ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10447843

STATE-AUGMENTED INFORMATION ROUTING IN COMMUNICATION SYSTEMS WITH GRAPH NEURAL NETWORKS

Sourajit Das[⋆] *Navid NaderiAlizadeh*[†] *Alejandro Ribeiro*[⋆]

[⋆]University of Pennsylvania
[†]Duke University

ABSTRACT

We consider the problem of routing network packets in a large-scale communication system where the nodes have access to only local information. We formulate this problem as a constrained learning problem, which can be solved using a distributed optimization algorithm. We approach this distributed optimization using a novel state-augmentation (SA) strategy to maximize the aggregate information packets at different source nodes, leveraging dual variables corresponding to flow constraint violations. The construction is based on graph neural networks (GNNs) that employ graph convolutions over the underlying communication network topology. We devise an unsupervised learning algorithm to transform the output of the GNN architecture into optimal routing decisions. The proposed method takes advantage of only the local information available at each node and efficiently routes the desired packets to the destination. We provide numerical results demonstrating the superiority of the proposed method over baseline routing algorithms.

***Index Terms*—** Routing, distributed optimization, graph neural networks, unsupervised learning, Lagrangian duality.

1. INTRODUCTION

Telecommunication technologies have seen a staggering rise in the requirement for large-scale intelligent systems and smart devices, both wired and wireless. With the outspread adoption of 5th generation (5G) communication systems, the next generation of communication technology forages the need for judicious use of accessible resources and delivery of services to an extremely greater precision. The research in telecommunication networks spans a wide range of complex challenges and network control targets, which could significantly impact the communication performance. There has been notable prior work to address such challenges in areas including stochastic network utility maximization (NUM) [1], radio resource management (RRM) [2, 3, 4], routing [5, 6, 7], and scheduling [8, 9, 10]. These methods are generally formulated as constrained optimization problems for a utility function, while taking into account the stochastic dynamics of user traffic and fading channel realizations.

Over the past few years, the onset of artificial intelligence (AI) and machine learning (ML) has been instrumental in mitigating the challenges of different communication networks with superior performance as compared to traditional methods. A comprehensive survey by Mao *et al.* highlights different ML algorithms in intelligent wireless networks that improve various network functions, such as traffic balancing, resource allocation, routing layer path search, etc. [11]. Most of the ML-based approaches in the communication networking literature require large training datasets to train neural networks that imitate "ground-truth" heuristic solutions that may be suboptimal. This limits the capability of these methods to exceed the performance of the heuristic algorithms using which they have been trained. Unsupervised learning is an alternative to solve problems in communication networks, treating such problems as statistical regression and then solving the resulting optimization problems directly instead of depending on explicit training sets [12, 13, 14, 15, 16].

One of the architectures to solving such unsupervised learning called called fully connected neural networks (FCNNs), provide universal approximation properties and scalability of the message signals in time and space has enabled CNNs to solve networks routing problems in the recent days [17, 12, 18, 19, 20]. Unfortunately, CNNs do not generalize well enough to large-scale networks which comprises of large network graphs with numerous system parameters. Thus we resort to Graph Neural Networks (GNNs) in our paper to address the bottlenecks of FCNNs and leverage the benefits such as scalability, transferability and permutation invariance [21].

In this paper, we consider a multi-node communication system at the network layer, where packets are generated at different nodes and tagged for delivery to assigned destination nodes. In addition to each packet being locally generated at a given node, each node also handles packets that it receives from its neighbors. This leads to a routing problem, where the objective of each node is to evaluate the next hop of suitable neighbors for each flow (where each flow comprises the set of packets targeted to a specific destination node) to ensure the successful delivery of the packets to the destination node. Similarly to [2, 3, 4, 6], we formulate the aforementioned routing problem as the maximization of a network utility function subject to a set of constraints. The objective function represents the aggregate amount of information generated at all nodes and for all flows. The constraints that the routing decisions need to satisfy, on the other hand, are two-fold: i) *flow* constraints, ensuring that each node has no fewer *outgoing* packets than *incoming* ones for each flow, and ii) *capacity* constraints, where the total number of packets routed on each link across the network is bounded by the link's capacity. The flow constraints, in particular, are pivotal to guarantee network stability by forcing the queue lengths to be stabilized over time.

