UNSUPERVISED LEARNING FOR ASYNCHRONOUS RESOURCE ALLOCATION IN AD-HOC WIRELESS NETWORKS

Zhiyang Wang[†] *Mark Eisen*[⋆] *Alejandro Ribeiro*[†]

[†] Department of Electrical and Systems Engineering, University of Pennsylvania, PA
[⋆]Intel Labs, Hillsboro, OR

ABSTRACT

We consider optimal resource allocation problems under asynchronous wireless network setting. Without explicit model knowledge, we design an unsupervised learning method based on Aggregation Graph Neural Networks (Agg-GNNs). Depending on the localized aggregated information structure on each network node, the method can be learned globally and asynchronously while implemented locally. We capture the asynchrony by modeling the activation pattern as a characteristic of each node and train a policy-based resource allocation method. We also propose a permutation invariance property which indicates the transferability of the trained Agg-GNN. We finally verify our strategy by numerical simulations compared with baseline methods.

Index Terms— Resource allocation, asynchronous, decentralized, graph neural networks

1. INTRODUCTION

Wireless communication systems are increasingly facing the challenge of allocating finite and interfering resource over large scale networks with limited coordination between devices. Resource allocation methods, generally speaking, are usually addressed through optimization methods. Because of their non-convex nature in wireless systems, standard centralized resource allocation policies are obtained through heuristic optimization methods [1–3] or data-driven machine learning methods [4–9]. The latter case is seeing growing interest due to its applicability in a wide range of application scenarios and lack of reliance on explicit or accurate system modeling.

Resource allocation problems are made more challenging, however, in the decentralized setting where devices are locally selecting resource levels, such as the case in ad-hoc networks. This framework significantly reduces communication overhead and be more robust to single-node failures in large networks. In fully localized methods, devices make their own decision solely with local state measurements. This formulation has been studied both with traditional heuristic methods [10] as well as machine learning methods [11]. More sophisticated approaches permit limited information exchanges among neighboring network nodes, which can greatly improve performance. Such is the case in the decentralized implementation of the WMMSE heuristic [12], as well as a variety of learning based methods that employ this broader local information structure [13–16].

While most existing decentralized methods consider synchronous algorithms, fully decentralized systems feature different working clocks across nodes in the network. Asynchronous decentralized resource allocation methods have been considered in the context of online optimization approaches [17, 18]. Alternatively, the learning-based methodologies may be used in which the parameters of a resource allocation policy are pre-trained under asynchronous working conditions offline.

In this paper we address the asynchronous decentralized wireless resource allocation problem with a novel unsupervised learning policy-based approach. By considering the interference patterns between transmitting devices as a graph [7–9], we capture the asynchrony patterns via the activation of the graph edges on a highly granular time scale. From this graph representation of interference and asynchrony, we implement a decentralized learning architecture as the Aggregation Graph Neural Networks (Agg-GNNs) [19]. The Agg-GNN policy leverages successive local state exchanges to build local embeddings of the global network state at each node, which is then processed through a convolutional neural network to determine resource allocation decisions. Such a policy is trained offline in an unsupervised manner that captures network performance under asynchronous conditions without explicit modeling. We further demonstrate a permutation invariance property of Agg-GNNs that facilitates transference across varying wireless network topologies. Numerical experiments are carried out to verify that our proposed policy outperforms other baseline heuristic methods.

2. ASYNCHRONOUS RESOURCE ALLOCATION

We consider an ad-hoc wireless system with m transmitters and n receivers with $m \geq n$, such that a receiver may be paired with multiple transmitters. We denote the paired receiver of the i-th transmitter as $r(i)$. Due to fast fading phenomenon, the quality of each link changes rapidly over time. At each time instance $t = 1, 2, ..,$ the channel states of all the links can be characterized by a matrix $\mathbf{H}(t) \in \mathbb{R}_{+}^{m \times m}$. Each element $|h_{ij}(t)| := [\mathbf{H}(t)]_{ij}$ represents the link state between transmitter j and receiver $r(i)$ at time slot t. We further consider a set of time-varying node states as $\mathbf{x}(t) \in \mathbb{R}^m$, with each entry $x_i(t) := [\mathbf{x}(t)]_i$ representing the state of the i-th node at time t. Both the link states $\mathbf{H}(t)$ and node states $\mathbf{x}(t)$ are considered as random, drawn from the joint distribution $m(\mathbf{H}, \mathbf{x})$.

In decentralized networks, it is challenging to maintain a shared

Supported by ARL DCIST CRA W911NF-17-2-0181 and Intel Science and Technology Center for Wireless Autonomous Systems.

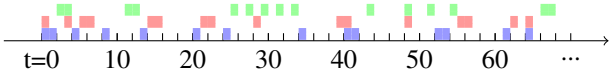

Fig. 1: Channel and node states change asynchronously in a granular time scale. The colored bars show the working time of three nodes.

synchronous clock across all the nodes without a centralized controller or frequent communication overhead. Figure 1 shows an example configuration in which channel and node states vary at each time instance in fast varying scenarios.To design resource allocation policies under asynchronous conditions, we may model heterogeneous and potentially time-varying "working patterns" for each node in the network relative to the more granular channel coherence time given by time index $t = 0, 1, \ldots$. In particular, at each time t, we can denote the set of active nodes as $\mathcal{A}(t) \subseteq \{1, 2, \ldots, m\}$, which indicates that only nodes $i \in \mathcal{A}(t)$ are capable of making decisions, such as updating its resource allocation strategy, and taking actions, such as sending information to neighboring nodes.

