DISTRIBUTED LINK SPARSIFICATION FOR SCALABLE SCHEDULING USING GRAPH NEURAL NETWORKS

Zhongyuan Zhao[*], Ananthram Swami[†], and Santiago Segarra[*]

[*]Rice University, USA   [†]US Army's DEVCOM Army Research Laboratory, USA

ABSTRACT

Distributed scheduling algorithms for throughput or utility maximization in dense wireless multi-hop networks can have overwhelmingly high overhead, causing increased congestion, energy consumption, radio footprint, and security vulnerability. For wireless networks with dense connectivity, we propose a distributed scheme for link sparsification with graph convolutional networks (GCNs), which can reduce the scheduling overhead while keeping most of the network capacity. In a nutshell, a trainable GCN module generates node embeddings as topology-aware and reusable parameters for a local decision mechanism, based on which a link can withdraw itself from the scheduling contention if it is not likely to win. In medium-sized wireless networks, our proposed sparse scheduler beats classical threshold-based sparsification policies by retaining almost 70% of the total capacity achieved by a distributed greedy max-weight scheduler with 0.4% of the point-to-point message complexity and 2.6% of the average number of interfering neighbors per link.

Index Terms— Independent set, graph neural networks, scheduling overhead, massive access, distributed scheduling.

1. INTRODUCTION

The proliferation of wireless devices and emerging machine-to-machine (M2M) traffic [1] bring new requirements to wireless networks, such as massive access, ultra-dense networks, better spectrum and energy efficiencies [2–4]. Wireless multi-hop communications is a promising solution for applications such as military communications, vehicular/UAV networks, wireless backhaul for xG, and Internet of Things (IoT) [2–6]. A fundamental problem in wireless multi-hop networks is distributed resource allocation without the help of infrastructure. This includes link scheduling, which determines which links should transmit and when should they transmit [7, 8]. The typical formulation of optimal scheduling in wireless multi-hop networks with orthogonal access is solving a maximum weighted independent set (MWIS) problem on a conflict graph [7–17], in which a vertex represents a link in the wireless network, an edge captures the interference relationship between two links, and the vertex weight is the utility of scheduling the corresponding link. The MWIS problem is known to be NP-hard [10, 15], and many approximate solutions have been proposed.
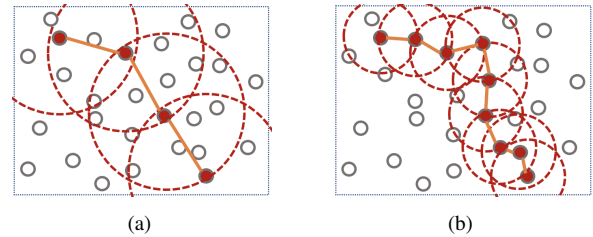
Fig. 1: Multi-hop wireless networks with neighborhoods of different sizes. (a) Dense network with large neighborhoods. (b) Sparse network with small neighborhoods. The overhead, collision, and starvation of scheduling increases with the neighborhood size, while the relays and thus overhearing increase in sparser networks.

Common approaches include distributed greedy algorithms [11, 12], carrier sense multiple access (CSMA) [18, 19], message passing algorithms [13, 14], and their hybrids [16, 17]. Being designed for throughput or utility maximization [9], the existing distributed schedulers introduce contention overheads or collision rates proportional to the average size of the neighborhood in the network, as illustrated in Fig. 1. This can lead to prohibitively high scheduling overhead in massive access [1, 4], as well as increased congestion, instability, energy consumption, radio footprint, and security vulnerability in network operations [4, 20–22].

Existing approaches to the aforementioned problems include topology control with antenna beams and/or transmit power [22–24], sleep scheduling [21, 24–26], and cross-layer optimization [27–29]. Topology control seeks to maintain a connected topology with minimal energy consumption, which however would increase overhearing and latency (e.g., larger hop distance as shown in Fig. 1(b)), as well as the risks of congestion and failure at some critical nodes. Sleep scheduling can reduce network density by putting nodes into periodic sleep mode in standalone or coordinated manner, and generally work well for low-duty-cycle devices with very low traffic demand. Cross-layer schemes jointly optimize power control, link scheduling, and routing for energy efficiency, but they generally have the same or higher control overhead than the above-mentioned MWIS schedulers, may need centralized computing [28, 29], and only apply to either time-slotted or random access networks.

In this work, we consider scalable scheduling in general wireless multi-hop networks with heterogeneous devices (e.g., with different importance, air-interfaces, and tasks), orthogonal access, and high connection density, where existing approaches [21–29] are less effective or not scalable. Since network capacity is mainly constrained by interference (connection density) [30] and given the growing bandwidth of air-interfaces [1], we can trade off latency for connectivity by scheduling fewer transmissions each with larger payloads for each link. For example, a desired subset of links could be excluded from contention by a global cut-off threshold derived from the empirical cumulative distribution function (eCDF) of per-link utility in the network. With a utility function based on queue

length [11,12] or sojourn time [31], each link will eventually join the contention as new packets arrive or time passes. Importantly, fewer concurrent transmission attempts will reduce scheduling overhead, collision, and idle listening. Furthermore, this approach could also be incorporated into existing solutions [21–29].

