ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-6874-1/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICASSP49660.2025.10888417

State-Augmented Opportunistic Routing in Wireless Communication Systems with Graph Neural Networks

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

[⋆]University of Pennsylvania
[†]Duke University

ABSTRACT

In this study, we address the challenge of packet based information routing in large-scale wireless communication networks. We approach this scenario by framing the problem as a statistical learning problem, where each node in the network relies only on the local data. Our exploration focuses on the idea of opportunistic routing, which exploits the broadcast nature of wireless communication to select the optimal relay node and transmit information packets to the destination node via multiple relay nodes. We present a distributed optimization method based on state augmentation (SA) that aims to maximize the total information on different source nodes of the network. Our formulation of the problem deploys graph neural networks (GNNs) to perform graph convolution on the topological connections between the network nodes. Using unsupervised learning, we derive optimal routing policies for the source nodes across multiple flows from the GNN output. Numerical results show the superiority of our proposed method by comparing a GNN model trained against standard algorithms.

***Index Terms*—** Opportunistic routing, Wireless communication networks, State augmentation, Graph neural networks, Unsupervised learning.

1. INTRODUCTION

Recent advances in communication technology have contributed significantly to the proliferation of wireless devices and smart systems, which require greater bandwidth, greater range and increased reliability. At the same time, the development of Artificial intelligence (AI), especially deep learning (DL), provides new solutions to complex problems that were previously unsolvable by conventional methods [1]. As wireless networks evolve to support large-scale smart infrastructures, they face increasing data traffic challenges, leading to latency and packet loss issues [2]. Opportunistic routing (OR) has emerged as an effective strategy to mitigate these challenges by exploiting the broadcast nature of wireless communication improve to transmission efficiency and reliability, especially in environments with high mobility or frequent link failures. The dynamic and uncertain nature of network topologies highlights the need for intelligent routing decisions based on deep learning techniques to adapt to these fluctuations.

In the context of communication networks, numerous features and control objectives have a significant impact on performance, including radio resource allocation, network congestion, and queue management. Extensive research has been carried out to focus on stochastic network utility maximization (NUM) [3], radio resource management (RRM) [4, 5, 6], routing [7, 8] and scheduling [9]. These efforts focus on constrained optimization problems considering the stochastic nature of user traffic and the fluctuating conditions of wireless channels. Mao *et al.* provide a comprehensive overview of various machine learning (ML) algorithms used to improve various network functions [1], highlighting the potential of ML to improve performance and functionality.

Supervised learning has been shown to be effective in solving network problems by using various ML algorithms to train neural networks that mimic system heuristics, thus improving execution efficiency [10, 11, 12, 13]. However, these methods often depend on large training datasets and sometimes fail to outperform heuristic benchmarks. Alternatively, unsupervised learning is evaluated for its suitability and potential to outperform heuristic solutions in network optimization scenarios. This paper adopts an unsupervised learning approach, considering the optimization of the communication network as a statistical regression problem, which ignores the need for large training datasets and aims to address the optimization directly [4, 5, 6, 7, 8, 9].

The architecture of ML-based systems have significant impact on network optimization. While fully connected neural networks (FCNNs) initially appeared promising due to their generalized approximation capabilities [4, 10, 14], their limitations in large-scale complex networks led to introduction of convolutional neural networks (CNN). CNNs have advanced network routing due to their ability to scale message signals [15, 16]. However, the limitations of CNN in terms of generalizability and transferability on large network graphs paved the way for graph neural networks (GNN), known for their scalability, portability and switch invariance [17, 18].

