gPerfIsol: GNN-based Rate-Limits Allocation for Performance Isolation in Multi-tenant Cloud

Kokouvi Benoit Nougnanke, Justin Loye, Jean-François Baffier, Simone Ferlin, Marc Bruyère, Yann Labit

gPerfIsol: GNN-based Rate-Limits Allocation for Performance Isolation in Multi-tenant Cloud

Benoit Nougnanke*, Justin Loye*, Jean-François Baffier*, Simone Ferlin†, Marc Bruyere*, Yann Labit‡
* *IIJ Research Laboratory, Japan*
† *Red Hat and Karlstad University, Sweden*
‡ *LAAS-CNRS, Université de Toulouse, CNRS, UPS, F-31400 Toulouse, France*
benoit@iij.ad.jp, jloye@iij.ad.jp, baffier@iij.ad.jp, sferlinr@redhat.com, marc@iij.ad.jp, ylabit@laas.fr

***Abstract*—Performance Isolation in Multi-Tenant Cloud Data Centers (MTCDCs) consists of a set of mechanisms to make sure tenants' use of resources does not impact other tenants. In this context, traffic shapers and rate limiters are fundamental to addressing the challenges of performance isolation in MTCDCs, which include predictable performance as minimum bandwidth guarantees, tenants-level fairness, and optimal resource utilization. However, the classical linear programming process to find the optimal rates to apply does not scale in terms of computing time, especially with the huge number of nodes, dominated mainly by virtual machines in an MTCDC environment. Motivated by this observation, this paper introduces gPerfIsol, a novel Graph Neural Network (GNN)-based approach designed to find near-optimal rates allocation in near-real-time to ensure performance isolation in MTCDC. gPerfIsol's key innovation leverages Heterogeneous GNNs to capture MTCDC-specific topological information and demand traffic matrix. Evaluations based on datasets generated through simulation demonstrate the effectiveness of gPerfIsol's binary classification model with a precision score of 0.964 and a recall score of 0.973. Ultimately, gPerfIsol offers a promising solution for near-optimal rate limit allocation for traffic shapers in multi-tenant environments, enhancing performance isolation.**

***Index Terms*—Multi-tenancy, Cloud, Performance Isolation, Rate Limiters, GNN, Network Optimization**

I. Introduction

Performance isolation is an important challenge in cloud environments serving multiple customers, more specifically, Multi-Tenant Cloud Data Centers (MTCDCs), where network resources are shared among different users [1]. This isolation consists of a set of mechanisms to ensure that resource usage by one tenant does not negatively impact others, a problem called the noisy neighbor problem [2]–[4]. For this reason, efficient performance isolation is critical in MTCDC networks to ensure the following three goals: predictable performance through minimum bandwidth guarantees, fair QoS impact during overload, and optimal resource utilization while meeting Service Level Objective (SLO) guarantees [5], [6].

Existing solutions for performance isolation commonly rely on rate limits consumed by traffic shapers [7]–[10]. The traditional way of finding these adequate rates at the tenant and virtual machine (VM) levels requires formulating the problem as an optimization problem and finding, through Linear Programming (LP) solvers, optimal rate allocations to satisfy the objective. However, in multi-tenant cloud data centers, unlike Wide Area Networks (WANs) or single-tenant data centers, the higher number of nodes, coupled with complex isolation constraints, can make the optimization challenging. Moreover, the control time granularity requirements of MTCDCs performance isolation (in seconds or milliseconds if possible) differ significantly from those of WANs traffic engineering (generally at 5 min time intervals) [11], [12].

In response to these challenges, this paper introduces gPerfIsol, a novel resource allocator based on Graph Neural Networks (GNNs) [13] and heterogenous graph transformation, designed to generate near-optimal rate limits allocation upon a demand traffic matrix of an MTCDC network. These rate limits provide adequate inputs to traffic shapers in addressing the performance isolation challenges in multi-tenant cloud environments. gPerfIsol constitutes an efficient near-real-time rate limit allocation mechanism by leveraging offline learning coupled with fast inference to enforce minimum bandwidth requirements for tenants while ensuring fair bandwidth allocation and high resource utilization, as would output an LP solver but in a much longer time.

This work presents several distinct contributions compared to existing research from the literature. First, to the best of our knowledge, this is the first work applying GNN to solve rate limit allocations for MTCDC traffic shapers. Previous works focus on shapers taking for granted the optimal rates they might consume. Indeed, these rates depend on Service Level Agreements (SLAs), expressed here as minimum and maximum bandwidths, and the traffic patterns. They must also satisfy high utilization and fairness objectives posed for MTCDC performance isolation. Second, while GNN-based techniques have been successfully applied to resource allocation for WANs traffic engineering (TE), their direct application to MTCDCs is challenging due to their specific characteristics, including heterogeneous nodes, especially VMs, tenant-level fairness, and guarantees. This work introduces the innovative use of heterogeneous GNNs to capture the specificities of MTCDCs in the input graph of GNN layers, a key differentiator from prior research. Overall, this paper marks the first application of heterogeneous GNN-based rate limit allocators to traffic shapers for performance isolation in MTCDCs.