In this paper, we go beyond the statistical approach discussed above, and propose a link sparsification scheme based on graph convolutional networks (GCNs), which assesses a link using both the utility value and topological information, and withholds it from a scheduling contention that it could not win. In a nutshell, a trainable GCN generates two topology-aware multipliers to respectively scale the per-link utility and the global cut-off threshold for each link, which drive the decision of whether or not a link should contest for scheduling. Inspired by [16, 17], the overall architecture of our sparse scheduler is composed of a GCN followed by a non-differentiable distributed contention process, which can be a distributed greedy scheduler [11, 12] for time-slotted networks, or weighted CSMA [18, 19] for random access networks. We propose a novel two-stage approach to train our GCN for link sparsification: 1) The GCN is trained to mimic the performance of a baseline (non-sparse) greedy scheduler while reducing the number of links considered in the contention, and 2) The GCN is trained to outperform the baseline of a sparse greedy scheduler with different sparsity thresholds. Although our method relies on centralized training, it can be deployed in a fully distributed manner thanks to the distributed nature of the GCN and the subsequent scheduler.

Contribution. The contributions of this paper are twofold: 1) We propose the first GCN-based distributed link sparsification scheme for wireless scheduling that exploits the topology of the interference graph, and 2) Through numerical experiments, we demonstrate the superior performance of the proposed method as well as its generalizability over different topologies.

2. SYSTEM MODEL AND PROBLEM STATEMENT

Consider a wireless multi-hop network, where an (undirected) link (i, j) implies that user i and user j can communicate with each other. A flow describes the stream of packets from a source user to a destination user, and may pass through multiple links determined by a routing scheme. In each link, there is a queuing system q for packets of all the flows as well as exogenous arrivals.

To describe the scheduling algorithm, we define *conflict graph*, $\mathcal{G}(\mathcal{V}, \mathcal{E})$, as follows: a vertex $v \in \mathcal{V}$ represents a link in the wireless network, and the presence of an undirected edge $e = (v_a, v_b) \in \mathcal{E}$ captures the interference relationship between links $v_a, v_b \in \mathcal{V}$, which is considered to follow a physical distance model [10]. For example, two links interfere with each other if their incident users are within a certain distance such that their simultaneous transmission will cause the outage probability to exceed a prescribed level, or they share the same user with only one radio interface. For the rest of this paper, we focus on the conflict graph $\mathcal{G}$, which we assume to be known; see, e.g., [32] for its estimation. In principle, the interference zone of each link (hence $\mathcal{G}$) depends on the transmit power and antenna directivity of the corresponding users. To avoid this dependency, we consider a simplified scenario in which all the users transmit at power levels that are time-invariant. From the definition of $\mathcal{G}$, a legal schedule must be a set of wireless links that can communicate simultaneously in time and frequency under orthogonal access, which forms an independent (vertex) set in $\mathcal{G}$ defined as a set of nodes with no edges connecting each other.

We describe the *network state* at time t by the tuple $(\mathcal{G}(t), \mathbf{u}(t))$ consisting of the conflict graph $\mathcal{G}(t)$ (potentially changing over time)

and a utility vector $\mathbf{u}(t)$ collecting $u(v, t) \in \mathbb{R}_+$ for all $v \in \mathcal{V}$. The utility $u(v, t)$ can capture, e.g., the hierarchical class, backlogs, sojourn time, and link rate of the wireless link v. We denote by $c(\cdot)$ the scheduling contention process (e.g., a distributed greedy scheduler) that maps every network state into an approximate MWIS of the graph. For notational simplicity, we henceforth omit t for operations in the same time slot, and denote the total utility of a vertex set $\boldsymbol{v}$ by $u(\boldsymbol{v}) = \sum_{v \in \boldsymbol{v}} u(v)$. Since our goal is to reduce the signaling overhead in the network, we want each link to decide if it will contest for scheduling based on its own information, without talking with its neighbors. Hence, we want to find vertex-specific functions h_v for all $v \in \mathcal{V}$ such that their application to the local utilities $u(v)$ determine whether a vertex should be considered for contention or not. Formally, we define our problem as follows.