In general, one can solve such problems by using a standard dual descent algorithm to reach optimal solutions, but they generally do not guarantee the satisfaction of the problem constraints. In this work, we propose an unsupervised *state-augmented* learning approach to ensure the feasibility of the resulting routing decisions. We leverage the fact that dual variables in constrained optimization reflect the degree of constraint violation or satisfaction over a given time period [22]. We utilize the above concept to augment the state of the communication system with dual variables corresponding to the flow constraints, to be used as dynamic inputs to the routing policy. We numerically verify that the proposed approach allows the routing policy to adapt its decisions to the instantaneous channel states, as well as constraint violations or satisfactions over time.

2. PROBLEM STATEMENT

We consider a communication network, modeled as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with a set of nodes, denoted by $\mathcal{V}$, and a set of edges or links between the nodes, denoted by $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. The capacity of a link $(i, j) \in \mathcal{E}$ is denoted as C_{ij} and the neighborhood of every node $i \in \mathcal{V}$ is defined as the set $\mathcal{N}_i = \{j \in \mathcal{V} | (i, j) \in \mathcal{E}\}$ of nodes j that can communicate directly to node i. The nodes communicate with each other by exchanging packets over different flows. We use $\mathcal{K}$ to represent the set of $F = |\mathcal{K}|$ information flows with the destination of flow $k \in \mathcal{K}$ being the node $o_k \in \mathcal{V}$. At any given time step t, node $i \neq o_k$ generates a certain number of packets at random, represented by $A_i^k(t)$, which is supposed to be delivered to destination o_k. We assume that the random variable(s), $A_i^k(t)$ are independent and identically distributed (*i.i.d.*) across time with the expected value $\mathbb{E}[A_i^k(t)] = A_i^k$. Within the same time instant, node i routes $r_{ij}^k(t)$ units of information packets to every neighboring node $j \in \mathcal{N}_i$ and also receives $r_{ji}^k(t)$ packets from neighboring node j. The difference between the total number of received packets $A_i^k(t) + \sum_{j\in\mathcal{N}_i} r_{ji}^k(t)$ and the total number of transmitted packets $\sum_{j\in\mathcal{N}_i} r_{ij}^k(t)$ is added to the local queue (or subtracted if the resultant quantity is negative). Therefore, we can iteratively define the queue length of the packets for flow k at node i, according to the following equation,

$$q_i^k(t+1) = \Big[q_i^k(t) + A_i^k(t) + \sum_{j\in\mathcal{N}_i} r_{ji}^k(t) - r_{ij}^k(t)\Big]^+, \quad (1)$$

where we add a projection onto the non-negative orthant to ensure the non-negativity of the number of packets in the queue. We need to account that (1) is expressed for all nodes $i \in \mathcal{V} \setminus \{o_k\}$, as packets routed to their final destination will be discarded from the network.

2.1. Problem Formulation

We consider the operation of the aforementioned communication network over a series of time instants $t \in \{0, 1, 2, .., T-1\}$, where at each time instant t, the set of channel capacities, or the *network state*, is represented as $\mathbf{C}_t \in \mathcal{C}$. Let $\mathbf{p}(\mathbf{C}_t)$ denote the vector of routing decisions across the above network, where $\mathbf{p} : \mathcal{C} \to \mathbb{R}^{n\times n\times F}$ denotes the routing function. These routing decisions ultimately give rise to a generic network performance vector $\mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t)) \in \mathbb{R}^b$, where $\mathbf{f} : \mathcal{C} \times \mathbb{R}^{n\times n\times F} \to \mathbb{R}^b$ represents the performance function.

We consider a concave utility function $\mathcal{U} : \mathbb{R}^b \to \mathbb{R}$ subject to a set of concave constraints $\mathbf{g} : \mathbb{R}^b \to \mathbb{R}^c$. We define the generic routing problem as follows,

$$\max_{\{\mathbf{p}(\mathbf{C}_t)\}_{t=0}^{T-1}} \quad \mathcal{U}\left(\frac{1}{T}\sum_{t=0}^{T-1} \mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t))\right) \quad (2a)$$

$$s.t. \quad \mathbf{g}\left(\frac{1}{T}\sum_{t=0}^{T-1} \mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t))\right) \geq 0. \quad (2b)$$

Note, the objective function as well as the constraints are derived based on the *ergodic average* of the network performance, *i.e.*, $\frac{1}{T}\sum_{t=0}^{T-1} \mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t))$. Thus, the goal of the routing algorithm is to determine the optimal vector of routing decisions $\mathbf{p}(\mathbf{C}_t)$ for any given network state $\mathbf{C}_t \in \mathcal{C}$ over time.