Through preliminary evaluations, we demonstrate that gPerfIsol, by effectively capturing topological information, node-

JSPS-CNRS Postdoctoral Fellowship for Research in Japan. This work was supported by JSPS KAKENHI Grant Number JP22KF0428.

edge features, and demands information, can output timely near-optimal rate allocations to satisfy the optimization problem objective while satisfying the constraints. gPerfIsol's binary classification model has a precision score of 0.964 and a recall score of 0.973. These results highlight the potential of gPerfIsol as a promising solution for near-real-time and near-optimal resource allocation for traffic control and rate limiting, offering enhanced performance isolation in multi-tenant environments.

The remainder of this paper is structured as follows. Section II presents background on MTCDC and the scenario considered for the study. Section III presents the mathematical formulation of the problem, the motivation of this work, and the introduction to GNNs. In Section IV, we present the design and architecture of gPerfIsol. The evaluation results are presented in Section V after providing detailed information on the simulation setup. Section VI discusses related works. Finally, Section VII summarizes our contributions and outlines potential future research works.

II. Background and Scenario

This section provides background on MTCDC with information on how virtualization and overlays are involved. It also describes the scenario considered throughout the paper.

A. MTCDC and Overlays

Practically speaking, the implementation of a multi-tenant cloud environment is done using networking overlays [14], [15], implemented over tunneling protocols such as Virtual Extensible LAN (VxLAN), Generic Network Virtualization Encapsulation (GENEVE), among others, on top of IP networks (underlay) [16] [17]. However, this only handles one aspect of isolation, namely, the logical isolation of traffic when it comes to security concerns [18]. It does not guarantee nor prevent performance interference between tenants [19] [20] [21]. Solving performance isolation requires creating new network-level isolation techniques with a tenant-centric perspective. These new solutions should consider that traditional 5-tuple-flow-centric methods are not suitable due to unfairness aspects [1], as well as overlay networking cannot guarantee performance isolation alone. Candidate solutions to tackle this challenge include traffic shapers and rate limiting [2], [7]–[10].

While traffic shapers are already central to achieving performance isolation, this work focuses on defining *optimal* rate limits implemented by these traffic shapers with the goal of improving performance isolation between tenants.

B. MTCDC Scenario

Scenario Topology. Figure 1 illustrates the topology considered for the MTCDC scenario. It is a leaf-spine topology, a 2-tier classical Clos topology used for medium-sized data center networks [22], [23]. For hyper-scale data centers, a 3-tier Clos topology involving super-spine switches is used. The physical hosts hold VMs for different tenants with their specific requirements (especially minimum bandwidth and upper bandwidth limit). This topology has the advantage of providing

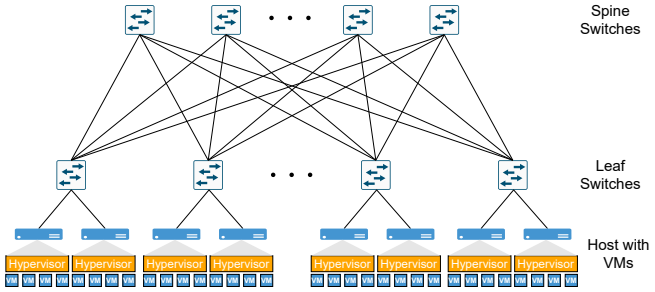

Fig. 1. Experimental Setup

several multiple paths and high bisection bandwidth [24]. The underlay is an L3 network from host to host through the switches leaf-spine-leaf, and the overlays (e.g., VXLAN) are deployed for tenant to isolate their VMs on a virtualized data center identified by a VNI offering L2 segments per tenant.

Service Model. In our work, we considered the hose model [25], in line with other related studies [5], [6], [8]. In this model, each tenant's virtual data center is represented through a single non-blocking switch, with each of the tenant's VMs connected through a dedicated link with a minimum bandwidth and maximum bandwidth envelope.

VMs Distribution. The total number of VMs in the MTCDC network ($N = \textit{number of hosts} \cdot \textit{maximum VM count}$) is divided into two equal-number groups: sending VMs and receiving VMs similar to the setup in the paper [26]. To achieve a representative VM distribution per tenant, we employed a Pareto-like distribution approach, where each tenant has at least a sending VM and a receiving VM.

Traffic Generation. Upon the network topology, we generate traffic matrices of the VM nodes essential for the MTCDC scenario. The traffic generation process, inspired by the approach used in *TMgen* [27], includes various parameters such as the `mean_traffic` (the average total volume of traffic) chosen randomly between zero and the `maximum_bandwidth` and `traffic_pattern` (e.g., many-to-many, incast, or random). These parameters are randomly selected for each tenant, resulting in diverse traffic patterns.

III. Mathematical Formulation and Motivation

A. Mathematical formulation

This section defines the mathematical formulation of the MTCDC performance isolation problem. We aim to optimize rate limit allocation in a multi-tenant cloud data center to ensure operational efficiency (high resource utilization) and good tenant experience (satisfied demand, minimum bandwidth guarantees, proportional and fair performance).

MTCDC Network: Let's consider a snapshot graph $G = (V, E, c)$ representing the network topology for the next time interval window, where:

- V is the set of nodes in the network, including VMs, hosts, leaf switches, and spine switches.
- E is the set of edges connecting these nodes.
- $c : E \to \mathbb{R}^+$ assigns capacities to the edges. c_{ij} represents the capacity of the edge ij.

Traffic Demands: Let D be a traffic matrix representing the demands between pairs of VMs (always within the same tenant). Each demand or commodity d_k corresponds to traffic from a source VM s_k to a destination VM t_k. Let K be the total number of demands in D. These demands are considered fixed for the next time interval and are provided as input to the rate limit allocation module.