In this paper, we consider a communication network at the network layer, where packets are generated at different source nodes and assigned to specific receivers. The source node processes both locally generated packets and those received from neighboring nodes. The objective is to identify the most suitable next-hop neighbors to forward packets to ensure efficient delivery. Our approach addresses a network utility maximization problem with multiple constraints, similar to the strategies described in [4, 5, 6, 7, 8, 9]. The objective function represents the aggregated information across all nodes and all flows. We consider two sets of constraints: i) the flow constraint, which guarantees that each node has fewer outgoing packets than incoming packets for each flow, and ii) the capacity constraint, which guarantees that the sum of the transmission probabilities of a node for all flows are limited by unity. These limits are essential to maintain network stability and stabilize queue lengths over time. Contrary to conventional dual-descent algorithms, which often fail to satisfy the constraints of the problem, we propose a state-augmented unsupervised learning model which ensures the feasibility of the resulting routing policies. This approach augments the network state with dual variables related to the flow constraints, which serve as dynamic inputs to the routing policy [19]. We numerically prove that the proposed method allows routing policies to dynamically adapt to instantaneous channel states, while systematically guaranteeing that constraints are met over time.

2. PROBLEM STATEMENT

We model a communication network as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents the set of nodes and $\mathcal{E} \subseteq |\mathcal{V}| \times |\mathcal{V}|$ defines the set of edges between pairs of nodes. In this network, nodes exchange information packets through several flows defined by the set $F = |\mathcal{K}|$, where each flow $k \in \mathcal{K}$ aiming at a destination node $o_k \in \mathcal{V}$. The neighborhood of a node i is defined as $\mathcal{N}_i = \{j \in \mathcal{V} | (i,j) \in \mathcal{E}\}$, indicating which nodes j can communicate directly with the source node i. We describe the channel characteristics by R_{ij} which denotes the probability that node i successfully decodes a packet sent by node j. The element R_{ij} belongs to the matrix $\mathbf{R} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$, which summarizes the channel state probabilities between pairs of nodes. At each time step t, the node $i \neq o_k$ generates a random number of packets $A_i^k(t)$, which are destined to the o_k node. We assume that these random variables $A_i^k(t)$ are independent and identically distributed over time, with an expected value $\mathbb{E}[A_i^k(t)] = A_i^k$.

In this study, we perform opportunistic routing, where we exploit the broadcast nature of the wireless medium to select the best forwarding nodes. Thus we consider two different routing variables. The first routing variable denoted as T_i^k, represents the probability that node i decides to transmit a packet for flow k. Once the packet is broadcasted, it is overheard by the neighboring nodes. The neighboring node $j \in \mathcal{N}_i$ may decide to keep the packet from node i or drop it, the probability of which is controlled by the second routing variable K_{ij}^k. Therefore, the number of packets received by node i from node j at time t for flow k is expressed as $\mathrm{T}_j^k(t)\ \mathrm{R}_{ij}(t)\ \mathrm{K}_{ij}^k(t)\ A_j^k(t)$. Node i is limited by its capability C_i, which limits the maximum number of packets it can transmit. The difference between the total number of packets received, $A_i^k(t) + \sum_{j \in \mathcal{N}_i} \mathrm{T}_j^k(t)\ \mathrm{R}_{ij}(t)\ \mathrm{K}_{ij}^k(t)\ A_j^k(t)$ and the total number of packets transmitted, $\mathrm{T}_i^k(t)\ \mathrm{C}_i$, determines the variation of the local queue length at node i. The queue length for flow k at node i, denoted as $q_i^k(t)$, is iteratively updated according to the equation:

$$q_i^k(t+1) = \Big[q_i^k(t) + A_i^k(t) + \sum_{j \in \mathcal{N}_i} \mathrm{T}_j^k(t)\ \mathrm{R}_{ij}(t)\ \mathrm{K}_{ij}^k(t)\ A_j^k(t) - \mathrm{T}_i^k(t)\ \mathrm{C}_i\Big]^+. \quad (1)$$

This equation involves a projection on the non-negative orthant, ensuring that the length of the row remains non-negative. In particular, this formulation (1) applies to all nodes $i \neq o_k$, because packets destined for the end node o_k are dropped from the network.

