GRLinQ: A Hybrid Model/Data-Driven Spectrum Sharing Mechanism for Device-to-Device Communications

Zhiwei Shan, *Student Member, IEEE*, Xinping Yi, *Member, IEEE*, Le Liang, *Member, IEEE*,
Chung-Shou Liao, *Senior Member, IEEE*, and Shi Jin *Fellow, IEEE*

Abstract—Device-to-device (D2D) spectrum sharing in wireless communications is a challenging non-convex combinatorial optimization problem, involving entangled link scheduling and power control in a large-scale network. The state-of-the-art methods, either from a model-based or a data-driven perspective, exhibit certain limitations such as the critical need for channel state information (CSI) and/or a large number of (solved) instances (e.g., network layouts) as training samples. To advance this line of research, we propose a novel hybrid model/data-driven spectrum sharing mechanism with graph reinforcement learning for link scheduling (GRLinQ), injecting information theoretical insights into machine learning models, in such a way that link scheduling and power control can be solved in an intelligent manner. Through an extensive set of experiments, GRLinQ demonstrates superior performance to the existing model-based and data-driven link scheduling and/or power control methods, with a relaxed requirement for CSI, a substantially reduced number of unsolved instances as training samples, a possible distributed deployment, reduced online/offline computational complexity, and more remarkably excellent scalability and generalizability over different network scenarios and system configurations.

I. INTRODUCTION

Device-to-device (D2D) communications are going to play a central role in future wireless networks, enabling direct message exchange between devices without resorting to cellular infrastructure. From an information theoretical perspective, the underlying physical model of D2D communications is a Gaussian interference channel (GIC), in which the capacity characterization has been a long-standing open problem. Among various sophisticated coding techniques for GIC, treating interference as noise (TIN) has been regarded as a low-complexity and robust one deployable in large-scale D2D networks. Remarkably, it has been proven that, with proper link scheduling, for a selected subset of D2D links

Z. Shan is with University of Liverpool, Liverpool L69 3GJ, United Kingdom, and also with National Tsing Hua University, Hsinchu 30013, Taiwan, ROC. Email: zshan@liverpool.ac.uk.

X. Yi, L. Liang, and S. Jin are with National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China. L. Liang is also with Purple Mountain Laboratories, Nanjing 211111, China. Email: {xyi, lliang, jinshi}@seu.edu.cn.

C.-S. Liao is with National Tsing Hua University, Hsinchu 30013, Taiwan, ROC. Email: csliao@ie.nthu.edu.tw.

with certain conditions satisfied, TIN at the receivers and power control at the transmitters can achieve the information-theoretical optimality in an asymptotic sense [2]–[10].

Nevertheless, D2D link scheduling and power control is still a challenging problem, aiming to maximize overall network throughput by activating a subset of D2D links with proper power allocation for each link. The primary challenge arises from the entangled nature of link scheduling and power control, where the former is a non-convex combinatorial optimization problem, particularly formidable in large-scale networks, and the latter can be seen as a continuous version of the former, yet optimizing the latter is critically dependent on the former. Due to the complex entanglement of link scheduling and power control, conventional wisdom suggests tackling the two problems separately, yielding a number of effective and low-complexity methods in the literature.

Current state-of-the-art link scheduling methods include traditional model-based approaches and emerging data-driven techniques. Model-based methods such as FlashLinQ [11], ITLinQ [12], and ITLinQ+ [13] design sequential link selection algorithms from an information-theoretical perspective. However, the adjustable parameters they use rely highly on network scenarios and system parameters. It is time/effort-consuming to find appropriate values for each different network scenario, while the same set of parameters apparently does not work across all networks. FPLinQ [14] approaches the optimization problem from a fractional programming standpoint. It relaxes the requirements of tuning parameters but requires centralized processing with accurate CSI, which is difficult to obtain in large-scale networks, especially in a distributed setting. When relaxing the CSI requirement, data-driven approaches hinge on the power of deep neural networks (DNNs), leaving aside insights obtained from model-based methods. As such, SpatialLinQ [15] involves learning interference patterns among neighboring transmitters/receivers through kernels, and then generating outcomes through DNNs. However, it requires hundreds of thousands of solved instances as training samples, i.e., different network layouts and the corresponding link scheduling solutions by e.g., FPLinQ. A graph-embedding-based method, GELinQ [16], reduces this number to hundreds of training network layouts, though at the cost of degraded performance. Although promising, the existing data-driven approaches such as SpatialLinQ and GELinQ simply put network layout or geographical location as node features, without considering the insights obtained

TABLE 1
SUMMARY OF STATE-OF-THE-ART LINK SCHEDULING METHODS

Method	FlashLinQ [11], ITLinQ [12], ITLinQ+ [13]	FPLinQ [14]	SpatialLinQ [15]	GELinQ [16]	GRLinQ (this work)
Driving Mechanism	Model-based	Model-based	Data-driven	Data-driven	Hybrid model/data-driven
Methodology	Heuristic sequential selection	Mathematical optimization	Machine learning	Machine learning	Machine learning
Key Idea	TIN conditions	Fractional programming	Kernel & DNN	GNN & DNN	RL & GNN
CSI	Yes	Yes	No	No	No
System Architecture	Decentralized	Centralized	Centralized	Distributed	Distributed
Number of Training Network Layouts	/	/	Hundreds of thousands	Hundreds	Hundreds
Generalizability	/	/	Strong	Good	Strong
Complexity	$O(N^2)$	$O(N^2)$	$O(N)$	$O(N)$	$O(N)$

from the model-based methods, rendering degraded sum rate performance and/or reduced generalizability.

Current state-of-the-art power control methods encompass traditional model-based approaches, and model/data-driven deep learning approaches. The previously mentioned model-based ITLinQ+ [13] provides a power control solution together with link scheduling, yet its effectiveness relies highly on the exponentially decayed channel strength. While the optimization-based FPLinQ [14] offers an effective power control solution to practical channels, its centralized processing requirement and high computational complexity make it challenging to be deployed in practice. The weighted minimum mean squared error (WMMSE) [17] method focuses on minimizing a reformulated weighted mean-squared-error cost function, yielding superior sum rate performance. Although these methods can yield results very close to the optimum, they are often criticized for their high complexity and the need for accurate CSI acquisition [18]–[20]. The model-driven method, unfolded WMMSE (UWMMSE) [19], unrolls the WMMSE iterations into a cascade of DNN layers, each of which retains the same update structure as the original algorithm, but with parameters learned by graph neural networks (GNNs) from data. Compared to the number of iterations required by WMMSE, UWMMSE can use fewer layers to achieve similar results, thereby reducing computational complexity. PCGNN [21], as a data-driven method, utilizes a GNN to directly learn the power allocation function, achieving near-optimal performance with much fewer training samples. However, all the aforementioned power control methods rely on exact CSI and still have relatively high computational cost, i.e., quadratic complexity with respect to the number of D2D links.

To advance this line of research, we propose a new hybrid model/data-driven intelligent spectrum sharing framework, named GRLinQ, addressing link scheduling and power control by leveraging advancements in graph reinforcement learning and some information theoretical insights. Specifically, we reformulate the link scheduling and power control problem as Markov decision processes (MDPs), breaking down the complex combinatorial optimization problem into multiple steps. By employing reinforcement learning (RL), we can learn and optimize the strategy for state transitions based on current network conditions. In doing so, it relaxes the need for labeled data, which is usually obtained from solved

instances by e.g., FPLinQ. As a result, GRLinQ achieves more efficient resource utilization and improved overall performance, making it highly adaptable and effective in real-world scenarios. Notably, GRLinQ employs GNNs as the policy and value networks within the RL framework, so as to effectively capture graph structures in interference graphs, demonstrating the significance and efficiency of topology awareness in both link scheduling and power control. Another key enabling factor is that GRLinQ incorporates insights derived from model-based approaches, e.g., FlashLinQ, ITLinQ, and ITLinQ+. These insights are carefully integrated into the node features of the GNNs, enriching GRLinQ with a deeper understanding of network behaviors and performance measures.

Through an extensive set of experiments, GRLinQ demonstrates superior sum rate performance compared to the state-of-the-art link scheduling and power control benchmarks, excellent scalability across different network scenarios (e.g., network sizes, network density), and strong generalization performance (e.g., unseen carrier frequency, D2D link distance). In addition, GRLinQ also relaxes the requirements of accurate CSI and a large number of solved instances as training samples, and enables potential distributed deployment. The comparison with state-of-the-art methods is summarized in Table 1 and 2. The proposed GRLinQ appears to be the most promising for dealing with D2D spectrum sharing with a hybrid model/data-driven machine learning methodology. Specifically, GRLinQ possesses the following advantages over the state-of-the-art link scheduling and power control methods, according to the comprehensive experiments.

- For the link scheduling problem, GRLinQ achieves comparable sum rate performance as FPLinQ, significantly improved over FlashLinQ, ITLinQ, and ITLinQ+, while requiring only pairwise distance information rather than accurate CSI. Meanwhile, GRLinQ surpasses SpatialLinQ and GELinQ by a large margin, whilst eliminating the need of solved instances as training samples and reducing the amount of required training data.
- For the power control problem, GRLinQ-pc also approaches the best-known FPLinQ-pc and WMMSE for different network sizes, and strictly surpasses the existing model/data-driven deep learning approaches such as UWMMSE and PCGNN, while requiring only distance information rather than accurate CSI and substantially

TABLE 2
SUMMARY OF STATE-OF-THE-ART POWER CONTROL METHODS

Method	WMMSE [17]	FPLinQ-pc [14]	UWMMSE [19]	PCGNN [21]	GRLinQ-pc (this work)
Driving Mechanism	Model-based	Model-based	Model-driven	Data-driven	Hybrid model/data-driven
Methodology	Mathematical optimization	Mathematical optimization	Machine learning	Machine learning	Machine learning
Key Idea	Minimum mean square error	Fractional programming	Unfolding WMMSE & GNN	GNN	RL & GNN
CSI	Yes	Yes	Yes	Yes	No
System Architecture	Distributed	Centralized	Distributed	Distributed	Distributed
Number of Training Network Layouts	/	/	Hundreds of thousands	Tens of thousands	Hundreds
Generalizability	/	/	Good	Strong	Strong
Complexity	$O(N^2)$	$O(N^2)$	$O(N^2)$	$O(N^2)$	$O(N)$

reduced training data.

- Thanks to topology-awareness (i.e., K-nearest interference graphs) and information-theory-inspired feature design, GRLinQ possesses excellent scalability from trained smaller-sized to testing large-scale network scenarios and strong generalizability from training to testing under different unseen system configurations. Moreover, GR-LinQ demonstrates great adaptiveness against network dynamics due to devices' mobility.

The rest of this paper is organized as follows. Section II describes wireless link scheduling and power control in D2D networks, and constructs K-nearest interference graphs. The existing link scheduling and power control methods are reviewed in Section III. The proposed GRLinQ framework is detailed in Section IV, and the extensive experiments are presented in Section V. We conclude the paper in Section VI.

II. PROBLEM STATEMENT

A. Wireless Link Scheduling and Power Control

Consider a D2D wireless network $\mathcal{D}$ with N unicast links, denoted by $\mathcal{D} = \{D_i\}_{i=1}^N$. Each D2D link D_i consists of a transmitter Tx$_i$ and a paired receiver Rx$_i$, each equipped with one single antenna. Modeled as a Gaussian interference channel, the received signal at Rx$_i$ can be written by

$$y_i = h_{ii}z_i + \sum_{j=1,j\neq i}^N h_{ji}z_j + n_i \tag{1}$$

where h_{ji} is the channel coefficient between Tx$_j$ and Rx$_i$ integrating both large-scale fading (i.e., path-loss) and small-scale fading coefficients, z_i is the transmitted signal from Tx$_i$ with the transmit power constraint $E[|z_i|^2] \leq p_i$, for all i, and n_i is the additive white Gaussian noise, i.e., $n_i \sim \mathcal{CN}(0, \sigma^2)$. We consider the block fading channel, where link scheduling is conducted within a coherence block, such that channel coefficients remain unchanged during communication.

