TRAFFIC ENGINEERING FOR BACKHAUL NETWORKS WITH WIRELESS LINK SCHEDULING

Nan Zhang[⋆]*, Wei-Cheng Liao*[†]*, Mingyi Hong*[♯]*, Hamid Farmanbar*[§]*, and Zhi-Quan Luo*[†‡]

[⋆]School of Mathematical Sciences, Peking University, China.
[†]Dept. of Electrical and Computer Engineering, University of Minnesota, USA.
[♯]Dept. of Industrial and Manufacturing Systems Engineering, Iowa State University, USA.
[§]Huawei Canada Research Center, Ottawa, Canada.
[‡]School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, China.

ABSTRACT

Traffic engineering (TE) problem is a central component of the next generation cloud-based wireless networks. In this paper, we study a new resource allocation scheme for effective traffic engineering under practical constraints such as the finite buffer size at each node. To reduce the computational effort required in the existing single-slot TE approaches and to deal with practical hardware limitations on link flows and buffers, we propose a two time-scale, low-complexity TE algorithm which incorporates a novel link scheduling component. The algorithm can be distributedly implemented. Simulation results demonstrate the effectiveness of the proposed algorithm.

Index Terms— Traffic Engineering, Link Scheduling, Two Time-Scale Approach.

1. INTRODUCTION

Central to the next generation cloud-based networks, traffic engineering (TE) problem in large-scale wireless backhaul networks has received much attention recently. In view of the increasing network scale and the existing hardware limitations, an effective scheme of traffic routing and link scheduling is very important. In wireless networks, the half-duplex nodes do not have the capability of simultaneously transmitting to and receiving from too many distinct nodes, as they all have limited number of RF chains. Since the interfering network nodes can not be activated simultaneously, a key challenge is to properly associate the transmitters and receivers in a slot-by-slot manner. The resulting problem can be formulated as an integer program [1, 3]. To deal with binary variables, most of the existing algorithms utilize certain forms of primal/dual decomposition with (sub)gradient update [1, 2], which can be slow. Meanwhile, many works focus on cross-layer design of routing, scheduling and power/flow control, and TE for multi-hop wireless networks [3, 4, 5, 6]. Among these works, a popular approach is to perform certain forms of relaxation to simplify the algorithm design. For example, references [3, 4] proposed to consider the dual problem and relax the flow conservation constraints by introducing Lagrangian multipliers. However, all of these works set up the model for only one time slot, during which it is impractical to apply iterative algorithms. Another limitation of these works is that they do not take into consideration of the practical hardware limitations, such as the limited buffer size per node. The well-known back pressure routing algorithm [7] routes flows with given demands and link capacity constraints. While performing well in general, it ignores the buffer capacity hence can not be directly applied to the case with finite-size buffers. There are a number of other link scheduling/routing algorithms which heuristically activate links while satisfying interference constraints [5, 6]. However, few works consider the effect of finite buffer size on multi-hop TE, and most of them do not utilize the real time buffer status to further improve the performance. In view of the limitations in these works, it is desirable to perform "predicted" routing in the long time-scale with a low-complexity algorithm, and perform link activation in the short time-scale while meeting the hardware constraints.

In this work, we propose a novel two time-scale joint flow routing and link scheduling algorithm. In our scheme, the slow time-scale performs traffic routing while the fast time-scale performs link scheduling. The link scheduler enforces the finite buffer size constraint while maximizing the overall system performance. Different from the existing single-slot approaches, our method requires less computation effort and can account for both the interference/hardware constraints and the finite buffer size constraint. The entire algorithm is distributedly implementable.

2. SYSTEM MODEL AND PROBLEM FORMULATION

The Network Model. Let us consider a wireless backhaul network represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{L})$, where $\mathcal{V} = \{v_i\}$ is the set of wireless nodes and $\mathcal{L} = \{l_{ij}\}$ is the set of directed links. We assume that each node is operated in the half-duplex mode, and there is one channel between each pair of connected nodes. M commodities are transported over the network in one long-term period, which consists of τ short-term periods, each being normalized as one unit time. For each commodity m, the source-destination pair is given as $(S(m), D(m))$. At each network node, there is one data buffer for each commodity.

THIS WORK IS SUPPORTED IN PART BY NSF, GRANT NUMBER CCF-1526434, AND BY NSFC, GRANT NUMBER 61571384.

