

Maintaining Routing Tree in IEEE 802.16 Centralized Scheduling Mesh Networks

Yanbin Lu^{*†} and Guoqing Zhang^{*}

^{*}Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, 100080

Email: {yanbinlu, gqzhang}@ict.ac.cn

[†]Graduate University of Chinese Academy of Sciences, Beijing, China, 100039

Abstract—The IEEE 802.16 mesh mode supports two scheduling mechanisms, namely centralized scheduling and distributed scheduling. Centralized scheduling is based on a routing tree to offer centralized control. As we will show, the structure of the routing tree has a significant impact on the system performance. In this paper, we propose a routing tree construction scheme for centralized scheduling to fully exploit spatial reuse. Our proposed method can be seamlessly integrated into IEEE 802.16 without any modification to the protocol. In particular, our scheme takes into account both interference and bandwidth request during tree construction. Performance evaluation shows that the routing tree generated by the proposed scheme achieves performance close to the optimal result and significantly outperforms existing routing tree construction methods.

I. INTRODUCTION

The IEEE 802.16 standard [1], also known as WiMax [2], enables easy and rapid deployment of low-cost BWA products. There are two modes of operation in WiMax, namely mandatory Point-to-Multi-Point (PMP) mode and optional Mesh mode. In the PMP mode, Subscriber Stations (SSs) can only interact with the base station (BS). While in mesh mode, SSs without direct links with Mesh BS may route their traffic through other SSs to BS.

In mesh mode, two specific scheduling mechanisms, centralized scheduling and distributed scheduling, are proposed to schedule traffic among links. Distributed scheduling is further classified into two types—coordinated and uncoordinated distributed scheduling. Coordinated distributed scheduling, by coordinating their transmissions in their extended two-hop neighborhood, suffers no collision. Uncoordinated distributed scheduling, used for setup of temporary bursts between a pair of neighboring nodes, acts more like an ad-hoc way.

On the contrary, centralized scheduling relies on BS to schedule transmissions for the SSs. For convenience of management, BS maintains a routing tree where BS is the root and SSs are the other nodes. Transmission can only occur along the links of the routing tree. Each SS regularly reports its bandwidth requests to BS which then determines flow assignments over the links in the routing tree.

Because different routing tree topology leads to different degree of interference and different density of traffic load, the topology of the routing tree has a direct impact on the throughput. As we will show in Section II-B, the structure

of the routing tree determines spatial reuse capability and as a result the throughput of the WiMax Mesh. Nevertheless, how to construct a high throughput routing tree as per both interference and bandwidth request of SSs has not been fully explored yet. In this paper, we will concentrate on routing tree construction to exploit spatial reuse potential.

Constructing a routing tree is equivalent to building each SS's route, the combination of which forms a tree. Choosing route according to link load in ARPANET is known to cause flapping because the traffic varies rapidly and routing is based on out-of-date linkstate information [3]. However it is justifiable for IEEE 802.16 centralized scheduling mesh for the reason that, before choosing route for SS, BS has the accurate knowledge of SS's bandwidth request. Jain *et al.* [4] explored both the upper and lower bound of capacity of multi-hop wireless network. Nevertheless its corresponding route is multi-path and cannot form a tree directly. If directional antenna is employed, the routing tree is constructed according to the direction of antenna and we can establish an optimal scheduling way [5], but, in real deployment, directional antenna is rarely used. Many prior researches [6], [7] have recognized the shortcomings of shortest-path routing in multi-hop wireless networks, so simply constructing a routing tree by shortest paths cannot maximize throughput. Wei and Haas [8] proposed a new metric based on interference degree for tree construction. But it does not take the bandwidth request of SSs into consideration when constructing routing tree. Unlike the mentioned work, we propose a heuristic algorithm to construct a near optimum routing tree. To the best of our knowledge, this is the first work that takes into account the impact of both interference and bandwidth requests to construct routing tree in IEEE 802.16 centralized scheduling mesh networks.

In short, followings are the major contributions of this paper:

- We develop a network model to characterize interference and traffic in 802.16 mesh mode.
- We give a heuristic algorithm to construct a near optimum routing tree by taking both interference and bandwidth request into account.