Link scheduling. Let us introduce $x_i \in \{0,1\}$ as an indicator variable for each link D_i, which equals 1 if the link is scheduled and 0 otherwise. Then, the achievable data rate of link D_i when treating interference as noise is given by

$$R_i(\boldsymbol{x}) = \log\left(1 + \frac{|h_{ii}|^2 p_i x_i}{\sum_{j\neq i}|h_{ji}|^2 p_j x_j + \sigma^2}\right), \tag{2}$$

where $\boldsymbol{x}$ is a collection of $\{x_i\}_{i=1}^N$, and for convenience, we assume full power transmission at the Txs. Due to mutual interference between links, activating too many links can lead to a decrease in data rate. Therefore, the task of wireless link scheduling is to activate a suitable set of links, aiming to maximize some utility function of the average rates.

In this paper, we formulate the link scheduling problem as maximizing the weighted sum rate over the N links for each scheduling slot, i.e.,

$$\max_{\boldsymbol{x}} \quad \sum_{i=1}^N w_i R_i(\boldsymbol{x}) \tag{3}$$
$$\text{subject to } x_i \in \{0,1\}, \ \forall i,$$

where the weight w_i signifies the priority assigned to link D_i.

Power control. To further enhance network performance, power control can be introduced as an additional mechanism. In this context, the indicator variable x_i is relaxed to $x_i \in [0,1]$, allowing for a continuous range of power allocation levels for each transmitter Tx$_i$. This adjustment enables more fine-grained control over the transmit power, which can significantly improve the achievable data rates.

B. Topology and K-nearest Interference Graphs

The D2D network can be represented by a topology graph, a bipartite graph that captures the Gaussian interference channel with edge weights being channel coefficients, as shown in Fig.1a. Each transmitter intends to send a message to its corresponding receiver, simultaneously causing interference to other receivers.
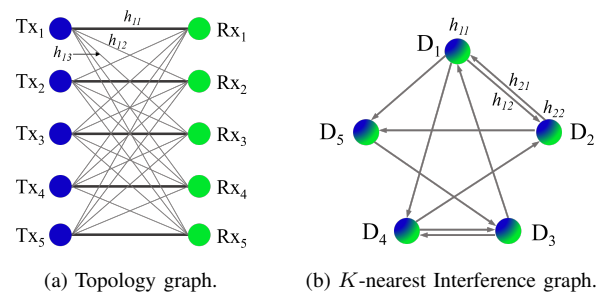

(a) Topology graph. (b) K-nearest Interference graph.

Fig. 1. A D2D network with 5 links. (a) topology graph, and (b) the corresponding K-nearest interference graph with $K = 2$.

The D2D network can also be represented by a K-nearest interference graph, as illustrated in Fig.1b. Such a graph captures the most significant interfering links of each receiver.

Definition 1 (K-nearest Interference Graph): Given an N-link D2D network $\mathcal{D}$, the corresponding K-nearest interference graph is defined as a directed graph, where each D2D link D_i serves as a vertex, and the directed edge (D_i, D_j) indicates the interfering link from Tx$_i$ to Rx$_j$. The number of incoming edges to any D_i is K, i.e., $|\mathcal{N}(i)| = K \le N-1$, for all i, where $\mathcal{N}(i)$ represents the set of K vertices exerting the most substantial influence to D_i, i.e., $|h_{ji}|^2 \ge |h_{ki}|^2$ for all $j \in \mathcal{N}(i)$ and $k \notin \mathcal{N}(i)$. In the following, unless otherwise specified, we use the distance d_{ji} between Tx$_j$ and Rx$_i$ to replace h_{ji} as the criterion for the K-nearest interference graph. Accordingly, the condition is modified to $d_{ji} \le d_{ki}$ for all $j \in \mathcal{N}(i)$ and $k \notin \mathcal{N}(i)$.

The K-nearest interference graph imposes a structural property that can be exploited in algorithm designs. When $K = N - 1$, the $(N - 1)$-nearest interference graph becomes a fully connected directed graph, accounting for all interfering links.

III. The State-of-the-Art

A. Link Scheduling

The state-of-the-art link scheduling mechanisms include model-based approaches, such as FlashLinQ [11], ITLinQ [12], ITLinQ+ [13], FPLinQ [14], and data-driven approaches, such as SpatialLinQ [15], GELinQ [16], to name a few.

1) Model-based approaches: Model-based approaches rely on theoretical frameworks and predefined models for algorithm designs. FlashLinQ, ITLinQ, and ITLinQ+ share a unified sequential selection framework. Denote the set of active links as $\mathcal{S}$, initially an empty set. The link scheduling mechanisms sequentially traverse all D2D links and evaluate each D2D link against predefined conditions. If a particular link satisfies the "condition", it is marked as active and included in $\mathcal{S}$; otherwise, it is labeled as inactive. FlashLinQ, ITLinQ, and ITLinQ+ distinguish themselves with different conditions.

For notational convenience, we define the signal-to-noise ratio (SNR), interference-to-noise ratio (INR), and signal-to-interference ratio (SIR) as:

$$\text{SNR}_i = \frac{|h_{ii}|^2 p_i}{\sigma^2}, \text{INR}_{ji} = \frac{|h_{ji}|^2 p_j}{\sigma^2}, \text{SIR}_{ji} = \frac{|h_{ii}|^2 p_i}{|h_{ji}|^2 p_j}. \quad (4)$$

FlashLinQ. The link selection criterion of FlashLinQ [11] is that, a link D_i can be selected if it does not experience significant interference from and does not cause significant interference to other links in the selected set $\mathcal{S}$, i.e.,

$$\text{FlashLinQ: } \begin{cases} \text{SIR}_{ij} \ge \theta, \ \forall j \in \mathcal{S}, \\ \frac{|h_{ii}|^2 p_i}{\sum_{j \in \mathcal{S}} |h_{ji}|^2 p_j} \ge \theta, \end{cases} \quad (5)$$

where θ is a predefined threshold. The effectiveness of FlashLinQ significantly hinges on the parameter θ, and selecting an appropriate θ can pose practical challenges.

ITLinQ & ITLinQ+. Both ITLinQ [12] and ITLinQ+ [13] are inspired by the TIN optimality conditions, which

are relaxed and translated into two criteria that can be locally assessed in a distributed manner:

$$\text{ITLinQ: } \begin{cases} M\text{SNR}_i^\eta \ge \max_{j \in \mathcal{S}} \text{INR}_{ji}, \\ M\text{SNR}_i^\eta \ge \max_{j \in \mathcal{S}} \text{INR}_{ij}, \end{cases} \quad (6)$$

and

$$\text{ITLinQ+: } \begin{cases} \text{SNR}_i^\eta \ge \max_{j \in \mathcal{S}} \left\{ \frac{\text{INR}_{ji}}{(\min_{k \in \mathcal{S}, k \ne j} \text{INR}_{jk})^\gamma} \right\}, \\ \text{SNR}_i^\eta \ge \max_{j \in \mathcal{S}} \left\{ \frac{\text{INR}_{ij}}{(\min_{k \in \mathcal{S}, k \ne j} \text{INR}_{kj})^\gamma} \right\}, \end{cases} \quad (7)$$

where M, η and γ are designing parameters. Similar to FlashLinQ, the performance of ITLinQ and ITLinQ+ is highly sensitive to the values of the design parameters. Nevertheless, it is difficult to choose the best parameters in practice.

FPLinQ. In FPLinQ [14], the original combinatorial problem in Eq-(3) is reformulated as an equivalent fractional programming form, by which a sub-optimal solution is proposed to iteratively solve the reformulated problem within a finite number of iterations. Each iteration analytically determines the link schedules with the help of auxiliary variables, thus creating a centralized mechanism for scheduling. FPLinQ is considered the state-of-the-art model-based link scheduling method in terms of data rate performance, although it incurs high computational complexity due to the iterative algorithm and the centralized computation. Its ability to achieve superior performance in comparison to other methods makes it highly valued in the development of data-driven approaches by providing high-quality solved instances of training datasets.

2) Data-driven approaches: In contrast to the above model-based methods, data-driven approaches learn a complex mapping/function, e.g., a link scheduler, from training data. In particular, SpatialLinQ and GELinQ are two typical data-driven link scheduling approaches, learning an end-to-end link scheduler either through unsupervised or supervised learning by extracting certain patterns from solved link selection instances generated from e.g., FPLinQ.

SpatialLinQ. SpatialLinQ [15] relaxes the need for CSI to the geographic locations of transmitters/receivers for link scheduling. By employing convolutional neural networks to learn the geographical locations of interfering or interfered nodes, SpatialLinQ determines if certain links are selected. While competitively effective compared to FPLinQ without requiring precise CSI, a notable challenge is a substantial demand for a large amount of training samples, posing constraints on both memory and time resources during training.

GELinQ. By representing the D2D network as an interference graph, a fully-connected directed graph, GELinQ [16] utilizes graph embedding to convert the D2D network into a low-dimensional space. Then it addresses the binary classification problem with a multi-layer classifier. Despite reducing the required number of training samples, this approach is trained with solved instances being labels generated by FPLinQ, resulting in some compromise in performance.

It is worth noting that both supervised and unsupervised training methods have been explored by SpatialLinQ [15] and GELinQ [16]. As the supervised versions employ datasets generated by FPLinQ for training, their sum rates are inherently limited by the best results of FPLinQ. While the unsupervised approach exhibits a marginal improvement in sum rates

compared to the supervised counterpart, there is still some gap to FPLinQ. These data-driven approaches hinge critically on training data and heuristics, leaving aside wisdom/insights obtained from model-based methods.

B. Power Control

State-of-the-art power control mechanisms can be classified into model-based approaches, such as WMMSE [17] and FPLinQ-pc [14], model-driven, such as UWMMSE [19], and data-driven approaches, such as PCGNN [21], among others.

1) Model-based approaches: WMMSE [17] and FPLinQ-pc [14] both solve power control problems using mathematical optimization techniques. WMMSE leverages the block coordinate descent method, iteratively minimizing a reformulated weighted mean-squared-error cost function. FPLinQ-pc, a version of FPLinQ specifically for power control, addresses the issue from the fractional programming perspective, sharing a similar procedure with the original FPLinQ approach.

2) Model-driven approaches: Model-driven deep learning techniques integrate iterative communication algorithms with deep learning tricks, thereby reducing computational resources and training duration [22]. For example, UWMMSE [19] "unfolded" the iterative WMMSE algorithm into a layered structure, where each layer mimics one iteration with certain adjustable parameters optimized through learning. Compared to the classical iterative WMMSE, UWMMSE leverages the learning-to-optimize mechanisms with GNNs to achieve faster convergence and comparable optimization performance.

3) Data-driven approach: PCGNN [21] leverages the message passing GNNs to solve the graph optimization problem of power allocation in an end-to-end data-driven manner. Each GNN layer consists of the aggregation and combination operations parameterized by multi-layer perceptrons (MLPs) to produce updated hidden representations from information obtained from the neighboring nodes. PCGNN is terminated after a number of GNN layers with an MLP layer to map hidden representations to power allocation variables.

Notably, these power control mechanisms, no matter model-based or model/data-driven, all require exact CSI, and their computational complexity is $O(N^2)$.