For the routing problem under consideration, we introduce the auxiliary optimization variables $a_i^k(t) \geq A_i^k(t)$ to ensure we generate as many packets at node i for flow k as possible, but, in reality, $A_i^k(t)$ is the actual number of packets present in the system, which is used to update the queue length defined in (1). To maximize the number of information packets generated at all nodes and over all flows, we define our objective as,

$$\mathcal{U}\left(\frac{1}{T}\sum_{t=0}^{T-1} \mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t))\right) = \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{V}} \log\left(\frac{1}{T}\sum_{t=0}^{T-1} a_i^k(t)\right). \quad (3)$$

We optimize the above objective function subject to three sets of constraints. To stabilize the queue lengths over time, the first set of constraints states that the sum of the total number of received packets and the locally-generated packets should be less than or equal to the total number of transmitted packets. More precisely, these *routing constraints* or *flow constraints* can be defined as,

$$a_i^k(t) + \sum_{j\in\mathcal{N}_i} r_{ji}^k(t) \leq \sum_{j\in\mathcal{N}_i} r_{ij}^k(t), \ \forall k \in \mathcal{K}, \forall i \in \mathcal{V}. \quad (4)$$

The second set of constraints, called the *capacity constraints*, ensure that the sum of the total number of packets transmitted over a link does *not* exceed the capacity of the channel, i.e.,

$$\sum_{k\in K} r_{ij}^k(t) \leq C_{ij}, \ \forall (i, j) \in \mathcal{E}. \quad (5)$$

Plugging (3)-(5) into the generic formulation of (2), alongside the third set of constraints on the auxiliary variables $a_i^k(t)$, the routing optimization problem can be written as,

$$\max_{\{a_i^k(t), r_{ij}^k(t)\}_{t=0}^{T-1}} \quad \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{V}} \log\left(\frac{1}{T}\sum_{t=0}^{T-1} a_i^k(t)\right), \quad (6a)$$

$$\sum_{j\in\mathcal{N}_i} r_{ij}^k(t) - \sum_{j\in\mathcal{N}_i} r_{ji}^k(t) - a_i^k(t) \geq 0, \ \forall k \in \mathcal{K}, \forall i \in \mathcal{V}, \quad (6b)$$

$$C_{ij}(t) - \sum_{k\in K} r_{ij}^k(t) \geq 0, \ \forall (i, j) \in \mathcal{E}, \quad (6c)$$

$$a_i^k(t) - A_i^k(t) \geq 0, \ \forall k \in \mathcal{K}, \forall i \in \mathcal{V}. \quad (6d)$$

3. AUGMENTED LAGRANGIAN & METHOD OF MULTIPLIERS

Generally, one can solve for (6) by formulating the Lagrangian and using a dual descent technique, where the Lagrangian is maximized over the primal variables and minimized over the dual variables. Unfortunately, such methods are known to have shortcomings, with the most prominent being its extremely slow convergence rate and the need for strictly convex objective functions [23, 24, 25]. Hence, we resort to the Augmented Lagrangian or the Method of Multipliers (MoM) to mitigate the above shortcomings of dual descent methods. To that end, we first convert the inequality constraint (6b) to an equality constraint as shown below,

$$\sum_{j\in\mathcal{N}_i} r_{ij}^k(t) - \sum_{j\in\mathcal{N}_i} r_{ji}^k(t) - a_i^k(t) - z_i^k(t) = 0 \quad (7)$$

where $z_i^k \geq 0$ is an auxiliary variable. The Augmented Lagrangian for the optimization problem in (6) can now be written as