2.1. Problem Formulation

In this paper, we consider the above communication network for a sequence of time steps $t \in \{0, 1, 2, .., T-1\}$. At each time step t, we define the network state as $\mathbf{R}_t \in \mathcal{R}$, which represents the channel state probabilities. Network routing decisions are characterized by two functions: $\mathbf{p}(\mathbf{R}_t)$ and $\mathbf{b}(\mathbf{R}_t)$. The function $\mathbf{p} : \mathcal{R} \to \mathbb{R}^{n \times n \times F}$ maps the network state to routing decisions regarding the acceptance of packets at a given node, while $\mathbf{b} : \mathcal{R} \to \mathbb{R}^{n \times F}$ maps the network state into routing decisions for packet transmission from a given node. These routing decisions directly affect network performance, which can be captured by the performance vector $\mathbf{f}(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t), \mathbf{b}(\mathbf{R}_t)) \in \mathbb{R}^b$, where $\mathbf{f} : \mathcal{R} \times \mathbb{R}^{n \times n \times F} \times \mathbb{R}^{n \times F} \to \mathbb{R}^b$ represents the performance function. The performance of the network is evaluated using a concave utility function $\mathcal{U} : \mathbb{R}^b \to \mathbb{R}$, which is subject to a set of concave constraints $\mathbf{g} : \mathbb{R}^b \to \mathbb{R}^c$. The general routing problem can now be defined as below,

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

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

Note that the objective function and constraints of the routing problem are evaluated based on the ergodic average of the system performance, expressed as $\frac{1}{T}\sum_{t=0}^{T-1} \mathbf{f}(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t), \mathbf{b}(\mathbf{R}_t))$. Thus the main goal of the routing algorithm is to determine the optimal routing decisions $\mathbf{p}(\mathbf{R}_t)$ and $\mathbf{b}(\mathbf{R}_t)$ for each network state $\mathbf{R}_t \in \mathcal{R}$. To solve the routing optimization problem in (2), we introduce an auxiliary variable $a_i^k(t) \geq A_i^k(t)$ to maximize the number of packets generated for flow k at node i. In reality, $A_i^k(t)$ represents the actual number of packets in the network, used to update the queue length as specified in (1).

The optimization problem aims to maximize the utility function, which represents the total number of information packets in all nodes i and flows k, given by:

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

This optimization is subject to three sets of constraints based on the current state of the communication network.

1. *Routing Constraints*: The sum of the total number of packets received and the packets generated locally $a_i^k(t)$ at node i cannot exceed the total number of packets transmitted by node i,

$$a_i^k(t) + \sum_{j \in \mathcal{N}_i} \mathrm{T}_j^k(t)\ \mathrm{R}_{ij}(t)\ \mathrm{K}_{ij}^k(t)\ a_j^k(t) \leq \mathrm{T}_i^k(t)\ \mathrm{C}_i. \quad (4)$$

2. *Capacity Constraints*: The sum of probabilities of packets transmitted by node i must *not* exceed the maximum probability value of 1,

$$\sum_{k \in K} T_i^k(t) \leq 1. \quad (5)$$

Substituting the (3)-(5) into the general routing formulation (2) along with the third set of constraints on the auxiliary variables $a_i^k(t)$, the routing optimization problem is expressed as follows:

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

$$\mathrm{T}_i^k(t)\ \mathrm{C}_i - a_i^k(t) - \sum_{j \in \mathcal{N}_i} \mathrm{T}_j^k(t)\ \mathrm{R}_{ij}(t)\ \mathrm{K}_{ij}^k(t)\ a_j^k(t) \geq 0 \quad (6b)$$

$$1 - \sum_{k \in K} \mathrm{T}_i^k(t) \geq 0 \quad (6c)$$

$$a_i^k(t) - A_i^k(t) \geq 0 \quad (6d)$$

3. AUGMENTED LAGRANGIAN & METHOD OF MULTIPLIERS

To solve the optimization problem of (6), conventional methods such as dual-descent (primal-dual) are usually used where the Lagrangian is maximized with respect to the primal variables and minimized with respect to the dual variables. However, such methods have obvious disadvantages, including slow convergence rates and the need for strictly convex objective functions [20, 21, 22]. Therefore, we adopt the augmented Lagrangian method, also known as the method of multipliers (MoM) to overcome these limitations of dual descent methods. We start by converting the inequality constraint in (6b) into an equality constraint:

$$\mathrm{T}_i^k(t)\,\mathrm{C}_i - a_i^k(t) - \sum_{j\in\mathcal{N}_i} \mathrm{T}_j^k(t)\,\mathrm{R}_{ij}(t)\,\mathrm{K}_{ij}^k(t)\,a_j^k(t) - z_i^k(t) = 0 \quad (7)$$