The rest of the paper is organized as follows. In Section II, we provide background on centralized scheduling mesh mode and motivation. Section III describes problem formulation, optimum routing tree and our proposed tree construction

¹This work was supported by 863 project (2006AA01Z207).

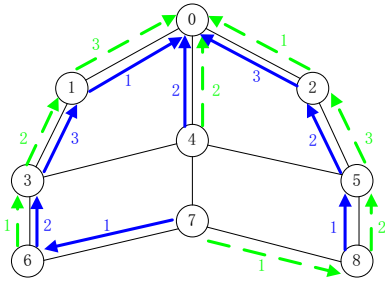


Fig. 1. Example of optimum routing tree

algorithm. Performance evaluation is presented in Section IV. Finally, Section V concludes the paper.

II. BACKGROUND

A. IEEE 802.16 Centralized Scheduling Mesh Mode

IEEE 802.16 builds on TDMA-based scheduling as compared to CSMA/CA-based 802.11 systems. The intrinsic difference between 802.16 and 802.11 MAC is that 802.16 only requires the receiver, rather than both receiver and transmitter, to be interference free.

In IEEE 802.16 Mesh mode, a Mesh base station (BS) provides backhaul connectivity of the mesh network and controls one or more subscriber stations (SS). The IEEE 802.16 Mesh mode supports both centralized scheduling and distributed scheduling. In centralized scheduling mode, the Mesh BS is responsible for coordinating the resource allocation within the mesh network. BS periodically broadcasts *MSH-CSCF* (Mesh Centralized Scheduling Configuration) messages to maintain a routing tree of which BS is the root and SSs are other nodes. The bandwidth request and grant process harness *MSH-CSCH* (Mesh Centralized Scheduling) messages. Each SS regularly issues its uplink and downlink bandwidth request through *MSH-CSCH: Request* to BS by the order of its appearance in the routing tree. BS carefully grants flow assignments over links in the routing tree by *MSH-CSCH: Grant* according to those requests from SSs. If the duration of all resulting minislot allocations exceeds available minislot space (in one or two frames depending on the *frame schedule flag* in *MSH-CSCH* message), all allocations should be reduced proportionally. The time between the first frame in which a node sends the request schedule and the last frame where a node receives the new grant schedule marks the validity of the previous grant schedule.

BS is able to adjust the routing tree by *MSH-CSCF* or *MSH-CSCH* which can update small number of tree links.

B. Motivation

The throughput of wireless network can be characterized by the minimum schedule length [9]. In this section, we show that different configuration of routing tree results in different minimum schedule length and that variation of bandwidth request leads to different optimum routing structure. As an example, consider the topology in Fig. 1 where node 0 represents BS. In the first case where SS 4, 7, 8 all require their

outgoing link to be active for 1 slot of a frame for their traffic to BS while other SSs have no traffic demand. The blue solid paths constitute the optimum tree to achieve the minimum total schedule length of 3 slots, with each link's active slot number labeled beside it. Any other tree cannot achieve a schedule length smaller than 3 slots. In the second case, we switch the demand of SS 6 and SS 8, the corresponding optimum tree will be the one marked with green dotted lines. This example shows that different bandwidth request will give rise to different optimum routing tree and it is easy to see that shortest path is far from optimum routing.

In this paper, we provide a heuristic algorithm to construct a near optimum routing tree. *MSH-CSCF* messages can be sent after the last request is received and before the grant schedule is transmitted by the Mesh BS. This makes it possible for our proposed algorithm to maintain the routing tree according to the variation of bandwidth requests from SSs.

III. IEEE 802.16 MESH MODE ROUTING TREE CONSTRUCTION

We first describe the constraints to model TDMA-based wireless network in Section III-A. Then based on the derived constraints, We employ ILP to model optimum routing tree in Section III-B. Finally, Section III-C gives our heuristic algorithm to construct a near optimum routing tree.

A. Network Model

1) *Interference Model*: We employ the *Protocol Model* introduced in [4] to incorporate the interference in problem formulation. We assume each node is equipped with one radio working on a common channel with transmission range R_i and larger interference range R'_i . There is a transmission link from i to j if the distance between them, d_{ij} , is smaller than R_i , and there is an interference link from i to j if d_{ij} is between R_i and R'_i . We refer to the resulting directed graph as $G = (V, L^T, L^I)$ where V represents the set of nodes, L^T represents the set of the transmission links and L^I represents the set of the interference links.