For decentralized resource allocation policies, it is necessary to define an asynchronous and localized information structure available to each node, which we denote as its *neighborhood*. In most large-scale network settings, large distance between certain devices causes only negligible channel quality and interference to its receiver. We therefore consider the single-hop neighborhood of node i at time t to consist of active devices with non-negligible interference channel quality, i.e.,

$$\mathcal{N}_i(t) := \{j \mid h_{ij}(t) \geq h_\epsilon, j \in \mathcal{A}(t)\}. \tag{1}$$

In addition to the information received by single-hop neighbors, a device can receive indirect information from larger neighborhoods with some delay to obtain more global state awareness. We define the k-hop neighborhood of node i as

$$\mathcal{N}_i^k(t) := \{j' \in \mathcal{N}_j(t-k+1), j \in \mathcal{N}_i^{k-1}(t) \cap \mathcal{A}(t)\}. \tag{2}$$

By imposing a limit of K-hop exchanges, the set of local available information of node i at time t is then defined as

$$\mathcal{H}_i(t) := \bigcup_{k=0}^{K-1} \Big\{ [\mathbf{H}(t-k+1)]_{jj'}, [\mathbf{x}(t-k)]_{j'} \mid j \in \mathcal{N}_i^{k-1}(t), j' \in \mathcal{N}_i^k(t) \Big\}. \tag{3}$$

Observe that the local information available at each node depends both on the channel quality with neighboring devices as well as the particular asynchronous activation patterns.

The goal of this work is to find a local resource allocation that maps a node's local history information to an instantaneous resource allocation level $\mathbf{p}_i(t) := \boldsymbol{\phi}_i(\mathcal{H}_i(t); \mathbf{A})$, where $\mathbf{A} \in \mathbb{R}^s$ is the parameter of some function family $\boldsymbol{\phi}(\cdot)$. At time t, together with state pairs $\mathbf{H}(t), \mathbf{x}(t)$, a collection of instantaneous global performance feedbacks $\mathbf{f}\left(\mathbf{P}(\mathcal{H}(t)), \mathbf{H}(t), \mathbf{x}(t)\right)$ is experienced in the network. In fast fading channels, users tend to experience the average performance across the states. This can be evaluated as an expectation over all random states with the assumption that these states are independent and stationary. The optimal policy then is defined as that which maximizes the total average performance across all the links while respecting local and shared resource budgets p_0 and P_{max}, respectively, i.e.

$$\begin{aligned} \mathbf{A}^* := \underset{\mathbf{A},\mathbf{r}}{\operatorname{argmax}} \quad & \mathbf{1}^T\mathbf{r}, \\ \text{s.t.} \quad & \mathbf{r} = \mathbb{E}\left[\mathbf{f}\left(\boldsymbol{\Phi}(\mathcal{H}, \mathbf{A}), \mathbf{H}, \mathbf{x}\right)\right], \\ & \mathbb{E}[\mathbf{1}^T\boldsymbol{\Phi}(\mathcal{H}, \mathbf{A})] \leq P_{max}, \\ & \phi_i(\mathcal{H}_i, \mathbf{A}) \in \{0, p_0\}, \quad i = 1, .., m. \end{aligned} \tag{4}$$

The optimization problem (4) is typically non-convex and intractable. This limitation and the lack of explicit model-knowledge has motivated the use of model-free unsupervised learning techniques to solve. Numerous decentralized learning architectures have been proposed to solve problems similar to (4), e.g., fully connected neural networks [13, 16]. In this work we develop the use of Aggregation Graph Neural Networks (Agg-GNNs) to perform decentralized and scalable resource allocation in asynchronous wireless network settings.

3. AGGREGATION GRAPH NEURAL NETWORKS

The Agg-GNN resource allocation utilized for decentralized resource allocation can be derived by contextualizing the interference and asynchronous working patterns of the networks as a time-varying graph. Consider a graph with m nodes corresponding to the transmitting devices, and edges (i, j) if $j \in \mathcal{N}_i(t)$. The transmitter states $\mathbf{x}(t)$ can be interpreted as signals supported on the nodes, while the instantaneous channel state $h_{ij}(t)$ as the weight of edge (i, j). The weighted graph adjacency matrix is then given by $\mathbf{H}_0(t)$, with $[\mathbf{H}_0(t)]_{ij} = h_{ij}(t)$ if $j \in \mathcal{N}_i(t)$ and $[\mathbf{H}_0(t)]_{ij} = 0$ otherwise based on the definition of $\mathcal{N}_i(t)$ in (1). Observe that this graph structure incorporates both the current channel state between devices as well as the asynchronous activation patterns of neighboring devices, and thus represents communicating devices at each time slot.

