Load-Balanced Routing for Hybrid Fiber/Wireless Backhaul Networks

Yan Yan, *Qiang Hu* and *Douglas M. Blough*
School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, 30332

***Abstract*—Dense deployment of small-cell base stations (BSs) requires a backhaul network to efficiently connect the BSs to the core network. In this paper, we focus on a hybrid backhaul architecture where some BSs connect with fiber to the core network and provide mmWave backhaul connections for the rest of the BSs. This architecture brings new challenges, e.g., how to prevent a large amount of traffic from becoming concentrated at certain egress BSs, thereby hurting overall backhaul performance. In this paper, we propose a load-balanced routing algorithm to address this challenge. We first define the concept of load balance factor (LBF) and address the challenge through a hill climbing procedure that attempts to minimize LBF. Results show that the proposed algorithm can distribute the dynamic traffic loads from different BSs nearly optimally among fiber-connected BSs for the simulated settings. We also present a variation of the algorithm that permits trade-offs between routing path length and load balance factor.**

I. Introduction

With the explosive growth of mobile data demand, capacity of both access and backhaul networks has become a critical issue. For access networks, dense deployment of small cell base stations (SBSs) is a key approach to enhance the system throughput. However, dense SBS deployments require the backhaul to provide flexible and reliable connections between SBSs and the core network. Fiber connections, where available, are ideal for backhaul traffic. However, connecting large numbers of SBSs via fiber is expensive and time consuming to deploy [1]. This is especially true in areas where fiber is still not widely available, such as much of North America.

Millimeter-wave (mmWave) communication, with its spectrum availability and multi-Gigabit-per-second (Gbps) data rates, is an attractive alternative to support the high demand in backhaul networks. However, there are challenges with mmWave, including higher propagation loss, link directivity, and susceptibility to blockage. These factors can limit the communication range of mmWave wireless links to a few hundred meters or less to achieve the promised multi-Gbps data rates. With wireless networks becoming more dense and heterogeneous, it is necessary to guarantee connectivity and capacity in a cost-effective and sustainable way for a diverse set of applications. Recently, hybrid backhaul architectures with a combination of fiber and mmWave connections have received considerable attention to address these challenges [2].

In a hybrid backhaul network, only a subset of SBSs in a given region connect directly to the core network via fiber (these are referred to as anchor BSs (ABSs)), while the other SBSs connect wirelessly to nearby SBSs/ABSs via mmWave links. If multiple ABSs exist in a given region, it is known as a distributed backhaul architecture [3], [4]. In [5], system-level simulations have shown that distributed backhaul achieves higher throughput and is more flexible than a centralized network architecture, where all SBSs connect to a single ABS [6], [7]. In the remainder of the paper, we focus on a hybrid distributed backhaul architecture.

In hybrid fiber-wireless backhauls, the ABSs relay all traffic to/from their assigned SBSs across the wireless channel, making the ABSs potential bottleneck points that limit backhaul performance [8], [9]. Balancing load across the ABSs is thus an important problem. While we believe the exact problem studied herein is novel, several works are tangentially related [10], [11], [12]. In [10], mmWave backhaul is targeted, but the problem considered is how to allocate resources between access and backhaul tiers in an integrated access backhaul architecture, which is quite different from our problem. In [11], load-balanced tree-based routing is considered for *wireless access networks* with a *single egress node*, whereas balancing load across the *multiple ABS egress nodes* in hybrid fiber/wireless backhaul networks is a critical aspect of our problem. In [12], multiple egress nodes are considered in MANETs. However, the algorithm in [12] requires exponential time and prioritizes minimizing path length, whereas our algorithm is a hill climbing procedure with polynomial time per step and with load balancing as the primary objective.

Our contribution begins with a definition of load balance factor (LBF) to describe how balanced (or unbalanced) the load is across ABSs in a given backhaul region. We then use the LBF to transform the problem into a load-balanced routing problem. Next, we develop and evaluate a hill-climbing procedure for selecting routes, which is used to optimize the LBF in the backhaul region. Finally, we consider a modification to our load-balanced routing procedure, which allows a trade-off between routing path length and load balance factor.

II. Network Model and Problem Formulation

In this section, we present the model and assumptions of the network, define the concept of load balancing factor, and formulate the load-balancing problem in our network scenario.