Problem 1. *Given a distribution $\mathcal{N}$ over network states $(\mathcal{G}, \mathbf{u})$, we want to obtain the optimal link sparsification functions $\{h_v^*\}$ for all $v \in \mathcal{V}$ as*

$$\{h_v^*\} = \operatorname*{argmax}_{\{h_v\}} \ \mathbb{E}_{\mathcal{N}}\Big(u(\hat{\boldsymbol{v}}^s) - \alpha|\mathcal{E}^s|\Big) \tag{1a}$$

$$s.t. \quad \mathcal{G}^s = \mathcal{G} \setminus \{v | v \in \mathcal{V}, h_v(u(v)) \leq 0\}, \tag{1b}$$

$$\mathbf{w}^s = [h_v(u(v))], \ \text{for all } v \in \mathcal{V}^s, \tag{1c}$$

$$\hat{\boldsymbol{v}}^s = c(\mathcal{G}^s, \mathbf{w}^s). \tag{1d}$$

To better understand Problem 1, first notice that constraint (1b) defines the sparsified conflict graph $\mathcal{G}^s(\mathcal{V}^s, \mathcal{E}^s)$ by removing from the original conflict graph $\mathcal{G}$ those nodes with non-positive value of $h_v(u(v))$. Thus, it is immediate that the form of functions h_v has a direct influence on the level of sparsity of $\mathcal{G}^s$. Constraint (1c) defines a modified utility vector $\mathbf{w}^s$ for the nodes of the sparsified graph $\mathcal{G}^s(\mathcal{V}^s, \mathcal{E}^s)$. Furthermore, constraint (1d) determines the scheduled vertices $\hat{\boldsymbol{v}}^s$ by applying a predefined scheduler $c(\cdot)$ to the sparsified graph. In (1a) we have two competing objectives: we want to maximize the utility of the scheduled vertices while minimizing the number of edges in our sparsified graph (since these determine the message complexity of the contention process). Consequently, our objective function linearly combines both terms with a relative weighting parameter α. Finally, it should be noted that we do not want to find sparsifying functions for a specific network state but rather find functions that generalize well across a whole distribution of network states. Hence, the inclusion of an expected value in (1a).

Finding an exact solution to Problem 1 is extremely challenging for several reasons including: i) The non-differentiable scheduler $c(\cdot)$ prevents direct application of gradient-based approaches, ii) The optimization is over the space of functions, which is infinite-dimensional, and iii) The sparsification functions should be valid for a distribution of network states, possibly associated with conflict graphs of different sizes and topologies. In the next section we present our solution to Problem 1, which addresses these challenges.

3. LINK SPARSIFICATION WITH GCNS

In order to (approximately) solve Problem 1, we restrain ourselves to a parametric family of functions $\{h_v\}$ given by

$$h_v(u(v); \mathbf{Z}) = z_0(v)u(v) H\Big(z_0(v)u(v) - z_1(v)u^{(\eta)}\Big), \tag{2}$$

where $z_0(v)$ and $z_1(v)$ are parameters employed by an individual link to decide on whether or not to join the scheduling contention, $\mathbf{Z} = [\mathbf{z}_0, \mathbf{z}_1] \in \mathbb{R}^{|\mathcal{V}| \times 2}$ is a matrix collecting the link parameters $z_0(v)$ and $z_1(v)$ for all $v \in \mathcal{V}$, $H(\cdot)$ is a Heaviside (step) function,
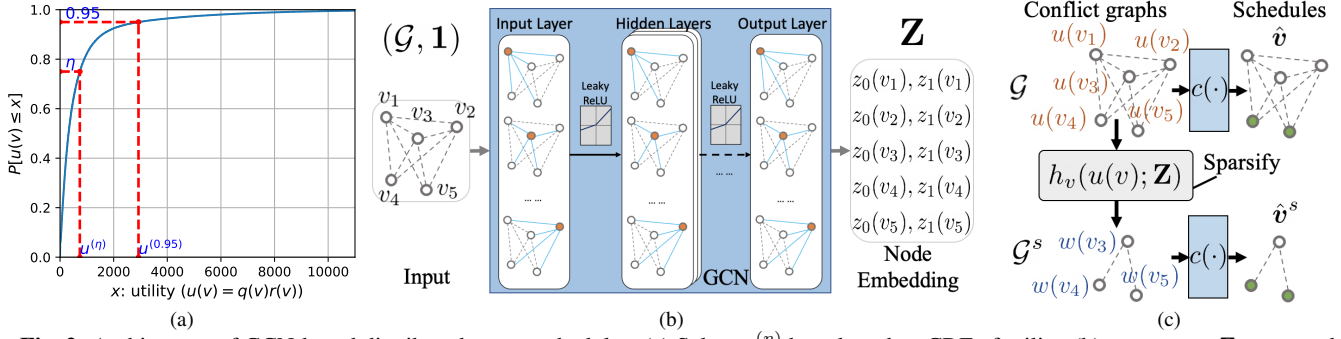
Fig. 2: Architecture of GCN-based distributed sparse scheduler: (a) Select $u^{(\eta)}$ based on the eCDF of utility, (b) parameters $\mathbf{Z}$ generated by a featureless GCN based on conflict graph $\mathcal{G}$, and (c) sparse graph $\mathcal{G}^s$ and topology-aware utility $\mathbf{w}^s$ created by the parameterized functions $\{h_v(u(v); \mathbf{Z})\}$ based on $(\mathcal{G}, \mathbf{u})$, then the distributed scheduler generates schedules $\hat{\boldsymbol{v}} = c(\mathcal{G}, \mathbf{u})$ and $\hat{\boldsymbol{v}}^s = c(\mathcal{G}^s, \mathbf{w}^s)$.