Now we introduce an auxiliary variable $z_i^k \geq 0$ and the augmented Lagrangian can be formulated as,

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

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 assume that the constraints in equations (6c) and (6d) are implicitly satisfied during the optimization, which means that the solutions are forced to satisfy these conditions automatically.

The MoM algorithm iteratively maximizes the augmented Lagrangian for different values of $\boldsymbol{\mu}$ and ρ. We start by maximizing $\mathcal{L}_\rho(\boldsymbol{a}, \mathbf{T}, \mathbf{K}, \boldsymbol{z}, \boldsymbol{\mu})$ with respect to $\boldsymbol{z}$:

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

Next, the algorithm maximizes $\mathcal{L}_\rho(\boldsymbol{a}, \mathbf{T}, \mathbf{K}, \boldsymbol{\mu})$ with respect to the primary variables ($\boldsymbol{a}$, $\mathbf{T}$, $\mathbf{K}$). After updating the primary variables, the minimization with respect to the dual variables is performed with gradient descent *i.e.*,

$$(\mu_i^k)^{m+1} = \left[(\mu_i^k)^m - \rho^m \left(\mathrm{T}_i^k\,\mathrm{C}_i \right. \right.$$
$$\left. \left. - \sum_{j\in\mathcal{N}_i} \mathrm{T}_j^k\,\mathrm{R}_{ij}\,\mathrm{K}_{ij}^k\,a_j^k - (a_i^k)^m - z_i^k \right) \right]^+ . \quad (10)$$

Running the above algorithm for a sufficient number of iterations ensures convergence to an optimal solution, regardless of the starting point. Although the MoM technique has theoretical advantages, its practical implementation is difficult due to the loss of decomposability in complex distributed systems and the difficulty of handling infinite dimensional spaces, for example, when the number of primary variables increases with the increasing time steps. Therefore, we resort to a parameterized routing model to enable a more practical optimization approach, which will be discussed next.

4. GNN PARAMETERIZATION WITH AUGMENTED LAGRANGIAN: METHOD OF MULTIPLIERS

Graph neural networks (GNNs) are an extension of convolutional neural networks (CNNs) designed specifically for graph-based data or signal processing [23, 24]. In our framework, we define a graph filter for GNN, where the state of the $\mathbf{R}_t$ network acts as the graph operator (GSO). At each time step t, the nodes of the graph are initialized with a feature matrix $\mathbf{Z}_t^0 \in \mathbb{R}^{|\mathcal{V}|\times F_0}$. GNN processes these input features and GSO through L layers, where each layer apply a point non-linearity to produce the output of layer l as follows:

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

where $\mathbf{H}_{lk} \in \mathbb{R}^{F_{l-1}\times F_l}$ represents the graph filter coefficients or GNN parameters in layer l.