IV. HYBRID MODEL/DATA-DRIVEN GRAPH REINFORCEMENT LEARNING SPECTRUM SHARING

A. Overview of Our Approach

As mentioned above, the model-based link scheduling and all previously reviewed power control approaches rely critically on the accuracy of CSI. The data-driven approaches require a large number of samples or labeled training samples generated from the solved instances by e.g., FPLinQ.

To relax the requirements of accurate CSI and massive training samples, we propose a hybrid model/data-driven framework that leverages the advantages of both model-based and data-driven methods. Our framework incorporates a specially designed Markov decision process (MDP) tailored for each task. We designate the approach for link scheduling as GRLinQ and for power control as GRLinQ-pc. Both GRLinQ and GRLinQ-pc models share the same network architectures

and feature designs, yet differ in their MDP components, allowing each to effectively tackle its specific interest.

Motivated by [23]–[29], we employ a graph reinforcement learning (GRL) framework with the actor-critic training method [30] to (1) disseminate the decision-making process across multiple iterations via reinforcement learning (RL); and (2) parameterize both the policy and value networks with graph neural networks (GNNs). The use of RL relaxes the need for obtaining solved instances as training data, where the network layouts without link scheduling solutions are sufficient. By translating some domain experts' knowledge into node features, GRLinQ achieves significant improvements over the state-of-the-art benchmarks.

Our approach takes the K-nearest interference graph $\mathcal{G}_d = (\mathcal{V}, \mathcal{E})$ as the input, where $\mathcal{V}$ represents the set of nodes and $\mathcal{E}$ represents the set of edges. When generating the K-nearest interference graph, we consider the Tx-Rx distance instead of the channel gain, hence eliminating the need for CSI. The node features are specified later. At each iteration, the agent (policy network) makes a decision to some of vertices and postpones the remaining vertices to later iterations. This process will be repeated until all vertices have been given decisions. The framework of GRLinQ is illustrated in Figure 2. Thanks to the inherent randomness of RL, the same test can be conducted in parallel without loss of robustness and reliability.

We introduce the proposed method in the following order. First, we present the RL framework within the context of Markov decision processes (MDPs), followed by an overview of the PPO algorithm employed in this study. Next, we introduce the proposed policy network and value network, both are GNNs. We then detail the specialized design of the initial feature set for the GNN inputs, which together constitute the complete GRLinQ model. Finally, we analyze the computational complexity of the proposed model.

B. Reinforcement Learning

We incorporate the K-nearest interference graphs along with their associated features as the environmental information. Due to the fundamental differences between the discrete nature of link scheduling and the continuous nature of power control, the MDP design is adapted accordingly to effectively capture these distinctions. In the following, we present the MDP formulations for each component.

1) Link scheduling: **State.** A state $\boldsymbol{s} = [s_i : i \in \mathcal{V}]$ is a sequence of operations on the given graph $\mathcal{G}_d$. Each node's state can be one of three possibilities: $\{active, inactive, pending\}$. Initially, the state $\boldsymbol{s}$ is set such that all elements are in the "pending" state. The algorithm terminates either when no nodes in the graph remain in the "pending" state or when the iterations limit, denoted as T, is reached.

Action. Given a state and the environmental information, the agent will produce an action $\boldsymbol{a} = [a_i : i \in \mathcal{V}]$. Each node's action also assumes one of the three previously mentioned states. This signifies that the agent has the flexibility to make decisions regarding "pending" nodes—either scheduling them as "active" or "inactive"—or to defer the decision by maintaining them in the "pending" state for future iterations.

This article has been accepted for publication in IEEE Transactions on Communications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2025.3582022
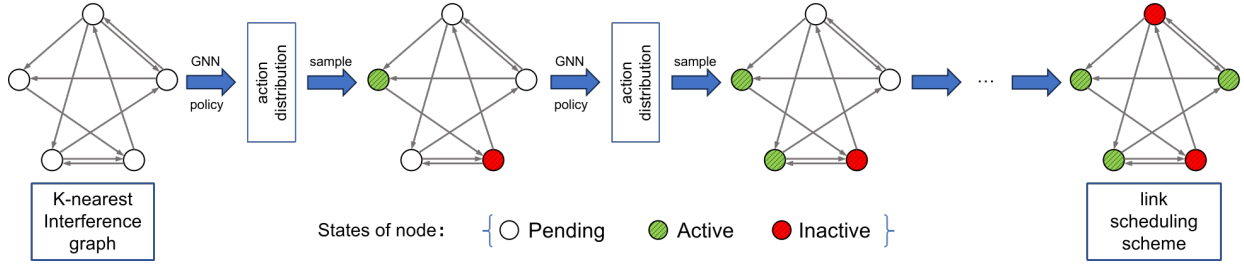
6

Fig. 2. Illustration of the proposed framework. All D2D pairs are initially set to a pending state. During each iteration, the policy network makes decisions based on the current state. The policy network has considerable flexibility to classify any number of D2D pairs as active, inactive, or to retain them in the pending state. The model terminates once all D2D pairs have exited the pending state.
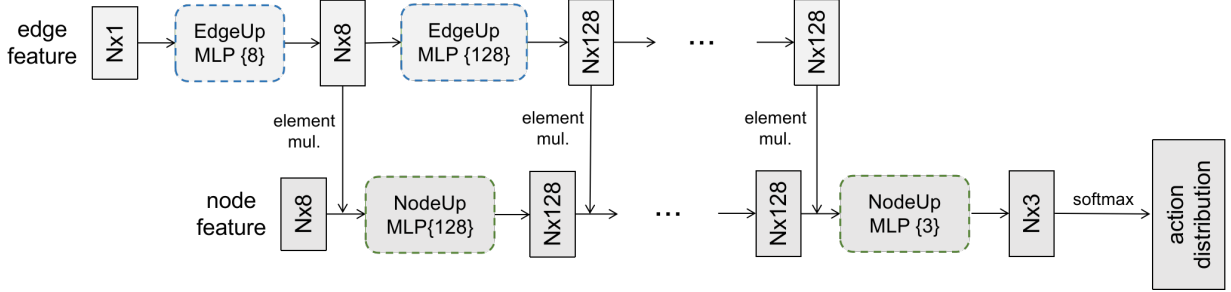

Fig. 3. **Policy network architectures:** The architecture of our proposed MPGNN policy network. Edge features are first updated by an EdgeUpdate block. Following the edge update, node features are combined with edge features and then updated by the NodeUpdate block. These processes repeat for L layers, and the final layer produces action distributions using the updated node features.
EdgeUp block: This block is a multi-layer perceptron (MLP) that updates the edge features. This transformation is applied at both the initial stage and during subsequent layers.
NodeUp block: This block aggregates messages from neighboring nodes and then applies an MLP to the concatenated node features and aggregated messages, resulting in updated node features.

Transition. The RL transits from state s to the next state s' through overwriting (or updating) the previously "pending" nodes with their corresponding actions in a.

Reward. The reward function, denoted as $r(s, a)$, is defined as the difference in the objective function Eq-(3) before and after applying action a and transitioning to a new state s'. An additional penalty term of $-\alpha$ ($\alpha > 0$) is added into the reward for each iteration to prompt the model to make decisions more expeditiously. Formally,

$$r(s, a) = \sum_{i=1}^{N} w_i R_i(x') - \sum_{i=1}^{N} w_i R_i(x) - \alpha, \qquad (8)$$

where x and x' are mapped from s and s' separately. $x_i = 1$ if s_i is "active", and $x_i = 0$ otherwise. The same applies to x' and s'. We set $\alpha = 1/T$, ensuring that the model receives a sum of the penalties no bigger than 1.

2) Power control: The MDP of GRLinQ-pc transforms the state of each node into a continuous number between 0 and 1 to control the transmission power. However, the actions are designed as discrete numbers of varying scales between -0.1 and 0.1, allowing for explicit adjustment and observation of power levels. Formally, we set the MDP components as:

State. A state $s = [s_i : i \in \mathcal{V}]$ is a sequence of continuous numbers, with $s_i \in [0, 1], \forall i \in \mathcal{V}$, corresponding to the indicator variable x_i defined in Section II-A. Initially, the state is set to $s_i = 0.5, \forall i \in \mathcal{V}$. The algorithm terminates either

when the iterations limit T is reached or an early stopping criterion is satisfied.

Action. Given a state and the environmental information, the agent will produce an action $a = [a_i : i \in \mathcal{V}]$, where

$$a_i \in \{-0.1, -0.01, -0.001, 0, 0.001, 0.01, 0.1\} \quad \forall i \in \mathcal{V}.$$

Transition. The RL process transitions from state s to the next state s' by adding the action a to the current state s. If the resulting state value exceeds the [0,1] range, the model clips it to ensure it remains within bounds.

Reward. The reward function is the same as before in Eq. (8), where x and x' correspond to s and s' respectively.

3) PPO Algorithm: The policy network aims to maximize the expected cumulative reward. To optimize the parameters of the policy network, we utilize proximal policy optimization (PPO) [31]. PPO is a widely used reinforcement learning algorithm designed to enhance the stability and efficiency of policy updates. It optimizes the policy such that new policies do not deviate too drastically from the old ones, through a clipped objective function to constrain the policy update within a specific range.

On the other hand, the value network estimates the expected cumulative reward for a given state, serving as a baseline that helps refine and improve the policy's performance.

C. GNN Policy and Value Network Architecture

To capture graph structures of the K-nearest interference graph, we employ a message passing GNN for policy and value networks, where the node features are well-crafted with experts' knowledge from model-based methods. This gives us a hybrid model/data-driven GNN architecture.

1) Data-driven message passing GNN (MPGNN): There exist numerous MPGNNs specifically designed to leverage edge features, such as [32]–[34]. These models typically employ specialized message passing mechanisms to effectively capture edge information within the graph, thereby enhancing the model's expressive capacity and predictive accuracy. Our proposed model first updates the edge features through a neural network, then propagates the message computed by a combination operation of edge and node features, and finally updates the node features through a neural network. The policy network architecture is showcased in Figure 3. At the l-th layer, our proposed MPGNN is structured as follows:

Edge Update: The edge feature is first updated by a multi-layer perceptron (MLP) $\boldsymbol{f}_e^{(l)}$:

$$\boldsymbol{e}_{ji}^{(l)} = \boldsymbol{f}_e^{(l)}(\boldsymbol{e}_{ji}^{(l-1)}), \tag{9}$$

where $\boldsymbol{e}_{ji}^{(l-1)}$ represents the edge feature of edge (j,i) at the $(l-1)$-th layer.

Node Update: Each node then sends a message to its outgoing neighbors. The message sent from v_j to v_i is computed by a message function $\mathrm{msg}^{(l)}$:

$$\boldsymbol{m}_{ji}^{(l)} = \mathrm{msg}^{(l)}(\boldsymbol{e}_{ji}^{(l)}, \boldsymbol{v}_j^{(l-1)}), \tag{10}$$

where $\boldsymbol{v}_j^{(l-1)}$ is the feature vector of node j at the $(l-1)$-th layer. For simplicity, an element-wise multiplication is applied for the $\mathrm{msg}^{(l)}(\cdot)$ function. Each node i aggregates the messages sent from its incoming neighbors $\mathcal{N}(i)$ with a summation:

$$\boldsymbol{a}_i^{(l)} = \sum_{j \in \mathcal{N}(i)} \boldsymbol{m}_{ji}^{(l)}. \tag{11}$$

Afterwards, each node updates its own feature through an MLP applied to the concatenation of its own feature and the aggregated messages, i.e.,

$$\boldsymbol{v}_i^{(l)} = \boldsymbol{f}_n^{(l)}(\mathrm{concat}(\boldsymbol{v}_i^{(l-1)}|\boldsymbol{a}_i^{(l)})), \tag{12}$$

where $\boldsymbol{f}_n^{(l)}$ is an MLP with the *ReLU* activation function at the l-th layer.