and $u^{(\eta)}$ is the η-quantile utility under network state distribution $\mathcal{N}$, as illustrated in Fig. 2(a). To better understand (2), notice that when $\mathbf{Z} = [\mathbf{1}, \mathbf{1}]$ the sparsification policy induced by h_v boils down to a baseline thresholding method. More precisely, the Heaviside function will compare the vertex utility $u(v)$ with the prescribed threshold $u^{(\eta)}$. If $u(v) \le u^{(\eta)}$, then $h_v(u(v)) = 0$ and the node is excluded from the sparse graph [cf. (1b)]. On the other hand, if $u(v)$ is larger than $u^{(\eta)}$, v is included in the sparse graph with the same utility $h_v(u(v)) = u(v)$. In this context, our proposed parameterization in (2) is a natural extension of this classical statistical baseline.

In order to define functions h_v that generalize well across different graphs $\mathcal{G}$, we define the parameters $\mathbf{Z}$ to be a function of the underlying topology. More precisely, we propose to have $\mathbf{Z} = \Psi_{\mathcal{G}}(\mathbf{1}; \boldsymbol{\omega})$, where $\Psi_{\mathcal{G}}$ is an L-layered GCN defined on the conflict graph $\mathcal{G}$, and $\boldsymbol{\omega}$ is the collection of trainable parameters of the GCN.

We define the output of an intermediate lth layer of the GCN as $\mathbf{X}^l \in \mathbb{R}^{|\mathcal{V}| \times g_l}$, and $\mathbf{X}^0 = \mathbf{1}^{|\mathcal{V}| \times 1}$, $\mathbf{Z} = \mathbf{X}^L$, then the lth layer of the GCN is expressed as:

$$\mathbf{X}^l = \sigma\left(\mathbf{X}^{l-1}\boldsymbol{\Theta}_0^l + \boldsymbol{\mathcal{L}}\mathbf{X}^{l-1}\boldsymbol{\Theta}_1^l\right), l \in \{1, \dots, L\}. \quad (3)$$

In (3), $\boldsymbol{\mathcal{L}}$ is the normalized Laplacian of $\mathcal{G}$, $\boldsymbol{\Theta}_0^l, \boldsymbol{\Theta}_1^l \in \mathbb{R}^{g_{l-1} \times g_l}$ are trainable parameters, and $\sigma(\cdot)$ is the activation function. The activation functions of the input and hidden layers are selected as leaky ReLUs whereas a linear activation is used for the output layer. The input and output dimensions are set as $g_0 = 1, g_L = 2$.

The downstream architecture of the entire distributed sparse scheduling is illustrated in Figs. 2. First, the prior knowledge of eCDF of the utility values from empirical data of network operations is collected, as shown in Fig. 2(a), based on which a global cut-off threshold is selected as $u^{(\eta)}$. Next, at the network level, the trained GCN observes the topology of the network and generates node embeddings as the local parameters $\mathbf{Z}$. Then, the sparse graph $\mathcal{G}^s$ and topology-aware utility vector $\mathbf{w}^s$ are obtained by the parameterized functions $\{h_v(u(v); \mathbf{Z})\}$ according to (1b) and (1c), respectively. Finally, the sparse schedule is obtained by the distributed scheduler as $\hat{\boldsymbol{v}}^s = c(\mathcal{G}^s, \mathbf{w}^s)$ with lower overhead.

Since $\boldsymbol{\mathcal{L}}$ in (3) is a local operator on $\mathcal{G}$, $z_0(v)$ and $z_1(v)$ can be computed in a distributed manner through neighborhood aggregation at v with L rounds of local exchanges between v and its neighbors. The local communication complexity (defined as the rounds of local exchanges between a node and its neighbors) of GCN is $\mathcal{O}(L)$. Hence, the local computational and communication costs can be controlled by modifying the number of layers L in the GCN. Notice that $\mathbf{Z}$ can be reused over time slots until the conflict graph $\mathcal{G}$ changes. Importantly, the constant local communication complexity

is a key aspect to promote scalability, whereas the reusability of $\mathbf{Z}$ is critical for overhead reduction.

To accurately control the sparsification ratio, we scale $\mathbf{z}_1$ by its expectation as $z_1(v) \leftarrow z_1(v)/\mathbb{E}_{\mathcal{N}}(\overline{z_1})$ after GCN, where $\mathbb{E}_{\mathcal{N}}(\overline{z_1})$ is the expected average of $z_1(v)$ over all $v \in \mathcal{G}$ under network state distribution $\mathcal{N}$. Since $\mathbb{E}_{\mathcal{N}}(\overline{z_1})$ can be computed offline, this operation does not introduce any additional communication complexity.