Let f_{ij}^m be the flow rate of commodity m on link l_{ij}, x_{ij} be the binary variable indicating whether link l_{ij} is active ($x_{ij} = 1$ if l_{ij} is active, otherwise $x_{ij} = 0$). The capacity of each link l_{ij} during the long-term period is assumed to be known and fixed as C_{ij}. This assumption is reasonable when the power allocation/precoder design is fixed and known with stable channel conditions during the long-term period. Under these settings, the total flow rates over link l_{ij} is upper bounded by $x_{ij}C_{ij}$, i.e.,

$$\sum_{m=1}^{M} f_{ij}^m \leq x_{ij}C_{ij},\ \forall\, l_{ij} \in \mathcal{L}. \tag{2.1}$$

To simplify the description, we denote the buffer for commodity m at v_i as "buffer (m, i)"; we also refer to the size of the data in a given buffer as its "buffer status". We assume that buffer (m, i) has a finite capacity except for the buffers at the source and destination nodes (i.e., $i \in \{S(m), D(m)\}$), which are assumed to be infinite. This assumption is reasonable since the received data at destination $D(m)$ can be quickly removed and the buffers $\{(m, S(m))\}$ at the source nodes can be assumed to be never empty. Let z_i^m be the amount of data in buffer (m, i), and for simplicity let $\bar{z}$ denote the buffer capacity for all finite-size buffers. Then the individual buffer size constraints are $0 \leq z_i^m \leq \bar{z},\ \forall\, i \neq S(m), D(m),\ \forall\, m$.

Let $z_i^{m,0}$ and $z_i^{m,1}$ be the status of buffer (m, i) just before and after a long-term period respectively. For each node i, denote the set of in-neighbors and out-neighbors as $\mathrm{In}(i)$ and $\mathrm{Out}(i)$, respectively. For each commodity m, the total data received by v_i during one long-term period is $\tau \sum_{j\in\mathrm{In}(i)} f_{ji}^m$, and the total data transmitted from v_i is $\tau \sum_{k\in\mathrm{Out}(i)} f_{ik}^m$. Note that the difference of these quantities equals to the change in the buffer status z_i^m, therefore we can write the flow conservation constraints (for a single long-term period) as

$$z_i^{m,0} + \tau \sum_{j\in\mathrm{In}(i)} f_{ji}^m = z_i^{m,1} + \tau \sum_{k\in\mathrm{Out}(i)} f_{ik}^m,\ \forall\, i,\ \forall\, m. \tag{2.2}$$

Also we have the following buffer capacity constraints

$$0 \leq z_i^{m,1} \leq \bar{z},\ \forall\, i \neq S(m), D(m),\ \forall\, m. \tag{2.3}$$

The Interference Model. We use the well-known conflict graph to model the co-channel interference [3]. Specifically, we construct a conflict graph $\mathcal{C}$ in which each vertex represents a directed link in $\mathcal{L}$, and two vertices are connected if the corresponding two links interfere with each other hence cannot be activated at the same time. Denote $\mathcal{N}_{ij}$ as the set of links whose corresponding vertices in the conflict graph $\mathcal{C}$ are neighbors of the vertex corresponding to link l_{ij}. Then by the conflict graph model, at any given time at most one link in $\mathcal{N}_{ij} \cup l_{ij}$ can be activated, i.e.,

$$x_{ij} + \sum_{l\in\mathcal{N}_{ij}} x_l \leq 1,\ \forall\, l_{ij} \in \mathcal{L}. \tag{2.4}$$

In view of the buffer capacity constraints and the interference constraints, the flow rates need to be further constrained to avoid buffer overflow. For a given short-term period, if l_{ij} ($i \neq S(m), j \neq D(m)$) is active, then the maximum amount of flow m that v_i can transmit or v_j can receive is $\bar{z}$. This is because v_i cannot receive from other nodes or transmit to other nodes at the same time. Therefore, we have the following upper bound for the transmitted flow rate

$$f_{ij}^m \leq x_{ij}\bar{z},\ \forall\, i \neq S(m), j \neq D(m),\ \forall\, m. \tag{2.5}$$

An Initial Problem Formulation. The achieved data rate of commodity m during a given long-term period is measured by the averaged change in the status of the buffer at the destination node:

$$r_m = \frac{1}{\tau}\left(z_{D(m)}^{m,1} - z_{D(m)}^{m,0}\right),\ \forall\, m. \tag{2.6}$$

Let us define the utility $U(r_1, \ldots, r_M) : \mathbb{R}^M \to \mathbb{R}$, which is a function of all the users' rates in the network. Such utility function can take the form such as the minimum rate, the sum rate or the weighted sum rate.