The Agg-GNN relies on gradually accumulating global state information through an aggregation process in which it collects state information from neighboring transmitters. In particular, node i receives delayed node states $x_j(t-1)$ from its neighbors $j \in \mathcal{N}_i(t)$. This neighborhood information is aggregated by node i into a first hop local feature $y_i^{(i)}(t)$ using its respective channel states $h_{ij}(t)$ as

$$y_i^{(1)}(t) = \sum_{j \in \mathcal{N}_i(t)} h_{ij}(t) x_j(t-1). \tag{5}$$

From the definition of $\mathcal{N}_i(t)$, this can be written globally as

$$\mathbf{y}^{(1)}(t) = \mathbf{H}_0(t)\mathbf{x}(t-1). \tag{6}$$

The aggregated signal $\mathbf{y}^{(1)}(t)$ in (5) only captures immediate neighborhood state information, but local decisions can be improved relative to the global problem in (4) by accumulating more global information. Thus, aggregated states are further aggregated at the subsequent time $t+1$ under the new adjacency matrix $\mathbf{H}_0(t+1)$. After a total of K such successive aggregations,

a sequence of aggregated signals $\mathbf{y}^{(0)}(t), \mathbf{y}^{(1)}(t), \ldots, \mathbf{y}^{(K-1)}(t)$ can be obtained, with the k-th element for $k > 1$ written as

$$\mathbf{y}^{(k)}(t) := \mathbf{H}_0(t)\mathbf{y}^{(k-1)}(t-1) = \left[\prod_{i=0}^{k-1} \mathbf{H}_0(t-i)\right] \mathbf{x}(t-k), \tag{7}$$

while $\mathbf{y}^{(0)}(t) := \mathbf{x}(t)$. By taking the i-th element of each element $\mathbf{y}^{(k)}(t)$, we obtain the *wireless information aggregation sequence* of node i, which can be written as:

$$\mathbf{y}_i(t) = [y_i^{(0)}(t); y_i^{(1)}(t); \ldots; y_i^{(K-1)}(t)]. \tag{8}$$

This is collected by node i with only the information contained in its local history set defined in (3). The aggregated information of each active node is transmitted once per time slot as an overhead message. Note that inactive nodes will not transmit signals but still receive aggregated information, which can then be forwarded to their neighboring nodes during their next active phase.

3.1. Graph Neural Networks

With the local aggregation process in (8), each node obtains a local sequence of state information with a temporal structure. Then it can be processed with a regular Convolution Neural Network (CNN) architecture with L layers.The architecture begins with a standard filter to produce an intermediate output, which is then passed through a pointwise nonlinear function.

For simplicity of presentation, we consider a single feature per layer network structure. We denote $\boldsymbol{\alpha}_l := [[\boldsymbol{\alpha}_l]_1; \ldots; [\boldsymbol{\alpha}_l]_{K_l}]$ as the coefficients of a K_l-tap linear filter which is used to process the feature of the $l-1$-th layer, which is then followed by a pointwise nonlinear function σ_l. With the initial layer input set as the local aggregation sequence $\mathbf{v}_{i0} := \mathbf{y}_i(t)$, the l-th layer output can be given by

$$\mathbf{v}_{il} = \sigma_l \left[\boldsymbol{\alpha}_l * \mathbf{v}_{i(l-1)}\right]. \tag{9}$$

The resource allocation is then given by the final layer output, i.e.

$$p_i(t) := \phi_i(\mathcal{H}_i(t), \mathbf{A}) = \mathbf{v}_{iL}. \tag{10}$$

The policy parameter is given by the collection of filter coefficients $\mathbf{A} = \{\boldsymbol{\alpha}_l\}_{l=1}^L$. Note that the filter parameters are shared across all the nodes, i.e. they implement the same local policy. Although the policy is the same across node, the resource allocations differ for each node due to different local history information. The detailed operation of this process is presented in Alg. 1.

Observe that the aggregated information sequence in (8) is obtained regardless of the dimension and shape of the underlying network—the policy is defined by $\mathbf{A}$ and is invariant to input dimension. Furthermore the sequence structure permits invariance to permutations of the corresponding graph. We restrict permutation matrices of dimension m as the permutation set:

$$\psi = \{\mathbf{\Pi} \in \{0,1\}^{m \times m} : \mathbf{\Pi}\mathbf{1} = \mathbf{1}, \mathbf{\Pi}^T\mathbf{1} = \mathbf{1}\}. \tag{11}$$

The theorem can be stated as follows:

Algorithm 1 Resource Allocation at Node i

for $t = 0, 1, 2..$ **do**
  if Node is active, $i \in \mathcal{A}(t)$ **then**
    Observe node state $x_i(t)$ and set $y_i^{(0)}(t) = x_i(t)$.
    Transmits sequence $\{y_i^{(k)}(t-1)\}_{k=0}^{K-2}$ to transmitter $j \in \mathcal{N}_i(t)$.
    Node i forms aggregation sequence $\mathbf{y}_i(t)$ based on information from its active neighbors.
    Updates resource level $p_i(t) = \phi(\mathcal{H}_i(t), \mathbf{A}) = \mathbf{y}_{iL}(t)$.
  else
    Receives information from its active neighbors and forms aggregation sequence $\mathbf{y}_i(t)$.
    Keeps resource level $p_i(t) = p_i(t-1)$.
  end if
end for

Theorem 1 *For any permutation $\mathbf{\Pi} \in \psi$, define the permuted graphs as $\hat{\mathbf{H}} = \mathbf{\Pi}^T\mathbf{H}\mathbf{\Pi}$. The permuted graph signal as: $\hat{\mathbf{x}} = \mathbf{\Pi}^T\mathbf{x}$. Further define the permuted joint state distribution $\hat{m}(\hat{\mathbf{H}}, \hat{\mathbf{x}})$ such that $\hat{m}(\hat{\mathbf{H}}, \hat{\mathbf{x}}) = m(\mathbf{H}, \mathbf{x})$. The solutions for* (4) *$\mathbf{A}^*$ and $\hat{\mathbf{A}}^*$ under $\hat{m}(\hat{\mathbf{H}}, \hat{\mathbf{x}})$ and $m(\mathbf{H}, \mathbf{x})$ respectively satisfy $\hat{\mathbf{A}}^* = \mathbf{A}^*$.*