Variables:

$$x_{ij} : \text{Flow on edge between node } i \text{ and node } j \quad \forall i, j \in E$$
$$y_k : \text{Fraction served from demand } d_k \quad \forall k \in \{1, \ldots, K\}$$

where x_{ij} and y_k are positive variables.

Objective Function:

$$\text{Maximize:} \quad \sum_k y_k \cdot d_k$$

Constraints:

Capacity Constraints :

$$x_{ij} \leq c_{ij} \quad \forall i, j \in E$$

Flow Conservation constraints:

$$\sum_j x_{ij} - \sum_j x_{ji} = \begin{cases} y_k \cdot d_k, & i = s_k \\ 0, & i \neq s_k, t_k \\ -y_k \cdot d_k, & i = t_k \end{cases}$$

Minimum bandwidth constraints:

$$\gamma_i \leq \sum_j x_{ij} - \sum_j x_{ji} = y_k \cdot d_k, \text{ for } i = s_k$$

where γ_i attached to each VM, represents a tenant metadata related to SLO/SLA, here the minimum bandwidth guarantee. The maximum bandwidth guarantee is encoded in the edge capacity from the VM to its upstream host C_{ij}. Equation (1) summarizes the MTCDC performance isolation rate limits allocations problem:

$$\text{Maximize:} \quad \sum_k y_k \cdot d_k$$

Subject to:

$$\begin{aligned} & x_{ij} \leq c_{ij} \quad \forall i, j \in E \\ & \sum_j x_{ij} - \sum_j x_{ji} = \begin{cases} y_k \cdot d_k, & i = s_k \\ 0, & i \neq s_k, t_k \\ -y_k \cdot d_k, & i = t_k \end{cases} \\ & \gamma_i \leq \sum_j x_{ij} - \sum_j x_{ji} = y_k \cdot d_k, \text{ for } i = s_k \end{aligned} \tag{1}$$

The objective is to maximize the total fractions of demand served and the constraints, including capacity constraints, flow conservation constraints, and minimum bandwidth constraints. Each constraint ensures that the network operates within its limits while satisfying the demands of each tenant with $G = (V, E, c)$. As such, this formulation falls under the general class of multi-commodity flow problems [28].

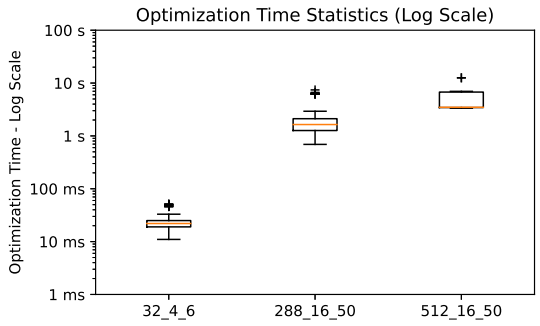

Fig. 2. Boxplot of optimization times in log scale. The figure displays the distribution of the optimization times (in milliseconds or seconds) for three different configurations: `32_4_6` (32 hosts, 4 VMs per host, 6 tenants), `288_16_50` (288 hosts, 16 VMs per host, 50 tenants), and `512_16_50` (512 hosts, 16 VMs per host, 50 tenants). The y-axis is in logarithmic scale, showing a wide range of optimization times from 1 ms to 12.5 s.

B. Scaling Challenges

Fig. 2 shows the optimization time observed from solving Equation (1) for three different configurations with the Pulp Linear Programming tool and the CBC solver [29]. It shows scalability challenges, particularly when faced with an increasing number of network nodes. In these scenarios, such as the one involving 512 hosts and 16 maximum VMs, meaning 8192 total VMs, the optimization process comes with increased complexity: As the total number of VMs, hosts, and tenants increases, the computational time also increases. More specifically, with an increase in the number of hosts from 288 to 512, almost double, the observed optimization time, as indicated by the median, rose from 1.64s to 3.48s. These observations highlight the need for efficient and scalable optimization algorithms, particularly in large-scale multi-tenant cloud data centers (e.g., in [30]–[32] targeting 100.000 hosts, going up to around 1.6 million VMs).

Thus, in order to mitigate these high optimization times, we explore in the rest of the paper the integration of GNNs. The neural network is trained offline by learning from optimal allocations derived from the LP solver and aims to generalize effectively to unobserved data with a fast online inference.

C. Graph Neural Networks (GNNs)

Graph Neural Networks (GNNs) represent a category of deep learning techniques specifically designed for performing inference on graph-structured data. During training, GNNs leverage both the graph structure and node/edge features to perform tasks such as regression and classification at various levels: node-level, edge-level, and graph-level [13], [33] [34]. At present, GNNs have found success in various domains, including social networks, recommendation systems, and chemistry [35].

The main construct of GNNs is GNN layers. Classical GNN layers like Graph Convolutional Networks (GCN) [36], GraphSAGE [37], Graph Attention Networks (GAT) [38], and computer networking domain-specific variants like RouteNet [39] and FlowGNN [40]. GNNs operate by enabling message

passing and aggregation, facilitating the integration of contextual information and relationships into node embedding. The stacking of multiple GNN layers enables the capture of increasingly complex patterns and dependencies that allow addressing various predictive challenges within graph-structured data. This way, GNNs extract relevant features from the underlying graph structure, thereby improving predictions for entities involved in graph interactions, as compared to models that consider entities in isolation.