With the network constraints listed above, a direct formulation of the long-term TE is given below:

$$\begin{aligned} \max_{\mathbf{f},\mathbf{x}} \quad & U(r_1, \ldots, r_M) \\ \text{subject to} \quad & (2.1) - (2.6), \\ & f_{ij}^m \geq 0, x_{ij} \in \{0, 1\},\ \forall\, m,\ \forall\, l_{ij} \in \mathcal{L}. \end{aligned} \tag{2.7}$$

There are a couple of problems with this formulation. First, problem (2.7) is a mixed integer linear program (MILP), which is generally NP-hard and therefore difficult to solve to global optimality. Second, problem (2.7) provides a single routing strategy for the entire τ short-term periods which may not be realizable given the buffer size constraints. The reason is that the flow conservation constraints (2.2) only conserve flow across one long-term period without considering the buffer status, nor the flow conservation constraint in each short-term period. To address these practical issues, a two time-scale formulation and a low-complexity, dynamic link scheduling and flow routing algorithm are needed.

3. THE TWO TIME-SCALE APPROACH

3.1. The Long-Term TE Controller

As we described before, it may be difficult to implement the flows and activation patterns obtained by solving (2.7) in a slot-by-slot manner. In this section, we propose a long-term TE controller to predict the preferred network flows and the frequency of link activation in a long period of time. In particular, we interpret x_{ij} in (2.7) as the time fraction that l_{ij} is active in the long-term period, and relax the integer variables to be continuous, i.e., $x_{ij} \in [0, 1], \forall\, l_{ij} \in \mathcal{L}$.

To improve the realizability of the relaxed long-term TE solution, we will add a few additional constraints to (2.7).

Adding Additional Constraints. Relaxing the binary variables in (2.7) makes the feasible region larger, potentially resulting in a solution that cannot be practically implemented. To mitigate the problem, we propose a few constraints that are redundant for (2.7) but can make the feasible region of the problem tighter.

Let us consider the 3-node cycle (i, j, k) in the network: v_i, v_j, v_k are connected with each other (see Fig. 1). Due to the half duplex constraint (HDC), any two directed links in the cycle can not be active simultaneously. Assuming all the possible six links exist in $\mathcal{L}$, the HDC with respect to the cycle is

$$x_{ij} + x_{ik} + x_{kj} + x_{ki} + x_{ji} + x_{jk} \leq 1, \tag{3.1}$$

which we call "triangular constraint". We can find out all 3-node cycles in $\mathcal{G}$ and add the corresponding constraints to (2.7). Similar

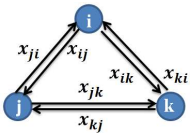

Fig. 1. A three-node cycle.

constraints that help enforce half-duplex can also be obtained, like the constraints for 5-node cycles.

Including the Virtual Capacity. The solution of problem (2.7) (or its modified version) predicts the rate for each commodity (r_m), and such rates can be very different across the commodities. Indeed, some commodities may achieve larger transmission rates due to favorable network conditions such as the relative closeness between its source and destination and/or the existence of relative high-capacity routes. In contrast, some other commodities can achieve very low transmission rates due to unfavorable network conditions. Then the following phenomenon can happen: some "preferred" paths and "preferred" intermediate nodes will be always fully loaded, and some commodities have lower achieved rates with higher transmission delay. To promote fair accessibility of network conditions among different commodities, we introduce a *virtual capacity* $\tilde{z}_i^m$ for each buffer (m, i). We modify (2.5) by replacing $\bar{z}$ by the corresponding virtual capacity $\tilde{z}_j^m$, then obtain

$$f_{ij}^m \le x_{ij}\tilde{z}_j^m,\ \forall\, i \ne S(m), j \ne D(m),\ \forall\, m, \tag{3.2}$$

which implies that the scheduled flow f_{ij}^m is at most $\tilde{z}_j^m$. By adjusting the virtual capacity $\tilde{z}_j^m$ for buffer (m, j) which is at the end node of l_{ij}, link flow f_{ij}^m can be adjusted through (3.2). When buffer (m, j) is quite full, there is little space for receiving data, $\tilde{z}_j^m$ should be small to discourage flows going into v_j. When buffer (m, j) is close to being empty, there is more storage space for incoming data, $\tilde{z}_j^m$ should be adjusted upward to encourage flows. We propose to adaptively update the virtual capacity according to the buffer status at the end of every long-term period, see Table 1. In the table, the "min" and the "max" operators are taken to ensure $\tilde{z}_i^m \in [0, \bar{z}]$, α and β are predetermined parameters satisfying $0 < \alpha < \beta < 1$, and Δz is the predefined stepsize for updating virtual capacity.