A. Network Model

Backhaul topology: We first define the network model we assume for connections between the core network and the mmWave backhaul network. Fig. 1 shows an example of our hybrid distributed network architecture, which includes a fiber connection between the core network and a subset of the base

stations that are referred to as anchor base stations (ABSs), and a mmWave mesh network connecting the remaining base stations to the ABSs. In the hybrid backhaul network

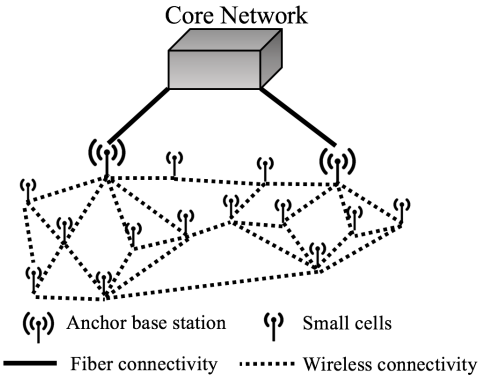

Fig. 1. Hybrid distributed network architecture

architecture, the traffic of non-fiber-connected small cell base stations (SBSs) is transmitted to/from one of the ABSs via mmWave wireless links. The traffic from ABSs to/from the core network is forwarded through fiber connections. Through the algorithm that we will present in the next section, each SBS is assigned to one ABS and then a virtual tree topology is constructed within the mesh network based on the SBS-ABS assignment (see Fig. 2 for an example). All traffic is then routed through this virtual tree topology.

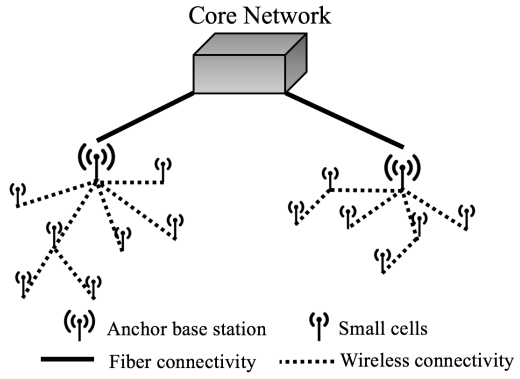

Fig. 2. Virtual tree topology for routing

As can be seen from Fig. 2, each ABS must process the aggregated traffic to/from all of its assigned SBSs through its wireless links. Thus, the wireless capacity of the ABSs becomes the main performance bottleneck in this hybrid architecture and, therefore, it is critical to balance the load as equally as possible among the ABSs. Since traffic demands are time-varying, the load balancing procedure that produces the virtual tree topology can be periodically re-executed (thereby changing both the SBS-ABS assignment and the routing paths) to maintain a balanced assignment as loads fluctuate.

mmWave backhaul in 3D urban areas: To support the high data rate requirement of mmWave links (around 10 Gbps) in backhaul networks, Line of Sight (LoS) paths must be utilized. However, LoS paths are susceptible to blocking by physical objects. When considering urban areas specifically, the LoS path between two BSs is often blocked by buildings, walls, trees, and other obstacles. Therefore, we use a 3D model of the environment like our earlier work [7], [13], [14] instead of 2D modeling, as it gives us a more practical view of the transmission environment. We used a 3D model of buildings in downtown Atlanta to provide a realistic evaluation environment and use it in the simulations described later. In the simulations, the SBSs are deployed on randomly selected top corners of buildings and ABSs with fiber connection are randomly chosen from these SBSs. Note that the following proposed problems and algorithms are not limited to this specific city model, which is used only for the purpose of producing a realistic environment for simulation.

B. Problem Formulation

We model the network architecture by a distributed hybrid backhaul topology such as Fig. 1. Let the N SBSs in the network be denoted by $B_S = \{B_{S0}, B_{S1}, \ldots B_{S(N-1)}\}$ and the M ABSs be denoted by $B_A = \{B_{A0}, B_{A1}, \ldots B_{A(M-1)}\}$. We denote the set of all base stations by $\mathcal{B} = B_S \cup B_A$. We represent the mesh network as an undirected graph $G(V, L)$, in which $V = \mathcal{B}$ and L is the set of wireless links in the mesh network.

