Demand-Aware Network Design with Minimal Congestion and Route Lengths

Chen Avin, Kaushik Mondal, and Stefan Schmid

***Abstract*—Emerging communication technologies allow to reconfigure the physical network topology at runtime, enabling *demand-aware networks (DANs)*: networks whose topology is optimized toward the workload they serve. However, today, only little is known about the fundamental algorithmic problems underlying the design of such demand-aware networks. This paper presents the first bounded-degree, demand-aware network, *cl-DAN*, which minimizes *both* congestion and route lengths. The degree bound Δ is given as part of the input. The designed network is provably (asymptotically) optimal in each dimension individually: we show that there do not exist any bounded-degree networks providing shorter routes (independently of the load), nor do there exist networks providing lower loads (independently of the route lengths). The main building block of the designed *cl-DAN* networks are *ego-trees*: communication sources arrange their communication partners in an optimal tree, *individually*. While the union of these ego-trees forms the basic structure of *cl-DANs*, further techniques are presented to ensure bounded degrees (for scalability).**

***Index Terms*—Reconfigurable networks, Network topology, Network design, Approximation algorithms, Load, Route length**

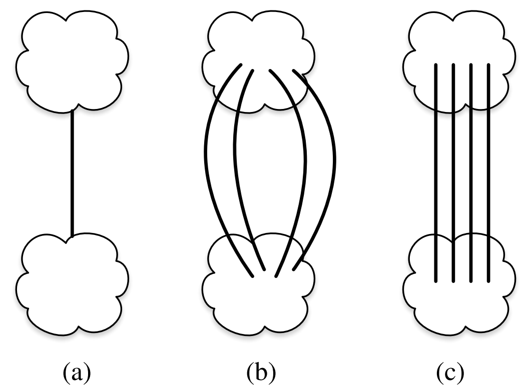


Fig. 1. Challenge of designing demand-aware networks: (a) Optimizing for route lengths only may result in bottlenecks and high loads. (b) Optimizing for congestion only, by distributing load across multiple paths, can result in long routes. (c) Ideally, we aim to design networks that minimize both congestion and route lengths, *using a small number of links* (constant degree).

I. Introduction

A. Motivation

Data center networks have become a critical infrastructure of our digital society. With the trend toward more data-intensive applications, data center network traffic is growing quickly [7], [31]. As much of this traffic is *internal* to the data center (e.g., traffic due to scatter-gather and batch computing applications), the design of efficient data center networks has received much attention over the last years [23].

Traditionally, data center designs are *demand-oblivious* and static: they are optimized for the "worst-case", e.g., they (almost) provide a full bisection bandwidth, allowing to serve dense, all-to-all communication patterns. Empirical studies however show that real communication patterns are usually far from all-to-all. Rather, traffic patterns feature spatial locality and are sparse [5], [9], [13], [21], [26]: only a small fraction of all possible source-destination pairs are involved in intensive communications at any time.

The advent of novel optical technologies which allow to reconfigure the physical network topology [10], [15], [20], [21], heralds a paradigm shift: using these technologies, data center designs can be reconfigured and optimized toward their demand, i.e., they become *demand-aware*. In particular, a demand-aware network design may connect frequently communicating nodes "better": the network provides shorter routes between such nodes (lower latency, energy consumption etc.) and aims to reduce congestion by keeping traffic local (lower load, less queuing delays, etc.).

However, only little is known today about the *algorithmic* challenge of designing demand-aware networks which provide low congestion ***and*** short routes (in the number of hops), for a given communication pattern. This is the topic of our paper (see also Figure 1).

At first sight, it may seem that designing networks providing both short routes and minimal load is hard and faces a tradeoff: to better balance loads, it may be necessary to route flows along longer paths. Yet, as we show in this paper, a solution can be efficiently computed which is almost optimal both in terms of route length and congestion, *independently* (i.e., without tradeoff).

B. The Demand-Aware Network Design Problem

Intuitively, the demand-aware network design problem can be stated as follows (a formal model will follow later). We are given a set of n nodes (e.g., top-of-rack switches [21]) interacting according to a certain *communication pattern*: a frequency distribution represented as matrix or (weighted) *demand graph*.

Our goal is to design a demand-aware network, *cl-DAN*, together with a routing scheme, which serves this communication pattern providing low congestion and short route lengths. The designed network should be scalable, i.e., of bounded degree: e.g., reconfigurable links may consume space

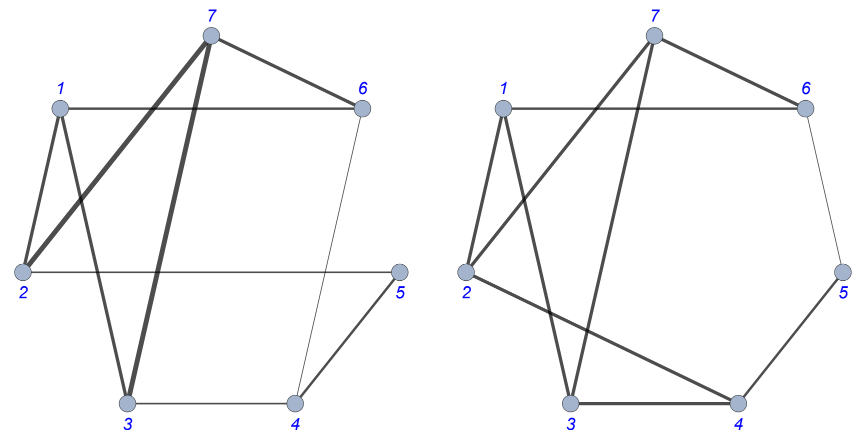
Chen Avin is with the School of Electrical and Computer Engineering, Ben Gurion University of the Negev, Israel, e-mail: avin@bgu.ac.il.

Kaushik Mondal is with Indian Institute of Technology Ropar, India, e-mail: kaushik.mondal@iitrpr.ac.in

Stefan Schmid is with TU Berlin, Germany, and University of Vienna, Austria, e-mail: stefan.schmid@tu-berlin.de.

	1	2	3	4	5	6	7
1	0	$\frac{3}{60}$	$\frac{4}{60}$	$\frac{1}{60}$	$\frac{1}{60}$	$\frac{1}{60}$	$\frac{1}{60}$
2	$\frac{3}{60}$	0	$\frac{2}{60}$	0	$\frac{1}{60}$	0	$\frac{4}{60}$
3	$\frac{4}{60}$	$\frac{2}{60}$	0	$\frac{2}{60}$	0	0	$\frac{4}{60}$
4	$\frac{1}{60}$	0	$\frac{2}{60}$	0	$\frac{3}{60}$	0	0
5	$\frac{1}{60}$	$\frac{1}{60}$	0	$\frac{3}{60}$	0	0	0
6	$\frac{1}{60}$	0	0	0	0	0	$\frac{3}{60}$
7	$\frac{1}{60}$	$\frac{4}{60}$	$\frac{4}{60}$	0	0	$\frac{3}{60}$	0

(a) Demand distribution



(b) Optimal route length (c) Optimal route congestion

Fig. 2. Example of the *demand-aware network design* problem for a network composed of seven nodes. (a) A given demand distribution $\mathcal{D}$ which describes the probability $p(i,j)$ of a source i to communicate with a destination j. In this case, the distribution is *symmetric*. (b) A demand-aware network, N, of degree $\Delta = 3$ which is optimal in terms of route lengths using a shortest paths routing scheme Γ_{sp}. The expected route length is optimal, $\mathrm{L}(\mathcal{D},\Gamma_{sp}(N)) = \mathrm{L}^*(\mathcal{D},\Delta) = \frac{35}{30}$ while the *link congestion* is not optimal $\mathrm{C}(\mathcal{D},\Gamma_{sp}(N)) = \frac{12}{60}$. (c) A solution N' which is optimal with respect to *link congestion* $\mathrm{C}(\mathcal{D},\Gamma_{sp}(N')) = \mathrm{C}^*(\mathcal{D},\Delta) = \frac{8}{60}$ but not optimal with respect to expected route length, $\mathrm{L}(\mathcal{D},\Gamma_{sp}(N')) = \frac{36}{30}$. The edge thickness represents the level of congestion.

and/or resources, and their number is limited [21]. To this end, the inputs to our problem include a degree bound Δ. *cl-DANs* provide particularly short routes between frequently communicating partners while accounting for the possible load. Throughout the paper we focus on the more challenging scenario of *unsplittable flows*: traffic for a given source-destination pair is routed along a single path. This scenario is also important in practice: unsplittable flows typically minimize issues with packet reorderings (which may for example harm TCP performance) and require less routing rules.

We are interested in *polynomial-time* algorithms to design *cl-DANs*, and we say that a *cl-DAN* achieves an *(α,β)-approximation* if: (1) the maximum congestion on any link is at most an α factor higher than the lowest possible (even when compared to a network design which can have arbitrary route lengths); (2) the weighted-average route length (henceforth also called the *expected route length*) is at most a factor β higher than optimal (even when compared to a network design which can have arbitrary loads).

C. Our Contributions

We initiate the study of the fundamental problem of demand-aware network designs which minimize both congestion and route length of unsplittable flows. The main result of this paper is a polynomial-time construction of a bounded-degree demand-aware network (together with a routing scheme) which provides a constant approximation of both the minimal congestion and minimal route lengths, for sparse demands (as they usually occur in practice): an *($O(1)$,$O(1)$)-approximate cl-DAN*. Our algorithm relies on an interesting algorithmic technique which connects the network design problem (where nodes communicate in pairs) to tree datastructures (where requests originate at the root). In particular, our construction is based on per-source optimal trees (henceforth called *ego-trees*) which are then combined in a manner which preserve bounded degrees. We believe that this technique is of independent interest and relevant for other (static and dynamic) network design problems. Furthermore, our paper leverages an interesting connection to information theory: while the diameter of demand-*oblivious* networks of bounded degree is inherently lower bounded by $\Omega(\log n)$, we are able to "encode" specific routing patterns in network topologies which match the entropy lower bounds of the demand: entropy is a natural measure to study what can and cannot be achieved by a demand-aware network.