Proof. Consider that as the underlying network changes in topology, the state distribution is then given by a transformed distribution $\hat{m}(\hat{\mathbf{H}}, \hat{\mathbf{x}})$ from which state pairs $(\hat{\mathbf{H}}, \hat{\mathbf{x}})$ are drawn randomly. With a similar way to (3), the history information $\hat{\mathcal{H}}$ can be formulated as

$$\hat{\mathcal{H}}_i(t) := \bigcup_{k=0}^{K-1} \Big\{ [\hat{\mathbf{H}}(t-k+1)]_{jj'}, [\hat{\mathbf{x}}(t-k)]_{j'} \mid j \in \hat{\mathcal{N}}_i^{k-1}(t), j' \in \hat{\mathcal{N}}_i^{k}(t) \Big\}, \tag{12}$$

where the neighboring set can be defined similarly as $\hat{\mathcal{N}}_i(t) = \{[\hat{\mathbf{H}}(t)]_{ij} \geq \eta_0, j \in \mathcal{A}(t)\}$ and $\hat{\mathcal{N}}_i^k(t) := \{j' \in \hat{\mathcal{N}}_j(t-k+1), j \in \hat{\mathcal{N}}_i^{k-1}(t) \cap \mathcal{A}(t)\}$.

Applying a permutation matrix $\mathbf{\Pi} \in \psi$ to a signal $\mathbf{x}$ as $\mathbf{\Pi}^T\mathbf{x}$ indicates a reordering of elements in the vector; likewise an application to matrix $\mathbf{H}$ as $\mathbf{\Pi}^T\mathbf{H}\mathbf{\Pi}$ indicates a corresponding reordering of columns and rows. In the setting of wireless networks, this can be interpreted as the permutation of locations or labels of the transmitters and paired receivers.

We first prove that the outputs of the same filter tensor is permutation equivariant, which can be stated as the follows.

Proposition 1 *Consider graphs $\mathbf{H}$ and $\hat{\mathbf{H}}$ together with signals $\mathbf{x}$ and $\hat{\mathbf{x}}$, we have $\hat{\mathbf{H}} = \mathbf{\Pi}^T\mathbf{H}\mathbf{\Pi}$ and $\hat{\mathbf{x}} = \mathbf{\Pi}^T\mathbf{x}$ for some permutation matrix $\mathbf{\Pi}$. The output of the Agg-GNN with filter $\mathbf{A}$ to the pairs $(\mathbf{H}, \mathbf{x})$ and $(\hat{\mathbf{H}}, \hat{\mathbf{x}})$ are such that:*

$$\mathbf{\Phi}(\hat{\mathcal{H}}, \mathbf{A}) = \mathbf{\Pi}^T\mathbf{\Phi}(\mathcal{H}, \mathbf{A}). \tag{13}$$

To prove this, we first begin with the simple version omitting the time stamps and a constant $\mathbf{H}$ matrix. We look at the layer $l = 1$ and take the input $\hat{\mathbf{y}}_{i0} = [[\hat{\mathbf{x}}]_i, [\hat{\mathbf{H}}\hat{\mathbf{x}}]_i, \ldots, [\hat{\mathbf{H}}^{K-1}\hat{\mathbf{x}}]_i]$. At

node i, the output of the first layer is given by:

$$\begin{aligned}\hat{\mathbf{y}}_{i1} &= \sigma_1[\boldsymbol{\alpha}_1 * \hat{\mathbf{y}}_{i0}] \\ &= \sigma_1[\boldsymbol{\alpha}_1 * [[\hat{\mathbf{x}}]_i; [\hat{\mathbf{H}}\hat{\mathbf{x}}]_i; \dots [\hat{\mathbf{H}}^{K-1}\hat{\mathbf{x}}]_i]].\end{aligned}$$

First we realize that for a general k, we have:

$$\hat{\mathbf{H}}^k = \mathbf{\Pi}^T\mathbf{H}\mathbf{\Pi}\mathbf{\Pi}^T\mathbf{H}\mathbf{\Pi}\dots\mathbf{\Pi}^T\mathbf{H}\mathbf{\Pi} = \mathbf{\Pi}^T\mathbf{H}^k\mathbf{\Pi},$$

due to the fact that $\mathbf{\Pi}\mathbf{\Pi}^T = \mathbf{I}$. By inserting the definition of $\hat{\mathbf{x}}$, we can get

$$\begin{aligned}\hat{\mathbf{y}}_{i1} &= \sigma_1[\boldsymbol{\alpha}_1 * [[\mathbf{\Pi}^T\mathbf{x}]_i; [\mathbf{\Pi}^T\mathbf{H}\mathbf{x}]_i; \dots [\mathbf{\Pi}^T\mathbf{H}^{K-1}\mathbf{x}]_i]] \\ &= \sigma_1[\boldsymbol{\alpha}_1 * [\mathbf{\Pi}^T\mathbf{Y}]_i] = \sigma_1[[\boldsymbol{\alpha}_1 * (\mathbf{\Pi}^T\mathbf{Y})]_i],\end{aligned}$$

where $\mathbf{Y}(t)$ stands for a matrix representation of the sequences of signals:

$$\mathbf{Y}(t) := [\mathbf{y}^{(0)}(t), \mathbf{y}^{(1)}(t), \dots, \mathbf{y}^{(K-1)}(t)]. \tag{14}$$