We assume that SBS $B_{Si},\ 0 \leq i \leq N-1$, has a load TL_i, which represents the traffic load between B_{Si} and the core network. Let $B_{Aj},\ 0 \leq j \leq M-1$, be any ABS, and let $S_j \subseteq B_S$ be the set of SBSs that are assigned to B_{Aj} through a load-balanced routing procedure. Then, the aggregated traffic load ATL_j of ABS B_{Aj} is

$$\mathrm{ATL}_j = \sum_{i:B_{Si} \in S_j} \mathrm{TL}_i \tag{1}$$

A primary metric of a load-balanced routing procedure is the load balance factor (LBF), defined by the following equation:

$$\mathrm{LBF} = \frac{\max_j(\mathrm{ATL}_j) - \min_j(\mathrm{ATL}_j)}{\max_j(\mathrm{ATL}_j)} \tag{2}$$

In Eq. (2), the maximum and minimum are taken over all ABSs, i.e. over $0 \leq j \leq M-1$. LBF represents the ratio of load difference between the ABS with maximum traffic load and the ABS with minimum traffic load. Note that if all ABSs are assigned identical loads (perfect load balancing), then $\mathrm{LBF} = 0$. Note also that the worst case for load balancing is that some ABS is not assigned any SBSs, i.e. $\min_j(TL_j) = 0$ and, in that case, $\mathrm{LBF} = 1$. Thus, the range of LBF is $[0, 1]$ with 0 being the best value (perfect load balancing) and 1 being the worst value (at least one ABS has no load). The problem of optimally balancing the load is, therefore, to assign SBSs to ABSs in order to minimize the value of Eq. (2), which amounts to distributing the SBS traffic load as evenly as possible among the ABSs.

III. Load-Balanced Tree Construction

In this section, we describe an algorithm based on a hill climbing procedure that works to minimize the load balance

factor (LBF) for a given set of SBSs, ABSs, connected graph representing the mesh network, and SBS traffic loads. Given an initial tree topology, the procedure works by adjusting the topology so as to move toward a smaller LBF based on Eq. (2). Since it is well known that one execution of a hill climbing procedure produces a local optimum, we repeat the procedure multiple times from different initial tree topologies and select the best LBF from among all the resulting local optima. Pseudocode for the algorithm implementing this load balancing optimization procedure is shown in Algorithms 1 and 2 and is described next.

Algorithm 1 Load-balanced tree construction procedure

Input: Graph $G(V, L)$, SBS set B_S, ABS set B_A, no. of SBSs N, no. of ABSs M, traffic load vector TL, no. of init. topologies n
Output: tree_paths

```
1: best_LBF = 1
2: for i = 1 to n do
3:    paths = Build_Initial_Virtual_Tree_Topology()
4:    curr_LBF = compute LBF using Eq. (1), (2)
5:    ctr = 0
6:    repeat
7:       paths = Adjust_Traffic_Load(paths, curr_LBF)
8:       new_LBF = compute LBF using Eq. (1), (2)
9:       if (new_LBF < curr_LBF) then
10:         curr_LBF = new_LBF
11:         ctr = 0
12:      else
13:         ctr++
14:   until (ctr = 10)
15:   if ((curr_LBF < best_LBF) OR (n == 1)) then
16:      best_LBF = curr_LBF
17:      tree_paths = paths
18: return
19: Build_Initial_Virtual_Tree_Topology()
20: for i = 0 to N − 1 do
21:    assign B_Si to a randomly chosen ABS B_Aj
22:    initial_paths[i] = shortest path between B_Si and B_Aj in G(V, L)
23: return initial_paths
```

Algorithm 1 shows the main outer loop of the procedure, along with the function that computes different initial trees on which the hill climbing procedure is executed. In addition to the inputs already mentioned, the number of initial trees to use is an input parameter n. The outer loop of the procedure runs n times and executes the hill climbing procedure once in each iteration starting from a different initial topology. Since the mesh network is assumed to be connected, we produce initial tree topologies by simply assigning each SBS to a random ABS and connecting the SBS to its assigned ABS via a shortest path. At the end of the procedure, the collection of paths that produced the lowest LBF is selected and these paths together form the set of trees used for routing.[1]

[1]Note that, since initially SBSs are assigned to ABSs randomly, some of the initial routing paths might be fairly long. In the algorithm of this section, we focus solely on minimizing LBF. In Sec. V, we modify the algorithm to include a maximum path length constraint so as to avoid long routing paths.