D. Paper Organization

The remainder of this paper is organized as follows. In Section II, we introduce our formal model. Section III presents an optimal network design and routing scheme for a single source, which is generalized in Section IV to arbitrary but sparse communication patterns. We give a detailed numerical example in Section V. After reviewing related work in Section VI, we conclude in Section VII. Some details are deferred to the appendix for better readability.

II. Model and Problem Definition

This paper considers the following fundamental demand-aware network design problem. The input is a set of n nodes $V = \{1,\ldots,n\}$ which communicate according to a given *communication pattern*, modeled as a discrete distribution $\mathcal{D}$ over $V \times V$: we represent the distribution of communication requests using a communication matrix $M_{\mathcal{D}}[\mathrm{p}(i,j)]_{n\times n}$ where the (i,j) entry indicates the communication frequency, $\mathrm{p}(i,j)$, from the (communication) source i to the (communication) destination j. The matrix is normalized, i.e., $\sum_{ij}\mathrm{p}(i,j) = 1$: we will hence interpret the matrix as a *probability distribution*. Furthermore, we will use $\mathrm{p}(i)$ to denote the total probability at which i serves as a source, i.e., $\mathrm{p}(i) = \sum_j \mathrm{p}(i,j)$. Similarly, $\mathrm{q}(i)$ denotes the frequency at which i is a destination.

We can also interpret the distribution $\mathcal{D}$ as a weighted directed *demand graph* $G_\mathcal{D}$, defined over the same set of nodes V: A directed edge $(u, v) \in E(G_\mathcal{D})$ exists iff $\mathrm{p}(u, v) > 0$. The edge weight is simply the communication frequency: $w(i, j) = \mathrm{p}(i, j)$. Throughout this paper, we are interested in the practically relevant case [21] where $M_\mathcal{D}$ and $G_\mathcal{D}$ are *sparse*, i.e., $G_\mathcal{D}$ has a linear number of communication edges (and $M_\mathcal{D}$ has a linear number of non zero entries).

Our goal is to design demand-aware networks N which provide both low congestion and short route lengths, henceforth called *cl-DANs*. We define both the routing lengths and congestion using a *routing scheme* (a.k.a. 'canonical paths' [17]) for a network N. A *routing scheme* for a network N is a set $\Gamma(N)$ of simple paths Γ_{uv}, one between each pair (u, v) of distinct vertices. In particular, we consider *unsplittable flows* and each Γ_{uv} is a sequence of edges connecting u to v. If a demand $\mathcal{D}$ is given, we can now define congestion and route lengths formally where for each edge $e \in \Gamma_{uv}$, Γ_{uv} contributes $\mathrm{p}(u, v)$ to the load of e, and the length of a route Γ_{uv} is defined as $\mathrm{d}_{\Gamma(N)}(u, v)$. Congestion is defined by the most loaded edge in $\Gamma(N)$:

Definition 1 (Congestion C*):* The congestion for a routing scheme $\Gamma(N)$ and a demand distribution $\mathcal{D}$ is defined as:

$$\mathrm{C}(\mathcal{D}, \Gamma(N)) = \max_{e \in \Gamma(N)} \sum_{e \in \Gamma_{uv}} \mathrm{p}(u, v)$$

The route length is defined as the *weighted-average route length* for $\Gamma(N)$:

Definition 2 (Route Length L*):* The weighted average route length for a routing scheme $\Gamma(N)$ and a demand distribution $\mathcal{D}$ is defined as:

$$\mathrm{L}(\mathcal{D}, \Gamma(N)) = \sum_{(u,v) \in \mathcal{D}} \mathrm{p}(u, v) \cdot \mathrm{d}_{\Gamma(N)}(u, v)$$

Furthermore, we require the designed *cl-DAN* networks to be scalable, i.e., of bounded (constant) degree Δ. We denote by $\mathcal{N}_\Delta$, the family of all Δ-bounded degree graphs and formally we require that $N \in \mathcal{N}_\Delta$.

We define *optimal congestion*, with respect to a design that is optimized toward congestion only; similarly, we define *optimal route lengths*, with respect to designs that are optimized toward route length only. Formally, for a given demand distribution $\mathcal{D}$ and a degree bound Δ, denote

$$\mathrm{C}^*(\mathcal{D}, \Delta) = \min_{N \in \mathcal{N}_\Delta, \Gamma(N)} \mathrm{C}(\mathcal{D}, \Gamma(N))$$

as the optimal congestion and

$$\mathrm{L}^*(\mathcal{D}, \Delta) = \min_{N \in \mathcal{N}_\Delta, \Gamma(N)} \mathrm{L}(\mathcal{D}, \Gamma(N))$$

as the optimal route length.

We can now state our optimization problem: the design of a network which minimizes both congestion and route lengths.

Definition 3 ((α, β) cl-DAN *Network Design):* Given a communication distribution, $\mathcal{D}$ and a maximum degree Δ, the (α, β)-*cl-DAN* network design problem is to design a network $N \in \mathcal{N}_\Delta$ and a routing scheme $\Gamma(N)$ such that both congestion and route lengths are bounded compared to the optimal:

$$\mathrm{C}(\mathcal{D}, \Gamma(N)) \leq \alpha \cdot \mathrm{C}^*(\mathcal{D}, \Delta) + \alpha'$$

Algorithm 1: EGOTREE$(s, \bar{\mathrm{p}}, \Delta)$

connect the source s to the root of Δ (empty) binary trees $T_1, T_2, \ldots, T_\Delta$
sort $\bar{\mathrm{p}}$ from large to small
add, one by one (in decreasing order according to their probability mass), the destinations to the currently minimal tree T_i: in an unoccupied node as close as possible to the root of T_i

and,

$$\mathrm{L}(\mathcal{D}, \Gamma(N)) \leq \beta \cdot \mathrm{L}^*(\mathcal{D}, \Delta) + \beta'$$

where α' and β' are constants independent of the problem parameters.

We emphasize that we aim to be optimal along each dimension (congestion and route lengths) even compared to a network which is optimized only along one dimension (and has slack in the other dimension). We also note that a $(1, 1)$-*cl-DAN* does not always exist. A more detailed example of our model and the challenge of minimizing both congestion and route length is given in Figure 2: for a given demand distribution, a network of optimal congestion may look different from a network with optimal route lengths. The main contribution of our paper is a $(O(1), O(1))$-*cl-DAN*.

III. THE EGO-TREE NETWORK

A fundamental building block of the *cl-DAN* network presented in this paper is the *ego-tree*: a congestion and route length optimized tree network for a *single node* (i.e., single source). The name ego-tree stems from the term ego-networks in social networks [22]: it describes the network of a user and her friends. Accordingly, in the following, we will first study the single-source multi-destination variant of our problem. In particular, we will present an algorithm EGOTREE$(s, \bar{\mathrm{p}}, \Delta)$ which, given a source s, a probability distribution $\bar{\mathrm{p}}$ across its neighbors and a bound Δ for the maximum degree, computes a demand-aware tree network of maximum degree Δ, with near optimal congestion and route length. More formally, the main result for this section is:

Theorem 1: Given a frequency distribution $\bar{\mathrm{p}}$ for a source s over its destinations, and a degree bound Δ, EGOTREE$(s, \bar{\mathrm{p}}, \Delta)$ is a (α, β) *cl-DAN* with $\alpha = 4/3$ and $\beta = \log^2(\Delta + 1)$.

That is, for a constant Δ, EGOTREE$(s, \bar{\mathrm{p}}, \Delta)$ achieves a constant approximation both in terms of the optimal congestion and the optimal average route length.

A. EGOTREE$(s, \bar{\mathrm{p}}, \Delta)$ *Algorithm*

Our algorithm to design an *ego-tree*, T, is a simple greedy algorithm (see pseudocode in Algorithm 1). Node s is the source and the root of T, its degree is Δ and it is connected to Δ *binary* trees [1] $T_1, T_2, \ldots, T_\Delta$ which will be defined shortly.

[1]Instead of binary trees, if one built the subtrees of Algorithm 1 as Δ-array trees, then the degree will increase to $O(\Delta^2)$ in *cl-DAN* while the congestion remains the same and the path length can be improved by a constant fraction only.

We note that $\textsc{EgoTree}(s, \bar{\mathrm{p}}, \Delta)$ could be further optimized for better α and β by using Δ-array trees rather than binary trees, as we do. However, as we will see, limiting ourselves to binary trees here is crucial to keep the degree low later, when we combine multiple trees to design general *cl-DANs*.

We sort the probabilities in $\bar{p} = \{\mathrm{p}_1, \mathrm{p}_2, \ldots, \mathrm{p}_k\}$ from large to small, and create T by placing destinations in T according to the order in the sorted list of probabilities. We choose and add the next destination v to the tree T_i which currently has the minimum probability mass (breaking ties arbitrarily); we place v in an unoccupied node as close as possible to the root of T_i (recall that T_i is a binary tree). We note that the resulting tree T may not be balanced since the sizes of the binary trees may be different. Let $\Pi_1, \Pi_2, \ldots, \Pi_\Delta$ denote the partition of $\bar{p}$ according to the binary subtrees and let S_i be the total probability mass of Π_i, i.e., $S_i = \sum_{p_j \in \Pi_i} p_j$. We will later show that this process creates a nearly balanced partition, i.e., the S_i are of similar values. See Figure 3 for a numerical example of the algorithm and the resulting tree.