To generate actions and value estimations in the final layer, the policy and value networks employ softmax and graph readout functions with sum pooling [35] instead of ReLU.

2) Model-driven feature design: Drawing from expert knowledge, we formulate three types of initial node features of GNNs $\boldsymbol{v}^{(0)}$, inspired by FlashLinQ, ITLinQ, and ITLinQ+.

First, we frame the selection criteria into node features, relating to SNR, INR, and SIR. To provide greater flexibility to the GNN, we eliminate manually designed variables as much as possible. For FlashLinQ, we take the logarithm of the left-hand side of Eq-(5) as the features. As for ITLinQ and ITLinQ+, we first take the logarithm of both sides of Eq-(6) and Eq-(7), exclude the designing parameters, and treat

both sides of the resulting expressions as features separately. We use the logarithmic transformation because neural network structures are more adept at learning linear mappings, and this transformation allows us to convert the exponential design parameters in the model-driven approach into linear design parameters. It is worth noting that in order to maintain the right-hand sides of Eq-(7) as an entity and avoid creating excessive initial features, we set $\gamma = 0.1$, consistent with [13].

Second, instead of considering channel gains $|h_{ij}|^2$, we consider the pairwise distance d_{ij} between transmitters and receivers as node features, as $|h_{ji}|^2$ is positively correlated with $\frac{1}{d_{ji}}$. As such, we replace $|h_{ii}|^2$ and $|h_{ji}|^2$ in node features with $\frac{1}{d_{ii}}$ and $\frac{1}{d_{ji}}$, respectively, where d_{ji} represents the distance between Tx_j and Rx_i. For simplicity, we keep p_i and σ fixed and exclude them from our feature set.

In addition, we include the current node state of the MDP, s_i, and the current iteration count, t, to the feature sets. The state s_i is mapped to a one-hot vector, and the current iteration t is normalized by the maximum number of iterations T. We have also included the model with no experts' knowledge (a.k.a. pure-data) as a baseline. As a result, the final sets of initial node features are shown in Table 3. Finally, the initial feature of edge (j,i) is set as $\log(\frac{1}{d_{ji}})$ for all models. The D2D distances in each network layout are normalized by the maximum distance for consistency.

D. Computational Complexity

The complexity of GRLinQ arises from the feature generation and GNN message passing, i.e., Eq-(11) and Eq-(12). Recall that N, L, T represent the numbers of D2D links, MPGNN layers, and the maximum MDP iterations, respectively. For GRLinQ with ITLinQ+ features, the complexity of feature generation is $O(K^2)$, since the K-nearest interference graph, instead of a fully connected graph, is adopted for MPGNN. The complexity of the entire feature generation process, where every node computes features in each MDP iteration, is given by $O(NTK^2)$, which simplifies to $O(N)$ since K and T are constants. Similarly, the complexity of the message passing process can be computed as $O(NTKL)$, which also simplifies to $O(N)$ as L is a constant as well. Therefore, the overall complexity of GRLinQ is $O(N)$.

TABLE 3
MODEL-DRIVEN NODE FEATURE DESIGNS

Inspiration	Node Features
pure-data	s_i, t, d_{ii}
FlashLinQ	$s_i, t, \log(d_{ii}), \min_{j \in \mathcal{S}}\{\log(d_{ij}) - \log(d_{jj})\},$ $\log(\frac{1}{d_{ii}}) - \log(\sum_{j \in \mathcal{S}} \frac{1}{d_{ji}})$
ITLinQ	$s_i, t, \log(d_{ii}),$ $\max_{j \in \mathcal{S}} \log(\frac{1}{d_{ji}}), \max_{j \in \mathcal{S}} \log(\frac{1}{d_{ij}})$
ITLinQ+	$s_i, t, \log(d_{ii}),$ $\max_{j \in \mathcal{S}}\{\log(\frac{1}{d_{ji}}) - \gamma \min_{k \in \mathcal{S} \setminus j} \log(\frac{1}{d_{jk}})\},$ $\max_{j \in \mathcal{S}}\{\log(\frac{1}{d_{ij}}) - \gamma \min_{k \in \mathcal{S} \setminus j} \log(\frac{1}{d_{kj}})\}$

TABLE 4
SYSTEM PARAMETERS

Parameter	Value
Area size	500 m $\times$ 500 m
Tx-Rx distance	2 m $\sim$ 65 m
Noise spectral density	-169 dBm/Hz
Bandwidth	5 MHz
Carrier frequency	2.4 GHz
Antenna height	1.5 m
Maximum transmit power	40/30/23 dBm

TABLE 5
AVERAGE SUM RATE RATIOS ACHIEVED BY LINK SCHEDULING AND
POWER CONTROL APPROACHES (HIGHER IS BETTER)

Method	CSI	Average Sum Rate Ratio
All scheduled	No	0.656
FlashLinQ [11]	Yes	0.776
ITLinQ [12]	Yes	0.840
ITLinQ+ [13]	Yes	0.877
GELinQ [16]	No	0.952
Greedy	Yes	0.971
SpatialLinQ [15]	No	0.984
FPLinQ [14]	Yes	1.000
GRLinQ	No	1.012
UWMMSE-4* [19]	Yes	1.034
PCGNN-1* [21]	Yes	1.047
PCGNN-2* [21]	Yes	1.049
UWMMSE-8* [19]	Yes	1.054
GRLinQ-pc*	No	1.055
FPLinQ-pc* [14]	Yes	1.059
WMMSE* [17]	Yes	1.060

V. EXPERIMENTS

In this section, we evaluate the performance and generalizability of our proposed GRLinQ and GRLinQ-pc. In the experiments, we set $w_i = 1$ for all i for simplicity.

Network setting. To facilitate comparison, we employ a setup consistent with that of [15], [16], as summarized in Table 4. Specifically, we consider N transmitters uniformly positioned within a 500 m $\times$ 500 m square area. The receivers are uniformly distributed within a range of 2 to 65 meters from their paired transmitters. Following [15], [16], we mainly focus on a path-loss channel model, whilst a more realistic channel model will be discussed in Section V-E. The transmitter-receiver channels follow the short-range outdoor model ITU-1411 with a distance-dependent path-loss [36], which is suited to D2D communication environments. For instance, in the street canyon UHF (Ultra High Frequency) communication scenarios, the path loss PL_{dB} of the Line-of-Sight (LoS) path can be computed as:

$$PL_{dB}(d) = L_{bp} + 6 + \begin{cases} 20\log_{10}\left(\frac{d}{R_{bp}}\right) & \text{for } d \leq R_{bp}, \\ 40\log_{10}\left(\frac{d}{R_{bp}}\right) & \text{for } d > R_{bp}. \end{cases} \quad (13)$$

Here, d is the distance between the transmitter and the receiver (in meters), $R_{bp} \approx \frac{4h_t h_r}{\lambda}$ is the breakpoint distance, where h_t and h_r are the antenna heights at the transmitter and the receiver (in meters), respectively, and λ is the wavelength (in meters). $L_{bp} = \left|20\log_{10}\left(\frac{\lambda^2}{8\pi h_t h_r}\right)\right|$ is a value for the basic transmission loss at the break point. As such, the channel coefficient in Eq-(1) can be specified as $h_{ji} = 10^{-\frac{PL_{dB}(d_{ji})}{20}}$. Unless otherwise specified, the carrier frequency is set to 2.4 GHz, and both antenna heights are set to 1.5 m. For fair comparison, we follow the same setting as the two main benchmarks in [15], [16], with a maximum transmit power of 40 dBm. The impact of the maximum transmit power will be discussed in Section V-F with comparison to the more commonly used transmit power levels of 23 dBm and 30 dBm in D2D communications.

GRLinQ settings. For the K-nearest interference graph adopted for MPGNN, we choose $K = 10$ for scalability, whilst a higher K does not make a significant difference, as discussed later in Section V-B. Unless otherwise specified, GRLinQ and GRLinQ-pc are equipped with node features inspired by ITLinQ+. The impact of different feature designs is detailed in Section V-C. We use a trivial map instead of an MLP in the edge update process, as discussed in Section V-D.

We set the number of MPGNN layers $L = 4$ and the hidden dimension $h = 128$. We set the RL iteration limit $T = 32$,

with an early stopping strategy. For each network setting, we generated 1,000 network layouts as a testing set. Each layout represents a topology graph with randomly distributed transmitters and receivers. Model-based methods are directly applied to the test set. Learning-based methods are trained on different numbers of layouts, according to their requirements, and subsequently evaluated on the test set. The proposed GRLinQ is trained with 500 network layouts. We report the average sum rate ratio, which denotes the ratio of the sum rate achieved by the method to that achieved by FPLinQ, averaged over the 1,000 test layouts.

A. Comparison with State-of-The-Art Methods

We compare the state-of-the-art link scheduling or/and power control approaches in three aspects:

- For link scheduling, we compare GRLinQ with other link scheduling approaches.
- For power control, we compare GRLinQ-pc with other power control approaches.
- Under the same network settings, we jointly compare link scheduling and power control approaches.

1) Link scheduling. We first compare our proposed GRLinQ against the state-of-the-art link scheduling benchmarks:

- **All scheduled.** Activating all the links.
- **FlashLinQ.** Run default FlashLinQ [11], using the selection criterion in Eq-(5).
- **ITLinQ.** Run default ITLinQ [12], using the selection criterion in Eq-(6).
- **ITLinQ+.** Run default ITLinQ+ [13], using the selection criterion in Eq-(7).
- **GELinQ.** Run default GELinQ [16], trained with 500 layouts solved by FPLinQ.
- **Greedy.** Sort links based on distance and activate one by one if increasing the overall sum rate.
- **SpatialLinQ.** Run default SpatialLinQ [15], trained with 800,000 layouts.
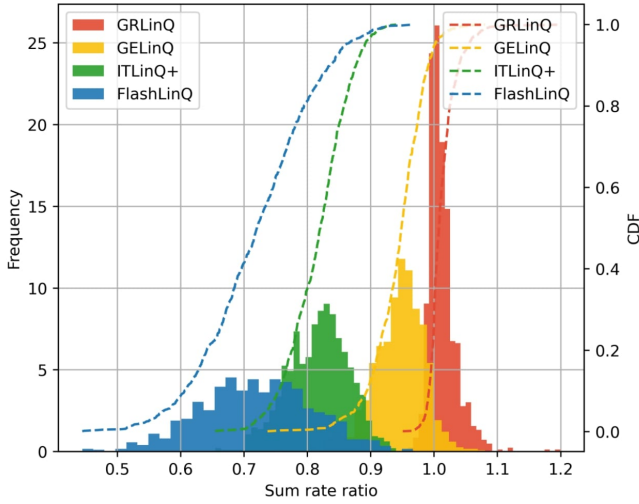
Fig. 4. CDF of sum rate ratios for GRLinQ, GELinQ, ITLinQ+, and FlashLinQ. The number of D2D links is set to be 50.

Fig. 5. Average sum rate ratios achieved by different power control approaches, GRLinQ-pc, FPLinQ-pc, WMMSE, PCGNN, UWMMSE.

- **FPLinQ.** Run FPLinQ [14] with all "ones" initialization of x and 100 iterations.

We set the number of D2D links to be 50. As shown in the upper section of Table 5, our proposed method, GRLinQ, even goes beyond the optimization-based approach FPLinQ by a margin of 1.2%, with a substantial reduction in CSI requirement. Moreover, compared with the state-of-the-art data-driven methods, SpatialLinQ and GELinQ, our proposed GRLinQ achieves almost 3% and 6% improvements, respectively, with the same number of training samples as GELinQ.