The heart of the hill climbing procedure takes place inside the Adjust_Traffic_Load() function, which is shown in Algorithm 2 and is described next. From Eq. (2), we can see that to lower the LBF, we can either reduce the maximum aggregated traffic load (ATL) on the ABSs or we can increase their minimum ATL. Adjust_Traffic_Load() searches for a modified SBS assignment that achieves one of these goals or moves in the direction of one of the goals if neither goal can be achieved with a single SBS reassignment. The first loop in Adjust_Traffic_Load() focuses on ABSs that currently have the maximum ATL. For each such ABS, we try reassigning each SBS currently assigned to it to every other ABS in the network. For each such reassignment, we compute the new LBF and, if any of the possible reassignments lowers the LBF from its current value, we select the new assignment that has the lowest LBF among all the reassignments considered. In the second loop, a similar procedure is repeated for the ABSs with minimum ATL, where we try to move an SBS from

Algorithm 2 Traffic load adjustment function

```
1: Adjust_Traffic_Load(paths, curr_LBF)
2: best_LBF = curr_LBF
3: adjusted = FALSE
4: for each B_Ak with max. load, i.e. ATL_k = max_l ATL_l do
5:    for each B_Si assigned to B_Ak do
6:       for each B_Aj ≠ B_Ak do
7:          old_max_ATL = max_l ATL_l
8:          temporarily reassign B_Si to B_Aj
9:          new_LBF = compute LBF using Eq. (1), (2)
10:         if (new_LBF < best_LBF) then
11:            best_LBF = new_LBF
12:            best_SBS = i
13:            best_p = shortest path from B_Si to B_Aj in G(V, L)
14:            adjusted = TRUE
15:         else if ((new_LBF == curr_LBF) AND (NOT adjusted)
            AND (ATL_j < old_max_ATL)) then
16:            best_SBS = i
17:            best_p = shortest path from B_Si to B_Aj in G(V, L)
18:            adjusted = TRUE
19:         cancel B_Si reassignment
20: for each B_Ak with min. load, i.e. ATL_k = min_l ATL_l do
21:    for each B_Aj ≠ B_Ak do
22:       for each B_Si assigned to B_Aj do
23:          old_min_ATL = min_l ATL_l
24:          temporarily reassign B_Si to B_Ak
25:          new_LBF = compute LBF using Eq. (1), (2)
26:          if (new_LBF < best_LBF) then
27:             best_LBF = new_LBF
28:             best_SBS = i
29:             best_p = shortest path from B_Si to B_Ak in G(V, L)
30:             adjusted = TRUE
31:          else if ((new_LBF == curr_LBF) AND (NOT adjusted)
             AND (ATL_j > old_min_ATL)) then
32:             best_SBS = i
33:             best_p = shortest path from B_Si to B_Ak in G(V, L)
34:             adjusted = TRUE
35:          cancel B_Si reassignment
36: if (adjusted) then
37:    paths[best_SBS] = best_p
38: return paths
```

another ABS *onto* the minimum load ABS. If we are able to lower the LBF with a single SBS reassignment, the collection of paths with the lowest resulting LBF from among all the possibilities over the maximum ATL and the minimum ATL ABSs is returned to the main procedure.

Note, however, that it might not be possible to lower the LBF with a single SBS reassignment. This can occur if there are multiple ABSs all having the maximum ATL and multiple ABSs all having the minimum ATL. In this case, multiple SBS reassignments will be needed to lower the LBF. If there is no single reassignment that lowers the LBF, Adjust_Traffic_Load() will return the first encountered reassignment that maintains the same LBF but reduces the number of maximum load ABSs or the number of minimum load ABSs. By reducing either the number of maximum or minimum load ABSs, the adjustment moves in the direction of reducing the LBF even if that cannot be accomplished in a single call to Adjust_Traffic_Load().

The time complexity of the core of our algorithm, namely Adjust_Traffic_Load(), is $O(N^3M)$, where N is the number of SBSs and M is the number of ABSs. Since N and M tend to be fairly small in practical scenarios, we have not tried to lower this complexity and, in fact, all executions of the procedure done to obtain the simulation results in the paper terminated very quickly.