Communication from s to any of its communication partners is routed along s's *ego-tree* (using a routing algorithm which runs in the background).

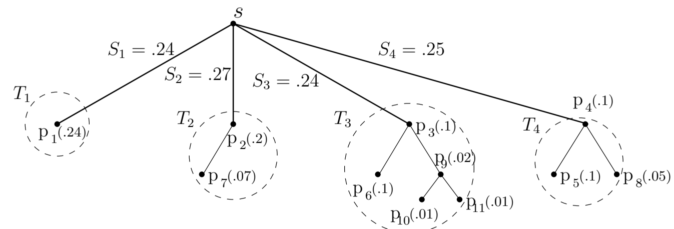


Fig. 3. $\textsc{EgoTree}(s, \{.24, .2, .1, .1, .1, .1, .07, .05, .02, .01, .01\}, 4)$: Source s is connected to 4 binary trees with cumulative frequency .24,.27, .24, .25, respectively.

B. Analysis

1) Analysis of Route Congestion: Since we design a tree network with a single source s, the most congested edge is clearly among the edges connected to the root s. Let e_i denote the edge that connects the root s to the tree T_i. The congestion in e_i is equal to S_i (the total probabilty mass of T_i) and therefore the congestion in T is $\max_i S_i$.

Minimizing $\max_i S_i$ is essentially a makespan scheduling problem where the goal is to assign jobs to Δ processors such that all the jobs can be completed as early as possible. While computing the optimal solution is NP-hard [30], we use a simple approximation method in Algorithm 1, which is known as the Longest Processing Time (LPT) [12] algorithm. LPT solves this problem by assigning the longest remaining unexecuted task to one of the free processors. It first sorts the jobs in decreasing order and then considers the jobs one at a time, placing them into the least working processors. The following well-known theorem from [12] gives an upper bound compared to the optimal solution:

Theorem 2 ([12], restated.): Let w_L be the maximum time a processor runs before completing all jobs, according to the greedy algorithm LPT, and let w_0 be the optimal processing time. Then,

$$\frac{w_L}{w_0} \leq \frac{4}{3} - \frac{1}{3\Delta}$$

The theorem can be easily extended from integers to rational numbers, as in our case, and we can claim the following (see Appendix for details):

Lemma 1: $\textsc{EgoTree}(s, \bar{\mathrm{p}}, \Delta)$ provides a 4/3 approximation of the minimum congestion w.r.t. an optimal Δ-ary tree which needs to serve a frequency distribution $\bar{\mathrm{p}}$ for a single source s.

2) Analysis of Route Length: For a set of probabilities $\bar{\mathrm{p}}$ and a tree T, where each node in T corresponds to a single element in T, let $\mathrm{L}(\bar{\mathrm{p}}, T)$ denote the average route length from nodes in the tree to the root (which has distance 0 to itself). For a distribution $\bar{\mathrm{p}}$, let $H(\bar{\mathrm{p}})$ be the binary entropy of $\bar{\mathrm{p}}$ and $H_\Delta(\bar{\mathrm{p}})$ be the entropy calculated using the logarithm of base Δ. We will prove the following.

Lemma 2: $\textsc{EgoTree}(s, \bar{\mathrm{p}}, \Delta)$ achieves a $\log^2(\Delta + 1)$ approximation on the minimum route length w.r.t. an optimal Δ-ary tree which needs to serve a frequency distribution $\bar{\mathrm{p}}$ for a single source s.

Proof: Recall that $\textsc{EgoTree}(s, \bar{\mathrm{p}}, \Delta)$ is based on binary trees T_i, each serving a set of probabilities $\Pi_i \subset \bar{\mathrm{p}}$. Let Π'_i be the normalized version of Π_i, i.e., Π'_i sums to 1: a probability distribution. We first bound the average route length to nodes in T_i.

Claim 1: $\mathrm{L}(\Pi_i, T_i) \leq S_i H(\Pi'_i)$

Proof:[of Claim 1] The distances of nodes in T_i from the root are according to their probabilities, where the root has the largest probability. Let q_k denote the probability of the node which has rank k, in the order of Π_i (i.e., q_k is the kth largest entity in Π_i and ties broken arbitrarily, the root has rank 1), and let q'_k the corresponding probability in Π'_i.

By definition $q'_k = q_k / S_i$. Clearly $q'_k \leq 1/k$, otherwise we get a contradiction that Π'_i is normalized. So $k \leq 1/q'_k$. The distance of a node with rank k from the root is exactly $\lfloor \log k \rfloor$. We can write $\mathrm{L}(\Pi_i, T_i)$ as follows:

$$\begin{aligned} \mathrm{L}(\Pi_i, T_i) &= \sum q_k \lfloor \log k \rfloor \leq \sum S_i q'_k \lfloor \log 1/q'_k \rfloor \\ &\leq \sum S_i q'_k \log 1/q'_k = S_i H(\Pi'_i) \end{aligned}$$

□

Let T_s denote the tree resulting from $\textsc{EgoTree}(s, \bar{\mathrm{p}}, \Delta)$. We can now bound $\mathrm{L}(\bar{\mathrm{p}}, T_s)$.

$$\begin{aligned} \mathrm{L}(\bar{p}, T_s) &= 1 + \sum_{i=1}^{\Delta} \mathrm{L}(\Pi_i, T_i) \leq 1 + \sum_{i=1}^{\Delta} S_i H(\Pi'_i) \\ &= 1 + H(\bar{\mathrm{p}}) - H(S_1, S_2, \ldots, S_\Delta) \leq H(\bar{\mathrm{p}}) \end{aligned} \quad (1)$$

The third step follows from the entropy grouping property and we note that the inequality of the last step may not always hold, however, we keep it for simplicity of presentation and since for $\Delta > 2$, large n and our partition method, $H(S_1, S_2, \ldots, S_\Delta) \geq 1$, holds.

We turn to the lower bound. Denote the optimal Δ-ary tree by T^*_Δ, a lower bound for $\mathrm{L}(\bar{\mathrm{p}}, T^*_\Delta)$ is given in [3]:

Lemma 3: Let T^*_Δ be an optimal Δ-ary tree built for the frequency distribution $\bar{\mathrm{p}}$. Then,

$$\mathrm{L}(\bar{\mathrm{p}}, T^*_\Delta) \geq \frac{1}{\log(\Delta+1)} H_\Delta(\bar{\mathrm{p}})$$

Using $H_\Delta(\bar{\mathrm{p}}) = (1/\log\Delta)\, H(\bar{\mathrm{p}})$, and combining Equation 1 and Lemma 3, we have:

$$\frac{H(\bar{\mathrm{p}})}{\log^2(\Delta+1)} \leq \mathrm{L}(\bar{\mathrm{p}}, T^*_\Delta) \leq \mathrm{L}(\bar{\mathrm{p}}, T_s) \leq H(\bar{\mathrm{p}}) \tag{2}$$

which concludes the proof of Lemma 2. □

3) Summary: Theorem 1 now directly follows from Lemma 1 and Lemma 2.

IV. Network Design for Sparse Distributions

We now describe a congestion and route-length optimized demand-aware network *cl-DAN* N for sparse demand distributions. Our construction will build upon the *ego-tree* technique above. We first present our algorithm and subsequently analyze it. For simplicity, we assume that the given matrix $\mathcal{D}$ is symmetric. Later we will show similar results for an arbitrary sparse matrix. In particular, we will derive the following main theorem:

Theorem 3: Let $\mathcal{D}$ be a symmetric communication request distribution where ρ is the average degree in $G_\mathcal{D}$ (so the number of edges is $\rho \cdot n/2$). Then, for a maximum degree $\Delta = 12\rho$. it is possible to generate an (α, β)-*cl-DAN* with $\alpha = 1 + (8/9)\Delta$ and $\beta = 1 + 4\log^2(\Delta+1)$. For constant ρ this provides a constant approximation for both the minimal congestion and optimal route length.

Since $\mathcal{D}$ is symmetric, we can view $G = G_\mathcal{D}$ as a weighted but undirected graph. We denote the normalized row (which is identical to the column) corresponding to any node u by $\overline{\mathcal{D}[u]}$. As mentioned earlier our main design method for *cl-DAN* relies on EgoTrees. We divide nodes in *high degree* and *low degree* nodes. For each *high degree* node $v \in V$, we construct its optimized tree based on $\mathrm{EgoTree}(v, \overline{\mathcal{D}[v]})$ and later take a union of them. Before taking union, we do some modifications on those trees which help to maintain the degree bound. In addition we keep all the edges between low degree nodes. A detailed explanation is given next followed by an analysis.

A. (α, β)-cl-DAN *Algorithm*

Recall that the total number of edges in G is $n\rho/2$ and we assume that the average degree ρ, is constant so $\mathcal{D}$ is sparse. Denote the nodes with the degree less or equal to 2ρ in G as *low degree* nodes and the rest as *high degree* nodes. Let $\mathcal{L}$ and $\mathcal{H}$ be the set of high degree and low degree nodes respectively such that $\mathcal{H} \cup \mathcal{L}$ include all the nodes. Note that each low degree node has a degree at most 2ρ. The construction of N will be done in two phases. In the first phase, we consider edges (u, v) between the high degree nodes u and v. We subdivide each such edge with two edges that connect u to v via a helping low degree node $\ell \in \mathcal{L}$, i.e., removing the undirected edge (u, v) and adding the edges (u, ℓ) and (v, ℓ). Note that there are at most $n\rho/2$ such edges, so we can distribute the help of low degree nodes in such a way that each low degree node helps at most ρ such edges. We keep a restriction on choosing ℓ. We use different ℓ for each different high degree neighbor of a high degree node u. This is feasible since a high degree node u can have at most $n/2 - 1$ high degree neighbors. Call the resulting graph G'.