To solve our optimization problem, we apply the Multi-Input Multi-Output (MIMO) graphical filter recursively as described in (11) to build the GNN architecture. The GNN structure is characterized by the operator $\mathbf{\Psi}(\mathbf{R}_t, \mathbf{Z}_t; \mathbf{H}) = \mathbf{Z}_t^L \in \mathbb{R}^{|\mathcal{V}|\times F_L}$, where $\mathbf{H} = \{\mathbf{H}_{lk}\}_{l=1,k=0}^{L,K-1}$ is the tensor of the filter. The initial input feature $\mathbf{Z}_t = \mathbf{Z}_t^0 = A_i^k(t)$ is fed into the GNN and the result is multiplied by the matrix $\boldsymbol{W}_r \in \mathbb{R}^{F_L\times F_L}$ to make packet acceptance decisions $\mathbf{p}(\mathbf{R}_t, \mathbf{Z}_t; \boldsymbol{\phi})$. This matrix is processed via a softmax function to ensure that the routing decision values are in the probabilistic interval from 0 to 1, *i.e.* $\mathbf{p}(\mathbf{R}_t, \mathbf{Z}_t) = \mathrm{Softmax}(\mathbf{Z}_t^L\ \boldsymbol{W}_r\ \mathbf{Z}_t^{L^T})$. Similarly, another intermediate matrix $w_s \in \mathbb{R}^{F_L}$ is used to make emission decisions $\mathbf{b}(\mathbf{R}_t, \mathbf{Z}_t; \boldsymbol{\phi})$, i.e. $\mathbf{b}(\mathbf{R}_t, \mathbf{Z}_t) = \mathrm{Softmax}(\mathbf{Z}_t^L\ \boldsymbol{W}_s\ \mathbf{Z}_t^{L^T})$. Also, by multiplying the GNN output by a column vector $\boldsymbol{W}_a \in \mathbb{R}^{F_L\times 1}$ and applying the ReLU function, we obtain $a_i^k(t)$.

Using this GNN architecture with the parameters shown in (11), we reformulate our *parameterized* optimization problem as follows:

$$\max_{\phi, w_r, w_s, w_a} \quad \mathcal{U}\left(\frac{1}{T} \sum_{t=0}^{T-1} \mathbf{f}\Big(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t; \boldsymbol{\phi}), \mathbf{b}(\mathbf{R}_t; \boldsymbol{\phi})\Big) \right) \quad (12a)$$

$$s.t. \quad \mathbf{g}\left(\frac{1}{T} \sum_{t=0}^{T-1} \mathbf{f}\Big(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t; \boldsymbol{\phi}), \mathbf{b}(\mathbf{R}_t; \boldsymbol{\phi})\Big) \right) \geq 0. \quad (12b)$$

where $\boldsymbol{\phi} = \{\mathbf{H}, \boldsymbol{W}_r, \boldsymbol{W}_s, \boldsymbol{W}_a\}$ represents the set of all GNN parameters involved in the optimization. Note that the maximization is now performed on the GNN filter tensor ϕ. The augmented Lagrangian for the network optimization in (12) can now be re-written as,

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\mu}) = \mathcal{U}\left(\frac{1}{T} \sum_{t=0}^{T-1} \mathbf{f}\Big(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t; \boldsymbol{\phi}), \mathbf{b}(\mathbf{R}_t; \boldsymbol{\phi})\Big) \right)$$
$$+ \boldsymbol{\mu}^T \left[\mathbf{g}\left(\frac{1}{T} \sum_{t=0}^{T-1} \mathbf{f}\Big(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t; \boldsymbol{\phi}), \mathbf{b}(\mathbf{R}_t; \boldsymbol{\phi})\Big) \right) - \boldsymbol{z} \right]$$
$$+ \frac{\rho}{2} \left\| \mathbf{g}\left(\frac{1}{T} \sum_{t=0}^{T-1} \mathbf{f}\Big(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t; \boldsymbol{\phi}), \mathbf{b}(\mathbf{R}_t; \boldsymbol{\phi})\Big) \right) - \boldsymbol{z} \right\|^2 . \quad (13)$$

Now we introduce an iteration length T_0, which represents the number of time steps between successive updates of the model parameters. Using slight abuse of the time notation t, we define the iteration index $m \in \{0, 1, 2, ..., M-1\}$, where $M = \lfloor T/T_0 \rfloor$, the model parameters ϕ are updated as follows:

$$\boldsymbol{\phi}_m = \arg\max_{\boldsymbol{\phi}\in\boldsymbol{\Phi}} \mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\mu}_m). \quad (14)$$

After updating the model parameters, the dual variables $\boldsymbol{\mu}$ are updated recursively:

$$\boldsymbol{\mu}_{m+1} = \left[\boldsymbol{\mu}_m - \rho \left\{ \mathbf{g}\left(\frac{1}{T} \sum_{t=mT_0}^{(m+1)T_0-1} \mathbf{f}(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t; \boldsymbol{\phi}), \mathbf{b}(\mathbf{R}_t; \boldsymbol{\phi})) \right) - \boldsymbol{z} \right\} \right]^+ . \quad (15)$$

It can be noted that the penalty parameter ρ plays the crucial role of a learning rate in the dual-variable update process. This augmented Lagrangian parameter method offers significant advantages over traditional dual subtraction algorithms, such as primal-dual methods. In particular, the updates given in (14), (15) allow the dynamic adjustment of the routing policy based on the dual variables during each iteration.