IV. Performance Evaluation

Here, we provide results to assess the routing paths that are constructed by the load-balanced routing algorithm presented in the previous section. As mentioned earlier, we use an actual 3D topology of a section of downtown Atlanta to drive the simulations using a custom C++ simulator, as was done in [7], [13], [14].

A. Simulation Setup and Example

This 3D topology contains 227 buildings higher than 5 meters. We consider the rooftop corners of every building with a height between 20 and 200 meters as candidate BS locations. We use a grid to represent this downtown Atlanta area with each grid element corresponding to a 200m×200m section of the area. We then place one base station within each grid element that has a suitable building located in it. The total area has 63 grid elements, out of which 48 have suitable buildings and are used as base station locations. From these 48 locations, in one execution of our algorithm, a given number of ABSs are randomly picked from these locations and the remaining base station locations are assigned to be SBSs. Since long-distance high-rate links rarely exist in a dense urban environment due to signal attenuation and blockages, we assume each base station has at most four neighbors, which are the base stations in its adjacent 200m×200m areas in the north, south, east, and west directions if they exist.[2] Fig. 3 shows an example with the 63 grid elements and the locations of the 48 base stations indicated by dots. The larger red dots are the (in this case three) randomly chosen ABSs and the 45 smaller dots represent the SBSs.

[2]If the LoS link between neighboring BSs is blocked by a building, our prior work showed that they can be connected by a high-throughput virtual link using 1–2 relays in almost all cases [13], [14].

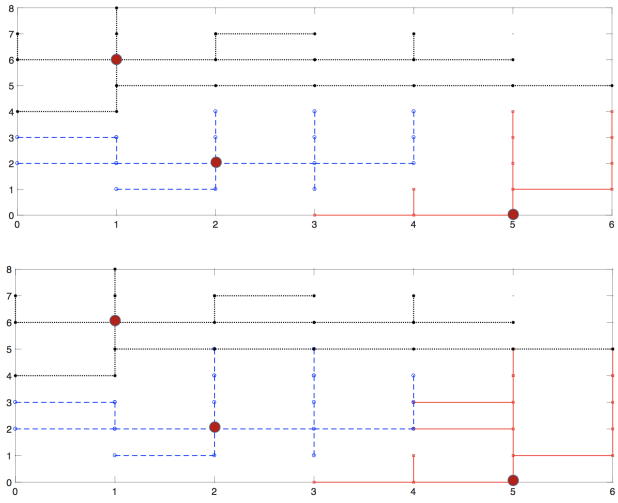

Fig. 3. (top) Initial virtual tree topology (bottom) Topology modified by the load balancing algorithm

Fig. 3 (top) shows an initial random tree topology and Fig. 3 (bottom) shows the modified virtual tree topology produced by our hill climbing procedure. To simplify the discussion of this example, let us assume that each SBS has one unit of traffic load. Under that assumption, the three ABSs in the initial randomly chosen virtual tree topology have aggregated loads of 19, 15, and 11 units, which results in a load balance factor of $8/19 \approx 0.42$. In the modified virtual tree topology, the aggregated load of each of the three ABSs is 15 units, resulting in a perfect load balance factor of 0.

B. Simulation Results

For each data point in the results of this section, we simulated 100 different random assignments of a given number of ABSs to the 48 possible locations and, for each of the 100 ABS assignments, the remaining locations were assigned to be SBSs. We considered scenarios with 3, 5, and 7 ABSs across the region. Initially, we consider a heterogeneous traffic load scenario, where we assign uniform random traffic loads in a range $[0, 2]$ to each SBS, which gives an average traffic load of 1. For each set of ABSs and SBSs with fixed positions, we constructed 100 different random initial topologies and executed our rebalancing procedure on each of them. We then compared the best LBF from the initial topologies (referred to as "initial multiple topologies" in the figures) to the best LBF from all of the rebalancing procedures.