A transmission from i to j is successful if and only if $d_{ij} \leq R_i$ and any node k with $d_{kj} \leq R'_k$ is not transmitting. We denote the links from i to j by l_{ij} . We refer to link l_{ij} as incoming link for node j and as outgoing link for node i . We denote by $L_{in(v)}$ the set of incoming links for node v and by $L_{out(v)}$ the set of outgoing links for node v . We always label BS with node 0. And we define uplink flow to be the flow from SS to BS, downlink flow to be the flow from BS to SS.

2) *Necessary and Sufficient Constraints*: In this section, we form the constraints, refined from [10], for WiMax Mesh. The major difference between following constraints and [10] is that following constraints are designed for 802.16 systems while [10]'s constraints are designed for 802.11 systems. Note the following constraints are not restricted to WiMax and can be applied to any TDMA-based networks. We define variable a_i^t to indicate whether the transmission link from node i to node

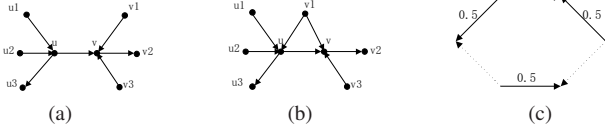


Fig. 2. Interference constraints. Solid line indicates transmission link and dotted line indicates interference link.

j is active in time slot t as follows:

$$a_l^t = \begin{cases} 1, & \text{if } l \text{ is active in time slot } t \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Note that $a_l^t = 0, \forall l \in L^I$.

Lemma 1: For any time slot t , a link schedule is valid if and only if it abides by following constraints:

$$\sum_{\forall l \in L_{in}(v) \cup L_{out}(v)} a_l^t \leq 1, \quad \forall v \in V, \forall t \quad (2)$$

$$\sum_{\forall l \in L_{out}(u) \cup L_{in}(v)} a_l^t \leq 1, \quad \forall l_{uv} \in L^T \cup L^I, \forall t \quad (3)$$

Proof: Constraint (2) holds because we assume one node has only one interface and every node is working on the same channel. Constraint (3) holds because any two of these links interfere with each other. This can be seen in Fig. 2(a) for $l_{u1u3}, l_{uv}, l_{v3v}$. Note that these constraints make feasible scheduling only if they are imposed on every link. In Fig. 2(b) for example, if we take l_{uv} as reference link, then l_{u1u} and l_{v1v} can be active simultaneously. But when we take l_{v1u} as reference link, Constraint (3) will obviate such scheduling. What's more, any schedule violating anyone of the constraints will not be feasible.

So any schedule satisfies these constraints will not cause collision and any valid schedule must satisfy these constraints. ■

3) *Necessary Constraints:* In this section, we develop necessary constraints from constraints in last section since the necessary and sufficient constraints lead to ILP (Integer Linear Programming) which is hard to solve. We relax the integer constraints by using link utilization:

$$u_{l_{ij}} = \frac{f_{l_{ij}}}{c_{l_{ij}}} = \frac{1}{T} \sum_{1 \leq t \leq T} a_{l_{ij}}^t \quad (4)$$

where $f_{l_{ij}}$ is the flow over l_{ij} , $c_{l_{ij}}$ is the capacity of l_{ij} , T is the length of a frame.

Thus constraints from (2) to (3) become:

$$\sum_{\forall l \in L_{in}(v) \cup L_{out}(v)} u_l \leq 1, \quad \forall v \in V, \forall t \quad (5)$$

$$\sum_{\forall l \in L_{out}(u) \cup L_{in}(v)} u_l \leq 1, \quad \forall l_{uv} \in L^T \cup L^I, \forall t \quad (6)$$

And it is obvious that any valid schedule must satisfy constraints (5) to (6).

However schedule satisfying constraints (5) to (6) is not necessarily feasible. Consider the directed graph in Fig. 2(c),

where a dotted line means interference link. The link utilization labeled on transmission links satisfies all the above constraints, resulting in a total utilization of 1.5, but actually only one transmission link can be active at any time, that is, the sum of link utilization is at most 1. Consequently, constraints from (5) to (6) are necessary rather than sufficient.