5. THE STATE AUGMENTATION ALGORITHM

The iterative optimization process described in (14), (15) faces practical challenges, especially because the optimal set of GNN parameters has to be re-calibrated for different dual variable vectors, $\boldsymbol{\mu}_m$

at each time step [6]. To solve this problem, we propose a *state augmentation* algorithm that augments the state of the network $\mathbf{R}_t$ at each time step t by including the corresponding set of dual variables $\boldsymbol{\mu}_{\lfloor t/T_0 \rfloor}$. This method inputs the dual variables along with the input features $A_i^k(t)$ in a GNN, thus allowing the generation of optimal routing decisions. These routing decisions, originally represented by $\mathbf{p}(\mathbf{R}_t)$ and $\mathbf{b}(\mathbf{R})$, are now expressed as $\mathbf{p}(\mathbf{R}_t, \boldsymbol{\mu}; \boldsymbol{\theta})$ and $\mathbf{b}(\mathbf{R}, \boldsymbol{\mu}; \boldsymbol{\theta})$, where $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ represents the set of tensors of the GNN filter parameterized by the state augmentation algorithm. Considering that a batch of dual variables is drawn from a probability distribution $p_{\boldsymbol{\mu}}$, we adapt the augmented Lagrangian formulation to:

$$\begin{aligned}\mathcal{L}_{\boldsymbol{\mu}}(\boldsymbol{\theta}) = \mathcal{U}\left(\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{f}\Big(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t, \boldsymbol{\mu}; \boldsymbol{\theta}), \mathbf{b}(\mathbf{R}_t, \boldsymbol{\mu}; \boldsymbol{\theta})\Big)\right) \\ + \boldsymbol{\mu}^T \mathbf{g}\left(\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{f}\Big(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t, \boldsymbol{\mu}; \boldsymbol{\theta}), \mathbf{b}(\mathbf{R}_t, \boldsymbol{\mu}; \boldsymbol{\theta})\Big)\right) \\ + \frac{\rho}{2}\left\|\mathbf{g}\left(\frac{1}{T}\sum_{t=0}^{T-1}\mathbf{f}\Big(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t, \boldsymbol{\mu}; \boldsymbol{\theta}), \mathbf{b}(\mathbf{R}_t, \boldsymbol{\mu}; \boldsymbol{\theta})\Big)\right)\right\|^2. \quad (16)\end{aligned}$$

The state-augmented routing policy is defined as the one that maximizes the expected value of the augmented Lagrangian over the probability distribution of all dual variables:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mathbb{E}_{\boldsymbol{\mu} \sim p_{\boldsymbol{\mu}}}\left[\mathcal{L}_{\boldsymbol{\mu}}(\boldsymbol{\theta})\right] \quad (17)$$

In this state augmented algorithm, parameterized by $\boldsymbol{\theta}^*$, the Lagrangian-maximized routing decisions $\mathbf{p}(\mathbf{R}, \boldsymbol{\mu}; \boldsymbol{\theta})$ and $\mathbf{b}(\mathbf{R}, \boldsymbol{\mu}; \boldsymbol{\theta})$ are evaluated for each dual variable $\boldsymbol{\mu} = \boldsymbol{\mu}_m$. This procedure, designed for the offline learning phase, consists of randomly drawing a batch of dual variables $\{\boldsymbol{\mu}_b\}_{b=1}^B$ from the probability distribution $p_{\boldsymbol{\mu}}$ and using gradient descent to find the optimal set of parameters θ^*. Starting with a arbitrary initialization set for θ_0, the model parameters are iteratively updated through $n = 0, ..., N-1$ as follows:

$$\boldsymbol{\theta}_{n+1} = \boldsymbol{\theta}_n + \frac{\eta_\theta}{B}\sum_{b=1}^{B-1}\nabla_\theta \mathcal{L}_{\boldsymbol{\mu}_b}(\boldsymbol{\theta}_n), \quad (18)$$

where η_θ is the learning rate for optimizing the model parameters. This approach systematically improves the model parameters to improve system performance based on the augmented Lagrangian gradient via (18). After the training is finished, the model parameters are stored as $(\boldsymbol{\theta}^*)$. During the execution phase, the dual variables are updated at each m iteration in (17) as below:

$$\begin{aligned}\boldsymbol{\mu}_{m+1} = \Bigg[\boldsymbol{\mu}_m - \eta_{\boldsymbol{\mu}}\mathbf{g}\bigg(\frac{1}{T_0}\times \\ \sum_{t=mT_0}^{(m+1)T_0-1}\mathbf{f}\Big(\mathbf{R}_t, \mathbf{p}(\mathbf{R}_t, \boldsymbol{\mu}_m; \boldsymbol{\theta}^*), \mathbf{b}(\mathbf{R}_t, \boldsymbol{\mu}_m; \boldsymbol{\theta}^*)\Big)\bigg)\Bigg]^+ \quad (19)\end{aligned}$$

The above dual update allows the simultaneous learning of a parameterized model and the determination of a state-augmented optimal routing policy, as described in (18). The proposed method allows the trained model to adapt to different sets of binary variables $\boldsymbol{\mu}_b$ in a set, which can be achieved by randomly choosing different instances of the network state $\{\mathbf{R}_{b,t}\}_{t=0}^{T-1}$.

6. NUMERICAL RESULTS

In our study, we model graphs with $N = |\mathcal{V}|$ nodes using the k-Nearest Neighbor (k-NN) approach to construct the graph topology. The nodes are randomly distributed on a unit circle. For our experiments, we set the parameters as follows: $k = 4$, $T = 100$, $\mathrm{C}_i = 100$, $T_0 = 5$. The probability matrix of the $\mathbf{R}$ channel for our GNN operation is estimated using a simple linear model:

$$\mathrm{R}_{ij} = 1 - \frac{d_{ij}}{d_c}, \quad (20)$$

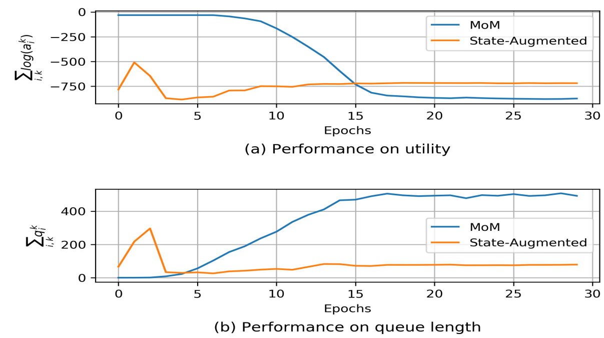

Fig. 1. Comparison of the state-augmented algorithm with MoM for a network with $N = 20$ nodes and $F = 5$ flows.

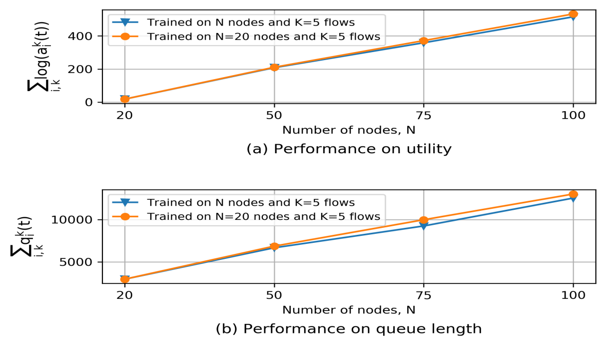

Fig. 2. Transferability of the proposed state-augmented algorithm on networks with different nodes while they were trained on a network with 20 nodes and 5 flows.