$$\mathcal{L}_\rho(\boldsymbol{a},\boldsymbol{r},\boldsymbol{z},\boldsymbol{\mu}) = \sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{V}} \log\left(\frac{1}{T}\sum_{t=0}^{T-1} a_i^k(t)\right)$$
$$+\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{V}}\frac{1}{T}\sum_{t=0}^{T-1}\Bigg\{\mu_i^k\Bigg(\sum_{j\in\mathcal{N}_i} r_{ij}^k(t)-\sum_{j\in\mathcal{N}_i} r_{ji}^k(t)-a_i^k(t)-z_i^k(t)\Bigg)$$
$$+\frac{\rho}{2}\Bigg\|\sum_{j\in\mathcal{N}_i} r_{ij}^k(t) - \sum_{j\in\mathcal{N}_i} r_{ji}^k(t) - a_i^k(t) - z_i^k(t)\Bigg\|^2\Bigg\} \quad (8)$$

where $\rho > 0$ is a penalty parameter and $\boldsymbol{\mu} \in \mathbb{R}_+^{n\times F}$ is the dual multiplier associated with the constraint (6b). We consider the constraints (6c) and (6d) implicit during the optimization, i.e., we force our solutions to the optimization problem to automatically satisfy them. MoM solves the above optimization problem for different values of $\boldsymbol{\mu}$ and ρ. The optimization algorithm starts by first maximizing $\mathcal{L}_\rho(\boldsymbol{a},\boldsymbol{r},\boldsymbol{z},\boldsymbol{\mu})$ with respect to $\boldsymbol{z}$, *i.e.*,

$$\mathcal{L}_\rho(\boldsymbol{a},\boldsymbol{r},\boldsymbol{\mu}) = \max_{\boldsymbol{z}} \mathcal{L}_\rho(\boldsymbol{a},\boldsymbol{r},\boldsymbol{z},\boldsymbol{\mu}) \quad (9)$$

The successive step proceeds by maximizing $\mathcal{L}_\rho(\boldsymbol{a},\boldsymbol{r},\boldsymbol{\mu})$ with respect to the primal variables $(\boldsymbol{a},\boldsymbol{r})$. Once the primal variables are updated, the subsequent step follows the update of dual variables. Upon convergence for the sequence $(a_i^k)^n$, the dual variable converges to $\boldsymbol{\mu}^*$, *i.e.*

$$\left[(\mu_i^k)^n - \rho^n\Bigg(\sum_{j\in\mathcal{N}_i} r_{ij}^k - \sum_{j\in\mathcal{N}_i} r_{ji}^k - (a_i^k)^n\Bigg)\right]^+ \to (\mu_i^k)^* \quad (10)$$

The above algorithm is run for a sufficient number of iterations and reaches convergence when (9) has an optimal solution irrespective of the initialization. A drawback of MoM is that it loses decomposability and efficiency in the case of complex distributed systems. Moreover, such optimization is hard to implement in practice, since it includes an *infinite-dimensional* space (as the number of primal variables grows with the number of time steps). Thus, we opt for a *parameterized* routing model, which we discuss next.

4. GNN PARAMETERIZATION WITH AUGMENTED LAGRANGIAN: METHOD OF MULTIPLIERS

Graph neural networks (GNNs), which can be treated as a generalization of convolutional neural networks (CNN), are special architectures developed to operate on graph data or signals [26, 27]. We define a graph filter,where the network state $\mathbf{C}_t$ acts as the graph shift operator (GSO). At each time instant t, let the graph nodes be equipped with an initial feature matrix, denoted by $\mathbf{Z}_t^0 \in \mathbb{R}^{|\mathcal{V}|\times F_0}$. The GNN processes the input node features and the GSO using L layers, where the intermediate layers are passed through a pointwise non-linearity function to give the output at the l^{th} layer,

$$\mathbf{Z}_t^l = \sigma\left[\sum_{k=0}^{K-1} \mathbf{C}_t^k\, \mathbf{Z}_t^{l-1}\, \mathbf{H}_{lk}\right], \quad (11)$$

where $\mathbf{H}_{lk} \in \mathbb{R}^{F_{l-1}\times F_l}$ denotes the set of graph filter coefficients, i.e., the GNN parameters, at the l^{th} layer.