B. Optimum Routing Tree

Since [1] specifies that if the bandwidth requests cannot be scheduled, SSs should reduce their bandwidth requests proportionally. So the optimum routing tree and scheduling can be modeled as a maximum concurrent flow problem, offering proportional fairness to all flows. We now formulate an Integer Linear programming (ILP) to find an optimum routing tree and flow assignment. Note, from now on, we assume all transmission links to be bidirectional, that is, if $l_{ij} \in L^T$, then $l_{ji} \in L^T$. This is because, in a tree structure, the uplink flow and downlink flow of one node will pass the same path with opposite direction. The resulting ILP (ILP1) is given below:

$$\max \lambda \quad (7)$$

subject to the constraints (2) to (4) and

$$\lambda d_{j0} + \sum_{i:l_{ij} \in L^T} y_{l_{ij}} f_{l_{ij}} = \sum_{i:l_{ji} \in L^T} y_{l_{ji}} f_{l_{ji}}, \quad \forall j \in V \setminus \{0\} \quad (8)$$

$$\sum_{i:l_{ji} \in L^T} y_{l_{ji}} f_{l_{ji}} = \lambda d_{0j} + \sum_{i:l_{ij} \in L^T} y_{l_{ij}} f_{l_{ij}}, \quad \forall j \in V \setminus \{0\} \quad (9)$$

$$\sum_{i:l_{0i} \in L^T} y_{l_{0i}} f_{l_{0i}} = \lambda \sum_i d_{0i}, \quad (10)$$

$$f_{l_{ij}} \leq c_{l_{ij}}, \quad \forall l_{ij} \in L^T \quad (11)$$

$$\sum_{i:l_{ji} \in L^T} y_{l_{ji}} = 1, \quad \forall j \in V \quad (12)$$

$$y_{l_{ij}} + y_{l_{ji}} \leq 1, \quad \forall l_{ij} \in L^T \quad (13)$$

$$y_{l_{ij}} \in \{0, 1\}, \quad f_{l_{ij}} \geq 0, \quad \forall l_{ij} \in L^T \quad (14)$$

where d_{j0} is the uplink bandwidth request of SS j , d_{0j} is the downlink bandwidth request of SS j , and $y_{l_{ij}}$ equals 1 if node j is node i 's parent in the routing tree.

Constraints (8), (9), (10) ensure that flow conserves and, coupled with (12), that there is only one outer link for every SS, therefore ensuring tree structure. The solution from above ILP gives both optimum scheduling and optimum routing tree derivable from $y_{l_{ij}}$.

C. Heuristic Solution

The ILP presented in section IV, though can attain an optimum solution to the problem, is NP-hard. In this section, we will develop a heuristic algorithm to construct a near optimum tree.

We derive our proposed tree in two steps: (a) we solve the maximum concurrent flow on the general directed graph and (b) we construct the tree based on the flow derived in step (a).

The maximum concurrent flow LP (LP1) is:

$$\max \lambda \quad (15)$$

Algorithm 1: Spanning Tree Algorithm

Input: Network with $w_{l_{ij}}$ on each link l_{ij}
Output: Spanning tree
1 $i \leftarrow 0; u_0 \leftarrow 0; S_0 \leftarrow \{0\}; L(v) \leftarrow -1 \forall v \in V \setminus \{0\};$
while $i < |V| - 1$ **do**
2 **foreach** $v \in V \setminus S_i$ **and** $l_{u_i v} \in L^T$ **do**
3 $\zeta = \max(L(v), w_{l_{u_i v}});$
4 **if** $\zeta > L(v)$ **then** $L(v) \leftarrow \zeta; p(v) \leftarrow u_i;$ **end**
5 **end**
6 $u_{i+1} \leftarrow \arg \max_{v \in V \setminus S_i} L(v);$
7 $S_{i+1} \leftarrow S_i \cup u_{i+1}; i \leftarrow i + 1;$
8 mark the link between $p(u_i)$ and u_i as
 transmission link in spanning tree;
9 **end**

