Exponential Average)

both topologies are given in the following subsections. For the results discussed in subsection IV-A1, IV-A2 and IV-A3, we set the number of control slots per frame to 10 during simulations and for the analysis under subsection IV-A4 and IV-A5, an exponentially distributed call arrival rate of 1/8000 (calls/milli-second) was used.

1) Call Blocking Probability vs. Call Arrival Rate: A comparison of call blocking probabilities at different call arrival rates is shown in Figure 3. For both topologies, at each call arrival rate, EQRP admits approximately 10% more VOIP calls in comparison with RFP. For every QoS request, RFP temporarily reserves multiple paths till the request validity period. These temporary reservations impacts reservation and forwarding of new QoS requests. EQRP solves this problem by using the "Greedy Forwarding" algorithm (see Algorithm 1) and temporarily reserves only one path for a QoS request.

2) Average Call Setup Delay vs. Call Arrival Rate: Figure 4 shows average call setup delay for EQRP and RFP at different call arrival rates. For both EQRP and RFP, the average call setup delay increases linearly with respect to call arrival rate. For both topologies, at each call arrival rate, average call setup delay of EQRP is less than that of RFP. EQRP took approximately 200 milli-seconds and 120 milli-seconds less call setup time for the simulated random and grid topology respectively.

3) Aggregate Signaling Overhead vs. Call Arrival Rate: It is clear from the Figure 5 that for duration of 1800 seconds, EQRP results in very less signaling overhead in comparison with RFP. The signaling overhead of EQRP increases with decrease in call arrival rate whereas the signaling overhead of RFP is approximately same at all call arrival rates. For the random topology (Topology 1), at a high call arrival rate of 1/2000 (calls/milli-seconds), the aggregate signaling overhead generated by EQRP is 76% less than that generated by RFP. As per the IEEE 802.16 standard, an operator can define the maximum number of data slots that can be scheduled by a scheduling algorithm. Therefore as per traffic demand, different signaling protocols may work simultaneously in a WiMAX mesh network. Less signaling overhead of EQRP shows that it is capable of efficiently sharing the control slots with other signaling protocols' control traffic whereas simultaneous use of other signaling protocol and RFP in network may require more control slots per frame.