3.1. Two-Stage Training

The parameters $\boldsymbol{\omega}$ in the GCN are trained on a set of random network states $(\mathcal{G}(i), \mathbf{u}(i))$ drawn from the distribution $\mathcal{N}$. To address the conflicting objectives in (1a), we break the training into two stages of relatively low-complexity training schemes. At stage 1, we employ $u^{(0.95)}$ in (2) and the vanilla (dense) scheduler $c(\mathcal{G}(i), \mathbf{u}(i))$ as the baseline. At stage 2, we employ a random cut-off quantile $0 < \eta < 1$ in (2), and the baseline is the statistical method, $h_v(u(v); [\mathbf{1}, \mathbf{1}])$. In each stage, we run the GCN-based sparse scheduler on the training dataset and collect experience tuples $(\mathcal{G}(i), \mathbf{u}(i), \hat{\boldsymbol{v}}^s(i), \boldsymbol{v}^r(i), \boldsymbol{\rho}_0(i), \boldsymbol{\rho}_1(i))$, for $i \in \{0, \dots, N\}$. $\boldsymbol{v}^r = \{v | v \in \mathcal{V}, h_v(u(v)) \le 0\}$ is the set of removed vertices. $\boldsymbol{\rho}_0$ and $\boldsymbol{\rho}_1$ are the target vectors for $\mathbf{z}_0$ and $\mathbf{z}_1$, respectively, which capture the reward signals to maximize the objective in (1a). Vector $\boldsymbol{\rho}_0$ is defined with respect to (w.r.t.) the baseline [16, 17]:

$$\boldsymbol{\rho}_0 = \varepsilon \mathbf{v}^s + \mathbf{z}_0 \odot (\mathbf{1} - \mathbf{v}^s), \ \varepsilon = u(\hat{\boldsymbol{v}}^s)/u(\hat{\boldsymbol{v}}^b), \quad (4)$$

where ε is the approximation ratio (AR) for total utility, $\hat{\boldsymbol{v}}^b$ is the baseline schedule, and $\mathbf{v}^s$ is the indicator vector of schedule $\hat{\boldsymbol{v}}^s$. (4) encourages (discourages) schedules that are better (worse) than the baseline. Vector $\boldsymbol{\rho}_1$ for stages 1 and 2 is respectively defined as:

$$\boldsymbol{\rho}_1 = \boldsymbol{\rho}_2, \ \boldsymbol{\rho}_2 = (b/u^{(\eta)})\mathbf{z}_0 \odot \mathbf{u} \odot \mathbf{v}^r + \mathbf{z}_1 \odot (\mathbf{1} - \mathbf{v}^r), \quad (5a)$$

$$\boldsymbol{\rho}_1 = \boldsymbol{\rho}_3/\overline{\rho_3}, \ \boldsymbol{\rho}_3 = \boldsymbol{\rho}_2 - 0.2\mathbf{z}_1 \odot \mathbf{v}^s, \quad (5b)$$

where $b = 0.9\mathbb{1}(\varepsilon < \delta) + 1.1\mathbb{1}(\varepsilon \ge \delta), 0 < \delta < 1$ is the target AR for ε (e.g., $\delta = 0.97$), $\mathbf{v}^r$ is the indicator vector of set $\boldsymbol{v}^r$, and $\overline{\rho_3} = (\mathbf{1}^\top \boldsymbol{\rho}_3)/|\boldsymbol{\rho}_3|$ is the average of the elements of vector $\boldsymbol{\rho}_3$. Intuitively, (5a) encourages removing more vertices as long as the target AR for utility w.r.t. the dense scheduler is met in stage 1. In stage 2, (5b) encourages the GCN to outperform the statistical baseline on total utility, with an average threshold equal to $u^{(\eta)}$. This drives the GCN to consistently promote peripheral vertices under any η, thus reducing the edges in sparse graphs, i.e., smaller $|\mathcal{E}^s|$ in (1a). Accordingly, a root-mean-square loss is adopted to train our GCN

$$\ell(\boldsymbol{\omega}; \mathcal{G}(i), \mathbf{u}(i)) = |\mathcal{V}|^{-\frac{1}{2}} \|\mathbf{Z}(i) - [\boldsymbol{\rho}_0(i), \boldsymbol{\rho}_1(i)]\|_2. \quad (6)$$