We use the multiple-input multiple-output (MIMO) graph filter in (11) recursively to create the GNN network for our optimization problem. If the filter has L layers, the GNN operator can be given as,

$$\boldsymbol{\Psi}(\mathbf{C}_t, \mathbf{Z}_t; \mathbf{H}) = \mathbf{Z}_t^L \in \mathbb{R}^{|\mathcal{V}|\times F_L}, \quad (12)$$

where $\mathbf{H} = \{\mathbf{H}_{lk}\}_{l=1,k=0}^{L,K-1}$. Here $\mathbf{Z}_t = \mathbf{Z}_t^0 = A_i^k(t)$ is used as the single input feature to the GNN. The output of the GNN is then multiplied by an intermediate square matrix $\boldsymbol{W}_r \in \mathbb{R}^{F_L\times F_L}$ to obtain the routing decisions $\mathbf{p}(\mathbf{C}_t, \mathbf{Z}_t; \boldsymbol{\phi})$. In order to ensure the entries of the routing decision matrix are between 0 and 1, we pass the resulting matrix through a Softmax function, i.e., $\mathbf{p}(\mathbf{C}_t, \mathbf{Z}_t) = \text{Softmax}(\mathbf{Z}_t^L\, \boldsymbol{W}_r\, \mathbf{Z}_t^{L^T})$. Similarly, we multiply the output of GNN with a column vector $\boldsymbol{W}_a \in \mathbb{R}^{F_L\times 1}$ followed by a ReLU non-linearity to obtain $a_i^k(t)$. Using the GNN parameterization described above in (11), we can reformulate the *parameterized* optimization problem in (6) as,

$$\max_{\phi, w_r, w_a} \quad \mathcal{U}\left(\frac{1}{T}\sum_{t=0}^{T-1} \mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t;\boldsymbol{\phi}))\right) \quad (13a)$$
$$s.t. \quad \mathbf{g}\left(\frac{1}{T}\sum_{t=0}^{T-1} \mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t;\boldsymbol{\phi}))\right) \geq 0, \quad (13b)$$

where $\boldsymbol{\phi} = \{\mathbf{H}, \boldsymbol{W}_r, \boldsymbol{W}_a\}$ is the collection of all the GNN parameters. Observe that the maximization is now happening over the GNN filter tensor ϕ. The augmented Lagrangian for the problem in (13) can be written as,

$$\mathcal{L}(\boldsymbol{\phi},\boldsymbol{\mu}) = \mathcal{U}\left(\frac{1}{T}\sum_{t=0}^{T-1} \mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}, \mathbf{x}_t;\boldsymbol{\phi}))\right)$$
$$+\boldsymbol{\mu}^T\mathbf{g}\left(\frac{1}{T}\sum_{t=0}^{T-1} \mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t;\boldsymbol{\phi}))\right)$$
$$+\frac{\rho}{2}\left\|\mathbf{g}\left(\frac{1}{T}\sum_{t=0}^{T-1} \mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t;\boldsymbol{\phi}))\right)\right\|^2. \quad (14)$$

Now we introduce an iteration duration T_0, which is the number of time steps between two consecutive model parameter updates. Using a slight abuse of notation for time t, we define an iteration index $m \in \{0, 1, 2, ..., M-1\}$, where $M = \lfloor T/T_0 \rfloor$ and model parameter, ϕ is updated as follows,

$$\boldsymbol{\phi}_m = \arg\max_{\boldsymbol{\phi}\in\boldsymbol{\Phi}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\mu}_m). \quad (15)$$

Subsequently the dual variables, $\boldsymbol{\mu}$ are updated recursively as

$$\boldsymbol{\mu}_{m+1} = \left[\boldsymbol{\mu}_m - \rho\, \mathbf{g}\left(\frac{1}{T}\sum_{t=mT_0}^{(m+1)T_0-1} \mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t;\boldsymbol{\phi}))\right)\right]^+ \quad (16)$$

Note that the penalty parameter ρ, serves as the learning rate for the dual variable update. The above parameterized augmented Lagrangian has an edge over regular dual descent algorithms like Primal-Dual methods since the parameter updates in (15), (16) adapts the routing policy to the dual variables for each iteration.