Figure 4 illustrates the cumulative distribution function (CDF) of sum rate ratio for GRLinQ, GELinQ[1], ITLinQ+ and FlashLinQ against FPLinQ. It can be seen that GRLinQ demonstrates a substantial gain over others. Remarkably, even without utilizing CSI during online testing, GRLinQ strictly outperforms FPLinQ in the majority (722 out of 1000) of samples. This observation implies that, GRLinQ could potentially discover more advantageous link scheduling policies than the best-known optimization-based methods, e.g., FPLinQ.

2) Power control. We then compare the proposed GRLinQ-pc with state-of-the-art power control benchmarks:

- **UWMMSE-4.** UWMMSE [19] with 4 unfolding layers, as suggested in [19].
- **UWMMSE-8.** UWMMSE [19] with 8 unfolding layers.
- **PCGNN-1.** PCGNN [21] trained with 5,000 layouts.
- **PCGNN-2.** PCGNN [21] trained with 50,000 layouts.
- **FPLinQ-pc.** Run FPLinQ-pc [14] with all "ones" initialization of x and 100 iterations.
- **WMMSE.** Run WMMSE [17] with 100 iterations.

For UWMMSE, in addition to the default settings (4 unfolding layers as suggested in [19]), we also increased the number of unfolding layers to 8 to provide improved solutions. We use 5,000 training layouts for UWMMSE, as more layouts did not bring any improvement. For PCGNN, we report results for models trained with 5,000 and 50,000 layouts.

[1]The results of GELinQ were obtained by running the code provided by [16] at https://github.com/ZhiweiShan/graph_embedding_link_scheduling.
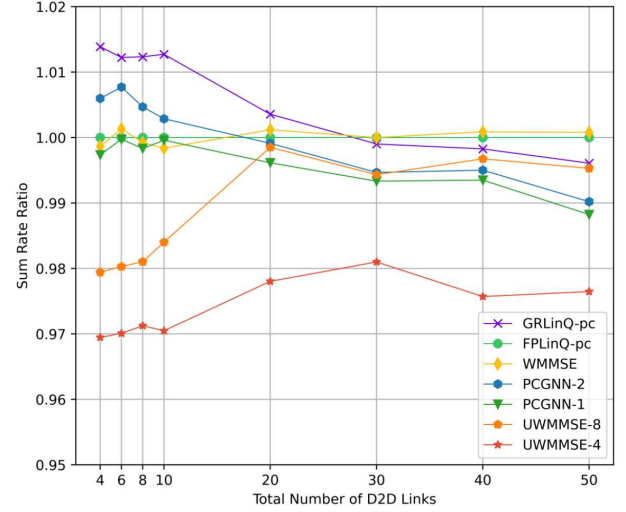
Power control can be viewed as a continuous version of link scheduling. Therefore, we aim to compare these methods together to provide a more comprehensive understanding of their relative performance. To provide a clear view, we have marked power control mechanisms with an asterisk (*) in Table 5. It can be immediately observed that, under this network setting with 50 D2D links, all power control methods outperform the link scheduling methods. GRLinQ-pc achieved remarkable performance using just 500 training layouts, outperforming other learning-based methods UWMMSE and PCGNN. It performs only about 0.5% worse compared to the optimization-based methods FPLinQ-pc and WMMSE, without explicitly knowing the channels and having lower computational complexity. On the other hand, GRLinQ stands out as the best link scheduling method in this context, being only 2.2% behind the power control method UWMMSE-4.

Since power control is particularly challenging, it is typically implemented on small networks (fewer than 30 D2D links) [17], [19], [21]. Therefore, we then compare power control methods on networks with 4 to 50 D2D links, as shown in Figure 5. It can be observed that when the network size is small (i.e., the number of D2D links is less than 20), GRLinQ-pc performs the best, even surpassing WMMSE and FPLinQ-pc. As the number of links increases, GRLinQ-pc's performance slightly declines but consistently outperforms other learning-based methods with CSI.

The decline in GRLinQ's performance is understandable, as the interactions between D2D pairs become more complex with an increasing graph size. Consequently, the mappings that the neural network needs to learn also become more intricate. This same phenomenon is observed in PCGNN as well. However, UWMMSE performs relatively poorly on small graphs, particularly when the number of links is 10, which seems contradictory to the performance of GRLinQ and PCGNN. We believe this phenomenon can be attributed to the fact that graph algorithms typically rely on the connections between a large number of nodes to extract meaningful

This article has been accepted for publication in IEEE Transactions on Communications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2025.3582022
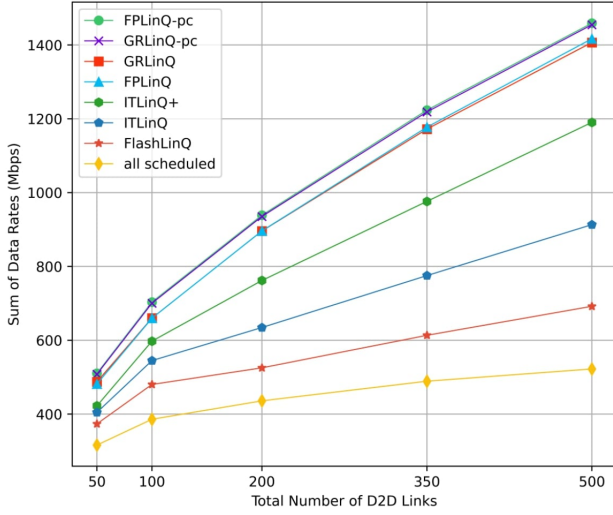
10

Fig. 6. Sum rates versus the total number of D2D links. GRLinQ and GRLinQ-pc are trained and tested under each number of links setting.

structural information. In small graphs, however, the available connections and information are limited, making it challenging for the UWMMSE algorithm to learn effective patterns. In contrast, GRLinQ and PCGNN avoid this issue with their unique designs. GRLinQ uses model-driven feature design to extract useful information from limited neighbor data (e.g., mutual distances), while PCGNN employs a one-layer MLP to expand feature dimensions before message passing, effectively learning from the limited data.

3) Joint link scheduling and power control. We then jointly apply the proposed GRLinQ and GRLinQ-pc in large-scale networks with D2D link counts of 50, 100, 200, 350, and 500, using FPLinQ, FPLinQ-pc, and other link scheduling methods as benchmarks. Figure 6 illustrates the quantitative metric of the average sum rate in **Mbps**. GRLinQ and GRLinQ-pc demonstrate performance that is comparable to or even better than FPLinQ and FPLinQ-pc, highlighting their effectiveness in large-scale network scenarios. Specifically, GRLinQ closely matches FPLinQ, and when the number of D2D links is smaller than 200, GRLinQ slightly outperforms FPLinQ. Although GRLinQ-pc performs marginally weaker than FPLinQ-pc across all network sizes, the difference consistently remains within 0.5%. This indicates that while GRLinQ-pc is slightly behind FPLinQ-pc in large-scale networks, it still maintains a highly competitive performance level with substantially reduced computational complexity.

B. Topology Awareness and Scalability

We use a K-nearest interference graph instead of a fully connected graph, significantly enhancing the scalability of the GRLinQ framework. This approach offers three main advantages:

1) It enhances topology awareness by ensuring that the graph structure more accurately reflects the local interactions and dependencies among nodes.

2) In a K-nearest interference graph, each node's number of in-neighbors (i.e., neighboring nodes that point to it) remains constant regardless of network size. This ensures that the amount of information each node receives during the GNN message passing process is fixed, facilitating scalability to large-scale networks.

3) It reduces both message passing and computational complexity from $O(N^2)$ to $O(KN)$.

Notably, by training GRLinQ with 50 D2D links and then testing it on networks with 10,000 links using $K = 10$, we reduced computational complexity by a factor of 1,000, while achieving 97% of FPLinQ's performance.

In what follows, we first examine the **impact of varying K** on the model's performance to demonstrate that the model is not sensitive to K. Subsequently, we evaluate the model's scalability by testing it on unseen **large-scale networks** with **similar network density** and **higher network density**.

TABLE 6
PERFORMANCE OF GRLinQ FOR DIFFERENT K, WHERE THE EXACT VALUE OF K IS SHOWN IN BRACKETS

N	$K = 10$	$K = 0.1N$	$K = 0.2N$	$K = 0.3N$
50	**1.012** (10)	1.010 (5)	**1.012** (10)	**1.012** (15)
100	1.001 (10)	1.001 (10)	1.005 (20)	**1.007** (30)
200	0.999 (10)	0.999 (20)	**1.002** (40)	1.000 (60)

TABLE 7
PERFORMANCE OF GRLinQ-pc FOR DIFFERENT K, WHERE THE EXACT VALUE OF K IS SHOWN IN BRACKETS

N	$K = 10$	$K = 0.2N$	$K = 0.3N$	$K = N - 1$
30	**0.999** (10)	0.993 (6)	**0.999** (9)	0.991 (29)
40	**0.998** (10)	0.997 (8)	0.997 (12)	0.993 (39)
50	**0.996** (10)	**0.996** (10)	0.990 (15)	0.989 (49)

1) Impact of K-nearest interference graph. In this section, we evaluate the impact of choosing different K on GRLinQ and GRLinQ-pc performance. We hypothesize that setting K to be proportional to the number of D2D pairs N is beneficial. Therefore, we set $K = 0.1N$, $0.2N$, and $0.3N$ respectively, and then retrain and test the model under these configurations on three network settings with $N = 50, 100,$ and 200.

As shown in Table 6, increasing K sometimes provides some benefits, with the most notable improvement observed when $N = 100$. However, for $N = 200$, increasing K from $0.2N$ to $0.3N$ results in a performance drop. This could be attributed to the declining performance of GNNs when handling very dense graphs. Overall, fixing $K = 10$ already achieves satisfactory results, and maintains computational complexity at a reasonable level. Table 7 shows the results of GRLinQ-pc, where a similar trend is observed. The impact of varying K on the model is marginal, with differences remaining below 1%. Therefore, we choose to set $K = 10$ for all of our experiments for GRLinQ and GRLinQ-pc.

2) Large-scale networks with similar network density. We first test the model's scalability to unseen networks with an increasing number of D2D links, while maintaining a similar network density. This is controlled by proportionally

TABLE 8
SCALABILITY OF GRLINQ FOR LARGE NETWORK SIZE BUT SIMILAR
NETWORK DENSITY

N	Area (m $\times$ m)	Average Sum Rate Ratio
50	500×500	1.012 (trained)
100	700×700	1.003
200	1000×1000	0.994
500	1500×1500	0.985
10,000	7000×7000	0.971

TABLE 9
SCALABILITY OF GRLINQ FOR LARGE NETWORK SIZE AND HIGHER
NETWORK DENSITY

N	Area (m $\times$ m)	Average Sum Rate Ratio
50		1.012 (trained)
100	500×500	1.001
200		0.969
500		0.910

increasing the area size. GRLinQ is equipped with ITLinQ+ features as well. We train the model on networks with 50 D2D links over a 500 m $\times$ 500 m square area, and then tested on networks with different numbers of D2D links and different area sizes, as shown in Table 8. It can be observed that the performance of GRLinQ remains competitive as FPLinQ by the increase in the number of D2D links. Notably, on a test set with 10,000 D2D pairs, which is 200 times larger than the training set, GRLinQ achieved an average sum rate ratio of 0.971, completing all decisions within $T = 32$ iterations. We report only the results of GRLinQ, omitting those of GRLinQ-pc, because they share the same scalability behavior.