With the loss in (6) and the collected experience tuples, we update the parameters $\boldsymbol{\omega}$ of the GCN through batch training, employing the Adam optimizer and exponentially decaying learning rates.
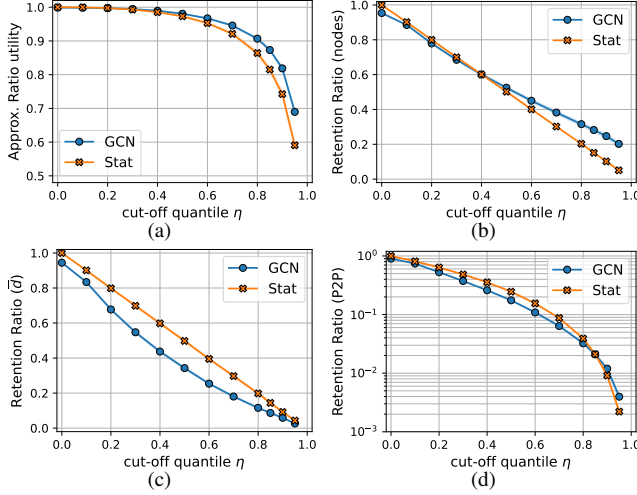
Fig. 3: The performance by cut-off quantile η, for (a) AR for total utility, (b) RR for number of vertices in $\mathcal{G}^s$, (c) RR for average degree of $\mathcal{G}^s$, and (d) RR for P2P message complexity of scheduling contention, of distributed sparse schedulers w.r.t. vanilla LGS [11] under identical input of $(\mathcal{G}, \mathbf{u})$. For (c) and (d), smaller RR is better.

4. NUMERICAL EXPERIMENTS

The GCN-based link sparsification is evaluated as a component of a sparse scheduler in synthetic random conflict graphs. The comparative baseline is the statistical link sparsification, $h_v(u(v); [\mathbf{1}, \mathbf{1}])$, described in the first paragraph of Section 3. The scheduling contention function $c(\cdot)$ is selected as the local greedy solver (LGS) in [11] for time-slotted networks, while the results can be easily extended to random access networks scheduled by weighted CSMA. The performance is presented as approximation ratio (AR) or retention ratio (RR), of which the nominator is a metric of the tested sparse scheduler and the denominator is that of the vanilla LGS applied to the original graph. The synthetic graphs for training and testing are generated from the Erdős–Rényi (ER) [33] model, which seeks to represent wireless networks with uniformly distributed users of identical omnidirectional transmit power (i.e., unit-disk interference model).

A single layer GCN ($L = 1$) is evaluated. The training settings include a batch size of 200 for experience replay, 25 epochs, and periodic gradient reset.[1] The training set comprises 5900 random graphs drawn from the ER model, including 5000 graphs of size $V = |\mathcal{V}| \in \{100, 150, 200, 250, 300\}$ and expected average degree $\bar{d} = Vp \in \{2, 5, 7.5, 10, 12.5\}$ (200 graphs per $(V, \bar{d})$), and 900 graphs of size $V \in \{30, 100\}$ and probability of edge-appearance $p \in \{0.1, 0.2, \ldots, 0.9\}$ (50 graphs per (V, p)). The utility values are drawn from an empirical distribution (Fig. 2(a)), collected from simulation of scheduling on the synthetic conflict graph based on vanilla LGS and a utility function of $u(v) = q(v)r(v)$, where $q(v)$ and $r(v)$ are the queue length and link rate (defined as the number of packets that can be transmitted in a time slot) on link v. A 1-hop flow is generated for each node in the conflict graph. The exogenous packets at each source user follow a Poisson arrival with a prescribed arrival rate λ. Our training and testing traffic load, defined as $\mu = \lambda/\mathbb{E}(\mathbf{r})$, is set to be unsaturated as $\mu \in [0.03, 0.05]$. The link rate $r(v)$ is drawn from a normal distribution $\mathbb{N}(50, 25)$ independently across time slots and links, and clipped to $[0, 100]$, to capture a wireless link based on constant transmit power and fading channel with log-

---

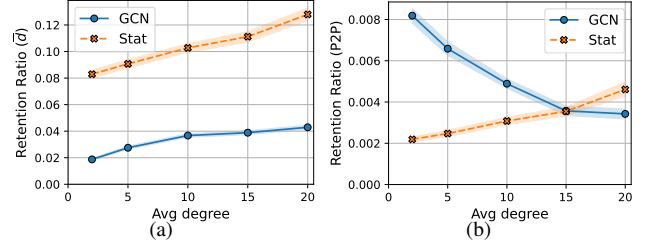[1]Training takes 2-3 hours on a workstation with a specification of 16GB memory, 8 cores, and Geforce GTX 1070 GPU. The source code is published at `https://github.com/zhongyuanzhao/gcn-sparsify`

Fig. 4: The RRs by the average degree of original graph $\mathcal{G}$, for (a) average degree of $\mathcal{G}^s$, and (b) P2P message complexity of scheduling contention, of distributed sparse schedulers w.r.t. the vanilla LGS [11] in 500 scheduling instances with ER graphs, traffic load $\mu \in [0.03, 0.05]$, $\eta = 0.95$, and 300 time slots. Smaller RR is better.

normal shadowing [34]. Our test set consists of 500 graphs drawn from the ER model with parameters of $V \in \{100, 150, \ldots, 300\}$, $\bar{d} \in \{2, 5, 10, 15, 20\}$, and 20 instances per $(V, \bar{d})$.