The training pipeline for GNNs involves processing the input graph through GNN layers to generate node embeddings. These embeddings are then used as input for prediction heads, which address node, edge, or graph-level tasks. Evaluation metrics, labels, and loss functions guide the training process.

D. Leveraging GNNs for Optimization and Rate Limit Allocation: Challenges

Applying GNNs to optimization and resource allocation introduces some challenges that demand careful consideration:

Firstly, achieving optimal rate limit allocation through GNNs requires the design of an effective input graph representation. This includes well-defined node features, topological information, and domain-specific attributes. Secondly, the judicious selection of suitable GNN layers is important. Each layer must align with the specific requirements of the task, as different rate-limiting settings may necessitate distinct GNN architectures. Thirdly, evaluating and exploiting the expressiveness of GNNs may be challenging. This involves determining how effectively GNNs can transform input data into informative node embeddings, and how to leverage these embeddings to complete the rate-limits allocation tasks.

Finally, addressing these challenges necessitates a certain comprehension of GNNs architecture and designs, which we will demonstrate applied to the problem of rate-limit allocation in multi-tenant clouds in the following.

IV. gPerfIsol Design and Implementation

A. Architecture

The general architecture of gPerfIsol, as depicted in Fig. 3 is composed of two main planes: the management-control plane where the gPerfIsol module operates and the data plane within the MTCDC network. The bandwidth demand predictor supplies the next interval (in a time-window-based control, e.g., each $T = 1s$) demands as traffic matrices to the gPerfIsol module, which leverages learning-based rate limit allocations to provide near-optimal allocations of the demands while complying with the constraints from Equation (1). The rate-limits allocation is performed independently for each time interval and its associated demand matrix.

The design of gPerfIsol accommodates a fixed physical underlay topology with variable and dynamic overlay networks composed of the VMs and their corresponding demands. This adaptability allows the system to adjust to varying demand sizes and tenant VM distributions.

The architecture of gPerfIsol is designed to address the critical challenge of performance isolation in multi-tenant

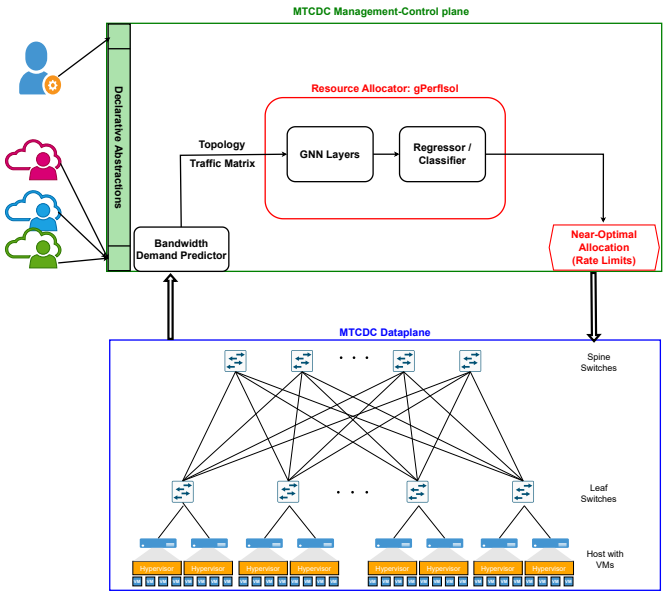

Fig. 3. gPerfIsol Architecture within MTCDC Management-Control Loop

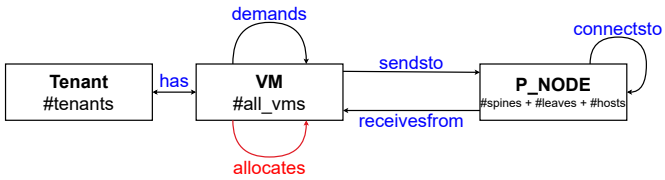

Fig. 4. Graph Representation: MTCDC as a Hetero-Graph

cloud environments. It is built upon a foundation of GNNs and is intended to allocate rate limits efficiently. At its core, gPerfIsol takes as input data a graph known as MTCDC with capacity information, VM-tenant metadata and a workload matrix representing demands between VMs. We represent this data with an heterogeneous graph (different types of nodes and edges) using the hetero-graph abstraction in Fig. 4. This graph modelization represents well the MTCDC with isolation requirements better than going directly with a simple graph. gPerfIsol leverages a sequence of GNN layers, with the first layer having 64 hidden channels and the output layer with a single layer. The primary objective of these GNN layers is to capture the essential features of the input graph and obtain node embeddings that represent the unique characteristics of each node, especially VM nodes.

The subsequent phase in the gPerfIsol architecture involves using a neural network to predict optimal allocations for the demand d_k by taking as inputs the embeddings of the source VM S_k and destination VM T_k of the demands. This final stage, which could have a binary classification function or a regression depending on the admission control goal or finding optimal allocation as a proportion to a demand that could be satisfied, is responsible for deriving edge-level predictions (allocation decisions) from these embeddings.