where d_{ij} is the distance between nodes i and j, and d_c (set to 0.3 in our experiment) is the threshold distance beyond which packet delivery fails. This model reflects the degradation of link performance with distance and is comparable to the more complex probabilistic models described in [25, 26]. Our GNN architecture consists of three layers with feature sizes $F_0 = 2, F_1 = 32$, and $F_2 = 16$. We optimize the primary model parameters using the ADAM optimizer with a learning rate $\eta_{\boldsymbol{\theta}} = 0.05$. The penalty term $\rho = 0.005$ undergoes an exponential decay for updates of dichotomous variables. The data includes 128 training samples and 16 testing samples for each network size. We train the model for 30 epochs with a set size of 16 samples, and during training, dichotomous variables are randomly sampled from a uniform distribution $U(1, 5)$.

In a random network with $N = 10$ nodes, as shown in Figure Fig. 1 , our state optimization algorithm outperforms the parameter-free multipliers (MoM) method to maximize network utility. Moreover, by exploiting the graph neural networks (GNN), our method effectively reduces the queue length, thereby ensuring the system stability faster than conventional MoM. We also explore the *transferability* of GNNs to larger and invisible networks. Fig. 2 shows that GNNs trained on smaller networks with $N = 20 \ll N'$ nodes can generate routing decisions whose performance is comparable to those trained directly on larger networks with $N' = \{50, 75, 100\}$ nodes. This highlights the ability of GNNs to be trained on smaller networks and then efficiently applied to larger networks, significantly reducing the computational costs associated with training larger network configurations in the training phase.

7. REFERENCES

[1] 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.

[2] Eric Rozner, Jayesh Seshadri, Yogita Mehta, and Lili Qiu, "Soar: Simple opportunistic adaptive routing protocol for wireless mesh networks," *IEEE transactions on Mobile computing*, vol. 8, no. 12, pp. 1622–1635, 2009.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

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

[9] 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.

[10] 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.

[11] Lei Lei, Lei You, Gaoyang Dai, Thang Xuan Vu, Di Yuan, and Symeon Chatzinotas, "A deep learning approach for optimizing content delivering in cache-enabled hetnet," in *2017 international symposium on wireless communication systems (ISWCS)*. IEEE, 2017, pp. 449–453.

[12] Di Xu, Xiaojing Chen, Changhao Wu, Shunqing Zhang, Shugong Xu, and Shan Cao, "Energy-efficient subchannel and power allocation for hetnets based on convolutional neural network," in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*. IEEE, 2019, pp. 1–5.

[13] Trinh Van Chien, Emil Bjornson, and Erik G Larsson, "Sum spectral efficiency maximization in massive mimo systems: Benefits from deep learning," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–6.

[14] Shengliang Peng, Hanyu Jiang, Huaxia Wang, Hathal Alwageed, and Yu-Dong Yao, "Modulation classification using convolutional neural network based deep learning model," in *2017 26th Wireless and Optical Communication Conference (WOCC)*. IEEE, 2017, pp. 1–5.

[15] 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.

[16] 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.

[17] Mikael Henaff, Joan Bruna, and Yann LeCun, "Deep convolutional networks on graph-structured data (2015)," *arXiv preprint arXiv:1506.05163*, 2015.

[18] Fernando Gama, Antonio G Marques, Geert Leus, and Alejandro Ribeiro, "Convolutional neural network architectures for signals supported on graphs," *IEEE Transactions on Signal Processing*, vol. 67, no. 4, pp. 1034–1049, 2018.

[19] 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.

[20] 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.

[21] 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.

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

[23] 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.

[24] 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.

[25] Daniel Mox, Miguel Calvo-Fullana, Mikhail Gerasimenko, Jonathan Fink, Vijay Kumar, and Alejandro Ribeiro, "Mobile wireless network infrastructure on demand," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 7726–7732.

[26] James Stephan, Jonathan Fink, Vijay Kumar, and Alejandro Ribeiro, "Concurrent control of mobility and communication in multirobot systems," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1248–1254, 2017.