Our first experiment compares the GCN-based link sparsification with the baseline under identical inputs of conflict graph and realization of random utilities. Each graph $\mathcal{G}(i)$ is tested with 12 cut-off quantiles $\eta \in \{0, 0.1, \ldots, 0.8, 0.85, 0.9, 0.95\}$, where a realization of $\mathbf{u}$ is drawn from the collected empirical distribution for each $(\mathcal{G}(i), \eta)$. The performances of the GCN and the baseline are compared in Fig. 3, as the ARs of achieved total utility (Fig. 3(a)), number of vertices in the sparse conflict graph $\mathcal{G}^s$ (Fig. 3(b)), average degree of $\mathcal{G}^s$ (Fig. 3(c)), and point-to-point (P2P) message complexity of scheduling contention (Fig. 3(d)). Both the baseline (Stat) and GCN can significantly reduce the network-wide message complexity to less than 0.5% while keep $59 \sim 69\%$ of total capacity. Compared to the baseline, GCN-based sparsification can achieve higher total utility with fewer message exchanges or collision rate (average degree of $\mathcal{G}^s$) by leveraging the topological information.

Next, we evaluate the GCN and the baseline in a scheduling experiment with $\eta = 0.95$ and 500 scheduling instances. Each instance contains a conflict graph from the ER test set, and realizations of random arrivals and link rates for $T = 300$ time slots. Even for the same scheduling instance, the tested schedulers generally have different input $\mathbf{u}(t)$ due to the dependency between network states and scheduling decisions. With unsaturated traffic of $\mu \in [0.03, 0.05]$, all three schedulers (vanilla LGS, GCN-based and statistical sparse schedulers) achieve the same long-term throughput. The ARs for the average degree and P2P message complexity of scheduling contention in the sparse graph $\mathcal{G}^s$ are presented in Figs. 4(a) and 4(b), respectively. The average degree of $\mathcal{G}^s$ can be reduced to $8.2 \sim 12.6\%$ of that of $\mathcal{G}$ by the baseline, and further down to $1.9 \sim 4.2\%$ by the GCN. Although the baseline has lower average P2P message complexity than the GCN, as shown in Fig. 3(d) ($\eta = 0.95$) and Fig. 4(b), their trends w.r.t. the graph density are opposite, and the GCN works better on denser graphs ($\bar{d} = 20$). These results show that GCN is superior to the baseline in reducing the scheduling overhead.

5. CONCLUSIONS

We presented a GCN-based distributed link sparsification scheme to reduce the scheduling overhead in wireless networks with orthogonal access. The GCN can encode the network topology into a parameterized local decision process to keep a link from contending if its chance of winning is low. The proposed scheme can significantly reduce the scheduling overhead for time-slotted and random access networks while retaining most of the network capacity. Thus, our approach can improve access, energy efficiency, and radio footprint of wireless multi-hop networks.

6. REFERENCES

[1] "Cisco annual internet report (2018–2023)," white paper, Cisco Systems, Inc., Mar. 2020.

[2] A. Kott, A. Swami, and B. J. West, "The internet of battle things," *Computer*, vol. 49, no. 12, pp. 70–75, 2016.

[3] I. F. Akyildiz, A. Kak, and S. Nie, "6G and beyond: The future of wireless communications systems," *IEEE Access*, vol. 8, pp. 133995–134030, 2020.

[4] X. Chen, D. W. K. Ng, W. Yu, E. G. Larsson, N. Al-Dhahir, and R. Schober, "Massive access for 5g and beyond," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 3, pp. 615–637, 2021.

[5] X. Lin, N. B. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1452–1463, 2006.

[6] S. K. Sarkar, T. G. Basavaraju, and C. Puttamadappa, *Ad hoc mobile wireless networks: principles, protocols and applications (2nd Ed)*. CRC Press, 2013.

[7] C. Joo, X. Lin, and N. B. Shroff, "Understanding the capacity region of the greedy maximal scheduling algorithm in multihop wireless networks," *IEEE/ACM Trans. Netw.*, vol. 17, no. 4, pp. 1132–1145, 2009.

[8] A. G. Marques, N. Gatsis, and G. B. Giannakis, "Optimal cross-layer design of wireless fading multi-hop networks," in *Cross Layer Designs in WLAN Systems* (N. Zorba, C. Skianis, and C. Verikoukis, eds.), Leicester, UK: Troubador Pub, 2011.

[9] S. Basagni, "Finding a maximal weighted independent set in wireless networks," *Telecomm. Systems*, vol. 18, no. 1-3, pp. 155–168, 2001.

[10] W. Cheng, X. Cheng, T. Znati, X. Lu, and Z. Lu, "The complexity of channel scheduling in multi-radio multi-channel wireless networks," in *IEEE Intl. Conf. on Computer Comms. (INFOCOM)*, pp. 1512–1520, 2009.

[11] C. Joo and N. B. Shroff, "Local greedy approximation for scheduling in multihop wireless networks," *IEEE Trans. on Mobile Computing*, vol. 11, no. 3, pp. 414–426, 2012.