The architecture leverages a heterogeneous graph representation to model the MTCDC as topology-capacities information and demand matrix effectively. This representation intro-

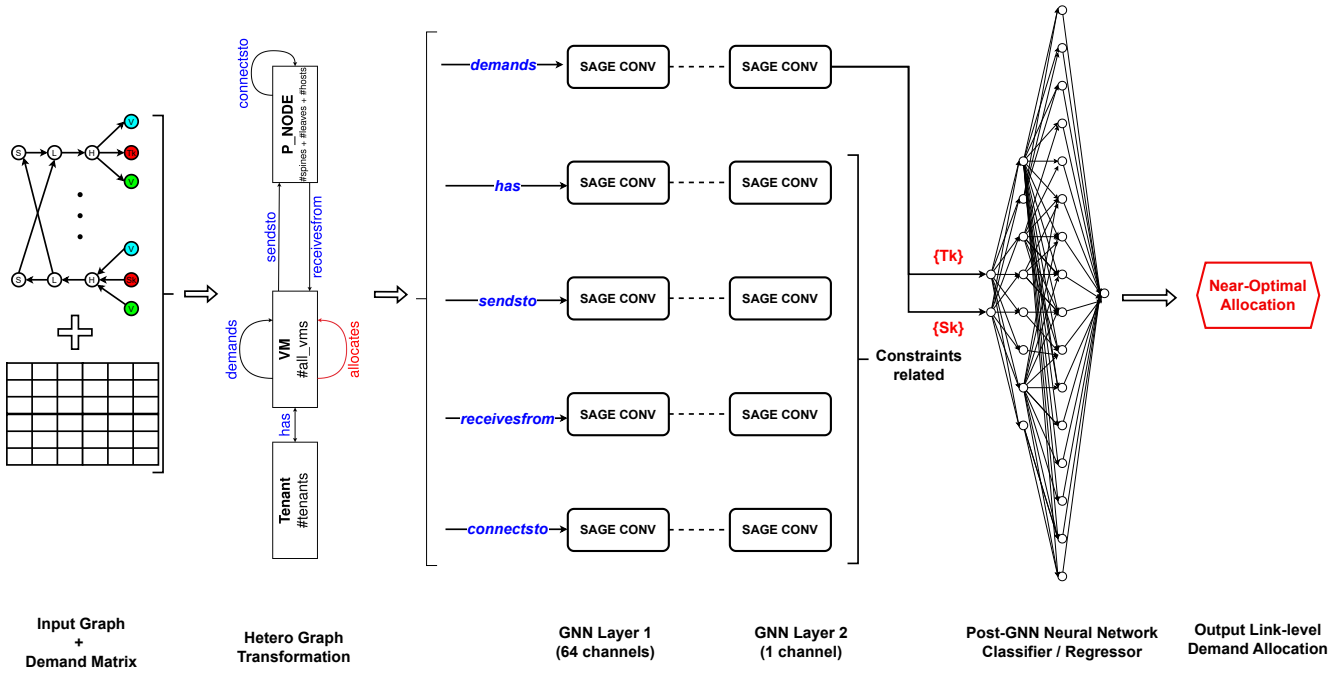

Fig. 5. gPerfIsol Workflow

duces specific edge types, such as VM-VM workload edges, providing a more expressive way to capture the relationships within the MTCDC. However, this expressiveness comes at the cost of increased computational and storage and a more complex implementation than traditional homogeneous GNNs.

B. gPerfIsol Resource Allocation Model Implementation

The implementation of the gPerfIsol rate limits allocation model utilizes the PyG (PyTorch Geometric) [41] and PyTorch libraries [42], a widely adopted tool for deep learning tasks. The model architecture comprises GNNs and a classifier. Specifically, we employ the SAGEConv layer for the GNN, which consists of two convolutional layers. The GNN processes input data from a heterogeneous graph, transforming it into node embeddings. These embeddings are then passed to the classifier, a neural network of multiple linear layers.

V. Evaluation

A. Evaluation Setup

Dataset. To generate a working dataset, we consider the scenario described in Section II-B with the topology Fig. 1 using Python simulations. For this configuration $number\_of\_host = 288$, $max\_vm\_count = 16$, $number\_of\_tenant = 50$, we create the corresponding MTCDC Leaf-Spine Clos topology. The numbers of spines and leaves are calculated based on the number of hosts, following practical considerations for Clos network topologies [22]. Upon the network topology, we generate VM pair traffic matrices as their estimated bandwidth demands for the next time interval, with the individual tenants having their own traffic patterns (many-to-many, incast, random, etc.).

Finally, taking the network topology information with the generated traffic matrix, we compute optimal resource allocations using the Pulp Linear Programming tool with the CBC solver [29]. These optimal allocations are collected with the input topology graph with capacity information and the demands to construct our dataset.

The considered dataset for the 288_16_50 configuration comprises 151 distinct graphs with their demands. It is split into 105 training graphs, 22 validation graphs, and 24 test graphs. After hetero data transformation, we count 4982 nodes for each graph and a mean of 107452 edges (with a standard deviation of 45864.4). The total number of demands across all 151 graphs is 14050862. Each demand's allocation (a value in $[0, 1]$) represents a fraction of the demand that can be satisfied. These demand allocations, when rounded, include 4125700 zeros (when the demand is rejected) and 9925162 ones (when all the demand is satisfied). These are the labels used during training. We consider here a binary classification setting corresponding to admission control-like bandwidth resource allocation for performance isolation [6].

Metrics. We consider binary-classification-related metrics, such as *precision*, *recall*, *F1-score*, and *accuracy* [43]. In this binary classification, we define the class with the labels equal to "one" as the positive class, and the "zeros" correspond to the negative class. The related metrics are defined based on four fundamental values: *true positives*, *true negatives*, *false positives*, and *false negatives*. True positives represent the instances correctly classified as positive, true negatives are instances correctly classified as negative, false positives are instances incorrectly classified as positive, and false negatives are instances incorrectly classified as negative.