Algorithm 2: Building *cl-DAN*

divide set of nodes to two subsets; $\mathcal{H}$ of high degree ($n/2$ nodes with highest degree) and $\mathcal{L}$ of low degree (remaining $n/2$ nodes)
find all the edges which are between two high degree nodes from $\mathcal{H}$
assign each edge (u, v) between high degree and helper node ℓ which is a low degree node from $\mathcal{L}$
construct *ego-trees* T_u according to Algorithm 1 for each high degree node u with the corresponding normalized rows $\overline{\mathcal{D}[u]}$ as the input distribution and $\Delta = 12\rho$
modify T_u to T'_u using the helper nodes (see text)
union all *ego-trees* T'_u and with the edges between low degree nodes

---

Accordingly, we also create a new (also symmetric) matrix $\mathcal{D}'$, which initially, is identical to $\mathcal{D}$, but we then change some entries according to G'. For every low degree node ℓ that helps an edge (u, v) we modify the corresponding entries in $\mathcal{D}'$:

$$\begin{aligned} \mathrm{p}'(u, v) &= \mathrm{p}'(v, u) = 0 \\ \mathrm{p}'(u, \ell) &= \mathrm{p}'(\ell, u) = \mathrm{p}(u, \ell) + \mathrm{p}(u, v) \\ \mathrm{p}'(\ell, v) &= \mathrm{p}'(v, \ell) = \mathrm{p}(\ell, v) + \mathrm{p}(u, v) \end{aligned} \tag{3}$$

In the second phase, we construct N from G'. We start with G by considering each node $u \in \mathcal{H}$ with high degree and create a tree T_u according to $\overline{\mathcal{D}[u]}$ using the method of Theorem 1 and with $\Delta = 12\rho$ as degree of the root, i.e., we generate $\mathrm{EgoTree}(u, \overline{\mathcal{D}[u]}, \Delta)$. The result is a constant approximation to the optimal tree built for $\overline{\mathcal{D}[u]}$ w.r.t. both congestion and route length.

But since routing between high degree nodes u and v is done via the helper node ℓ, a slight change to T_u and T_v is needed. We modify T_u to create T'_u in the following way. If $\ell \notin T_u$ ($\mathrm{p}(u, \ell) = 0$), then node ℓ takes the position of node v in T'_u. If $\ell \in T_u$ ($\mathrm{p}(u, \ell) > 0$) then there are two cases: if $\mathrm{p}(u, \ell) > \mathrm{p}(u, v)$, we remove v from the tree; else when $\mathrm{p}(u, \ell) \leq \mathrm{p}(u, v)$, ℓ takes the position of v in the tree. In all cases, any communication from u to v is routed first to ℓ in T'_u and the forwarded to v on T'_v. Note that by definition ℓ will be in both trees.

To construct N we take the union of all these *ego-trees*, T'_u, for high degree nodes together with the low degree to low degree edges in G, i.e., (u, v) edges where both $u, v, \in \mathcal{L}$. This completes the construction of N. We present the pseudo-code in Algorithm 2 and the analysis in the next section.

B. *Analysis*

This section is devoted to prove Theorem 3. We first analyze the degree bound, and then study the congestion and expected route length in turn.

1) Analysis of Degree Bound: Each high degree node appears only in its optimal tree, hence has degree Δ. Each low degree node has degree at most 2ρ in $G_{\mathcal{D}}$, hence can be a part of 2ρ trees assuming all its neighbors are high degree. Additionally a low degree node may need to help at most ρ edges between high degree nodes and hence appears in another 2ρ trees. So a low degree node may appear in 4ρ trees resulting a degree $\Delta = 12\rho$ (since its degree in each tree is at most 3).

2) Analysis of Congestion, C*:* We start with a lower bound. In the optimal network, each node must have to carry all the outgoing and incoming communications (sum of its row/column in $\mathcal{D}$) via at most Δ edges to its neighbors. For a node u, at best, any algorithm performs the optimal Δ-way partitioning over $\mathcal{D}[u]$ and divides the load of u optimally via those Δ edges to the neighbors of u. Let C^*_u be the optimal solution for Δ-way partitioning for $\mathcal{D}[u]$. A lower bound for the minimum congestion is therefore $\mathrm{C}^* = \max_u \mathrm{C}^*_u$.

We now turn to the congestion in our algorithm. Consider a node u and the EGOTREE we built for it, T'_u. Let C_u be the congestion when we use $\mathcal{D}[u]$ to build T_u according to Algorithm 1. We know from Theorem 1 that $\mathrm{C}_u < (4/3)\mathrm{C}^*_u$. But, in Algorithm 2 we modify T_u to T'_u. Let C'_u denote the congestion in T'_u. Next we show a connection between C'_u and C_u.

Lemma 4: The congestion on T'_u is bounded such that C'_u less or equal to $2\mathrm{C}_u$.

Proof: Recall that, for a high degree node u, different low degree helper nodes are used for different high degree neighbors. That is, for the communication between u and v, where v is a high degree node, a low degree helping node ℓ is used such that ℓ is not being used for communications between u and any high degree node other than v. Let the left subtree of u be T_i in T_u as shown in Figure 4, where T'_u be the modified tree. Also assume that T_i in T_u has total probability S_i.

There are three ways to modify T_u to T'_u as per our construction. First, look at Figure 4.(a) that shows the case when $\mathrm{p}(u,\ell) > 0$ with $\mathrm{p}(u,\ell) \le \mathrm{p}(u,v)$. In this case, ℓ takes the position of v in the tree. Since ℓ was in the other sub tree, when it appears in T_i in the modified tree T'_u, S_i could at most be doubled as $\mathrm{p}(u,\ell) \le \mathrm{p}(u,v)$ itself. Second, look at Figure 4.(b) that shows the case when $\mathrm{p}(u,\ell) > 0$ with $\mathrm{p}(u,\ell) > \mathrm{p}(u,v)$. We remove v from the tree and communication happens via ℓ. As both v and ℓ were in T_i, hence S_i remains unchanged. Also if v would have been in the other subtree, then S_i would be increased at most by $\mathrm{p}(u,v) > 0$ which is again less than $\mathrm{p}(u,\ell)$ itself. Hence S_i could be at most doubled. Third, if $\ell \notin T_u$ ($\mathrm{p}(u,\ell) = 0$), then node ℓ takes the position of node v in T'_u; it is clear that no change in S_i happens in this case what so ever.

So, by construction of T'_u each element in T_i can stay in the same partition (but replaced by a helper node), removed or double its mass (since nodes with lower probability can only move to the location of higher probability nodes). So for each i, $S'_i \le 2S_i$, and the claim follows. □

From Lemma 4, the congestion on T'_u is such that $\mathrm{C}'_u \le 2\mathrm{C}_u \le 8/3\mathrm{C}^*_u$. So according to our construction, any edge may carry at most $(8/3)\mathrm{C}^*$ amount of traffic on a single tree constructed for high degree nodes where C^* is the optimal congestion. Low degree nodes may be present in at most 4ρ trees, so corresponding edges subsequently may carry loads from at most 4ρ trees for high degree roots. Accordingly the congestion is bounded by $4\rho \cdot (8/3)\mathrm{C}^* = (8/9)\Delta\mathrm{C}^*$. Additionally the original communication between two low degree nodes can be at most C^*. Hence,

$$\mathrm{C}(\mathcal{D}, \Gamma(N)) \le \mathrm{C}^*(1 + \frac{8}{9}\Delta) = \mathrm{C}^*(\mathcal{D}, \Delta)(1 + \frac{8}{9}\Delta)$$

i.e., $\alpha = 1 + (8/9)\Delta$ which is constant as Δ is constant.

$$\mathrm{L}(\mathcal{D}, \Gamma(N)) = \sum_{(u,v)\in\mathcal{D}} \mathrm{p}(u,v) d_{\Gamma(N)}(u,v)$$

(Since routes between all possible pairs are unique in $\Gamma(N)$)

$$= \sum_{u\in\mathcal{L}}\sum_{v\in\mathcal{L}} \mathrm{p}(u,v) + \sum_{u\in\mathcal{H}}\sum_{v\in\mathcal{L}} \mathrm{p}(u,v) d_{T'_u}(u,v)$$

$$+ \sum_{u\in\mathcal{L}}\sum_{v\in\mathcal{H}} \mathrm{p}(u,v) d_{T'_v}(u,v) + \sum_{u\in\mathcal{H}}\sum_{v\in\mathcal{H}} \mathrm{p}(u,v) d_{\Gamma(N)}(u,v)$$

(Sum over all possible pairs)

$$\le 1 + \sum_{u\in\mathcal{H}}\sum_{v\in\mathcal{L}} \mathrm{p}(u,v) d_{T'_u}(u,v) + \sum_{u\in\mathcal{L}}\sum_{v\in\mathcal{H}} \mathrm{p}(u,v) d_{T'_v}(u,v)$$

$$+ \sum_{u\in\mathcal{H}}\sum_{v\in\mathcal{H}} \mathrm{p}(u,v) \left[d_{T'_u}(u,\ell) + d_{T'_v}(\ell,v)\right]$$

(Route between u, v in $\Gamma(N)$ goes via ℓ when u and v are high degree)

$$= 1 + \left[\sum_{u\in\mathcal{H}}\sum_{v\in\mathcal{L}} \mathrm{p}(u,v) d_{T'_u}(u,v) + \sum_{u\in\mathcal{H}}\sum_{v\in\mathcal{H}} \mathrm{p}(u,v) d_{T'_u}(u,\ell)\right]$$

$$+ \left[\sum_{u\in\mathcal{L}}\sum_{v\in\mathcal{H}} \mathrm{p}(u,v) d_{T'_v}(u,v) + \sum_{u\in\mathcal{H}}\sum_{v\in\mathcal{H}} \mathrm{p}(u,v) d_{T'_v}(\ell,v)\right]$$

$$= 1 + \sum_{u\in\mathcal{H}}\sum_{v\in V} \mathrm{p}'(u,v) d_{T'_u}(u,v) + \sum_{u\in V}\sum_{v\in\mathcal{H}} \mathrm{p}'(u,v) d_{T'_v}(u,v)$$