As the convolution and matrix permutation are linear operations, we can get:

$$\begin{aligned}\hat{\mathbf{y}}_{i1} &= \sigma_1[[\mathbf{\Pi}^T(\boldsymbol{\alpha}_1 * \mathbf{Y})]_i] = \sigma_1[\mathbf{\Pi}^T[\boldsymbol{\alpha}_1 * \mathbf{Y}]_i] \\ &= \mathbf{\Pi}^T\sigma_1[\boldsymbol{\alpha}_1 * \mathbf{y}_{i0}] = \mathbf{\Pi}^T\mathbf{y}_{i1},\end{aligned} \tag{15}$$

where $\mathbf{y}_{i1}$ is the output of the first layer under unpermuted inputs. (15) is derived based on the pairwise operation σ. As we have shown the output of a single layer is permutation equivalent to its input, it can be concluded that the output of layers $l = 2, 3, \dots, L$ is also permutation equivalent.

Moreover, the exponent of the constant $\mathbf{H}$ matrix can be replaced with a product sequence of time varying $\mathbf{H}(t)$ matrix with the equalities still hold, i.e.

$$\prod_{i=1}^{k-1}\hat{\mathbf{H}}(t-i)\hat{\mathbf{x}}(t-k) = \mathbf{\Pi}^T\prod_{i=1}^{k-1}\mathbf{H}(t-i)\mathbf{x}(t-k). \tag{16}$$

Therefore this comes to the conclusion that $\mathbf{\Phi}(\hat{\mathcal{H}}, \mathbf{A}) = \mathbf{\Pi}^T\mathbf{\Phi}(\mathcal{H}, \mathbf{A})$.

This proposition states the inherent permutation equarvariance property of the Agg-GNN due to the invariant manner in which the aggregation sequence is formed and the resulting convolution structure. These results imply that an appropriately permuted output is obtain from a permuted input.

This permutation equivariance is not only a valuable structure for the parameterization to hold, but is moreover a fundamental property of the wireless resource allocation problem itself. We may then study the effect that a permutation brings to the optimal resource allocation problem. For problem (4), suppose that we have the optimal solution for distribution $m(\mathbf{H}, \mathbf{x})$ as $\mathbf{A}^*$ with the optimal strategy and reward denoted as $\mathbf{\Phi}(\mathcal{H}, \mathbf{A}^*)$ and $\mathbf{r}^*$ respectively. Permute the network with matrix $\mathbf{\Pi} \in \psi$, based on Proposition 1, the parameterized allocation strategy satisfies:

$$\mathbf{\Phi}(\hat{\mathcal{H}}, \mathbf{A}^*) = \mathbf{\Pi}^T\mathbf{\Phi}(\mathcal{H}, \mathbf{A}^*). \tag{17}$$

The performance function then satisfies:

$$\hat{\mathbf{r}} = \int \mathbf{f}\left(\mathbf{\Phi}(\hat{\mathcal{H}}, \mathbf{A}^*), \hat{\mathbf{H}}, \hat{\mathbf{x}}\right) \mathrm{d}\hat{m}(\hat{\mathbf{H}}, \hat{\mathbf{x}}) \tag{18}$$

$$= \int \mathbf{f}\left(\mathbf{\Pi}^T\mathbf{\Phi}(\hat{\mathcal{H}}, \mathbf{A}^*), \hat{\mathbf{H}}, \hat{\mathbf{x}}\right) \mathrm{d}\hat{m}(\hat{\mathbf{H}}, \hat{\mathbf{x}}) \tag{19}$$

$$= \int \mathbf{f}\left(\mathbf{\Pi}^T\mathbf{\Phi}(\hat{\mathcal{H}}, \mathbf{A}^*), \mathbf{\Pi}^T\mathbf{H}\mathbf{\Pi}, \mathbf{\Pi}^T\mathbf{x}\right) \mathrm{d}\hat{m}(\mathbf{\Pi}^T\mathbf{H}\mathbf{\Pi}, \mathbf{\Pi}^T\mathbf{x}) \tag{20}$$

Based on the condition that the performance function $\mathbf{f}$ here we involves is the capacity function, which is permutation equivariant. We have:

$$\mathbf{f}\left(\mathbf{\Pi}^T\mathbf{\Phi}(\hat{\mathcal{H}}, \mathbf{A}^*), \mathbf{\Pi}^T\mathbf{H}\mathbf{\Pi}, \mathbf{\Pi}^T\mathbf{x}\right) = \mathbf{\Pi}^T\mathbf{f}\left(\mathbf{\Phi}(\hat{\mathcal{H}}, \mathbf{A}^*), \mathbf{H}, \mathbf{x}\right). \tag{21}$$

This indicates:

$$\hat{\mathbf{r}} = \int \mathbf{\Pi}^T\mathbf{f}\left(\mathbf{P}(\mathcal{H}), \mathbf{H}, \mathbf{x}\right) \mathrm{d}m(\mathbf{H}, \mathbf{x}) \tag{22}$$

$$= \mathbf{\Pi}^T\mathbf{r} = \int \mathbf{f}\left(\mathbf{\Phi}(\mathcal{H}, \mathbf{A}^*), \mathbf{H}, \mathbf{x}\right) \mathrm{d}m(\mathbf{H}, \mathbf{x}). \tag{23}$$