The *precision* metric is defined as $precision =$

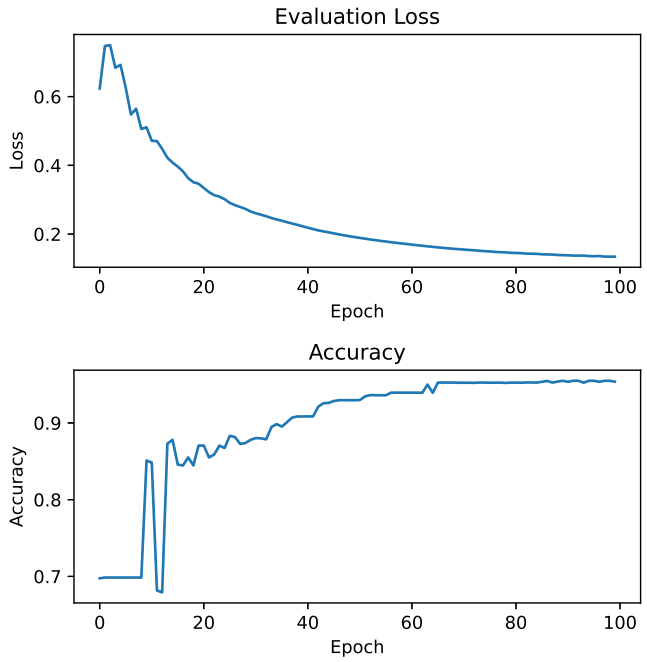

Fig. 6. Model Evaluation Results

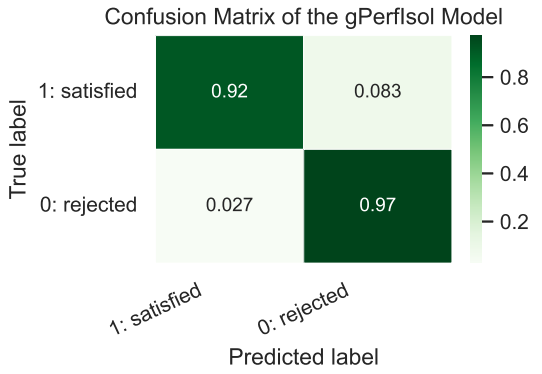

Fig. 7. Evaluation Results: Confusion Matrix

Label	Precision	Recall	F1-score	Support
1: satisfied	0.94	0.92	0.93	769,699
0: rejected	0.96	0.97	0.97	1,743,274
Accuracy			0.96	2,512,973
Macro avg	0.95	0.94	0.95	2,512,973
Weighted avg	0.96	0.96	0.96	2,512,973

TABLE I
CLASSIFICATION REPORT

$\frac{true\ positives}{true\ positives + false\ positives}$ as the ratio of true positives to the total instances predicted as positive, while $recall = \frac{true\ positives}{true\ positives + false\ negatives}$, also known as sensitivity or true positive rate, is the ratio of true positives to the total actual positive instances. These metrics comprehensively evaluate the model's performance, balancing correctness in positive predictions (precision) and capturing all positive instances (recall).

The *F1-score* combine *precision*, *recall* in their harmonic mean $F1\text{-}score = 2.\frac{precision \cdot recall}{precision + recall}$, considering *precision*, *recall* are evenly weighted. These weights could be adjusted depending on the overall objective. Indeed, for the performance isolation case, the weights' adjustment may depend on the weight put on overutilization that may lead to congestion or underutilization, which corresponds to resource inefficiency, but where admitted demands will not suffer congestion. In other words, missing a "one" may lead to underutilization, while missing "zero" may lead to network congestion.

B. gPerfIsol Model Training

We train the gPerfIsol model using the Adam optimizer and binary cross entropy loss on 100 epochs. The validation loss and accuracy are shown in Fig. 6. It shows how the model learns through the epochs before stabilizing around a classification accuracy of 0.95 and loss error of 0.13 on the validation set.

C. Evaluation Results

The gPerfIsol model's binary classification approach addresses the resource allocation challenge outlined in Equation (1). This transforms the resource allocation problem into an admission control version with interesting results: With a precision score of 0.964, a recall score of 0.973, and an F1 score of 0.968, it effectively balances the trade-offs between precision and recall. Overall, this binary classification capability of the model demonstrates its effectiveness in optimizing resource allocation with good classification accuracy. The detailed information on the classification performance is presented in Table I. We also analyze atomic prediction times, where the minimum time is 0.0093 seconds, the mean time is 0.0239 seconds, the median time is 0.0210 seconds, the standard deviation is 0.0149 seconds, and the maximum time is 0.0626 seconds. The time granularity is interesting for the MTCDC performance isolation time requirement. They could be improved using parallelization and specialized hardware like GPUs that are useful for accelerating neural network computing.

VI. RELATED WORKS

In this section, we provide an overview of related work on network optimization and traffic engineering, mainly focusing on those leveraging machine learning (ML) techniques.