subject to constraints (4) to (6), and

$$\sum_{i:l_{ji} \in L^T} x_{l_{ji}}^{j0} = \lambda d_{j0}, \quad \sum_{i:l_{0i} \in L^T} x_{l_{0i}}^{0j} = \lambda d_{0j}, \quad \forall j \in V \setminus \{0\} \quad (16)$$

$$\sum_{i:l_{ij} \in L^T} x_{l_{ij}}^{q0} = \sum_{i:l_{ji} \in L^T} x_{l_{ji}}^{0q}, \quad \forall j \neq q, \forall q \in V \setminus \{0\} \quad (17)$$

$$\sum_{i:l_{ij} \in L^T} x_{l_{ij}}^{0q} = \sum_{i:l_{ji} \in L^T} x_{l_{ji}}^{0q}, \quad \forall j \neq q, \forall q \in V \setminus \{0\} \quad (18)$$

where $x_{l_{ij}}^{q0}$ indicates the uplink flow of SS q over l_{ij} , $x_{l_{ij}}^{0q}$ indicates the downlink flow of SS q over l_{ij} .

By running LP1 on G , we obtain each SS's uplink and downlink flow over each link. We then assign weight to links according to the following criteria:

$$w_{l_{ij}} = \sum_{q \in V \setminus \{0\}} (\beta_q x_{l_{ij}}^{0q} + x_{l_{ij}}^{q0}) \quad (19)$$

where

$$\beta_q = \begin{cases} d_{0q}/d_{q0}, & \text{if } d_{q0} \neq 0 \\ 1, & \text{otherwise} \end{cases}$$

is the ratio of downlink bandwidth request to uplink bandwidth request of node q . Then we use a greedy algorithm depicted in Algorithm 1 to construct the tree.

Algorithm 1 is like Prim's minimum spanning tree algorithm except that the graph is directed and we choose, at each step, a maximum weighted link out of link candidates. Link bi-direction assumption guarantees that there is a solution. Choosing maximum weighted link at each step means augmenting the probability of concurrency in the resulting tree as much as possible. Furthermore, given that all flows occur between SSs and BS, the links closer to BS will contain larger amounts of flow. So choosing links beginning at BS will give links closer to BS higher priority to be selected.

For networks with moderate size, we can directly use LP1's result as Algorithm 1's input. For networks composed of large number of SSs, directly solving LP1 maybe time consuming. In this case, we can use a FPTAS (Fully Polynomial Time Approximation Scheme) [11], which computes a $(1 - \epsilon)^{-3}$ optimal solution to LP1, to accelerate computing speed. By

choosing ϵ appropriately, we can balance the tradeoff between running time and accuracy. Moreover, SSs are supposed to be stationary in [1] so the traffic variation will not be great between two rounds of bandwidth request-grant process. The Mesh BS therefore can choose to adjust the routing tree every several rounds if the cost of computation is large. In addition, if the network designer can obtain the long-term bandwidth request distribution in a mesh network, the above algorithm can be run just once before deployment.

D. Feasible scheduling

By running LP1 on the generated tree, we can only get an upper bound on the scaling factor due to the reasons stated in Section III-A.3. However feasible traffic scheduling is NP-hard even under centralized scheduling [12].

In order to get feasible scheduling factor to make evaluation in the next section, we employ the greedy coloring method as used by [8], [10]. First we multiply all flows by the frame duration, $T_f * \tau$, where T_f is the total number of time-slots in a frame and τ is the time-slot duration. Then we apply following procedure.

- 1) Select a link, l , with highest residual flow.
- 2) Assign the smallest time slot, t , which doesn't violate condition (2) and (3), to l .
- 3) Reduce the flow by $c_l \tau$.
- 4) Repeat until all flows are scheduled.

If total number of time slots needed is T_t , the feasible scaling factor is

$$\lambda = \frac{T_f}{T_t} \lambda', \quad (20)$$

where λ' is the scaling factor achieved by running LP1 on the resulting tree. In our simulation, we choose T_f to be 256 and τ to be $39\mu s$.

Greedy coloring is just one kind of scheduling algorithms. In fact our tree construction algorithm does not depend on any specific scheduling algorithm. Our objective is to construct a tree having potential of high spatial reuse, therefore improving the throughput for any scheduling algorithm.