Therefore, the objective function satisfies: $\mathbf{1}^T\mathbf{r} = \mathbf{1}^T\hat{\mathbf{r}}$. Similarly, we suppose the optimal solution for distribution $\hat{m}(\hat{\mathbf{H}}, \hat{\mathbf{x}})$ is $\hat{\mathbf{A}}^*$ and we have that:

$$\mathbf{\Phi}(\hat{\mathcal{H}}, \hat{\mathbf{A}}^*) = \mathbf{\Pi}^T\mathbf{\Phi}(\mathcal{H}, \hat{\mathbf{A}}^*). \tag{24}$$

$$\hat{\mathbf{r}}^* = \int \mathbf{f}\left(\mathbf{\Phi}(\hat{\mathcal{H}}, \hat{\mathbf{A}}^*), \hat{\mathbf{H}}, \hat{\mathbf{x}}\right) \mathrm{d}\hat{m}(\hat{\mathbf{H}}, \hat{\mathbf{x}}) \tag{25}$$

$$= \mathbf{\Pi}^T\mathbf{r} = \int \mathbf{f}\left(\mathbf{\Phi}(\mathcal{H}, \hat{\mathbf{A}}^*), \mathbf{H}, \mathbf{x}\right) \mathrm{d}m(\mathbf{H}, \mathbf{x}). \tag{26}$$

We also have: $\mathbf{1}^T\mathbf{r} = \mathbf{1}^T\hat{\mathbf{r}}^*$. $\hat{\mathbf{r}}^*$ here is optimal for $\hat{m}(\hat{\mathbf{H}}, \hat{\mathbf{x}})$ and $\mathbf{r}$ is feasible. Based on the properties of optimization, we have:

$$\mathbf{1}^T\hat{\mathbf{r}}^* \geq \mathbf{1}^T\hat{\mathbf{r}} = \mathbf{1}^T\mathbf{r}^*, \mathbf{1}^T\mathbf{r}^* \geq \mathbf{1}^T\mathbf{r} = \mathbf{1}^T\mathbf{r}^*. \tag{27}$$

Therefore, the inequalities must be equalities, which means $\mathbf{A}^*$ is optimal for $\hat{m}(\hat{\mathbf{H}}, \hat{\mathbf{x}})$ and $\hat{\mathbf{A}}^*$ is optimal for $m(\mathbf{H}, \mathbf{x})$. This concludes the proof.

■

Theorem 1 indicates we can train an Agg-GNN with state distribution $m(\mathbf{H}, \mathbf{x})$ and transfer to a permuted wireless network with state distribution $\hat{m}(\hat{\mathbf{H}}, \hat{\mathbf{x}})$ without loss of optimality (see [20] for proof). This result, along with dimension invariance, suggests potential for transference of an Agg-GNN policy to other networks of varying size, which we study numerically in Section 5.

4. ASYNCHRONOUS PRIMAL-DUAL TRAINING

Finding the optimal GNN filter tensor $\mathbf{A}$ requires the solving of a constrained optimization problem in (4). The constrained optimization problems are often solved by converting to the Lagrangian form, i.e.

$$\begin{aligned}\mathcal{L}(\mathbf{A}, \mathbf{r}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathbf{1}^T\mathbf{r} &+ \boldsymbol{\lambda}^T\left[\mathbb{E}\left[\mathbf{f}\left(\mathbf{\Phi}(\mathcal{H}, \mathbf{A}), \mathbf{H}, \mathbf{x}\right)\right] - \mathbf{r}\right] \\ &+ \boldsymbol{\mu}^T\left[\mathbb{E}\left[\mathbf{1}^T\mathbf{\Phi}(\mathcal{H}, \mathbf{A})\right] - P_{max}\mathbf{1}^T\right].\end{aligned} \tag{28}$$

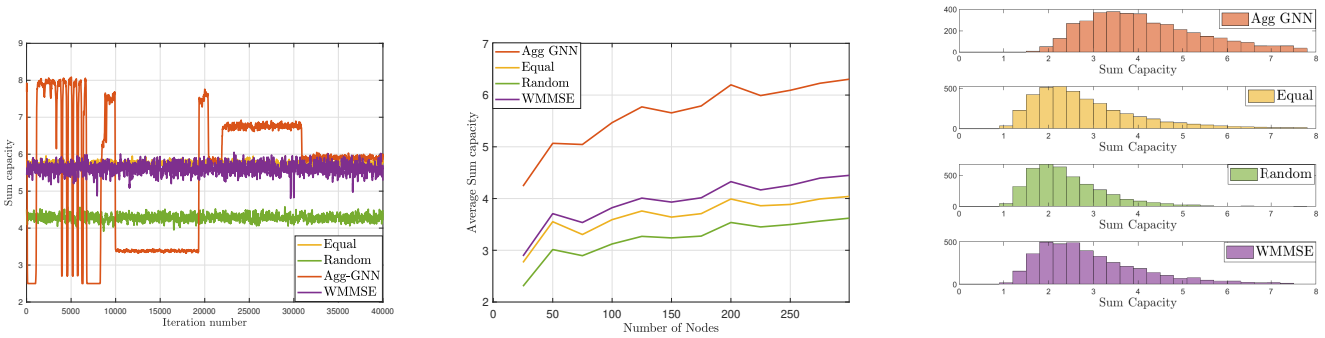

Fig. 2: (left) Performance comparison during training for 25 nodes with 5 hops. (middle) Performance comparison when transferring to networks with the same size. (right) Performance comparison when transferring to larger networks.