Fig. 4 and Fig. 5 compare the LBFs and path lengths that are produced by these two different methods. As seen from Fig. 4, the LBFs produced by our load balancing procedure are substantially smaller than the best LBF from 100 random topologies. For example, when the number of ABSs is 7, the average LBF of the best initial tree topologies is almost 0.8, while the average LBF of the best rebalanced topologies is

less than 0.02, which is nearly perfect load balancing. Thus, using our load-balanced routing algorithm provides a very large improvement in LBF. There are a few exceptions in Fig. 4, where the LBF is 1. These are cases where one of the ABSs cannot connect to any SBS, i.e. its only neighbor is another ABS. In that case, no matter how rebalancing is done, the $\min(\mathrm{ATL}_j)$ is always 0 and the LBF is always 1.

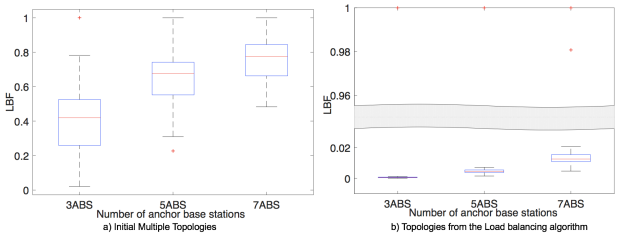

Fig. 4. LBFs among heterogeneous traffic load scenario a) initial multiple topologies b) topologies from the load balancing algorithm

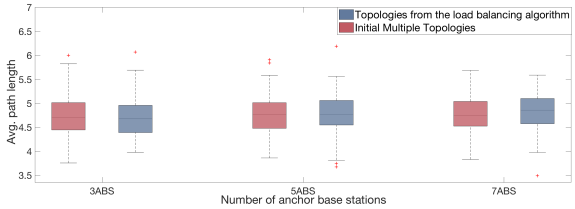

Fig. 5. Avg. path lengths among heterogeneous traffic load scenario a) initial multiple topologies b) topologies from the load balancing algorithm

We also evaluated the path lengths that are produced by the two methods. We took the best LBF topologies, as discussed in the prior paragraph, and we computed the average of the path lengths from every SBS to their assigned ABS. Fig. 5 reports the distribution of these average path lengths across the 100 random BS assignments for each experiment. The figure shows that the topologies modified by the load-balanced routing algorithm have similar average path lengths as the best initial topologies. Together, Fig. 4 and Fig. 5 demonstrate that our load-balanced routing algorithm can achieve very good load balancing with very little impact on path lengths.

In the heterogeneous traffic load scenario, there is no easy way to compare the LBF produced by our algorithm to the best achievable LBF. Therefore, we also evaluated a homogeneous traffic load scenario, where each SBS is assigned one unit of load. In this case, load balancing simply tries to equalize the *number* of SBSs assigned to each ABS and the best possible LBF corresponds to distributing integer numbers of SBSs as equally as possible. Here, the best possible such distribution can be determined easily. Fig. 6 and Fig. 7 show the LBFs and path lengths under this homogeneous traffic load scenario with all other experimental parameters the same as in the heterogeneous load experiments.

With 48 BSs, the optimal values of LBF with homogeneous loads can be easily determined to be 0, 0.125 and 0.1667 for 3ABS, 5ABS and 7ABSs, respectively. Fig. 6 shows that most of the LBFs produced by our algorithm are equal to the best possible achievable value. Most of the non-optimal cases are the aforementioned ones where one ABS is isolated from the

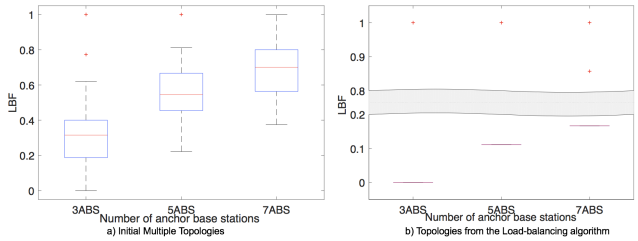

Fig. 6. LBFs among homogeneous traffic load scenario a) initial multiple topologies b) topologies from the load balancing algorithm

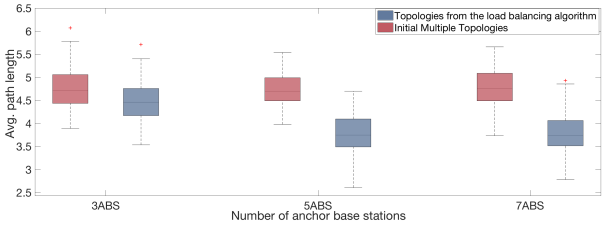

Fig. 7. Avg. path lengths among homogeneous traffic load scenario a) initial multiple topologies b) topologies from the load balancing algorithm

rest of the network. There is only a single case across all our simulations where a non-optimal LBF was produced without an isolated ABS. Somewhat surprisingly, Fig. 7 also shows that the path lengths of our optimized topologies are shorter than from the multiple initial topologies. This indicates that equalizing the numbers of SBSs assigned to ABSs also tends to reduce path lengths compared to more random assignments.