ML for Network Optimization. In the domain of network optimization, ML has become increasingly relevant. Notable contributions in this area include TEAL [12] and RouteNet [44] [45] that use GNNs, which concern learning-based accelerated traffic engineering (TE) and routing optimizing in the WAN respectively. RouteNet [44] also tackles performance modeling. CFR-RL [46] focuses on critical flow identification in traffic matrices on ISP topologies using ML, specifically deep neural networks (DNN) as part of the TE pipeline. Additionally, the work [47] proposes ML performance modeling of data center traffic (including incast and elephant) and leverages the ML-based performance model in

bayesian optimization for smart switch buffer management and traffic optimization.

ML in Wide Area Network Traffic Engineering.. TEAL [12] proposes FlowGNN that alternates between GNN layers to capture capacity constraints and DNN layers for capturing demand constraints for addressing traffic engineering on a cloud WAN composed of data center nodes. It assumes the next interval demand matrix is available through a bandwidth broker. DOTE [11] proposes rethinking WAN TE leveraging a neural network to tackle TE in a single process, bypassing demand traffic matrix prediction used as input for TE optimizer. It directly optimizes TE objectives using only historical demand matrices. Some other works use ML coupled with Deep Reinforcement Learning for intradomain traffic engineering in WANs [48] [49].

Network Optimization in Intra-Cloud Data Centers. Several works have provided valuable insights in the context of intra-cloud data centers, particularly those employing Clos leaf-spine topologies. MicroTE [50] addresses the segregation of predictable and non-predictable flows within single-tenant university data centers. This fine-grained approach aims to optimize resource allocation. Joint VM Placement and Routing [51] presents strategies for efficient resource management within data centers.

These studies contribute to the evolving landscape of network optimization, incorporating ML techniques and novel approaches to address the challenges in modern networking environments. However, while GNN-based techniques have been successfully applied to resource allocation for WANs TE, especially TEAL [12], their direct application to multi-tenant intra-cloud data centers is challenging due to their specific characteristics, including heterogeneous nodes (VMs, hosts, and switches), tenant-level fairness, and guarantees. Our approach introduces the innovative use of heterogeneous GNNs to capture the specificities of MTCDCs in the input graph of GNN layers, a key difference. This work marks the first application of heterogeneous GNN-based rate limit allocators to traffic shapers for performance isolation in MTCDCs.

VII. Conclusion

This paper introduces gPerfIsol, a GNN-based approach that addresses the scalability challenge of the rate limit allocation optimization problem for performance isolation in MTCDC. By applying heterogeneous GNNs to capture MTCDC-specific topological information and training them offline with optimal allocations from an LP solver, gPerfIsol provides near-optimal rate limits in near-real-time. The promising results obtained in our preliminary evaluations provide evidence of gPerfIsol's efficacy, emphasizing its potential to enhance performance isolation by providing good inputs to traffic shapers and rate limiters.

Future research works can build upon this foundation, exploring more sophisticated GNN architectures, experiments on larger-scale MTCDC, and investigating alternative problem formulation of the performance isolation problem. Additionally, an interesting exploration may bypass the optimal solver process that provides labels in the supervised learning used here.

References

[1] J. Chen, H. Zhang, W. Zhang, L. Luo, J. S. Chase, I. Stoica, and D. Zhuo, "Nethint: White-box networking for multi-tenant data centers," in *Symposium on Networked Systems Design and Implementation*, 2022.

[2] M. Noormohammadpour and C. S. Raghavendra, "Datacenter traffic control: Understanding techniques and tradeoffs," *IEEE Communications Surveys & Tutorials*, vol. 20, pp. 1492–1525, 2017.

[3] A. Ather, "Noisy neighbors," https://medium.com/@amerather_9719/noisy-neighbors-fa167359d39e, 2023.

[4] G. H. Benoit Rostykus, "Predictive cpu isolation of containers at netflix," https://netflixtechblog.com/predictive-cpu-isolation-of-containers-at-netflix-91f014d856c7, 2019.

[5] L. Popa, A. Krishnamurthy, S. Ratnasamy, and I. Stoica, "Faircloud: sharing the network in cloud computing," in *CCRV*, 2011.

[6] S. Wang, K. Gao, K. Qian, D. Li, R. Miao, B. Li, Y. Zhou, E. Zhai, C. Sun, J. Gao, D. Zhang, B. Fu, F. P. Kelly, D. Cai, H. H. Liu, and M. Zhang, "Predictable vfabric on informative data plane," *Proceedings of the ACM SIGCOMM 2022 Conference*, 2022.

[7] A. Saeed, N. Dukkipati, V. Valancius, V. The Lam, C. Contavalli, and A. Vahdat, "Carousel: Scalable traffic shaping at end hosts," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, 2017, pp. 404–417.

[8] P. Kumar, N. Dukkipati, N. Lewis, Y. Cui, Y. Wang, C. Li, V. Valancius, J. Adriaens, S. D. Gribble, N. Foster, and A. Vahdat, "Picnic: predictable virtualized nic," *Proceedings of the ACM Special Interest Group on Data Communication*, 2019.

[9] V. Jeyakumar, M. Alizadeh, D. Mazières, B. Prabhakar, A. G. Greenberg, and C. Kim, "Eyeq: Practical network performance isolation at the edge," in *Symposium on Networked Systems Design and Implementation*, 2013.

[10] H. Rodrigues, J. R. Santos, Y. Turner, P. Soares, and D. O. Guedes, "Gatekeeper: Supporting bandwidth guarantees for multi-tenant datacenter networks." *WIOV*, vol. 1, no. 3, pp. 784–789, 2011.