(Using Equation (3))

$$= 1 + 2\sum_{u\in\mathcal{H}}\sum_{v\in V} \mathrm{p}'(u,v) d_{T'_u}(u,v) \quad \text{(Since } \mathcal{D}' \text{ is symmetric)}$$

$$= 1 + 2\sum_{u\in\mathcal{H}} \mathrm{p}'(u) \sum_{v\in V} \mathrm{p}'(v|u) d_{T'_u}(u,v)$$

(W.r.t. marginal distribution of $u \in \mathcal{H}$)

$$\le 1 + 2\sum_{u\in\mathcal{H}} \mathrm{p}'(u) \mathrm{L}(\overline{\mathcal{D}'[u]}, T'_u) \quad \text{(By definition of } \mathcal{D}'[u]\text{)}$$

$$\le 1 + 2\sum_{u\in\mathcal{H}} \mathrm{p}(u) 2\mathrm{L}(\overline{\mathcal{D}[u]}, T_u) \quad \text{(By Lemma 5 and Equation 6)}$$

$$\le 1 + 4\sum_{u\in\mathcal{H}} \mathrm{p}(u) H(\overline{\mathcal{D}[u]}) \quad \text{(Using Equation 1)}$$

$$\le 1 + 4H(Y \mid X) \tag{4}$$

3) Analysis of Route Length, L*:* We show that the expected route length is also optimal on this construction. We start with a lower bound that relates the expected route length to the conditional entropy $H(Y \mid X)$ of the joint distribution. We note that for symmetric distributions $H(Y \mid X) = H(X \mid Y)$. Formally we show:

Theorem 4: Consider a symmetric joint frequency distribution $\mathcal{D}$. Let X, Y be the random variables distributed according to

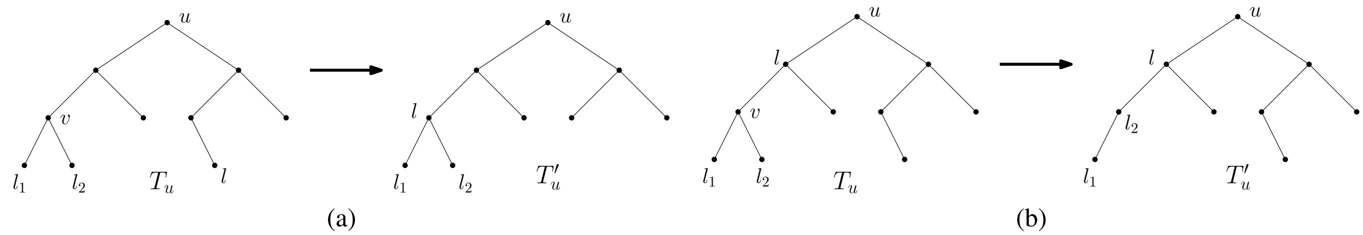


Fig. 4. (a) $\mathrm{p}(u,\ell) \leq \mathrm{p}(u,v)$: ℓ takes the position of v . (b) $\mathrm{p}(u,\ell) > \mathrm{p}(u,v)$: v is removed from the tree

the marginal distribution of the sources and destinations in $\mathcal{D}$, respectively. Then

$$\mathrm{L}^*(\mathcal{D},\Delta) \geq H(Y \mid X)/\log^2(\Delta+1) \tag{5}$$

Proof: Let $\Gamma^*(N)$ be the solution for optimal route length on $\mathcal{D}$. If we consider the union of optimal trees (T_u^*) of bounded degree Δ (w.r.t. route length) for each normalized row $\overline{\mathcal{D}[u]}$ of $\mathcal{D}$, the route length on this construction constitutes a valid lower bound on the route length, although the degree bound Δ is no more true. Therefore, using Equation 2 we can write,

$$\begin{aligned}\mathrm{L}^*(\mathcal{D},\Delta) &\geq \sum \mathrm{p}(u)\mathrm{L}(\overline{\mathcal{D}[u]}, T_u^*)\\ &\geq \sum \mathrm{p}(u)H_\Delta(\overline{\mathcal{D}[u]})/\log(\Delta+1)\\ &= H(Y \mid X)/\log^2(\Delta+1)\end{aligned}$$

□

Similarly, when considering the union of trees for each column:

$$\mathrm{L}^*(\mathcal{D},\Delta) \geq H(X \mid Y)/\log^2(\Delta+1)$$

We turn to our upper bound. Before the technical proof, we start with the intuitive discussion. One can view the bounded degree network N as a union of optimal trees (*ego-trees* T_u for u) built for the high degree nodes according to the construction presented in the proof of Theorem 1. But recall the construction of T'_u which is the modification of T_u. To preserve optimality we can first show:

Lemma 5: The expected route length on T'_u is bounded by $\mathrm{L}(\mathcal{D}'[u], T'_u) \leq 2\mathrm{L}(\mathcal{D}[u], T_u)$.

Proof: First note that

$$\mathrm{p}(u) = \mathrm{p}'(u), \tag{6}$$

that is, after the modification of $\mathcal{D}$ to $\mathcal{D}'$, u has the same probability mass. Next since a node in T_u may only move to a location with larger mass, this can increase by at most twice: the contribution of each node to the expected route length. □

For each request (u, v) in $\mathcal{D}$ there are two possibilities for the route on $\Gamma(N)$: either the edge $(u, v) \in N$ is a direct route, or the route goes via T'_u or T'_v or both. We can now prove the upper bound.

Lemma 6: The expected route length on N built using Algorithm 2 and $\Gamma(N)$ is bounded by

$$\mathrm{L}(\mathcal{D}, \Gamma(N)) \leq 1 + 4H(Y \mid X).$$

Proof: The analysis is shown in Equation 4. □

We conclude the proof of Theorem 3 by combining Theorem 4 and Lemma 6 to get

$$\mathrm{L}(\mathcal{D}, \Gamma(N)) \leq 1 + 4\log^2(\Delta+1)\mathrm{L}^*(\mathcal{D},\Delta). \tag{7}$$

We note that the result can be extended to *asymmetric* matrices and show (skipping some details due to space).

Theorem 5: Let $\mathcal{D}$ be an arbitrary but sparse distribution with ρ being the average degree in $G_\mathcal{D}$. It is possible to generate a *cl-DAN* of maximum degree $\Delta = 12\rho$ which achieves a $(\alpha = 1 + (8/9)\Delta, \beta = 1 + 4\log^2(\Delta+1))$-approximation.

The basic idea is to convert an asymmetric matrix $\mathcal{A}$ to a symmetric matrix $\mathcal{D}$ and design the network based on $\mathcal{D}$. We do not lose much by doing this. Let $\mathcal{A}$ be an asymmetric matrix. Let its corresponding symmetric matrix be $\mathcal{D} = (\mathcal{A} + \mathcal{A}^T)/2$.

It is easy to see that the degree bound Δ and the congestion bounds do not change as a result of this operation. First we discuss on Δ. The total number of edges in $G_\mathcal{A}$, $G_\mathcal{D}$ are the same and so the average degree ρ. Accordingly the set of low degree nodes, high degree nodes and their neighbors are no different in both $G_\mathcal{A}$ and $G_\mathcal{D}$. Therefore, the degree bound Δ remains the same. Next we discuss effects on congestion bound. Consider any $a(i,j) \in \mathcal{A}$ and the corresponding $d(i,j) \in \mathcal{D}$. Notice that $a(i,j) \leq 2d(i,j)$. Hence, when we create an *ego-tree* for any high degree node according to $\mathcal{D}$, each of the subtrees may have a total probability mass bounded by at most twice according to the original entities in $\mathcal{A}$. For the route length, we prove the following result on the tight relation between conditional entropies of $\mathcal{A}$ and $\mathcal{D}$. Let $H^*_{\text{con}} = \max(H^\mathcal{A}(Y \mid X), H^\mathcal{A}(X \mid Y))$, the maximum of both possible conditional entropies. Then we can state the following:

Lemma 7: The conditional entropy of the symmetric matrix $\mathcal{D}$ is in the order of the maximal conditional entropy of $\mathcal{A}$.

$$H^\mathcal{D}(Y \mid X) = H^\mathcal{D}(X \mid Y) \leq H^*_{\text{con}} + 1$$

To prove the theorem, we first show the following lemma.

Lemma 8: Let $\overrightarrow{\mathrm{p}}$ and $\overrightarrow{\mathrm{q}}$ be two probability (frequency) distributions for the same set. Let $H^* = \max(H(\overrightarrow{\mathrm{p}}), H(\overrightarrow{\mathrm{q}}))$. Then

$$\frac{1}{2}H^* \leq \frac{1}{2}H(\overrightarrow{\mathrm{p}}) + \frac{1}{2}H(\overrightarrow{\mathrm{q}}) \leq H(\frac{\overrightarrow{\mathrm{p}} + \overrightarrow{\mathrm{q}}}{2}) \leq H^* + 1$$

Proof: The lower bound is implied by the concavity of entropy [8], i.e., $H((1/2)\overrightarrow{\mathrm{p}} + (1/2)\overrightarrow{\mathrm{q}}) \geq (1/2)H(\overrightarrow{\mathrm{p}}) +$

$(1/2)H(\overrightarrow{\mathrm{q}})$. For the upper bound, we have:

$$\begin{aligned}
H(\frac{\overrightarrow{\mathrm{p}}+\overrightarrow{\mathrm{q}}}{2}) &= \sum \frac{\mathrm{p}_i+\mathrm{q}_i}{2}\log\frac{2}{\mathrm{p}_i+\mathrm{q}_i}\\
&= \frac{1}{2}\sum \mathrm{p}_i\log\frac{2}{\mathrm{p}_i+\mathrm{q}_i}+\frac{1}{2}\sum \mathrm{q}_i\log\frac{2}{\mathrm{p}_i+\mathrm{q}_i}\\
&\le \frac{1}{2}\sum \mathrm{p}_i\log(\frac{1}{\mathrm{p}_i})+\frac{1}{2}+\frac{1}{2}\sum \mathrm{q}_i\log(\frac{1}{\mathrm{q}_i})+\frac{1}{2}\\
&= \frac{1}{2}H(\overrightarrow{\mathrm{p}})+\frac{1}{2}H(\overrightarrow{\mathrm{q}})+1 \le H^*+1
\end{aligned}$$

□

We now prove Lemma 7.