IV. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed routing tree by comparing it to the optimum routing tree, interference-aware tree [8] and the shortest-path tree (SPT) in terms of feasible scaling factor λ obtained in Section III-D. To be accurate, we solve ILP1 and LP1 by *LINGO* [13].

A. Grid Topology

We consider a 100×100 grid square with nodes distributed at intersections while keeping the graph's connectivity. The most upper right node is selected as the BS and all the other nodes serve as SSs. Each node has a transmission range of 1 unit, which means the node degree is at most 4. All links are transmission links in the configuration. The bandwidth request of each SS is randomly distributed between $[0, 3]$ units. Each link has a capacity of 100 units. We measure two parameters

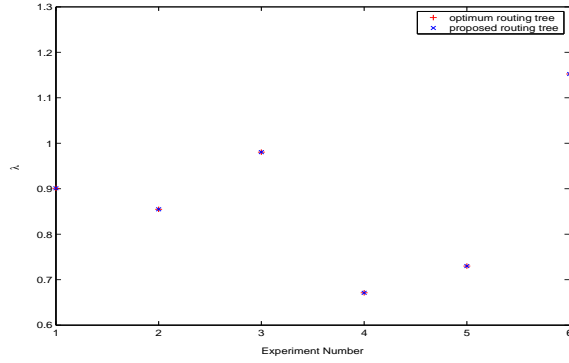


Fig. 3. 20-node example

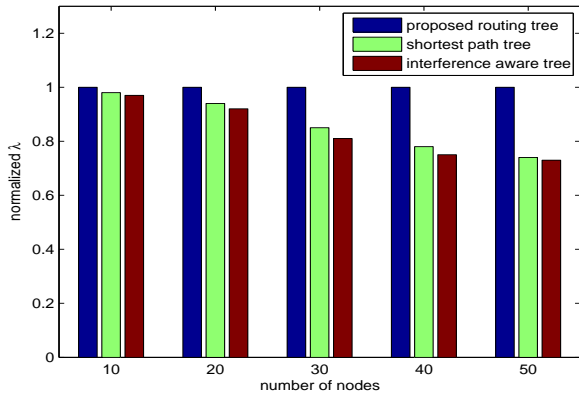


Fig. 4. Comparison to SPT and interference-aware tree

for each experiment: (a) optimum routing tree feasible scaling factor, (b) proposed routing tree feasible scaling factor.

Since ILP1 is only solvable when the problem scale is small, we vary the number of nodes from 10 to 20. We observe the proposed routing tree scaling factor is always equal to that of the optimum routing tree in our experiments. We only present the 20-node result in Fig. 3 due to space limit.

B. Random Topology

We randomly place nodes in a 1000×1000 square following uniform distribution. All nodes are assumed to have the same transmission range. All links are transmission links. Two nodes are connected if they are within each other's transmission range. We evaluated the bandwidth request distribution of both uniform distribution and exponential distribution, and observed similar results. We present the exponential distribution results here. The probability density function (PDF) of exponential distribution is $\frac{1}{\mu} e^{-\frac{x}{\mu}}$. We control the average node degree, which is rounded to its nearest integer, by adjusting each nodes' transmission range. Each result in the following simulation is averaged over 5 runs.

First, we set average node degree to be 6, exponential parameter $\frac{1}{\mu}$ to be 5. We compare our proposed routing tree to interference-aware tree and SPT in terms of feasible