5. THE STATE AUGMENTATION ALGORITHM

The caveat with the iterative optimization in (15), (16) is that it becomes infeasible in practice, since the optimal set of GNN parameters at each time step needs to be recalculated for different vectors of dual variables $\boldsymbol{\mu}_m$ [4]. In light of the above challenges, we propose a *state augmentation* algorithm, where the key idea lies in augmenting the network state $\mathbf{C}_t$ at each time step t with the corresponding set of dual variables $\boldsymbol{\mu}_{\lfloor t/T_0 \rfloor}$. The dual variables are fed as input in parallel with the input node features to the GNN model, $A_i^k(t)$ to obtain the optimal routing decisions. We consider a separate parameterization for the state augmented routing policy, where the routing decisions $\mathbf{p}(\mathbf{C}_t)$ are represented as $\mathbf{p}^{\boldsymbol{\theta}}(\mathbf{C}_t, \mathbf{x}_t; \boldsymbol{\theta})$, and $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ is the set of GNN filters tensors parameterized by the state-augmented routing algorithm. Next, we assume a batch of dual variables to be drawn from a probability distribution $p_{\boldsymbol{\mu}}$. Substituting the new parameterization $\boldsymbol{\theta}$ in (14), the augmented Lagrangian can be rewritten as, (13) can be written as,

$$\mathcal{L}_{\boldsymbol{\mu}}(\boldsymbol{\theta}) = \mathcal{U}\left(\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}, \mathbf{x}_t; \boldsymbol{\theta}))\right) + \boldsymbol{\mu}^T\mathbf{g}\left(\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t; \boldsymbol{\theta}))\right) + \frac{\rho}{2}\left\|\mathbf{g}\left(\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{f}(\mathbf{C}_t, \mathbf{p}(\mathbf{C}_t; \boldsymbol{\theta}))\right)\right\|^2. \quad (17)$$

We define the state-augmented routing policy as the one that maximizes the expectation of the augmented Lagrangian over the probability distribution of all dual variables, *i.e.*,

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathbb{E}_{\boldsymbol{\mu}\sim p_{\boldsymbol{\mu}}}\left[\mathcal{L}_{\boldsymbol{\mu}}(\boldsymbol{\theta})\right]. \quad (18)$$

The state augmented policy in (18) parameterized by $\boldsymbol{\theta}^*$ allows us to determine the Lagrangian maximized routing decision $\mathbf{p}(\mathbf{C}, \mathbf{x}_t; \boldsymbol{\theta})$ for every dual variable $\boldsymbol{\mu}_m$. This optimization in (18) is presumed to be carried out in the offline training phase. During the training phase, we randomly sample a batch of dual variables, $\{\boldsymbol{\mu}_b\}_{b=1}^B$ from the probability distribution $p_{\boldsymbol{\mu}}$ and opt for a gradient ascent technique to learn the optimal set of parameters θ^*. With a random initialization of θ_0, the model parameters can be updated over the iteration index $n = 0, ..., N-1$ as

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n + \frac{\eta_\theta}{B}\sum_{b=1}^{B-1}\nabla_\theta \mathcal{L}_{\boldsymbol{\mu}}(\boldsymbol{\theta}_n), \quad (19)$$

where η_θ denotes the learning rate for the model parameters θ. The update of the primal variables $(\boldsymbol{a}, \boldsymbol{r})$ is invoked by the GNN after the model parameters are updated using (19). The final set of converged parameters can be stored as $(\boldsymbol{\theta}^*)$ upon completion of training the model. When we enter the execution phase, the dual variable update for the iteration m in (18) can be updated as below.

$$\boldsymbol{\mu}_{m+1} = \left[\boldsymbol{\mu}_m - \eta_{\boldsymbol{\mu}}\mathbf{g}\left(\frac{1}{T_0}\sum_{t=mT_0}^{(m+1)T_0-1}\mathbf{f}(\mathbf{C}_t, \mathbf{p}^{\boldsymbol{\theta}}(\mathbf{C}_t; \boldsymbol{\theta}^*))\right)\right]^+, \quad (20)$$

The above dual update enables us to learn a parameterized model in parallel while solving for the optimal state-augmented routing policy in (18). The trained model above can be generalized further for every set of dual variable $\boldsymbol{\mu}_b$ in the batch by randomly sampling different realizations of the network state $\{\mathbf{C}_{b,t}\}_{t=0}^{T-1}$.