Proof:[Proof of Lemma 7] From Lemma 8 we have $H^{\mathcal{D}}(X,Y) \le H\mathcal{D}(X,Y)+1$ and $H^{\mathcal{D}}(X) \ge (1/2)H^{\mathcal{A}}(X)+(1/2)H^{\mathcal{A}}(Y)$. Now we can bound the conditional entropy.

$$\begin{aligned}
H^{\mathcal{D}}(Y \mid X) &= H^{\mathcal{D}}(X,Y) - H^{\mathcal{D}}(X)\\
&\le H^{\mathcal{A}}(X,Y)+1-\frac{1}{2}H^{\mathcal{A}}(X)-\frac{1}{2}H^{\mathcal{A}}(Y)\\
&= \frac{1}{2}H^{\mathcal{A}}(Y \mid X)+\frac{1}{2}H^{\mathcal{A}}(X \mid Y)+1\\
&\le H^*_{\mathrm{con}}+1
\end{aligned}$$

By the symmetry of the matrix, we have that $H^{\mathcal{D}}(Y \mid X) = H^{\mathcal{D}}(X \mid Y)$. □

V. Numerical Example

This section gives a more detailed numerical example for our construction. We start with a given demand distribution in the form of a graph and its edge weights. We construct a bounded degree network that serves the given demands. Though our theoretical results target large-scale networks, here we take a small initial network and construct the final network following our idea of a bounded-degree network construction. This is to provide an intuition to the reader about how our algorithm works.

The Given Network: In the given network G (see Figure 5), h_i, $1 \le i \le 12$ are the high degree nodes. The nodes l_i^j, $1 \le j \le 12$ are the low degree neighbors of h_i and together they form a block B_{h_i}. Each block has exactly the same structure. One such block B_{h_4} is shown. The edges between the high degree nodes are colored in red. The maximum degree is 23 and it is of h_4. All the remaining high degree nodes have degree 13. Low degree nodes l_7^j and l_9^j have degree 2 for all j. All the remaining low degree nodes have degree 1. The total number of nodes is 156 and the total number of edges is 167. We want to serve a distribution where each edge $h_i l_j^i$ for all i and even j serves demand p. We have 72 such edges. Let each of the other edges of the form $h_i l_j^i$ for all i and odd j serve demand p'. Also each $l_7^i l_9^i$ for all i serves demand p'. There are 12 such edges. Among the edges between high degree nodes, each $h_4 h_i$ for odd i serves demand p. There are 6 such edges. The remaining 5 red edges each serves demand p'. Take $p = 2/245$ and $p' = 1/245$ and notice that this forms a probability distribution.

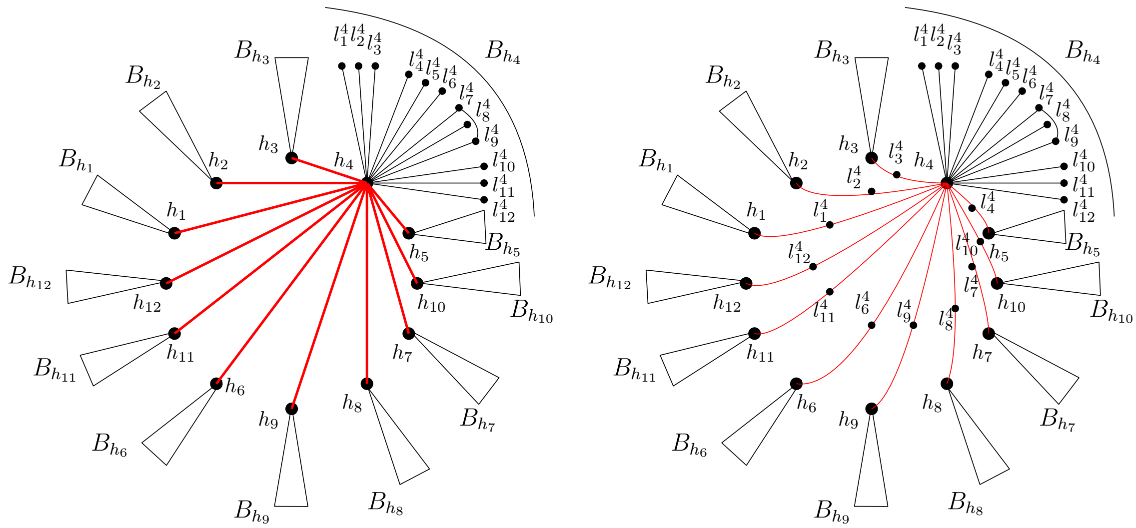
So, in this network, almost half (78) of the edges serve communication 2/245 whereas other edges (89) serve 1/245. The average degree is $\rho = 2.14$. We can enlarge this example by adding new high degree nodes (with their respective blocks) connected to h_4. So, roughly we can understand that in this network there are $O(\sqrt{n})$ many high degree nodes with their degrees being also $O(\sqrt{n})$. We finally construct a network of maximum degree 4.

A Modified Network: As our aim is to keep only low degree neighbors for each high degree node, each edge $h_4 h_j$ between two high degree nodes in G is replaced by two edges $h_4 l_j^4$ and $l_j^4 h_j$, as shown in Figure 5(b). We call this the modified network G'. Demands are also modified according to Equation 3. We can observe that, no low degree node will be a part of more than two ego-trees. Also, only l_i^4, $1 \le i \le 12$ will be part of two modified ego-trees, the remaining low degree nodes will be part of only one modified ego-tree. Due to the distribution that the initial network serves, we can see that the modified ego-trees can be made in such a way that none of l_i^4, $1 \le i \le 12$ will have a cumulative degree more than 4 in the final network. This is because, on the modified ego-tree of h_4 the degree of any of these nodes (except l_4^4) can be 3 but in the other modified ego-tree, its degree can be kept to 1. Hence in this example we keep the degree of high degree nodes to 4 since according to our theory, the degree of high degree nodes in its ego-tree needs to be Δ where Δ is the bound on the degree of the low degree nodes in the final network. As we already mentioned, this does not completely match with our theory, however we provide an example considering a small network that shows the idea behind our construction.

Modified Ego-trees: We show the modified ego-trees of h_4 and h_6 (see Figure 6) based on the modified network G' and the modified demands. According to the modified distribution, in G', each edge between h_4 and l_i^4 (except for $i = 4$) serves $(p + p')$ amount of traffic and l_4^4 serves only p'. Hence in the modified ego-tree of h_4, l_4^4 must be a leaf node. As discussed above we take $\Delta = 4$. The remaining nodes can be there in the tree arbitrarily as the corresponding edges serve the same demands. Similarly in the modified ego-tree of h_6, l_j^6 for all even j along with l_6^4 serve p whereas other neighbors of h_6 serve p'. As there are 5 internal nodes in this modified ego-tree and 7 edges corresponding to 7 nodes carry p, it is possible to keep l_6^4 in a leaf and its degree would be 1. Here also $\Delta = 4$. Note that, l_6^4 is present in both the trees. Similarly the modified ego-trees corresponding to all the high degree nodes are computed. Observe that, all these ego-trees will have a similar structure as of the ego-tree of h_6.

The Final Construction: Finally, we take the union of all the modified ego-trees corresponding to the high degree nodes. Along with that, we add the original edges between low degree nodes. This gives us the final network. Here we show in Figure 7, a part of the final network by taking the union of ego-trees of h_4 and h_6, the blue dashed edge joining two copies of l_6^4 helps to unite them. Also the edges (colored in blue) between the low degree neighbors l_7^i and l_9^i of these two high degree nodes are kept in the final network. Observe that the maximum degree of the final network does not depend on n. So we can roughly say that we brought the degree down from $O(\sqrt{n})$ to 4.