3) Large-scale networks with higher network density. We then evaluate the scalability of GRLinQ to larger and denser networks. We conducted training on the networks with 50 links as well, and subsequently tested on larger size and denser networks, maintaining the area size of 500 m $\times$ 500 m. The results in Table 9 reveal that the scalability of GRLinQ to high-density networks is acceptable. However, in extremely dense networks, specifically with $N = 500$, which is ten times denser compared to the training set, GRLinQ exhibits a decline in performance. This decline may be related to the dramatically increased mutual interference in such highly dense network environments, suggesting that retraining the models with a denser training set might be necessary.

TABLE 10
PERFORMANCE OF GRLINQ WITH DIFFERENT MODEL-DRIVEN NODE
FEATURE DESIGNS

N	GRLinQ-PD	GRLinQ-FL	GRLinQ-IT	GRLinQ-IT+
50	0.940	1.006	**1.012**	**1.012**
100	0.974	1.002	**1.003**	1.001
200	0.990	0.997	0.998	**0.999**
500	0.984	0.989	**0.995**	0.993

C. Feature Design and Generalizability

In this section, we assess the effectiveness of **model-driven features** together with the generalizability of the model with respect to **carrier frequency** and **user distribution**.

TABLE 11
PERFORMANCE OF GRLINQ-PC WITH DIFFERENT MODEL-DRIVEN NODE
FEATURE DESIGNS

N	GRLinQ-pc-PD	GRLinQ-pc-FL	GRLinQ-pc-IT	GRLinQ-pc-IT+
10	0.981	1.001	1.012	**1.013**
20	0.916	0.996	0.994	**1.004**
30	0.900	0.963	0.996	**0.999**
40	0.881	0.972	0.993	**0.998**

TABLE 12
GENERALIZATION OF GRLINQ WITH DIFFERENT NODE FEATURES OVER
DIFFERENT NETWORK SIZES

N	GRLinQ-PD	GRLinQ-FL	GRLinQ-IT	GRLinQ-IT+
50 (trained)	0.940	1.006	**1.012**	**1.012**
100	0.956	0.996	0.998	**1.001**
200	0.941	0.963	0.964	**0.969**
500	0.828	0.860	**0.910**	**0.910**

1) Effectiveness of model-driven features. We first compare the performance of GRLinQ equipped with different feature designs:

- **GRLinQ-PD.** GRLinQ with pure-data-driven features.
- **GRLinQ-FL.** GRLinQ with FlashLinQ features.
- **GRLinQ-IT.** GRLinQ with ITLinQ features.
- **GRLinQ-IT+.** GRLinQ with ITLinQ+ features.

The models are trained and tested on the same network settings, with 50, 100, 200, and 500 D2D links. Based on the performance evaluation results presented in Table 10, we observe several noteworthy trends and insights. For the smallest network size, $N = 50$, both GRLinQ-IT and GRLinQ-IT+ achieve the highest average sum rate ratio of 1.012, outperforming GRLinQ-PD and GRLinQ-FL, which exhibit sum rate ratios of 0.940 and 1.006, respectively. As the network size increases to $N = 100$ and $N = 200$, the performance gap between the models narrows. GRLinQ-IT and GRLinQ-IT+ slightly outperform other models. In the largest network size of $N = 500$, GRLinQ-IT achieves the highest average sum rate ratio of 0.995, with GRLinQ-IT+ following at 0.993. This indicates that the TIN-condition-inspired designs provide superior performance in smaller network settings, although the diminishing performance gap suggests that their advantages become less pronounced as the network size increases.

Overall, the results demonstrate that models inspired by TIN-condition (GRLinQ-IT and GRLinQ-IT+) consistently outperform the pure-data-driven (GRLinQ-PD) and FlashLinQ-inspired (GRLinQ-FL) models across all network sizes. This performance superiority is particularly pronounced in smaller networks but persists even as the network size increases. These findings underscore the effectiveness of ITLinQ-inspired designs in optimizing the average sum rate performance in D2D networks. A similar result is obtained when testing the performance of GRLinQ-pc equipped with different features, as shown in Table 11. Node features inspired by the TIN-condition have been shown to significantly enhance power control as well.

Afterward, we evaluate the generalizability of GRLinQ with different features, as shown in Table 12. GRLinQ-IT and GRLinQ-IT+ consistently achieve better generalizability compared to GRLinQ-PD and GRLinQ-FL across all tested network sizes. At $N = 100$, GRLinQ-IT+ achieves the highest sum rate ratio of 1.001, followed closely by GRLinQ-IT at

0.998. This trend continues at $N = 200$, where GRLinQ-IT+ leads with a ratio of 0.969. In larger and denser networks, GRLinQ-PD and GRLinQ-FL performance diminishes more significantly compared to GRLinQ-IT and GRLinQ-IT+. These findings demonstrate that TIN-condition-inspired models, GRLinQ-IT and GRLinQ-IT+, exhibit better generalizability and maintain higher performance across various network densities and sizes. However, in extremely dense networks, e.g., at $N = 500$, which is ten times denser than the training set, GRLinQ shows certain performance degradation.

2) Generalizability for different carrier frequencies. We then examine how changes in carrier frequency affect its performance. The model has been trained under a carrier frequency of 2.4 GHz with $N = 50$. We assess the model when the carrier frequency is adjusted to 6 GHz and 26 GHz, and the results are presented in Table 13. FPLinQ continues to serve as the baseline, which recalculates the scheduling scheme based on the new carrier frequency. In the scenario where the carrier frequency changes from 2.4 GHz to 6 GHz and 26 GHz, our model's performance only decreases by 0.5% and 1.1%, respectively, still outperforming FPLinQ. This implies that GRLinQ has significant potential to handle various exceptional situations without generating unreasonable solutions.

TABLE 13
GENERALIZATION PERFORMANCE FOR DIFFERENT CARRIER FREQUENCY

Carrier frequency (GHz)	2.4 (trained)	6	26
Average sum rate	1.012	1.007	1.001

3) Generalizability for different user distributions. Then, we evaluate the generalization performance of the model with respect to different user distributions. The model is trained on networks with D2D distances ranging from $2\sim65$ meters and then tested on other D2D distance scenarios, as shown in Table 14. Surprisingly, it performs better on networks with D2D distances ranging from $10\sim50$ meters and $30\sim70$ meters, *without further training*. Remarkably, on networks where all D2D distances are 30 meters, GRLinQ achieves 94.1% of the performance of FPLinQ. This indicates that even when all D2D pairs have identical initial information (i.e., distance), GRLinQ can still perform scheduling effectively, likely due to the well-designed model-driven features.

TABLE 14
GENERALIZATION PERFORMANCE FOR DIFFERENT USER DISTRIBUTIONS

D2D distance (m)	2$\sim$65 (trained)	10$\sim$50	30$\sim$70	all 30
Average sum rate	1.012	1.073	1.099	0.941

D. Ablation Study

In this section, we perform an ablation study to evaluate the contributions of various components within the GRLinQ framework. Specifically, we analyze the following aspects: the role of **edge feature updates**, **training convergence**, and the effects of **transformation learning**.

1) Impact of edge feature update. To evaluate the effectiveness of the edge feature update mechanism, we compare the following two models:

- **GRLinQ (EdgeUp):** This model updates edge features according to Eq-(9).
- **GRLinQ:** This model uses a trivial map, replacing Eq-(9) with $e_{ji}^{(l)} = e_{ji}^{(l-1)}$, and broadcasts it to the corresponding dimensions before performing element-wise multiplication with the node information.

We report their performance in Table 15. It can be observed that while using an MLP for edge feature updates brings some improvements, employing a trivial mapping instead of a complex MLP yields comparable performance. This phenomenon can be attributed to the sufficiency of the node features, especially when these features are derived from theoretically sound model designs.

Incorporating a complex edge update mechanism, such as an MLP, introduces additional parameters, thereby increasing the model's complexity and computational cost. In scenarios where the dataset is limited or the variation in edge features is minimal, a sophisticated edge update mechanism may induce overfitting. Conversely, a simpler mapping approach can enhance generalization capabilities. As such, unless otherwise specified, we use the model without edge feature updates in our paper. Nonetheless, the performance of these models may be enhanced by incorporating edge feature updates.

TABLE 15
COMPARISON OF EDGE FEATURE UPDATE METHODS

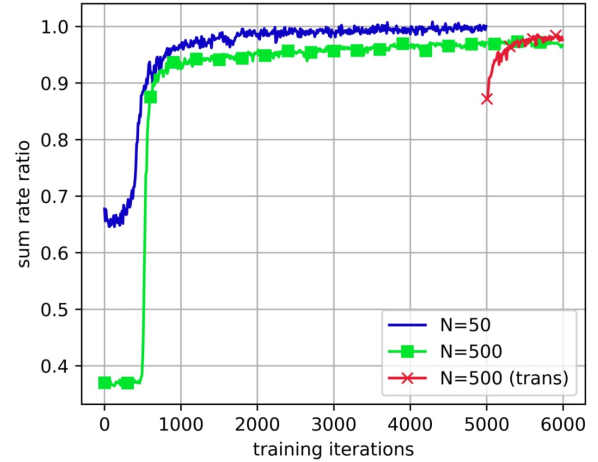| N	GRLinQ	GRLinQ (EdgeUp)
50	1.012	1.013
100	1.001	1.006
200	0.999	1.001
500	0.993	0.987

Fig. 7. Convergence performance of GRLinQ is evaluated under two training conditions: starting from scratch with network sizes of $N = 50$ and $N = 500$ separately. Additionally, the model trained with $N = 50$ serves as the initial point for further training with $N = 500$.

2) Training convergence and transfer learning. In this section, we evaluate the training convergence on both small and large networks. Additionally, we employ transfer learning on large networks to accelerate the training process:

- $N = 50$: Train GRLinQ on networks with 50 D2D links for 5,000 training iterations.

- $N = 500$: Train GRLinQ on networks with 500 D2D links for 6,000 training iterations.
- $N = 500$ (trans): Start with the pre-trained model from $N = 50$, then train for an additional 1,000 iterations on networks with 500 D2D links.

The results are shown in Figure 7. For $N = 50$, the model converges quickly: during the initial 400 iterations, the agent engages in extensive random exploration and experience accumulation within the environment, achieving significant performance improvement around 400 to 500 iterations. After approximately 2,000 iterations, the performance stabilizes and no longer shows significant growth.

Compared to the $N = 50$ case, the training process for $N = 500$ is noticeably more challenging. In the initial phase, the agent also performs random exploration and experience accumulation, but due to the increased dataset size, the exploration space is larger, requiring more time to accumulate sufficient experience and learn the strategy. After undergoing a rapid improvement in performance, the model continues to steadily enhance its performance until the 5,000th iteration.

The transfer training method starts with significantly higher initial performance compared to models trained from scratch because it leverages the strategies and experiences already learned from the model trained with $N = 50$. It can be seen that the performance quickly improves and outperforms the model trained from scratch with $N = 500$. Therefore, in our experiments, we use transfer training from the pre-trained model with 50 links for networks with more than 50 links.

TABLE 16
PERFORMANCE COMPARISON WITH DIFFERENT INPUTS ON REALISTIC CHANNEL MODEL

N \ Input	distance	CSI	CSI gen.
50	0.814	1.022	1.022
100	0.803	1.025	1.023
200	0.782	1.019	1.015
500	0.770	1.001	0.995

E. Discussion on Realistic Channels

In the previous sections, our proposed model, along with prior data-driven models such as SpatialLinQ [15] and GELinQ [16], considered the ITU-1411 [36] path loss as channel coefficient h_{ji} in Eq-(1). The success of these data-driven models, as well as that of the present work in reducing CSI requirements, can be partly attributed to the fact that path loss is a deterministic function of distance.