The primal-dual method is used to find the saddle-point of the Lagrangian form by updating primal and dual variables alternatively. Despite having a decentralized network structure, the training of the Agg-GNN parameters is done offline and we can thus leverage a centralized, or federated, learning architecture. This is in contrast to online methods, e.g. [18], which make resource allocation decisions during a continuous optimization process and thus require decentralized learning.

The primal-dual method is derived by alternating gradient steps on the primal variables $\mathbf{A}, \mathbf{r}$ and dual variables $\boldsymbol{\mu}, \boldsymbol{\lambda}$. We denote τ as an iteration index and ϵ as the stepsize. Although computing gradients of (28) requires explicit knowledge of $\mathbf{f}$ and the distribution $m(\mathbf{H}, \mathbf{x})$, model-free learning methods such as stochastic gradient and policy gradient methods [21] are typically used to bypass this requirement by sampling the performance under different policies—see [8]. However, because of the asynchronous working patterns of nodes, not all devices will utilize the current parameter tensor $\mathbf{A}(\tau)$ at current index τ. While the centralized learner keeps an iterate $\mathbf{A}(\tau)$, we further define the local copy at node i, $\mathbf{A}_i(\tau)$, and store all together in $\mathbb{A}(\tau) := \{\mathbf{A}_i(\tau)\}_{i=1}^m$. The local copies can be expressed as

$$\mathbf{A}_i(\tau) := \begin{cases} \mathbf{A}(\tau) & \text{if } i \in \mathcal{A}(\tau), \\ \mathbf{A}_i(\tau-1) & \text{if } i \notin \mathcal{A}(\tau). \end{cases} \tag{29}$$

The primal updates with gradient ascent are given by

$$\mathbf{r}(\tau+1) = \mathbf{r}(\tau) + \epsilon[\mathbf{1} - \boldsymbol{\lambda}(\tau)], \tag{30}$$

$$\mathbf{A}(\tau+1) = \mathbf{A}(\tau) + \epsilon \nabla_{\mathbf{A}} \mathbb{E}\left[\mathbf{f}\left(\boldsymbol{\Phi}(\mathcal{H}, \mathbb{A}), \mathbf{H}, \mathbf{x}\right)\right] \boldsymbol{\lambda}(\tau) + \epsilon \nabla_{\mathbf{A}} \mathbb{E}\left[\mathbf{1}^T \boldsymbol{\Phi}(\mathcal{H}, \mathbb{A})\right] \boldsymbol{\mu}(\tau) \tag{31}$$

The dual updates are then likewise given by

$$\boldsymbol{\mu}(\tau+1) = \left[\boldsymbol{\mu}(\tau) - \epsilon\left[\mathbb{E}\left[\mathbf{1}^T \boldsymbol{\Phi}(\mathcal{H}, \mathbb{A})\right] - P_{max} \mathbf{1}^T\right]\right]^+, \tag{32}$$

$$\boldsymbol{\lambda}(\tau+1) = \boldsymbol{\lambda}(\tau) - \epsilon\left[\mathbb{E}\left[\mathbf{f}\left(\boldsymbol{\Phi}(\mathcal{H}, \mathbb{A}), \mathbf{H}, \mathbf{x}\right)\right] - \mathbf{r}(\tau)\right] \tag{33}$$

We emphasize in (30)-(33), that the model-free evaluations of gradients are taken with respect to the *asynchronized* versions of the policies as given by the paramter tensors $\mathbb{A}(\tau)$, rather than the centralized iterate $\mathbf{A}(\tau)$.

5. SIMULATION RESULTS

In this section, we provide a numerical study for the problem in (4). We focus on the wireless ad-hoc networks with $m = n = 25$. We first drop m transmitters randomly uniformly within the range of $\mathbf{a}_i \in [-m, m]^2, i = 1, 2 \ldots, m$. Each paired receiver is located randomly within $\mathbf{b}_i \in [\mathbf{a}_i + [-m/4, m/4]]^2$. The fading channel state is composed of a large-scale pathloss gain and a random fast fading gain, which can be written as: $h_{ij} = h_{ij}^l h_{ij}^f$, $h_{ij}^l = \|\mathbf{a}_i - \mathbf{b}_j\|^{-2.2}$, $h_{ij}^f \sim$ Rayleigh(2). The local and total resource budgets are set as: $p_0 = 2$ and $P_{max} = m$. We model the active patterns of nodes by considering a collection of $N_{act} = 5$ active subsets denoted as $\{\mathcal{A}_n\}_{n=1}^{N_{act}}$, where $\mathcal{A}_n \subseteq \{1, 2, \ldots, m\}$ for all n. We set the number of active nodes at each time step to be a Poisson distributed random variable with $\lambda = 12$. At each time t, we randomly draw a set of active nodes $\mathcal{A}(t) \in \{\mathcal{A}_n\}_{n=1}^{N_{act}}$. We construct a CNN with $L = 10$ hidden layers, each with a filter (hop) with length $K_l = 5$ and a standard ReLu non-linear activation function, i.e. $\boldsymbol{\sigma}(\mathbf{z}) = [\mathbf{z}]_+$. The final layer normalizes the outputs through a sigmoid function.

We compare our algorithm with three existing heuristic methods for solving the original optimization problem: (i) WMMSE [12] in a distributed setting with limited communication exchanges, (ii) Equal allocation, i.e. assign P_{max}/m to all nodes, and (iii) random allocation, i.e. each node transmits at p_0 with probability $P_{max}/(p_0 m)$ to meet the total average power constraint in (4). Both the proposed Agg-GNN and WMMSE require state information exchange; in comparisons we keep the exchanges complexity equal between these methods, i.e. $K = 5$.