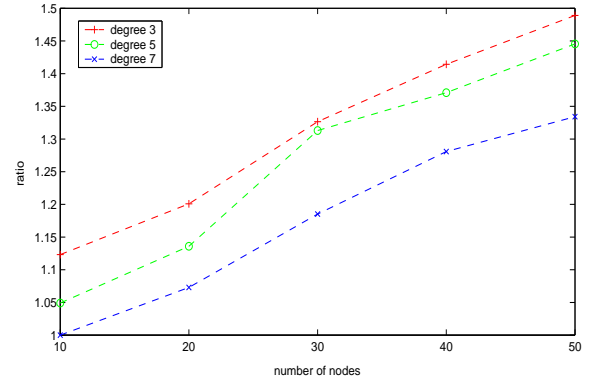


Fig. 5. Impact of node degree

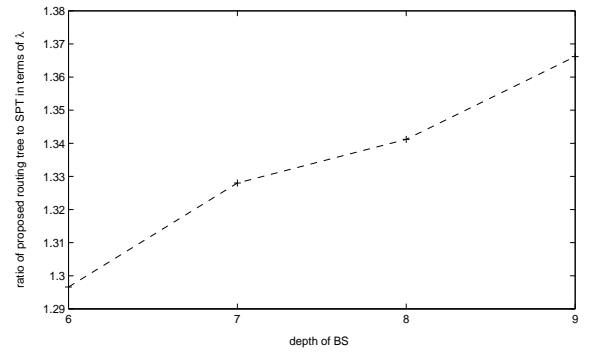


Fig. 6. Impact of BS position

scaling factor. Interference-aware tree is constructed according to [8] with node joining order following breadth-first order. We present the result in Fig. 4. Each result is averaged over 10 experiments and normalized to our proposed routing tree. From Fig. 4, we see our proposed routing tree is the best out of the three schemes and SPT is slightly better than interference-aware tree.

Then we evaluate the correlation between such parameters as average node degree, node numbers, BS position, $\frac{1}{\mu}$ of exponential distribution and link capacity distribution to the performance of our proposed routing tree in random graphs. Each data point in Fig. 5, 6, 7, 8 is the ratio of the proposed routing tree to the SPT in terms of feasible scaling factor λ and is averaged over 10 random graph topologies. The higher the ratio is, the better our proposed routing tree is than the SPT.

1) *Impact of Node Degree and Number:* We vary the number of nodes from 10 to 50 and the average node degree from 3 to 7. The $\frac{1}{\mu}$ is fixed at 5. We partition the links in decreasing order in terms of their physical length into three sets and assign the links in these three sets with capacity of 40, 80 and 120 units respectively. These capacities approximately corresponds to QPSK, 16QAM and 64QAM in 802.16.

From Fig. 5, we can see that the ratio increases when there are more nodes and decreases when the node degree increases. This is because more routing choices are available for nodes

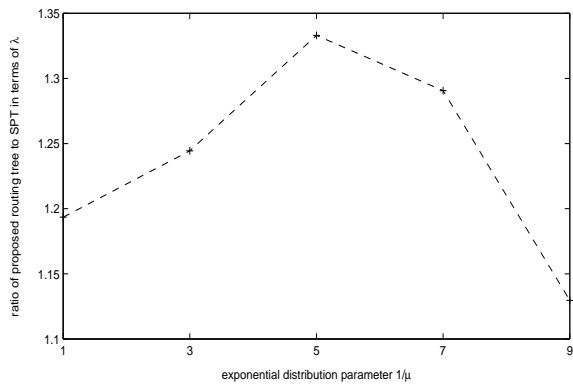


Fig. 7. Impact of $1/\mu$

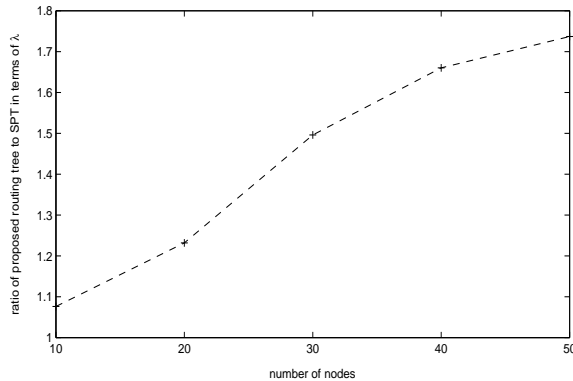


Fig. 8. Impact of link capacity

when the number of nodes increases. And our proposed algorithm is good at choosing advantageous route. When the node degree increases, the corresponding interference also increases so it becomes harder for our proposed algorithm to find routing tree better than SPT.

2) *Impact of BS Position:* We consider random graphs of 30 nodes with average node degree of 5. Other configuration is the same as IV-B.1. We measure the BS position in terms of the depth. The depth of a node is the number of hops of the longest of the shortest paths from it to all other nodes. We plot the results with respect to the depth of BS in Fig. 6.