V. Load Balancing with Path Length Constraint

From. Fig. 5 and Fig. 7, we see that average path lengths produced by our load-balanced topologies are between 4.5 and 5 for heterogeneous loads and slightly lower for homogeneous loads. Since it has been observed that throughput in wireless multihop paths decreases with path length [15], maintaining relatively short path lengths is another factor to consider in optimizing network performance in addition to the load balance factor, which has been our primary consideration so far. Thus, in this section, we consider adding a maximum path length constraint to our load-balanced routing procedure.

To provide some perspective on the path lengths produced by our basic load balancing procedure, we being by comparing it to shortest path trees, which are the best possible topology in terms of path lengths for a tree-based routing procedure. We also consider a load-balanced modification of the shortest path tree topology (referred to as "load balanced SPT" in the figures), where we run our hill climbing procedure to reduce the LBF starting from a shortest path tree. Fig. 8 and Fig. 9 compare the topologies produced by these three methods for the same parameters as the prior experiments under the heterogeneous load scenario.

Fig. 8, shows that the average path lengths of the shortest path trees in our simulated scenarios are between 2.5 and 3 with 3 ABSs and slightly more than 2 with 5 and 7 ABSs, which are significantly lower than the average path lengths from our load balancing procedure. Note also that the load balanced modification of the shortest path tree maintains

average path lengths in the 2–3 range. In comparing LBFs, Fig. 9 shows that the load balanced SPT significantly lowers the LBF compared to the initial shortest path trees, which have a very poor LBF. However, note that the LBF of the load-balanced SPT is still substantially higher than the extremely low LBFs produced by our basic algorithm. These results motivate us to explore the trade-off possible between LBF and path length within our load balanced routing procedure.

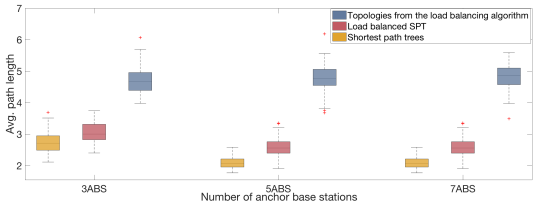

Fig. 8. Avg. path length comparison of different topologies

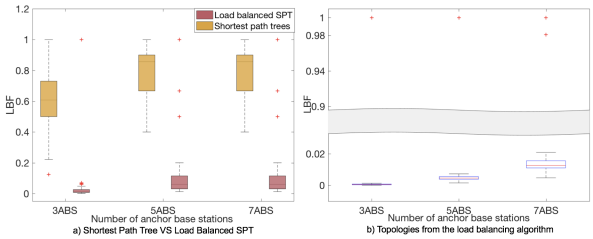

Fig. 9. LBF comparison of different topologies

To permit this trade-off study, we modified our load balancing algorithm to include a maximum path length constraint h. This was included in initial topology construction where the random SBS assignment is done only over ABSs that are within a distance of h or less from the SBS.[3] Also, in the topology adjustment procedure, any adjustments that produce path lengths longer than h are ignored. Fig. 10 shows the results of this path length constrained load balancing procedure compared with the load balanced SPT topologies already discussed. The figure shows that the LBF for the path

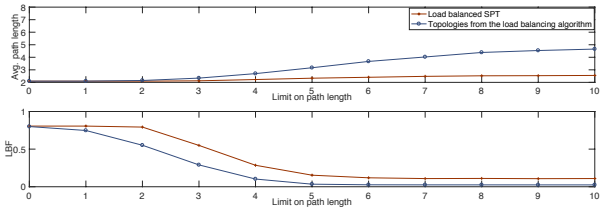

Fig. 10. Avg. path lengths and LBFs for the load balancing algorithm with path length constraint and load balanced SPT for 5 ABSs

length constrained algorithm approaches the LBF without any constraint as h increases. As already discussed, the path length increases with h also but the figure clearly shows how the two metrics can be traded off by simply choosing an appropriate value of h. Smaller h prioritizes shorter path lengths and larger h prioritizes LBF. Intermediate values can produce good values of both. For example, for $h = 3$, the path lengths of our algorithm and the load balanced SPT are 2.34 and 2.13, respectively, which represents only a 10% increase, while the LBFs of the two methods are 0.29 and 0.55, which represents almost a 50% decrease in LBF from our algorithm.