In Fig. 2 we evaluate the performance of the Agg-GNN policy and learning method in the asynchronous network. In the left figure, we show the performance relative to baselines during the training procedure on a fixed network and see that, after convergence, the Agg-GNN narrowly exceeds the performance of the model-based WMMSE method. We next evaluate transference capabilities of the learned Agg-GNN policy on new randomly drawn networks. In the middle figure, we show a histogram of total capacity of the network using the policy in new networks of size 25 compared to baselines. In the right figure, we plot the performance of the learned network and baselines on new networks with growing size. We can see that the proposed Agg-GNN outperforms other heuristic

algorithms in all cases. This transference across networks indicates that decentralized strategies for large scale wireless networks can be obtained by training an Agg-GNN on a similar network or a representative smaller network due to their permutation invariance.

6. CONCLUSION

We consider the problem of asynchronous resource allocation in wireless networks. We train an Aggregation Graph Neural Network with primal-dual model-free unsupervised learning method. Each node aggregates information from its active neighbors with certain delay. The information sequence incorporates the underlying network structure as well as the asynchrony of this system. We propose a policy based on Aggregation Graph Neural Networks with permutation invariance to network structure. The performance of this policy is verified with numerical simulation results compared with heuristic baseline methods.

7. REFERENCES

[1] J. Zhang and D. Zheng, "A stochastic primal-dual algorithm for joint flow control and mac design in multi-hop wireless networks," in *Information Sciences and Systems, 2006 40th Annual Conference on*. IEEE, 2006, pp. 339–344.

[2] A. K. Sangaiah, A. A. R. Hosseinabadi, M. B. Shareh, S. Y. Bozorgi Rad, A. Zolfagharian, and N. Chilamkurti, "Iot resource allocation and optimization based on heuristic algorithm," *Sensors*, vol. 20, no. 2, p. 539, 2020.

[3] M. B. Gawali and S. K. Shinde, "Task scheduling and resource allocation in cloud computing using a heuristic approach," *Journal of Cloud Computing*, vol. 7, no. 1, p. 4, 2018.

[4] D. Xu, X. Che, C. Wu, S. Zhang, S. Xu, and S. Cao, "Energy-efficient subchannel and power allocation for hetnets based on convolutional neural network," *arXiv preprint arXiv:1903.00165*, 2019.

[5] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for wireless resource management," *IEEE Transactions on Signal Processing*, vol. 66, no. 20, pp. 5438–5453, 2018.

[6] M. Eisen, C. Zhang, L. F. Chamon, D. D. Lee, and A. Ribeiro, "Learning optimal resource allocations in wireless systems," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2775–2790, 2019.

[7] M. Lee, G. Yu, and G. Y. Li, "Graph embedding based wireless link scheduling with few training samples," *arXiv preprint arXiv:1906.02871*, 2019.

[8] M. Eisen and A. R. Ribeiro, "Optimal wireless resource allocation with random edge graph neural networks," *IEEE Transactions on Signal Processing*, 2020.

[9] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "Graph neural networks for scalable radio resource management: Architecture design and theoretical analysis," *arXiv preprint arXiv:2007.07632*, 2020.

[10] A. Chavez, A. Moukas, and P. Maes, "Challenger: A multiagent system for distributed resource allocation," in *Proceedings of the first international conference on Autonomous agents*, 1997, pp. 323–331.

[11] U. Challita, L. Dong, and W. Saad, "Proactive resource management in lte-u systems: A deep learning perspective," *arXiv preprint arXiv:1702.07031*, 2017.

[12] Q. Shi, M. Razaviyayn, Z.-Q. Luo, and C. He, "An iteratively weighted mmse approach to distributed sum-utility maximization for a mimo interfering broadcast channel," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4331–4340, 2011.

[13] H. Lee, S. H. Lee, and T. Q. Quek, "Deep learning for distributed optimization: Applications to wireless resource management," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2251–2266, 2019.

[14] H. Ye, G. Y. Li, and B.-H. F. Juang, "Deep reinforcement learning based resource allocation for v2v communications," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3163–3173, 2019.

[15] Y. S. Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2239–2250, 2019.

[16] N. Naderializadeh, J. Sydir, M. Simsek, and H. Nikopour, "Resource management in wireless networks via multi-agent deep reinforcement learning," *arXiv preprint arXiv:2002.06215*, 2020.

[17] K. Rajawat, N. Gatsis, and G. B. Giannakis, "Cross-layer designs in coded wireless fading networks with multicast," *IEEE/ACM Transactions on Networking*, vol. 19, no. 5, pp. 1276–1289, 2011.

[18] A. S. Bedi and K. Rajawat, "Asynchronous incremental stochastic dual descent algorithm for network resource allocation," *IEEE Transactions on Signal Processing*, vol. 66, no. 9, pp. 2229–2244, 2018.

[19] F. Gama, A. G. Marques, G. Leus, and A. Ribeiro, "Convolutional neural network architectures for signals supported on graphs," *IEEE Transactions on Signal Processing*, vol. 67, no. 4, pp. 1034–1049, 2019.

[20] "Unsupervised learning for asynchronous resource allocation in ad-hoc wireless networks." [Online]. Available: https://zhiyangw.com/Papers/unsupervised-icassp21.pdf

[21] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Advances in neural information processing systems*, 2000, pp. 1057–1063.