Table 1. Virtual Capacity Updates

if $z_i^{m,1} \le \alpha\bar{z},\ \tilde{z}_i^m = \min\{\bar{z}, \tilde{z}_i^m + \Delta z\}$;
if $z_i^{m,1} \ge \beta\bar{z},\ \tilde{z}_i^m = \max\{0, \tilde{z}_i^m - \Delta z\}$.

The Long-Term TE Formulation. With the newly introduced constraints described above, we propose the following formulation of the long-term TE problem:

$$\begin{array}{ll} \max\limits_{\mathbf{f},\mathbf{x}} & U(r_1, \ldots, r_M) \\ \text{subject to} & (2.1) - (2.4), (2.6), (3.1), (3.2), \\ & f_{ij}^m \ge 0, x_{ij} \in [0, 1],\ \forall\, m,\ \forall\, l_{ij} \in \mathcal{L}. \end{array} \tag{3.3}$$

The initial buffer statuses are given as input to the problem and the buffer statuses will be updated every long-term period. The solution f_{ij}^{m*} is actually the average rate of commodity m over link l_{ij} during the long-term period, and the flow rate in one active short-term period is actually f_{ij}^{m*}/x_{ij}^* when $x_{ij}^* \ne 0$. The number of short-term periods during which l_{ij} is active can be approximated by $\lfloor x_{ij}^*\tau \rfloor$, where $\lfloor \cdot \rfloor$ represents the floor function.

Problem (3.3) is a linear program with $\{f_{ij}^m\}$ and $\{x_{ij}\}$ as variables, so it can be effectively solved by the distributed asynchronous methods introduced in [8].

3.2. The Short-Term Link Scheduler

The TE controller predicts the flow rate for each commodity on each link (i.e., f_{ij}^{m*}), as well as the fraction of time that each link should be active (i.e., x_{ij}^*). It is then the task of the link scheduler to come up with per-time slot link activation and routing schemes. In this section, we propose the short-term link scheduler based on the greedy edge-coloring algorithm.

An edge coloring for a multi-graph is an assignment of colors to the edges such that no two edges incident to the same node are colored the same. The problem of determining the minimum number of colors needed for edge coloring is called the edge-coloring problem. Despite being NP-hard, this problem can be well solved by heuristic greedy edge-coloring algorithms in practice. For example, the simple *first fit greedy* edge-coloring algorithm colors the nodes in sequence using the first available color, and is known to provide a 2-approximate solution, i.e., the number of colors used by the algorithm is at most 2Δ where Δ is the maximum degree of the nodes in the graph [9] (which is a lower bound of the minimum color).

To formulate our link activation problem into the edge-coloring problem, we first construct a multi-graph that characterizes the overall link activation frequency. Let the multi-graph be $\mathcal{G}_e = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the same node set as in the network $\mathcal{G}$, $\mathcal{E}$ is the set of directed edges. Denote the number of directed edges from v_i to v_j by n_{ij}. In our scheduling context, the edges with the same color can be activated in the same time slot, therefore the total number of colors used should not exceed the number of short-term periods τ. We use the *first fit greedy algorithm* for scheduling. Since this algorithm is 2-approximate, in our scheduling context this means that an upper bound of the required number of colors is 2Δ, that is, $2\Delta \le \tau$. Further, the number of active short-term periods for l_{ij} is approximated by $\lfloor x_{ij}^*\tau \rfloor$, which implies n_{ij} should be proportional to the time fraction x_{ij}^*. Notice that due to HDC, at any time for any given network node at most one link incident on the node can be active, and one link $l_{ij} \in \mathcal{L}$ results in n_{ij} edges in $\mathcal{E}$. Thus at any time, only a subgraph of $\mathcal{G}_e$ is activated (available). The effective maximum degree of the nodes in $\mathcal{G}_e$ can be counted by $\Delta = \max n_{ij}$. Based on the discussion above, we set the value of edge numbers as

$$n_{ij} = \left\lfloor \frac{\tau}{2} x_{ij}^* \right\rfloor,\ \forall\, l_{ij} \in \mathcal{L}. \tag{3.4}$$

Such settings satisfy $2\Delta \le \tau$ and n_{ij} is roughly proportional to x_{ij}^*.