From Fig. 6, we can see the performance increases as the depth rises. The reason is that, as the depth rises, more routes are available for nodes to choose.

3) *Impact of $\frac{1}{\mu}$:* We consider a random graph of 30 nodes with average node degree of 5. Other configuration is the same as IV-B.1 except that $\frac{1}{\mu}$ varies from 1 to 9. We present the result in Fig. 7.

From Fig. 7, we can see that at first the ratio rises with the increase of $\frac{1}{\mu}$ and reaches the summit when $\frac{1}{\mu} = 5$. Then it begin to decline as $\frac{1}{\mu}$ increases. This is because when the deviation of bandwidth request is very small, the SPT is just close to the optimum routing tree. And when the deviation is very large, it is hard to select paths with consistent remaining bandwidth from BS to SS, so the advantage is not obvious either.

4) *Impact of Link Capacity:* We consider random graphs of 30 nodes with average node degree of 5. Other configuration is the same as IV-B.1 except the link capacity assignment. We measure the largest hops, H , to the BS of all nodes. Links whose endpoints are within $H/3$, $2H/3$ and H hops from the BS are assigned capacity of 120, 80 and 40 units respectively. The result is shown in Fig. 8.

From Fig. 8, we can see that the gain of proposed tree over SPT is significantly larger than Fig. 5. This is because assigning less capacity to links farther from BS makes them easier or as easy to incur scaling as links nearer to BS. But far links have less influence on proposed routing tree since more routes to BS are available for them and our algorithm is good at choosing advantageous route when multiple routes are available. So the advantage of our algorithm is more obvious in this scenario.

V. CONCLUSION

We studied the impact of routing tree on the performance of WiMax Mesh. By modeling the WiMax Mesh TDMA-based network, we presented a tree construction algorithm accounting for both interference and bandwidth request. Simulation results showed that the proposed routing tree can achieve throughput close to optimum routing tree and significantly outperforms existing routing tree construction methods. The proposed algorithm is also useful for future consideration in the design of wireless mesh networking in 802.16.

REFERENCES

- [1] *IEEE Standard for Local and metropolitan Area Networks Part 16: Air Interface for Fixed Broadband Wireless Access Systems*, IEEE Std. 802.16, 2004.
- [2] Wimax forum. [Online]. Available: <http://www.wimaxforum.org/>
- [3] A. Khanna and J. Zinky, "The revised arpanet routing metric," in *Proc. ACM SIGCOMM*, Sept. 1989, pp. 45–56.
- [4] K. Jain, J. Padhye, N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," in *Proc. ACM MOBICOM*, Sept. 2003, pp. 66–88.
- [5] Y. Lu and G. Zhang, "Optimum fair bandwidth allocation scheme for ieee 802.16 mesh mode with directional antenna," in *Proc. 64th VTC'2006 Fall*, Sept. 2006.
- [6] B. Awerbuch, D. Holmer, and H. Rubens, "High throughput route selection in multi-rate ad hoc wireless networks."
- [7] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks," in *Proc. SenSys*, Nov. 2003.
- [8] H. Wei and J. Haas, "Interference-aware ieee 802.16 wimax mesh networks," in *Proc. 61st VTC'2005 Spring*, vol. 5, May 2005, pp. 3102–3106.
- [9] S. Ramanathan and E. L. Lloyd, "Scheduling algorithms for multihop radio networks," *IEEE/ACM Trans. Networking*, vol. 1, no. 2, pp. 166–177, Apr. 1993.
- [10] M. Kodialam and T. Nandagopal, "Characterizing the capacity region in multi-radio multi-channel wireless mesh networks," in *Proc. ACM MOBICOM*, Aug. 2005, pp. 73–87.
- [11] G. Karakostas, "Faster approximation schemes for fractional multicommodity flow problems," in *Proc. Foundations of Computer Science*, Nov. 1998, pp. 300–309.
- [12] E. Arıkan, "Some complexity results about packet radio networks," *IEEE Trans. Inform. Theory*, vol. 30, pp. 681–685, July 1984.
- [13] "Lingo," <http://www.lindo.com/>.