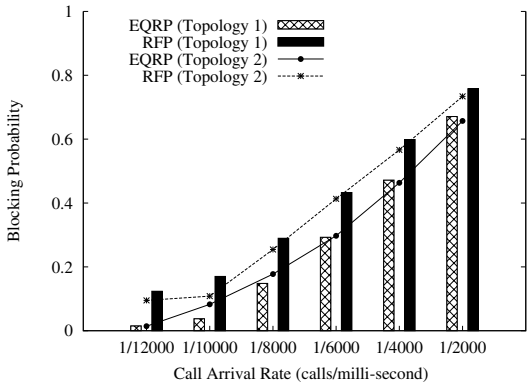

Fig. 3. Call Blocking Probability vs. Call Arrival Rate

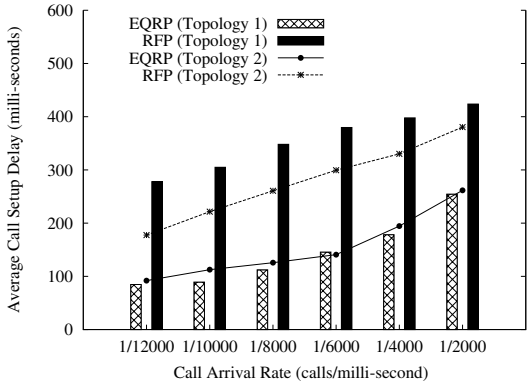

Fig. 4. Average Call Setup Delay vs. Call Arrival Rate

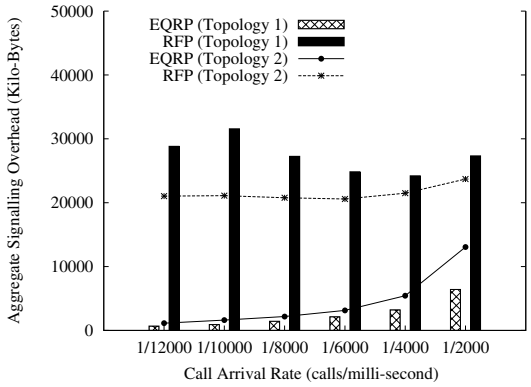

Fig. 5. Aggregate Signaling Overhead vs. Call Arrival Rate

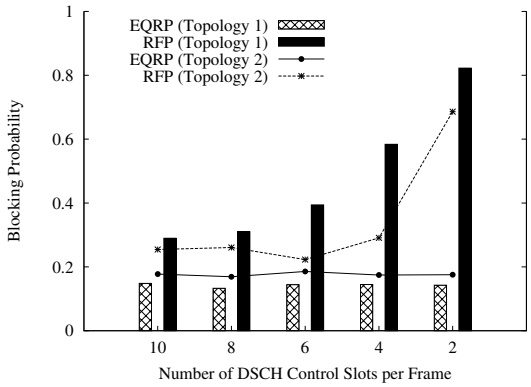

Fig. 6. Call Blocking Probability vs. Number of Control Slots

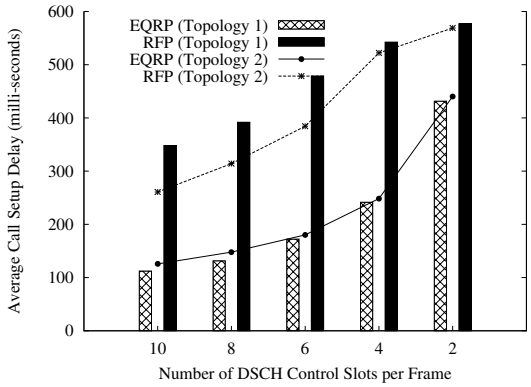

Fig. 7. Average Call Setup Delay vs. Number of Control Slots

4) Call Blocking Probability vs. Number of Control Slots: The effect of number of control slots per frame on call blocking probability at call arrival rate of 1/8000 (calls/milli-seconds) is shown in Figure 6. We reduce the number of control slots per frame from 10 to 2. For RFP, a decrease in the number of control slots per frame results in an increase of call blocking probability whereas call blocking probability of EQRP does not get affected with the decrease in number of control slots. The results indicate that only 2 control slots per frame are sufficient for EQRP to provide end-to-end QoS in the simulated topologies.

5) Average Call Setup Delay vs. Number of Control Slots: Figure 7 shows that for both EQRP and RFP, a decrease in the number of control slots per frame results in an increase in call setup delay since less number of control sots requires requests and replies to wait more in pending and outgoing queues at the intermediate routers.

V. Conclusion

Simulations proved that for a static WiMAX mesh network, the "Greedy Forwarding" algorithm used by EQRP is more efficient than the flooding approach used by RFP. The less signaling overhead of EQRP helps it efficiently share the control slots with other signaling traffic. The distributed call admission control provided by the proposed scheduling framework using EQRP is capable of admitting more calls (in less setup delay) in comparison with RFP.

Though simulations show that EQRP performs better than RFP on a grid and a random topology, there exist some scenarios where RFP may perform better than EQRP. Suppose a new mesh router joins an existing mesh, and there are four unique (having no common link) paths (say of two, three, four and five hops) towards a destination. If two, three and four hops paths are under congestion, EQRP will result in drop of three calls meant for this destination before concluding that only the longest hop path is capable of providing QoS. RFP on other hand may be able to satisfy all four requests. If these scenarios are common in network, then RFP may give better results than EQRP.

This work focused on CBR VOIP traffic. Further research is required to make EQRP capable of efficiently handling VBR QoS requests. Another possible extension of this work is to support the mobility of clients by modifying EQRP to be aware of handoffs in a network.

References

[1] S. Chen and K. Nahrstedt, "Distributed quality-of-service routing in ad hoc networks," *IEEE J. Sel. Areas Commun.*, vol. 17, no. 8, pp. 1488–1505, 1999.

[2] C.-C. Liu, "An On-Demand QoS Routing Protocol for Mobile Ad Hoc Networks," in *ICON '00: Proceedings of the 8th IEEE International Conference on Networks*. Washington, DC, USA: IEEE Computer Society, 2000, p. 160.

[3] C. Lin and J. Liu, "QoS routing in ad hoc wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 17, no. 8, pp. 1426–1438, 1999.

[4] W. Liao, Y. Tseng, and K. Shih, "A TDMA-based bandwidth reservation protocol for QoS routing in a wireless mobile ad hoc network," in *IEEE international Conference on Communications*, vol. 5. Citeseer, 2002, pp. 3186–3190.

[5] I. Jawhar and J. Wu, "A race-free bandwidth reservation protocol for QoS routing in mobile ad hoc networks," in *Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS-04), IEEE Computer Society*, vol. 9, 2004.

[6] P. S. Mogre, M. Hollick, and R. Steinmetz, "The IEEE 802.16-2004 MeSH Mode Explained, Technical Report," TU-Darmstadt, Germany, Tech. Rep., 2007.

[7] IEEE, "IEEE standard for local and metropolitan area networks part 16: Air interface for fixed broadband wireless access systems," 2004.

[8] C. Cicconetti, V. Gardellin, L. Lenzini, E. Mingozzi, and A. Erta, "End-to-end bandwidth reservation in IEEE 802.16 mesh networks," in *Proc. IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS), Pisa, Italy*, 2007, pp. 1–6.

[9] C. Cicconetti, I. Akyildiz, and L. Lenzini, "Bandwidth balancing in multi-channel IEEE 802.16 wireless mesh networks," in *Proc. IEEE Infocom*, 2007, pp. 6–12.

[10] B. Rajesh, "Distributed Scheduling for IEEE 802.16 Wireless Mesh Networks," Master's thesis, Department of Computer Science and Engineering, IIT Delhi, New Delhi, May 2007.

[11] C. Cicconetti, A. Erta, L. Lenzini, and E. Mingozzi, "Performance evaluation of the mesh election procedure of IEEE 802.16/wiMAX," in *MSWiM '07: Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*. New York, NY, USA: ACM, 2007, pp. 323–327.

[12] K. Fall and K. Varadhan, *The ns manual*, The VINT Project, 2009.

[13] C. Cicconetti, I. Akyildiz, and L. Lenzini, "WiMsh: a simple and efficient tool for simulating IEEE 802.16 wireless mesh networks in ns-2," in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2009, p. 7.