6. NUMERICAL RESULTS

We consider graphs with $N = |\mathcal{V}|$ nodes, where the graph topology is created using the k-Nearest Neighbor (k-NN) method. The node locations are generated uniformly at random inside the unit circle. We consider the following values for our experiments: $k = 4$, $T = 100$ and $T_0 = 5$. We consider the capacity of all existing links $(i,j) \in \mathcal{E}$ to be $C_{ij}(t) = 10, \forall t$. We use a 3-layer GNN with $F_0 = 2, F_1 = 32$, and $F_2 = 8$ features. We use the ADAM optimizer with a primal learning rate of $\eta_{\boldsymbol{\theta}} = 0.05$ to optimize the primal model parameters, and we set the penalty term $\rho = 0.005$, which is decayed exponentially to optimize the dual variables. We generate 128 training samples and 16 testing samples for each network size with a batch size set to 16. We train the model for 40 epochs. Furthermore, we drew random samples of dual variables for the training phase of the state-augmented model from $U(0,1)$.

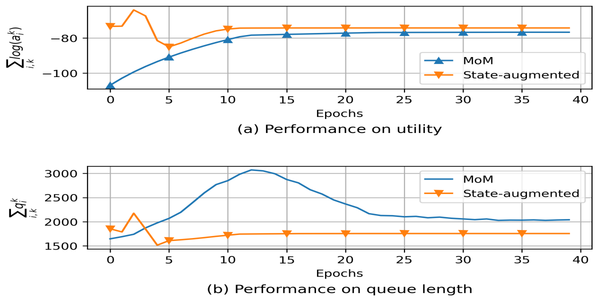

Fig. 1. Comparison of the state-augmented algorithm with MoM for a network with $N = 20$ nodes and $F = 5$ flows.

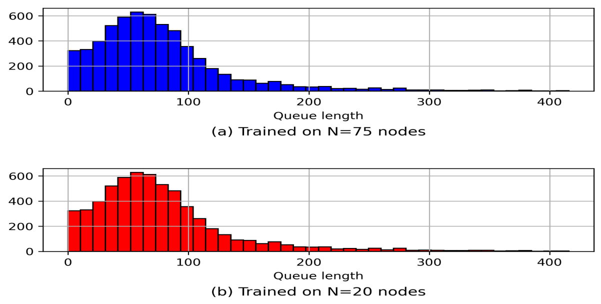

Fig. 2. Transferability of a GNN trained on networks with 20 nodes and tested on networks with 75 nodes (bottom) vs. a GNN that has been both trained and tested on 75-node networks (top).

For random networks with $N = 20$ nodes, Fig. 1 demonstrates that the state-augmented algorithm performs pretty well compared to the unparameterized MoM in terms of maximizing the utility. Our proposed method utilizing the GNN does an even superior job when it comes to minimizing the queue length, thereby ensuring the queue stability of the system faster than MoM.

We further study the *transferability* properties of GNNs on networks of larger size. In particular, Fig. 2 demonstrates the distribution of the final queue lengths during inference in networks with $N' = 75$ nodes using the routing decisions created by a GNN that has been trained on networks of size $N = 20 \ll N'$. As the figure shows, the performance of the GNN trained on the smaller network of $N = 20$ nodes almost perfectly matches that of a GNN trained on larger networks with $N' = 75$ nodes. This highlights the transfer capability of GNNs, which allows us to train the models on smaller networks, and then implement them on larger networks, thereby reducing the computational overhead incurred in training large networks during the training phase.

7. REFERENCES

[1] Jia Liu, Ness B Shroff, Cathy H Xia, and Hanif D Sherali, "Joint congestion control and routing optimization: An efficient second-order distributed approach," *Ieee/acm transactions on networking*, vol. 24, no. 3, pp. 1404–1420, 2015.

[2] Mark Eisen and Alejandro Ribeiro, "Optimal wireless resource allocation with random edge graph neural networks," *ieee transactions on signal processing*, vol. 68, pp. 2977–2991, 2020.

[3] Navid NaderiAlizadeh, Mark Eisen, and Alejandro Ribeiro, "Learning resilient radio resource management policies with graph neural networks," *IEEE Transactions on Signal Processing*, vol. 71, pp. 995–1009, 2023.

[4] Navid NaderiAlizadeh, Mark Eisen, and Alejandro Ribeiro, "State-augmented learnable algorithms for resource management in wireless networks," *IEEE Transactions on Signal Processing*, vol. 70, pp. 5898–5912, 2022.

[5] Shuang Xia, Pu Wang, and Hyuck M Kwon, "Utility-optimal wireless routing in the presence of heavy tails," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 780–796, 2018.