[3]If no such ABS exists, the SBS is assigned to the closest ABS.

VI. Conclusion

In this paper, we explored load-balanced routing in hybrid fiber-wireless backhaul networks. We presented a hill climbing procedure that produces a load-balanced virtual topology for routing, which was shown to provide near-optimal load balancing in simulated settings. We also modified the algorithm to include a maximum path length constraint, which allows trade offs between path length and load balance factor. Future work will consider issues related to dynamic traffic loads, namely how frequently the virtual routing topology should be adjusted and quantifying performance under dynamic conditions, accounting for both the quality of load balancing and the overheads of virtual topology adjustment.

Acknowledgement

This research was supported in part by the National Science Foundation through Award CNS—1813242.

References

[1] "Backhaul Technologies for Small Cells: Use Cases, Requirements, and Solutions," *Small Cell Forum Press Releases,* Feb. 2013.

[2] López. O.L, et al, "Hybrid Wired-Wireless Backhaul Solutions for Heterogeneous Ultra-Dense Networks," *2018 IEEE 87th Vehicular Technology Conference,* 2018, pp. 1-5.

[3] S. Sarabjot, et al,. "Tractable model for rate in self-backhauled millimeter wave cellular networks," *IEEE Journal on Selected Areas in Communications,* Vol. 33, no. 10 (2015): 2196-2211.

[4] X. Xiangxiang, et al,. "Joint deployment of small cells and wireless backhaul links in next-generation networks," *IEEE Communications Letters 19,* no. 12 (2015): 2250-2253.

[5] X. Ge, et al,. "5g wireless backhaul networks: Challenges and research advance," *Proc. of IEEE network,* 28(6), 6-11.

[6] R. J. Weiler, et al,. "Enabling 5G Backhaul and Access with Millimeter-Waves," *Proc. of IEEE EuCNC'14,* June 2014.

[7] H. Qiang, and D. Blough. "On the Feasibility of High Throughput mmWave Backhaul Networks in Urban Areas," *Proc. of International Conference on Computing, Networking and Communications (ICNC),* pp. 572-578. IEEE, 2020.

[8] B, Iman, et al,. "Two new biasing load balancing algorithms in distributed systems," *Proc. of First Asian Himalayas International Conference on Internet,* pp. 1-5. IEEE, 2009.

[9] I. Ivanisenko and T. Radivilova. "Survey of major load balancing algorithms in distributed system," *Proc. of information technologies in innovation business conference (ITIB),* pp. 89-92. IEEE, 2015.

[10] S. Chiranjib, et al,. "Load Balancing in 5G HetNets with Millimeter Wave Integrated Access and Backhaul," arXiv preprint arXiv:1902.06300 (2019).

[11] H. Pai-Hsiang, et al,. "Load-balancing routing for wireless access networks," *Proc. of IEEE Int'l Conference on Computer Communications,* pp. 986-995, 2001.

[12] Y. Yonghang, et al,. "Load balancing routing algorithm among multiple gateways in MANET with Internet connectivity," *Proc. of Int'l Conf. on Advanced Communication Technology,* pp. 388–392. IEEE, 2014.

[13] Y. Yan, Q. Hu, and D. Blough, "Optimal Path Construction with Decode and Forward Relays in mmWave Backhaul Networks," *Proc. of IEEE International Conference on Computing, Networking and Communications,* pp. 579-585, 2020.

[14] Y. Yan, et al,. "Path Selection with Amplify and Forward Relays in mmWave Backhaul Networks," *Proc. of IEEE Int'l Symposium on Personal, Indoor, and Mobile Radio Communications,* 2018.

[15] J. Li, et al,. "Capacity of Ad Hoc Wireless Networks," *Proc. of ACM MobiCom,* pp. 61-69, July 2001.