However, a more realistic channel model should account for both large-scale fading (e.g., path loss, antenna gain, and shadowing effects) and small-scale fading (e.g., multipath fading). We collectively refer to this as the "realistic channel model", consistent with those used in model-based studies. On one hand, the large scale fading is modeled by

$$10 \log_{10} \beta_{ji} = PL_{dB}(d_{ji}) - AG + SD, \quad (14)$$

where AG represents antenna gain, set to 2.5 dB, and SD denotes shadowing, modeled as a Gaussian noise with a standard deviation of 10 dB. On the other hand, the multipath

fading g_{ji} is modeled as a Gaussian distributed complex-valued random variable, i.e., $g_{ji} \sim \mathcal{CN}(0,1)$. Thus, the realistic channel coefficient can be specified as $h_{ji} = \sqrt{\beta_{ji}} g_{ji}$.

The realistic channel model presents significant challenges for data-driven methods that rely solely on distance. For example, when SpatialLinQ was tested on channel with path loss and Rayleigh fast fading, its performance significantly deteriorated [15]. This outcome is understandable because, under such conditions, the channel becomes a stochastic function of distance, complicating the neural network's objective function.

To adapt our proposed model to the realistic channel, we use CSI as the input, and use the realistic channel gain to compute the data rate. In practice, we use the z-score normalization method for CSI, and add an offset to make it non-negative. The results presented in Table 16 demonstrate the average sum rate ratio of GRLinQ on different network sizes when using two types of input: distance and CSI. Here, "CSI gen." refers to the model trained only with $N = 50$ but tested across various network sizes to evaluate its generalizability. It can be seen that for networks utilizing the realistic channel model, merely using the transmitter-receiver distance as inputs is insufficient, achieving only about 80% of the sum rate relative to FPLinQ, where the latter always has access to exact CSI. If CSI is provided as input to GRLinQ, our model can achieve superior performance. Additionally, even with a significant increase in network density, the experiments demonstrate that our model exhibits strong generalization when CSI is used as input.

We compare the GRLinQ framework with multiple models on a large-scale realistic channel. We additionally introduce a method called Joint GRLinQ, which first uses GRLinQ to generate link scheduling results and then fine-tunes using GRLinQ-pc. As shown in Fig. 8, both GRLinQ and Joint GRLinQ outperform FPLinQ across all network sizes, and significantly surpass ITLinQ+, ITLinQ, and FlashLinQ, with the results presented in **Mbps** highlighting their superior performance. This demonstrates that the GRLinQ framework has a strong capability for solving real channel spectrum sharing. However, there is still a gap between them and FPLinQ-pc, which remains a future direction for improvement.

F. Discussion on Maximum Transmit Power

In D2D network scenarios, user equipment typically communicates directly with each other, and the maximum transmit power is generally between 23 dBm and 30 dBm. However, for the sake of comparison, we have adopted a maximum transmit power of 40 dBm, as used in [15], [16]. In this section, we investigate the impact of different transmit power settings on the model performance.

We assume $N = 50$ users in the network. As shown in Table 17, the results indicate that, under both the path-loss and realistic channel models, all algorithms FPLinQ, GRLinQ, FPLinQ-pc, and GRLinQ-pc achieve *consistent* sum rate performance across different maximum transmit power settings. In other words, the link scheduling and power control mechanisms demonstrate insensitivity to variations in the maximum transmit power. In fact, we observe that the optimized power allocation levels $\{x_i\}$ (i.e., the ratio between actual and

TABLE 17
SUM RATE (MBPS) PERFORMANCE UNDER DIFFERENT MAXIMUM TRANSMIT POWERS

Problem	Method	Path-loss channel			Realistic channel		
		40 dBm	30 dBm	23 dBm	40 dBm	30 dBm	23 dBm
Link Scheduling	GRLinQ	487.4843	487.9651	487.7418	563.5189	563.1899	563.9611
	FPLinQ	481.6085	481.8924	481.8869	557.9475	557.8484	557.5375
Power Control	GRLinQ-pc	506.6041	506.8127	506.8387	565.7961	565.9971	565.7788
	FPLinQ-pc	510.5294	510.8978	510.8907	595.6953	595.6942	595.6895
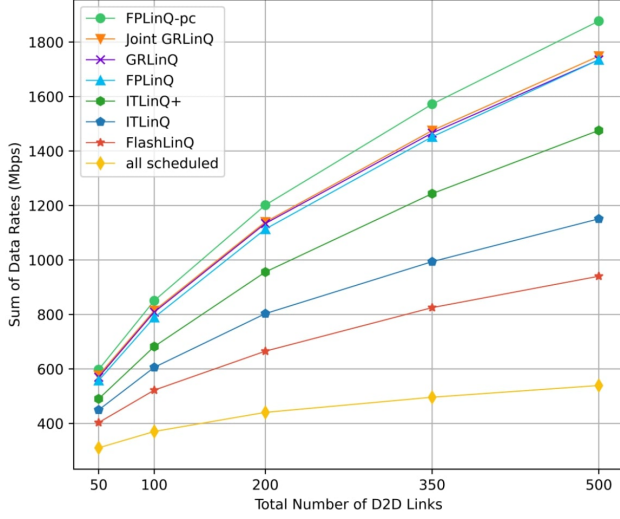
Fig. 8. Sum rate achieved by different approaches on realistic channels.
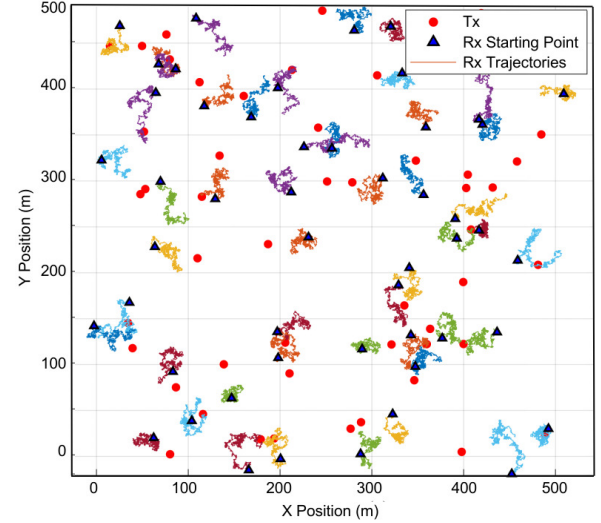

Fig. 9. A dynamic D2D network with 50 fixed transmitters and 50 moving receivers. Each receiver moves by 1 meter in a random direction at every time step, and the trajectories depict the mobility of 1000 time steps.

maximum power) produced by these algorithms remain nearly identical across different transmit power constraints $\{p_i\}$. This implies that the system operates in an interference-limited regime, where performance is dominated by interference levels rather than the power constraints $\{p_i\}$. Consequently, the achievable sum rate is primarily governed by the optimized power allocation levels $\{x_i\}$.

G. Dynamic Link Scheduling

While spectrum sharing can be conducted for each coherence time or frequency block in the block fading channels, dynamic spectrum sharing is a pivotal enabler in establishing adaptive communication for high-mobility D2D networks. To evaluate the effectiveness of the proposed GRLinQ framework in dynamic D2D networks, we consider the following scenario with the receiver's mobility. Based on the previously considered static network, the locations of all transmitters remain fixed, while each receiver moves by 1 meter in a random direction at every time step. Figure 9 presents an illustrative example, assuming a total of 1000 time steps with $N = 50$.

Thanks to the MDP state transition structure of the proposed method, GRLinQ can dynamically adapt to user mobility and CSI variations by updating its computations and state transitions instantaneously based on the new pairwise distances or CSI. Specifically, given a current MDP state (derived from the original environmental information) and the updated environmental information, the agent generates an action and
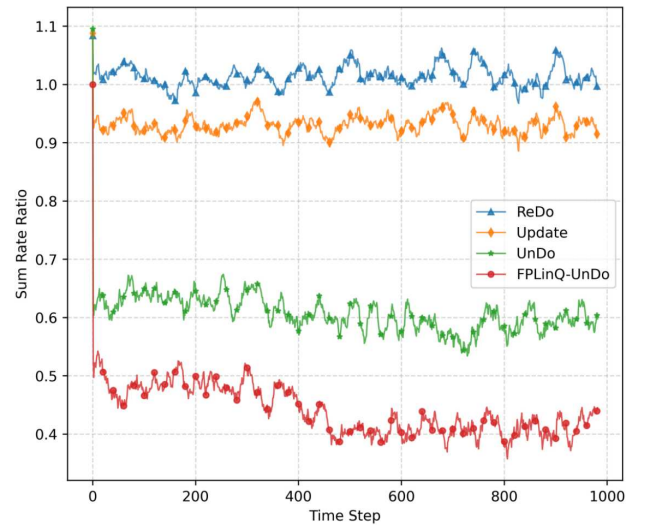
Fig. 10. The comparison of sum rate ratio over 1000 time steps for different link scheduling strategies.

This article has been accepted for publication in IEEE Transactions on Communications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2025.3582022

15

transitions to a new state. After T' iterations on the updated environment, this process results in an updated link scheduling or power control scheme.

Finally, we evaluate GRLinQ for the dynamic link scheduling problem under the above realistic channel model and compare the following methods:

- **ReDo**: Reapply GRLinQ at each time step.
- **Update**: Run GRLinQ at the first time step, then apply the GRLinQ with $T' = 1$ iteration to update the scheme for the remaining time steps.
- **UnDo**: Run GRLinQ at the first time step and maintain the obtained link scheduling scheme.
- **FPLinQ-UnDo**: Run FPLinQ at the first time step and maintain the obtained link scheduling scheme.

We emphasize that GRLinQ operates without additional training and recall that the maximum number of iterations, T, is set to 32. The ratio of the obtained sum rate to that of reapplying FPLinQ with 100 iterations at each time step is defined as the sum rate ratio, with results presented in Figure 10. It can be observed that with just $T' = 1$ updating iteration, Update GRLinQ achieves approximately 90% to 95% of the performance of reapplying FPLinQ. While there is about a 10% gap compared to ReDo, it only requires $\frac{1}{32}$ of the computation time. More importantly, the sum rate ratio remains stable over time. In contrast, maintaining the previous link scheduling scheme (referred to as UnDo) leads to rapid performance deterioration with a downward trend. By comparing the performance of GRLinQ-UnDo and FPLinQ-UnDo, it can be observed that FPLinQ experiences a more significant degradation. This indicates that FPLinQ is more sensitive to dynamic user mobility in the network.
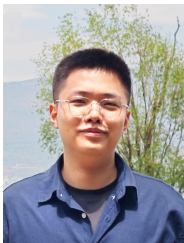
VI. CONCLUSION

This work proposed a novel hybrid model/data-driven intelligent D2D link scheduling (GRLinQ) and a power control (GRLinQ-pc) mechanism leveraging graph reinforcement learning and experts' knowledge from information theory. GRLinQ demonstrates superior sum rate performance compared to the optimization-based FPLinQ, with significantly reduced CSI requirements, and to existing data-driven machine learning approaches for link scheduling and/or power control with substantially improved scalability and generalization performance. Further studies include the theoretical analysis of the convergence and generalization performance of GRLinQ, as well as its comprehensive deployment in practical dynamic wireless networks.

REFERENCES

[1] Z. Shan, X. Yi, L. Liang, C.-S. Liao, and S. Jin, "GRLinQ: A distributed link scheduling mechanism with graph reinforcement learning," in *IEEE Int. Symp. Inf. Theory (ISIT)*, 2024, pp. 2436–2441.