Congestion in the initial network was $p = 2/245$. Congestion in our constructed network can be calculated by computing the congestion on the out-going edges from h_4 in its modified



(a) Initial network, (b) Modified network

Fig. 5. (a) Network generated from the given demand (b) Modified network after using helper nodes

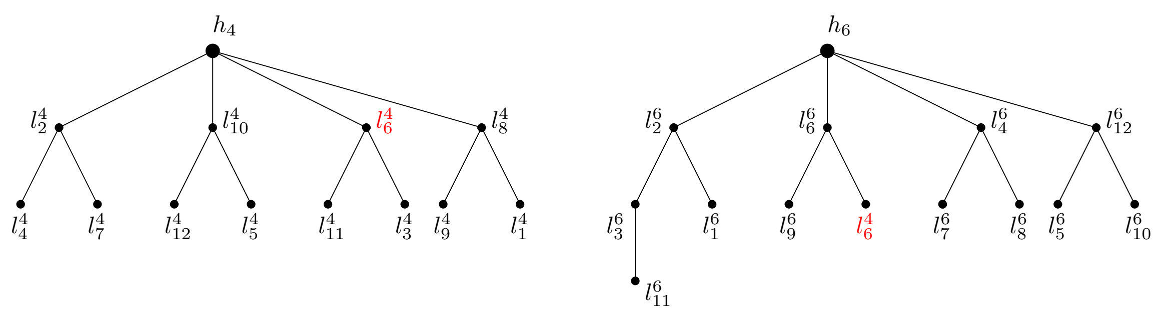


Fig. 6. Modified ego-trees of h_4 and h_6 are shown

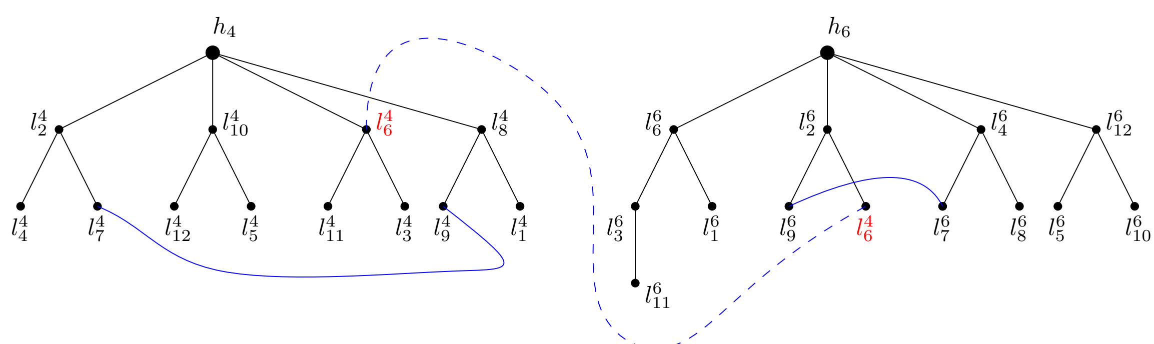


Fig. 7. A part of the final network

ego-tree and the congestion is $3p + 3p' = 9/245$, which is an 4.5-approximation of the original.

Remark: In the above example, we have shown a possible construction of modified ego-trees that can come up using our algorithm. We have chosen the position of the low degree nodes in the ego-trees purposefully such that the degree of any node remains at most 4 in the final network. This helps us to have a better view of the ego-tree structure where the root can have more than two children but all the sub-trees are binary. However, if one exactly follows our algorithm, it will produce a final network where the degree bound can be as large as 7 since a low degree node can be a part of at most 2 ego-trees and some low-degree nodes have a low-degree neighbor. Note that, this obeys our theoretical bound 12ρ as here we have $\rho > 2$.

VI. Related Work

The advent of technologies for reconfigurable networks has motivated much research recently [**?**], [6], [10], [14], [15], [18], [20], [21], [34], [35] Empirical studies confirm that communication patterns are often *sparse* and of *low entropy*, which can be exploited in demand-aware networks: in [21], it is shown that a high percentage of rack pairs does not exchange any traffic at all, while less than 1% of them account for 80% of the total traffic. The study of reconfigurable networks is not limitedh to data center networks. Interesting use cases also arise in the context of wide-area networks [18], [32] and, more traditionally, in the context of overlays [25], [28].

In contrast to most existing work, we in this paper are mainly interested in the algorithmic aspects of demand-aware network designs, see [4] for a recent taxonomy and survey of the field. Related to our perspective, Avin et al. [3] presented algorithms to design bounded-degree demand-aware networks providing an almost optimal expected route length under sparse communication patterns. The algorithms in [3] build upon initial insights on SplayNets [29] (later extended to *distributed* SplayNets [24]). Foerster et al. [11] presented algorithms to design networks for a model based on emerging optical switches providing flexible matchings on top of an otherwise static network. However, these works focus on networks providing short route lengths and do not account for the congestion introduced by multiple commodities. The study of congestion however is of prime importance as it directly affects the network performance, but also renders the algorithmic problem different in nature and more challenging. Other solutions in the literature, such as [33], [34], either rely on integer programming which can result in super-polynomial run times, or on heuristics which do not provide any provable guarantees.

Finally, we note that our approach of reconfiguring network topologies to reduce communication costs, is orthogonal to approaches changing the traffic matrix itself (e.g., [27]) or migrating communication endpoints on a fixed topology [1], [2], [16].

VII. Conclusion

We presented the first demand-aware networks which provide provable guarantees on both congestion and route length, the two main objectives in traffic engineering. The proposed networks are of bounded degree and hence scalable.

our work as a first step and believe that it opens several interesting avenues for future research. In particular, it will be interesting to investigate more fault-tolerant designs as well as dynamic demand-aware networks which can self-adjust over time to temporally changing traffic patterns. It would also be interesting to explore the use of demand-aware networks in other application domains, such as sensor networks [36], [37].

Acknowledgments.

Research supported by the ERC Consolidator grant *AdjustNet* (agreement no. 864228).

References

[1] Chen Avin, Louis Cohen, Mahmoud Parham, and Stefan Schmid. Competitive clustering of stochastic communication patterns on a ring. In Journal of Computing, 2018.
[2] Chen Avin, Andreas Loukas, Maciej Pacut, and Stefan Schmid. Online balanced repartitioning. In Proc. 30th International Symposium on Distributed Computing (DISC), 2016.
[3] Chen Avin, Kaushik Mondal, and Stefan Schmid. Demand-aware network designs of bounded degree. In 31st International Symposium on Distributed Computing, DISC 2017, 2017.
[4] Chen Avin and Stefan Schmid. Toward demand-aware networking: A theory for self-adjusting networks. In ACM SIGCOMM Computer Communication Review (CCR), 2018.
[5] Theophilus Benson, Ashok Anand, Aditya Akella, and Ming Zhang. Understanding data center traffic characteristics. In Proc. ACM Workshop on Research on Enterprise Networking, 2009.
[6] Kai Chen, Ankit Singla, Atul Singh, Kishore Ramachandran, Lei Xu, Yueping Zhang, Xitao Wen, and Yan Chen. Osa: An optical switching architecture for data center networks with unprecedented flexibility. IEEE/ACM Transactions on Networking (TON), 2014.
[7] Cisco. Cisco global cloud index: Forecast and methodology, 2015-2020. White Paper, 2015.
[8] Thomas M Cover and Joy A Thomas. Elements of information theory. John Wiley and Sons, 2012.
[9] Christina Delimitrou, Sriram Sankar, Aman Kansal, and Christos Kozyrakis. Echo: Recreating network traffic maps for datacenters with tens of thousands of servers. In Workload Characterization (IISWC), 2012 IEEE International Symposium on, 2012.
[10] Nathan Farrington, George Porter, Sivasankar Radhakrishnan, Hamid Hajabdolali Bazzaz, Vikram Subramanya, Yeshaiahu Fainman, George Papen, and Amin Vahdat. Helios: a hybrid electrical/optical switch architecture for modular data centers. Proc. ACM SIGCOMM Computer Communication Review (CCR), 2010.
[11] Klaus-Tycho Foerster, Monia Ghobadi, and Stefan Schmid. Characterizing the algorithmic complexity of reconfigurable data center architectures. In Proc. ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS), 2018.
[12] Ronald L. Graham. Bounds on multiprocessing timing anomalies. SIAM Journal of Applied Mathematics, 17(2), 1969.
[13] Albert Greenberg, James R Hamilton, Navendu Jain, Srikanth Kandula, Changhoon Kim, Parantap Lahiri, David A Maltz, Parveen Patel, and Sudipta Sengupta. Vl2: a scalable and flexible data center network. In Proc. ACM SIGCOMM Computer Communication Review (CCR), volume 39, 2009.
[14] D. Halperin, S. Kandula, J. Padhye, P. Bahl, and D. Wetherall. Augmenting data center networks with multi-gigabit wireless links. In Proc. ACM SIGCOMM, 2011.
[15] Navid Hamedazimi, Zafar Qazi, Himanshu Gupta, Vyas Sekar, Samir R Das, Jon P Longtin, Himanshu Shah, and Ashish Tanwer. Firefly: A reconfigurable wireless data center fabric using free-space optics. In Proc. ACM SIGCOMM Computer Communication Review (CCR), volume 44, 2014.
[16] Monika Henzinger, Stefan Neumann, and Stefan Schmid. Efficient distributed workload (re-)embedding. In Proc. ACM SIGMETRICS / IFIP PERFORMANCE, 2019.
[17] Mark Jerrum and Alistair Sinclair. The markov chain monte carlo method: an approach to approximate counting and integration. Approximation algorithms for NP-hard problems, 1996.
[18] Su Jia, Xin Jin, Golnaz Ghasemiesfeh, Jiaxin Ding, and Jie Gao. Competitive analysis for online scheduling in software-defined optical wan. In Proc. IEEE INFOCOM, 2017.
[19] Srikanth Kandula, Sudipta Sengupta, Albert Greenberg, Parveen Patel, and Ronnie Chaiken. The nature of data center traffic: measurements and analysis. In Proc. ACM Conference on Internet Measurement (IMC). ACM, 2009.
[20] He Liu, Feng Lu, Alex Forencich, Rishi Kapoor, Malveeka Tewari, Geoffrey M Voelker, George Papen, Alex C Snoeren, and George Porter. Circuit switching under the radar with reactor. In Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI), volume 14, 2014.
[21] M. Ghobadi et al. Projector: Agile reconfigurable data center interconnect. In Proc. ACM SIGCOMM, 2016.
[22] Julian J. McAuley and Jure Leskovec. Learning to discover social circles in ego networks. In Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States., 2012.
[23] Mohammad Noormohammadpour and Cauligi S Raghavendra. Datacenter traffic control: Understanding techniques and trade-offs. IEEE Communications Surveys and Tutorials, 2017.
[24] Bruna Peres, Otavio Souza, Olga Goussevskaia, Stefan Schmid, and Chen Avin. Distributed self-adjusting tree networks. In Proc. IEEE INFOCOM, 2019.
[25] Sylvia Ratnasamy, Mark Handley, Richard Karp, and Scott Shenker. Topologically-aware overlay construction and server selection. In Proc. IEEE INFOCOM, volume 3, 2002.