[12] C. Joo, X. Lin, J. Ryu, and N. B. Shroff, "Distributed greedy approximation to maximum weighted independent set for scheduling with fading channels," *IEEE/ACM Trans. Netw.*, vol. 24, no. 3, pp. 1476–1488, 2015.

[13] P. Du and Y. Zhang, "A new distributed approximation algorithm for the maximum weight independent set problem," *Mathematical Problems in Engineering*, vol. 2016, 2016.

[14] I. C. Paschalidis, F. Huang, and W. Lai, "A message-passing algorithm for wireless network scheduling," *IEEE/ACM Trans. Netw.*, vol. 23, p. 1528–1541, Oct. 2015.

[15] C. Joo, G. Sharma, N. B. Shroff, and R. R. Mazumdar, "On the complexity of scheduling in wireless networks," *EURASIP Journal on Wireless Commun. and Netw.*, vol. 2010, no. 1, p. 418934, 2010.

[16] Z. Zhao, G. Verma, C. Rao, A. Swami, and S. Segarra, "Distributed scheduling using graph neural networks," in *IEEE Int. Conf. on Acoustics, Speech and Signal Process. (ICASSP)*, pp. 4720–4724, 2021.

[17] Z. Zhao, G. Verma, C. Rao, A. Swami, and S. Segarra, "Link scheduling using graph neural networks," in *submitted to IEEE J. Sel. Topics Signal Process., arXiv:2109.05536*, 2021.

[18] J. Ni, B. Tan, and R. Srikant, "Q-CSMA: Queue-length based CSMA/CA algorithms for achieving maximum throughput and low delay in wireless networks," in *IEEE Intl. Conf. on Computer Comms. (INFOCOM)*, pp. 1–5, 2010.

[19] L. Jiang and J. Walrand, "A distributed CSMA algorithm for throughput and utility maximization in wireless networks," *IEEE/ACM Trans. Netw.*, vol. 18, no. 3, pp. 960–972, 2010.

[20] E. Testi and A. Giorgetti, "Blind wireless network topology inference," *IEEE Trans. Commun.*, vol. 69, no. 2, pp. 1109–1120, 2021.

[21] W. Ye, J. Heidemann, and D. Estrin, "Medium access control with coordinated adaptive sleeping for wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 12, no. 3, pp. 493–506, 2004.

[22] P. Santi, "Topology control in wireless ad hoc and sensor networks," *ACM Comput. Surv.*, vol. 37, p. 164–194, June 2005.

[23] R. Ramanathan, "Antenna beamforming and power control for ad hoc networks," in *Mobile Ad Hoc Networking* (S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic, eds.), pp. 139–173, Hoboken, NJ, USA: John Wiley & Sons, Ltd, 2004.

[24] N. K. Ray and A. K. Turuk, "A hybrid energy efficient protocol for mobile ad hoc networks," *Journal of computer netw. and commun.*, vol. 2016, 2016.

[25] S. Guha, P. B. Basu, C.-K. C. Chau, and R. Gibbens, "Green wave sleep scheduling: Optimizing latency and throughput in duty cycling wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 8, pp. 1595–1604, 2011.

[26] J. Long and O. Büyüköztürk, "Collaborative duty cycling strategies in energy harvesting sensor networks," *Comput.-Aided Civ. Infrastruct. Eng.*, vol. 35, p. 534–548, May 2020.

[27] L. Lin, X. Lin, and N. B. Shroff, "Low-complexity and distributed energy minimization in multihop wireless networks," *IEEE/ACM Trans. Netw.*, vol. 18, no. 2, pp. 501–514, 2010.

[28] X. Xiang, C. Lin, and X. Chen, "Energy-efficient link selection and transmission scheduling in mobile cloud computing," *IEEE Commun. Lett.*, vol. 3, no. 2, pp. 153–156, 2014.

[29] W. Wu, W. Xu, Z. Chen, and M. Yang, "Energy-efficient link scheduling in time-variant dual-hop 60GHz wireless networks," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 23, p. e5903, 2020.

[30] J. Li, C. Blake, D. S. De Couto, H. I. Lee, and R. Morris, "Capacity of ad hoc wireless networks," in *IEEE Intl. Conf. on Mobile Computing And Netw. (MobiCom)*, pp. 61–69, 2001.

[31] L. Hai, Q. Gao, J. Wang, H. Zhuang, and P. Wang, "Delay-optimal back-pressure routing algorithm for multihop wireless networks," *IEEE Trans. Vehicular Tech.*, vol. 67, no. 3, pp. 2617–2630, 2018.

[32] J. Yang, S. C. Draper, and R. Nowak, "Learning the interference graph of a wireless network," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 3, no. 3, pp. 631–646, 2016.

[33] P. Erdős and A. Rényi, "On random graphs I," *Publ. Math. Debrecen 6*, pp. 290–297, 1959.

[34] H. Mousavi, I. S. Amiri, M. Mostafavi, and C. Choon, "LTE physical layer: Performance analysis and evaluation," *Applied Computing and Informatics*, vol. 15, no. 1, pp. 34 – 44, 2019.