[2] C. Geng, N. Naderializadeh, A. S. Avestimehr, and S. A. Jafar, "On the optimality of treating interference as noise," *IEEE Trans. Inf. Theory*, vol. 61, no. 4, pp. 1753–1767, Apr. 2015.

[3] C. Geng and S. A. Jafar, "On the optimality of treating interference as noise: Compound interference networks," *IEEE Trans. Inf. Theory*, vol. 62, no. 8, pp. 4630–4653, Aug. 2016.

[4] C. Geng, H. Sun, and S. A. Jafar, "On the optimality of treating interference as noise: General message sets," *IEEE Trans. Inf. Theory*, vol. 61, no. 7, pp. 3722–3736, Jul. 2015.

[5] H. Sun and S. A. Jafar, "On the optimality of treating interference as noise for k-user parallel Gaussian interference networks," *IEEE Trans. Inf. Theory*, vol. 62, no. 4, pp. 1911–1930, Apr. 2016.

[6] S. Gherekhloo, A. Chaaban, C. Di, and A. Sezgin, "(Sub-)optimality of treating interference as noise in the cellular uplink with weak interference," *IEEE Trans. Inf. Theory*, vol. 62, no. 1, pp. 322–356, Jan. 2015.

[7] H. Joudeh and B. Clerckx, "On the optimality of treating inter-cell interference as noise in uplink cellular networks," *IEEE Trans. Inf. Theory*, vol. 65, no. 11, pp. 7208–7232, Nov. 2019.

[8] X. Yi and H. Sun, "Opportunistic treating interference as noise," *IEEE Trans. Inf. Theory*, vol. 66, no. 1, pp. 520–533, Jan. 2019.

[9] H. Joudeh, X. Yi, B. Clerckx, and G. Caire, "On the optimality of treating inter-cell interference as noise: Downlink cellular networks and uplink-downlink duality," *IEEE Trans. Inf. Theory*, vol. 66, no. 11, pp. 6939–6961, Nov. 2020.

[10] S. Gherekhloo, A. Chaaban, and A. Sezgin, "Expanded GDoF-optimality regime of treating interference as noise in the $m \times 2$ X-channel," *IEEE Trans. Inf. Theory*, vol. 63, no. 1, pp. 355–376, Nov. 2016.

[11] X. Wu, S. Tavildar, S. Shakkottai, T. Richardson, J. Li, R. Laroia, and A. Jovicic, "FlashLinQ: A synchronous distributed scheduler for peer-to-peer ad hoc networks," *IEEE/ACM Trans. Networking*, vol. 21, no. 4, pp. 1215–1228, Aug. 2013.

[12] N. Naderializadeh and A. S. Avestimehr, "ITLinQ: A new approach for spectrum sharing in device-to-device communication systems," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1139–1151, Jun. 2014.

[13] X. Yi and G. Caire, "Optimality of treating interference as noise: A combinatorial perspective," *IEEE Trans. Inf. Theory*, vol. 62, no. 8, pp. 4654–4673, Aug. 2016.

[14] K. Shen, W. Yu, L. Zhao, and D. P. Palomar, "Optimization of MIMO device-to-device networks via matrix fractional programming: A minorization–maximization approach," *IEEE/ACM Trans. Networking*, vol. 27, no. 5, pp. 2164–2177, Oct. 2019.

[15] W. Cui, K. Shen, and W. Yu, "Spatial deep learning for wireless scheduling," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1248–1261, Jun. 2019.

[16] M. Lee, G. Yu, and G. Y. Li, "Graph embedding-based wireless link scheduling with few training samples," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2282–2294, Apr. 2020.

[17] 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 Trans. Signal Process.*, vol. 59, no. 9, pp. 4331–4340, May 2011.

[18] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "Graph neural networks for scalable radio resource management: Architecture design and theoretical analysis," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 1, pp. 101–115, Jan. 2020.

[19] A. Chowdhury, G. Verma, C. Rao, A. Swami, and S. Segarra, "Unfolding WMMSE using graph neural networks for efficient power allocation," *IEEE Trans. Wireless Commun.*, vol. 20, no. 9, pp. 6004–6017, Sep. 2021.

[20] Q. Hu, Y. Cai, Q. Shi, K. Xu, G. Yu, and Z. Ding, "Iterative algorithm induced deep-unfolding neural networks: Precoding design for multiuser MIMO systems," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1394–1410, Feb. 2020.

[21] Y. Shen, J. Zhang, S. Song, and K. B. Letaief, "Graph neural networks for wireless communications: From theory to practice," *IEEE Trans. Wireless Commun.*, vol. 22, no. 5, pp. 3554–3569, May 2022.

[22] H. He, S. Jin, C.-K. Wen, F. Gao, G. Y. Li, and Z. Xu, "Model-driven deep learning for physical layer communications," *IEEE Wireless Commun.*, vol. 26, no. 5, pp. 77–83, Oct. 2019.

[23] Z. Liu, J. Zhang, E. Shi, Z. Liu, D. Niyato, B. Ai, and X. S. Shen, "Graph neural network meets multi-agent reinforcement learning: Fundamentals, applications, and future directions," *IEEE Wireless Commun*, vol. 31, no. 6, pp. 39–47, 2024.

[24] V.-A. Darvariu, S. Hailes, and M. Musolesi, "Graph reinforcement learning for combinatorial optimization: A survey and unifying perspective," *arXiv preprint arXiv:2404.06492*, 2024.

[25] Z. Shan, X. Yi, H. Yu, C.-S. Liao, and S. Jin, "Revisiting topological interference management: A learning-to-code on graphs perspective," *arXiv preprint arXiv:2502.09344*, 2025.

[26] L. Liang, H. Ye, G. Yu, and G. Y. Li, "Deep-learning-based wireless resource allocation with application to vehicular networks," *Proc. IEEE*, vol. 108, no. 2, pp. 341–356, Feb. 2020.

[27] S. Ahn, Y. Seo, and J. Shin, "Learning what to defer for maximum independent sets," in *International Conference on Machine Learning*. PMLR, 2020, pp. 134–144.

This article has been accepted for publication in IEEE Transactions on Communications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCOMM.2025.3582022

16

[28] L. Liang, H. Ye, and G. Y. Li, "Spectrum sharing in vehicular networks based on multi-agent reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2282–2292, Oct. 2019.

[29] E. Khalil, H. Dai, Y. Zhang, B. Dilkina, and L. Song, "Learning combinatorial optimization algorithms over graphs," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[30] V. Mnih, "Asynchronous methods for deep reinforcement learning," *arXiv preprint arXiv:1602.01783*, 2016.

[31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[32] S. Cui, B. Yu, T. Liu, Z. Zhang, X. Wang, and J. Shi, "Edge-enhanced graph convolution networks for event detection with syntactic relation," *arXiv preprint arXiv:2002.10757*, 2020.

[33] E. Isufi, F. Gama, and A. Ribeiro, "EdgeNets: Edge varying graph neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 11, pp. 7457–7473, Nov. 2021.

[34] T. Monninger, J. Schmidt, J. Rupprecht, D. Raba, J. Jordan, D. Frank, S. Staab, and K. Dietmayer, "Scene: Reasoning about traffic scenes using heterogeneous graph neural networks," *IEEE Robot. Autom. Lett.*, vol. 8, no. 3, pp. 1531–1538, Mar. 2023.

[35] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *International Conference on Learning Representations*, 2019.

[36] International Telecommunication Union, "Recommendation ITU-R P.1411-8," 2015.

Le Liang (S'13-M'19) received the B.E. degree in information engineering from Southeast University, Nanjing, China, in 2012, the M.A.Sc degree in electrical engineering from the University of Victoria, Victoria, BC, Canada, in 2015, and the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, in 2018. From 2019 to 2021, he was a Research Scientist at Intel Labs, Hillsboro, OR. Since 2021, he has been with the National Mobile Communications Research Laboratory, Southeast University. His main research interests are in wireless communications and machine learning.

Dr. Liang serves as an Associate Editor for IEEE Transactions on Cognitive Communications and Networking and China Communications. He was as an Associate Editor for IEEE Journal on Selected Areas in Communications (JSAC) Series on Machine Learning in Communications and Networks from 2020 to 2022 and an Editor for IEEE Communications Letters from 2019 to 2023.

Chung-Shou Liao is a Distinguished Professor with Dept. Industrial Engineering and Engineering Management, National Tsing Hua University (NTHU), Taiwan. Before joining NTHU, he had worked in the Algorithms and Computation Laboratory at the Institute of Information Science, Academia Sinica for eight years. He received his Ph.D. from Dept. Computer Science and Information Engineering, National Taiwan University in 2009. His research mainly focuses on designing efficient algorithms that can be used to solve difficult combinatorial optimization problems from real applications.

Dr. Liao received the Outstanding Research Award of NSTC, Taiwan in 2023 and the Outstanding Young Researcher Award of IICM (ACM Taiwan) in 2014. He is also a Fulbright Senior Research Scholar for 2018 & 2019. He served as the Program Committee Chair of AAAC 2021 (the 14th Annual Meeting of Asian Association for Algorithms and Computation) and ISAAC 2018 (the 29th International Symposium on Algorithms and Computation). He is the board member of AAAC, the steering committee member of ISAAC, and the Associate Editor of Journal of Combinatorial Optimization and International Journal of Foundations of Computer Science. In recent years, his team has collaborated with world-class industrial companies such as MediaTek, UMC, TSMC, Unimicron, etc. and conducted the MOST AI research project.

Dr. Liao is Senior Member of ACM and IEEE.

Zhiwei Shan (S'23) received the B.Sc. degree in Statistics from Shandong University, China, in 2018, and the M.Sc. degree in Financial Mathematics from Loughborough University, United Kingdom, in 2020. He is currently pursuing the Ph.D. degree in Computer Science with the University of Liverpool, United Kingdom, and National Tsing Hua University, under a joint doctoral program. His research focuses on graph machine learning and its applications in wireless communications.

Shi Jin (S'06-M'07-SM'17-F'24) received the B.S. degree in communications engineering from Guilin University of Electronic Technology, Guilin, China, in 1996, the M.S. degree from Nanjing University of Posts and Telecommunications, Nanjing, China, in 2003, and the Ph.D. degree in information and communications engineering from the Southeast University, Nanjing, in 2007. From June 2007 to October 2009, he was a Research Fellow with the Adastral Park Research Campus, University College London, London, U.K. He is currently with the Faculty of the National Mobile Communications Research Laboratory, Southeast University. His research interests include wireless communications, random matrix theory, and information theory.

Dr. Jin is serving as an Area Editor for the IEEE Transactions on Communications and IET Electronics Letters. He was an Associate Editor for the IEEE Transactions on Wireless Communications, and IEEE Communications Letters, and IET Communications. Dr. Jin and his coauthors have been awarded the IEEE Communications Society Stephen O. Rice Prize Paper Award in 2011, the IEEE Jack Neubauer Memorial Award in 2023, The IEEE Marconi Prize Paper Award in Wireless Communications in 2024, and the IEEE Signal Processing Society Young Author Best Paper Award in 2010 and Best Paper Award in 2022.

Xinping Yi (S'12-M'15) received his Ph.D. degree in Electronics and Communications in 2015 from Télécom ParisTech, Paris, France. He is currently a Professor at the National Mobile Communications Research Laboratory, Southeast University, China. Prior to that, he has been an Assistant Professor at University of Liverpool, United Kingdom, a Postdoctoral Research Associate at Technische Universität Berlin, Germany, and a Research Assistant at EURECOM, France. His main research interests include network information theory, statistical learning theory, graph machine learning, and their applications in wireless communications and trustworthy artificial intelligence.