[26] Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, and Alex C Snoeren. Inside the social network's (datacenter) network. In Proc. ACM SIGCOMM Computer Communication Review (CCR), volume 45, pages 2015.

[27] Nadi Sarrar, Steve Uhlig, Anja Feldmann, Rob Sherwood, and Xin Huang. Leveraging zipf's law for traffic offloading. Proc. ACM SIGCOMM Computer Communication Review (CCR), 42(1), 2012.

[28] [28] Christian Scheideler and Stefan Schmid. A distributed and oblivious heap. Proc. International Conference on Automata, Languages and Programming (ICALP), 2009.

[29] Stefan Schmid, Chen Avin, Christian Scheideler, Michael Borokhovich, Bernhard Haeupler, and Zvi Lotker. Splaynet: Towards locally selfadjusting networks. IEEE/ACM Transactions on Networking (ToN), to appear.

[30] Ethan L Schreiber. Optimal Multi-Way Number Partitioning. University of California, Los Angeles, 2014.

[31] Arjun Singh, Joon Ong, Amit Agarwal, Glen Anderson, Ashby Armistead, Roy Bannon, Seb Boving, Gaurav Desai, Bob Felderman, Paulie Germano, et al. Jupiter rising: A decade of clos topologies and centralized control in google's datacenter network. Proc. ACM SIGCOMM Computer Communication Review (CCR), 45(4), 2015.

[32] Rachee Singh, Manya Ghobadi, Klaus-Tycho Foerster, Mark Filer, and Phillipa Gill. Radwan: rate adaptive wide area network. In Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication. 2018.

[33] Ankit Singla. Fat-free topologies. In Proc. ACM Workshop on Hot Topics in Networks (HotNets),2016.

[34] Ankit Singla, Atul Singh, Kishore Ramachandran, Lei Xu, and Yueping Zhang. Proteus: a topology malleable data center network. In Proc. ACM Workshop on Hot Topics in Networks (HotNets), 2010.

[35] Xia Zhou, Zengbin Zhang, Yibo Zhu, Yubo Li, Saipriya Kumar, Amin Vahdat, Ben Y Zhao, and Haitao Zheng. Mirror mirror on the ceiling: Flexible wireless links for data centers. Proc. ACM SIGCOMM Computer Communication Review (CCR), 2012.

[36] Mohammed Hasan, Hussain Al-Rizzo, and Fadg Al-Turjman. A survey on multipath routing protocols for QoS assurances in real-time wireless multimedia sensor networks. IEEE Communications Surveys & Tutorials, 2017.

[37] Esra Ceylan, Klaus-Tycho Foerster, Stefan Schmid, and Katsiaryna Zaitsava. Demand-Aware Plane Spanners of Bounded Degree. Proc. IFIP Networking, 2021.

APPENDIX A
PROOF OF LEMMA 1

Before going to prove Lemma 1, we first show that LPT indeed works on rational numbers too.

Let $\{\mathrm{f}_1, \mathrm{f}_2, \ldots, \mathrm{f}_n\}$ be a set of integer frequencies indicating number of times (f_i) a node (i) communicates with the source s. Let $\bar{p} = \{\mathrm{p}_1, \mathrm{p}_2, \ldots, \mathrm{p}_n\}$ be the corresponding set of frequency distributions such that $\mathrm{p}_j = \mathrm{f}_j / \sum \mathrm{f}_i$. Clearly all p_i are rational numbers. We show that LPT generates equivalent partitions on both the above sets. LPT first sorts the inputs in non-increasing order. W.l.o.g., we assume that, $\mathrm{f}_1 \leq \mathrm{f}_2 \leq \ldots \leq \mathrm{f}_n$. Consequently $\mathrm{p}_1 \leq \mathrm{p}_2 \leq \ldots \leq \mathrm{p}_n$. Since LPT takes elements from this order and puts in one of the Δ subsets which has the minimum subset sum, the first Δ elements $\mathrm{f}_1, \mathrm{f}_2, \ldots, \mathrm{f}_\Delta$ must fall into different subsets namely, $\Pi_1^{(\mathrm{f})}, \Pi_2^{(\mathrm{f})}, \ldots, \Pi_\Delta^{(\mathrm{f})}$. This represents the subsets of the partition uniquely. Similarly we define the subsets generated on partitioning $\bar{\mathrm{p}}$ by $\Pi_1^{(\mathrm{p})}, \Pi_2^{(\mathrm{p})}, \ldots, \Pi_\Delta^{(\mathrm{p})}$. To show these two partitions are equivalent, we state the following lemma.

Lemma 9: For any j, if f_j is in $\Pi_k^{(\mathrm{f})}$ $(k \leq \Delta)$, then the corresponding frequency $\mathrm{p}_j = \mathrm{f}_j / \sum \mathrm{f}_i$ would be in $\Pi_k^{(\mathrm{p})}$.

Proof: We prove it by induction. For $j = 1$, f_1 belongs to $\Pi_1^{(\mathrm{f})}$ and p_1 belongs to $\Pi_1^{(\mathrm{p})}$. For $j = 2$, f_2 belongs to $\Pi_2^{(\mathrm{f})}$ and p_2 belongs to $\Pi_2^{(\mathrm{p})}$. By induction, let this be true for $j = m - 1$, and let, before processing f_m, the sums of the frequencies in the subsets of the corresponding partitions to be $\mathrm{S}_1^{(\mathrm{f})}, \mathrm{S}_2^{(\mathrm{f})}, \ldots, \mathrm{S}_\Delta^{(\mathrm{f})}$. Assume that $\mathrm{S}_k^{(\mathrm{f})}$ is minimum among them. At the same time, we denote the sum of the subsets in the partition of $\bar{\mathrm{p}}$ as $\mathrm{S}_1^{(\mathrm{p})}, \mathrm{S}_2^{(\mathrm{p})}, \ldots, \mathrm{S}_\Delta^{(\mathrm{p})}$. Since the property is true for $j = m - 1$, and $\mathrm{S}_i^{(\mathrm{p})} = \mathrm{S}_i^{(\mathrm{f})} / \sum \mathrm{f}_i$, so if $\mathrm{S}_k^{(\mathrm{f})}$ attains the minimum sum, so does $\mathrm{S}_k^{(\mathrm{p})}$. □

Hence LPT puts p_m in to $\Pi_k^{(\mathrm{p})}$. The partitions w.r.t. a set of integer frequencies and the corresponding set of frequency distributions have equivalent subsets. Now we want to leverage the fact [12] that LPT partitions any set of integers within an $4/3 - 1/(3\Delta)$-approximation of the optimal (Theorem 2).

Corollary 1: Theorem 2 holds for partitioning frequency distribution $\bar{\mathrm{p}}$.

Proof: The proof is immediate from Lemma 9. □

Now we prove Lemma 1.

Proof:[of Lemma 1] Let $\Pi_1, \Pi_2, \ldots, \Pi_\Delta$ denote the partition of $\bar{p}$ according to the binary subtrees, and let S_i denote the total probability mass of Π_i, i.e., $S_i = \sum_{p_j \in \Pi_i} p_j$. Let $\max_i S_i$ be the maximum subset sum where S_{opt} be the maximum sum in the optimum partition. From Corollary 1, we can say, $\max_i S_i \leq (4/3 - 1/(3\Delta))\, \mathrm{S}_{opt} \leq (4/3)\mathrm{S}_{opt}$. Since source s can have at most Δ neighbors, and all the communications must flow through those, then there must be one edge from s to some neighbor in the optimum tree through which S_{opt} amount of communication flows, which is the optimal congestion. We construct an *ego-tree* where congestion is less than equal to $(4/3)\mathrm{S}_{opt}$ since no edge in the *ego-tree* carries communications more than $\max_i S_i$. This completes the proof. □

Chen Avin received a B.Sc. degree in Communication Systems Engineering from Ben Gurion University, Israel, in 2000. He received the M.S. and Ph.D. degrees in computer science from the University of California, Los Angeles (UCLA) in 2003 and 2006 respectively. He is an associate professor at the Department of Communication Systems Engineering, School of Electrical and Computer Engineering, at the Ben Gurion University of the Negev, Israel. His current research interests are: data-driven graphs and networks algorithms, modeling and analysis with emphasis on demand-aware networks, distributed systems, social networks and randomize algorithms for networking.

Kaushik Mondal is an Assistant Professor at Indian Institute of Technology Ropar. He received his MSc (2008) from Visva Bharati, a Central University in India and PhD (2015) from Indian Institute of Technology Guwahati. Then from August 2015, he worked as postdoc at various institutes namely IIT Guwahati, Ben Gurion University of Negev, Israel and Indian Statistical Institute, Kolkata till June 2019. Then he worked as an Assistant Professor at Indian Institute of Information Technology, Vadodara from July 2019 to January 2020 before joining IIT Ropar. His research interests include network algorithms, distributed algorithms for swarm robots, graph algorithms etc.

Stefan Schmid is a Full Professor at TU Berlin, Germany, and research fellow at the University of Vienna, Austria. He received his MSc (2004) and PhD (2008) from ETH Zurich, Switzerland. Stefan Schmid worked as postdoc at TU Munich and the University of Paderborn (2009), from 2009 to 2015, he was a senior research scientist at T-Labs in Berlin, Germany, from 2015 to 2018 an Associate Professor at Aalborg University, Denmark, and from 2018 to 2021 a Full Professor at the University of Vienna.