[6] Michael Zargham, Alejandro Ribeiro, and Ali Jadbabaie, "Accelerated backpressure algorithm," in *2013 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2013, pp. 2269–2275.

[7] Guillermo Bernárdez, José Suárez-Varela, Albert López, Xiang Shi, Shihan Xiao, Xiangle Cheng, Pere Barlet-Ros, and Albert Cabellos-Aparicio, "Magnneto: A graph neural network-based multi-agent system for traffic engineering," *IEEE Transactions on Cognitive Communications and Networking*, 2023.

[8] Shuang Xia and Pu Wang, "Stochastic network utility maximization in the presence of heavy-tails," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–7.

[9] Shuang Xia, *Distributed throughput optimal scheduling for wireless networks*, Ph.D. thesis, Wichita State University, 2014.

[10] Zhongyuan Zhao, Gunjan Verma, Ananthram Swami, and Santiago Segarra, "Delay-oriented distributed scheduling using graph neural networks," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 8902–8906.

[11] Qian Mao, Fei Hu, and Qi Hao, "Deep learning for intelligent wireless networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2595–2621, 2018.

[12] Mark Eisen, Clark Zhang, Luiz FO Chamon, Daniel D Lee, and Alejandro Ribeiro, "Learning optimal resource allocations in wireless systems," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2775–2790, 2019.

[13] Paul de Kerret, David Gesbert, and Maurizio Filippone, "Team deep neural networks for interference channels," in *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2018, pp. 1–6.

[14] Fan Meng, Peng Chen, Lenan Wu, and Julian Cheng, "Power allocation in multi-user cellular networks: Deep reinforcement learning approaches," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6255–6267, 2020.

[15] Shuai Zhang, Bo Yin, Weiyi Zhang, and Yu Cheng, "Topology aware deep learning for wireless network optimization," *IEEE Transactions on Wireless Communications*, 2022.

[16] Wei Cui, Kaiming Shen, and Wei Yu, "Spatial deep learning for wireless scheduling," *ieee journal on selected areas in communications*, vol. 37, no. 6, pp. 1248–1261, 2019.

[17] Haoran Sun, Xiangyi Chen, Qingjiang Shi, Mingyi Hong, Xiao Fu, and Nikos D Sidiropoulos, "Learning to optimize: Training deep neural networks for wireless resource management," in *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, 2017, pp. 1–6.

[18] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675–678.

[19] Jiawei Zhang, Zhuo Chen, Bojun Zhang, Ruikun Wang, Huangxu Ma, and Yuefeng Ji, "Admire: collaborative data-driven and model-driven intelligent routing engine for traffic grooming in multi-layer x-haul networks," *Journal of Optical Communications and Networking*, vol. 15, no. 2, pp. A63–A73, 2023.

[20] Jiayuan He, Jeonghun Lee, Sithamparanathan Kandeepan, and Ke Wang, "Machine learning techniques in radio-over-fiber systems and networks," in *Photonics*. MDPI, 2020, vol. 7, p. 105.

[21] Yifei Shen, Jun Zhang, SH Song, and Khaled B Letaief, "Graph neural networks for wireless communications: From theory to practice," *IEEE Transactions on Wireless Communications*, 2022.

[22] Miguel Calvo-Fullana, Santiago Paternain, Luiz FO Chamon, and Alejandro Ribeiro, "State augmented constrained reinforcement learning: Overcoming the limitations of learning with rewards," *arXiv preprint arXiv:2102.11941*, 2021.

[23] Jian Su, Shiming Yu, Bin Li, and Yinghui Ye, "Distributed and collective intelligence for computation offloading in aerial edge networks," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[24] Nikolaos Chatzipanagiotis and Michael M Zavlanos, "On the convergence of a distributed augmented lagrangian method for nonconvex optimization," *IEEE Transactions on Automatic Control*, vol. 62, no. 9, pp. 4405–4420, 2017.

[25] Dimitri Bertsekas and John Tsitsiklis, *Parallel and distributed computation: numerical methods*, Athena Scientific, 2015.

[26] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al., "Relational inductive biases, deep learning, and graph networks," *arXiv preprint arXiv:1806.01261*, 2018.

[27] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip, "A comprehensive survey on graph neural networks," *IEEE transactions on neural networks and learning systems*, vol. 32, no. 1, pp. 4–24, 2020.