Given the multi-graph $\mathcal{G}_e$, we then need to color the edges. It is well-known that the scheduling results generated by the greedy edge-coloring algorithm are heavily dependent on the order in which we visit and color the nodes [10]. If the visiting order is poorly chosen, some activation opportunities may be wasted. Notice that when l_{ij} is activated in one short-term period, the transmission data size for commodity m from v_i to v_j is $\min\{f_{ij}^{m*}/x_{ij}^*, z_i^m, \bar{z} - z_j^m\}$. If z_i^m is small or z_j^m is large, it is possible that only a small amount of data will be transmitted despite a large scheduled rate f_{ij}^{m*}/x_{ij}^*.

Moreover, activation opportunities may be wasted due to the possible successive activation of the same link, because the earlier activation may have filled (or exhausted) the receive (or transmit) buffers of the connecting nodes. In the following, we propose an edge-coloring algorithm which includes buffer status in link scheduling.

Assume that none of the M commodities has one-hop path. In this case the maximum data size that can be transmitted by node v_i in one short-term period is given by $B_i = \sum_{m=1}^{M} b_i^m$, where

$$b_i^m = \begin{cases} \min\{\bar{z}, z_i^m\}, & \text{if } i = S(m), \\ z_i^m, & \text{otherwise.} \end{cases} \tag{3.5}$$

We can easily see that B_i represents the maximum transmission data size from v_i. If v_i is not a source node, then B_i is exactly the total size of data in v_i. To realize the link flows predicted by the TE controller, the link activation should give priority to the nodes who can contribute more to the network throughput. Thus we propose to visit the nodes in the decreasing order of B_i in each short-term period, so that the nodes that have relatively large potential transmission data size can have higher priority. When v_i is visited, we use the following strategy to pick an incident edge to color:

S0. While satisfying the interference constraints, and if there are incident edges corresponding to outgoing links, we choose the one whose destination node has the smallest B_j; otherwise choose the edge whose source node has the largest B_j.

Our strategy of coloring the edges (scheduling links) is summarized in Table 2. Further, this algorithm can be easily implemented in a distributed way. We omit the details due to space limitation.

Table 2. Short-Term Link Scheduler (Centralized Implementation).

Construct $\mathcal{G}_e$: choose edge numbers according to (3.4);
For short-term period $t = 1 : \tau$
  If $\mathcal{G}_e = \emptyset$, stop; Else,
    Calculate B_i according to (3.5);
    Visit the nodes in the decreasing order of B_i, choose an incident edge to color according to **S0** for each visited node;
    Active the corresponding links of the colored edges;
    Update buffer status, update $\mathcal{G}_e$ by deleting the colored edges;
End
If $t < \tau$, repeat the previous schedule in the remaining periods.

3.3. The Overall Algorithm

Our joint long-term TE and short-term link scheduling algorithm is presented in Table 3. Notice that long-term TE and short-term scheduling can both be solved distributedly, the computation complexity is much lower than the existing short-term based approaches.

Table 3. The Proposed Two Time-Scale Approach

For $n = 1 : T$
  Solve TE problem (3.3) for the nth long-term period;
  Schedule link flows by the link scheduler in Table 2;
  Update virtual capacity according to Table 1;
End

4. NUMERICAL EXPERIMENTS

In this section, we present numerical results on the performance of the proposed algorithm. We consider a wireless backhaul network with 25 network nodes, which are split into 2 categories - the macro base station (BS) and the micro BS. The macro (reps. micro) BS has a maximum transmit power of 20dBm (resp. 10dBm). The topology of this wireless network is shown in Fig. 2. We assume that each BS can interact with BSs within 360 meters, the BSs use their full power to transmit, and the background noise is assumed to be -20dBm. The source and the destination nodes of each data flow are randomly selected among all the BSs and the path has at least 2 hops. We take

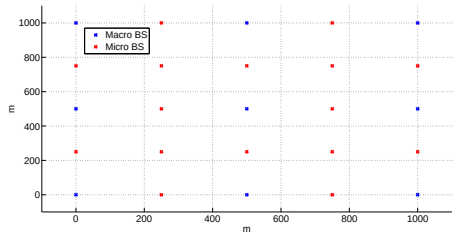

Fig. 2. The considered network topology with 25 wireless nodes.

$M = 30$, $\tau = 20$, $\bar{z} = 5$, $\alpha = 0.25$, $\beta = 0.75$, $\Delta z = 0.1\bar{z}$, $T = 100$ and set $U = \sum_m r_m$. We define the average achieved rate in the first T long-term period as $\frac{1}{T}\sum_{t=1}^{T} R_t$, where R_t is the sum of the achieved rate of all commodities in the t-th long-term period. We also compare with the simple greedy edge-coloring based link scheduler which uses the first-fit edge-coloring algorithm to color the edges in the multi-graph $\mathcal{G}_e$ [9]. For convenience, we shall refer the proposed two time-scale approach as "2TS-approach".

We inject x_m bits of data for commodity m every 20 long-term periods, where x_m is uniformly randomly chosen from (0, 240). From Fig. 3 we can see that the realized flows tend to be stable as time index increases, and most (over 94%) of the injected data are well transmitted. The proposed link scheduler can improve the network throughput by 30% over the greedy edge-coloring based scheduler. Moreover, excluding either triangular constraints (Δ-constraints) or virtual capacity (vc) will decrease the overall network throughput. Specifically, the throughput decreases by 12% without involving virtual capacity and triangular constraints.

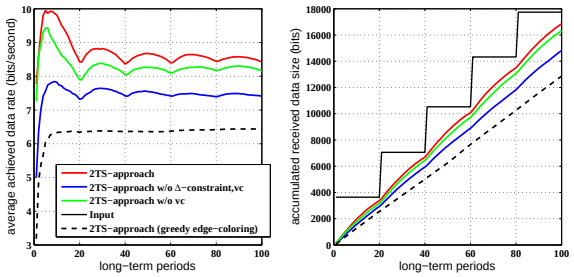

Fig. 3. Comparison of the system throughput performance: (1) 2TS-approach (2) 2TS-approach w/o vc (3) 2TS-approach w/o Δ-constraint nor vc. Left: the average achieved data rate; Right: the accumulated received data size.

5. REFERENCES

[1] Q. Ye, B. Rong, Y. Chen, M. Al-Shalash, C. Constantine, and J. G. Andrews. User association for load balancing in heterogeneous cellular networks. *IEEE Trans. Wireless Communications*, vol. 12, no. 6, pp. 2706–2716, Jun. 2013.

[2] H. Boostanimehr and V. Bhargava. Unified and distributed qos-driven cell association algorithms in heterogeneous networks. *IEEE Trans. Wireless Communications*, vol. 14, no. 3, pp. 1650–1662, Mar. 2015.

[3] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle. Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks. *Proceedings IEEE INFOCOM 2006. 25th IEEE International Conference on Computer Communications*, pp. 676–688, Apr. 2006.

[4] R. L. Cruz and A. V. Santhanam, "Optimal routing, link scheduling and power control in multihop wireless networks," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*. vol. 1, pp. 702–711, 2003.

[5] H. Su and X. Zhang, "Joint link scheduling and routing for directional-antenna based 60 ghz wireless mesh networks," in *Global Telecommunications Conference, 2009. GLOBECOM2009. IEEE*, pp. 1–6.

[6] M. Kodialam and T. Nandagopal. Characterizing achievable rates in multi-hop wireless mesh networks with orthogonal channels. *IEEE/ACM Transactions on Networking (TON)*, vol. 13, no. 4, pp. 868–880, Aug. 2005.

[7] B. Awerbuch and T. Leighton, "A simple local-control approximation algorithm for multicommodity flow," in *Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on. IEEE*, pp. 459–468, Nov. 1993.

[8] W. C. Liao, M. Hong, H. Farmanbar, and Z. Q. Luo, "Semi-asynchronous routing for large scale hierarchical networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on. IEEE*, pp. 2894–2898, Apr. 2015.

[9] W. Klotz. Graph coloring algorithms. *Mathematics Report*, vol. 5, no. 2002, pp. 1–9, 2002.

[10] L. Kučera. The greedy coloring is a bad probabilistic algorithm. *Journal of Algorithms*, vol. 12, no. 4, pp. 674–684, Dec. 1991.