[11] Y. Perry, F. V. Frujeri, C. Hoch, S. Kandula, I. Menache, M. Schapira, and A. Tamar, "Dote: Rethinking (predictive) wan traffic engineering," in *20th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2023, Boston, MA, April 17-19, 2023*, M. B. 0001 and M. Ghobadi, Eds. USENIX Association, 2023, pp. 1557–1581. [Online]. Available: https://www.usenix.org/conference/nsdi23/presentation/perry

[12] Z. Xu, F. Y. Yan, R. Singh, J. T. Chiu, A. M. Rush, and M. Yu, "Teal: Learning-accelerated optimization of wan traffic engineering," *Proceedings of the ACM SIGCOMM 2023 Conference*, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:257079293

[13] A. Daigavane, B. Ravindran, and G. Aggarwal, "Understanding convolutions on graphs," *Distill*, 2021, https://distill.pub/2021/understanding-gnns.

[14] V. Del Piccolo, A. Amamou, K. Haddadou, and G. Pujolle, "A survey of network isolation solutions for multi-tenant data centers," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2787–2821, 2016.

[15] T. Narten, E. Gray, D. Black, L. Fang, L. Kreeger, and M. Napierala, "Problem statement: Overlays for network virtualization," Internet Engineering Task Force, Tech. Rep. RFC 7364, 2014.

[16] T. Narten, E. Gray, D. L. Black, L. Fang, L. Kreeger, and M. Napierala, "Problem statement: Overlays for network virtualization," *RFC*, vol. 7364, pp. 1–23, 2014.

[17] M. Mahalingam, D. G. Dutt, K. J. Duda, P. Agarwal, L. Kreeger, T. Sridhar, M. Bursell, and C. Wright, "Virtual extensible local area network (vxlan): A framework for overlaying virtualized layer 2 networks over layer 3 networks," *RFC*, vol. 7348, pp. 1–22, 2014.

[18] K. Thimmaraju, S. Hermak, G. Rétvári, and S. Schmid, "Mts: Bringing multi-tenancy to virtual networking," in *USENIX Annual Technical Conference*, 2019.

[19] W. Lin, C. Xiong, W. Wu, F. Shi, K. Li, and M. Xu, "Performance interference of virtual machines: A survey," *ACM Computing Surveys*, vol. 55, pp. 1 – 37, 2022.

[20] R. S. Kannan, M. Laurenzano, J. Ahn, J. Mars, and L. Tang, "Caliper: Interference estimator for multi-tenant environments sharing architectural resources," *ACM Trans. Archit. Code Optim.*, vol. 16, pp. 22:1–22:25, 2019.

[21] R. Yang, I. S. Moreno, J. Xu, and T. Wo, "An analysis of performance interference effects on energy-efficiency of virtualized cloud environments," in *2013 IEEE 5th International Conference on Cloud Computing Technology and Science*, vol. 1. IEEE, 2013, pp. 112–119.

[22] D. G. Dutt, *Cloud native data center networking: architecture, protocols, and tools*. O'Reilly Media, 2019.

[23] A. Abhashkumar, K. Subramanian, A. Andreyev, H. Kim, N. K. Salem, J. Yang, P. Lapukhov, A. Akella, and H. Zeng, "Running bgp in data centers at scale," in *Symposium on Networked Systems Design and Implementation*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:233377949

[24] P. Namyar, S. Supittayapornpong, M. Zhang, M. Yu, and R. Govindan, "A throughput-centric view of the performance of datacenter topologies," *Proceedings of the 2021 ACM SIGCOMM 2021 Conference*, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:236206940

[25] N. G. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramakrishnan, and J. E. van der Merive, "A flexible model for resource management in virtual private networks," in *Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*, 1999, pp. 95–108.

[26] B. M. Maggs and R. K. Sitaraman, "Algorithmic nuggets in content delivery," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 3, pp. 52–66, 2015.

[27] "Tmgen: Traffic matrix generation tool," https://github.com/progwriter/TMgen/tree/master.

[28] "Multicommodity maximum flow problems," https://personal.utdallas.edu/ chandra/documents/networks/net7.pdf.

[29] S. Mitchell, A. Kean, A. Mason, M. O'Sullivan, A. Phillips, and F. Peschiera, "Optimization with pulp," https://coin-or.github.io/pulp/index.html.

[30] M. Chowdhury, Z. Liu, A. Ghodsi, and I. Stoica, "{HUG}:{Multi-Resource} fairness for correlated and elastic demands," in *13th USENIX symposium on networked systems design and implementation (NSDI 16)*, 2016, pp. 407–424.

[31] M. Moshref, M. Yu, A. Sharma, and R. Govindan, "Scalable rule management for data centers," in *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, 2013, pp. 157–170.

[32] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, "Vl2: A scalable and flexible data center network," in *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*, 2009, pp. 51–62.

[33] B. Sanchez-Lengeling, E. Reif, A. Pearce, and A. B. Wiltschko, "A gentle introduction to graph neural networks," *Distill*, 2021, https://distill.pub/2021/gnn-intro.

[34] "Cs224w: Machine learning with graphs stanford / fall 2023," http://web.stanford.edu/class/cs224w/, 2023.

[35] J. Zhou, G. Cui, S. Hu, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph neural networks: A review of methods and applications," *AI open*, vol. 1, pp. 57–81, 2020.

[36] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[37] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[38] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[39] K. Rusek, J. Suárez-Varela, P. Almasan, P. Barlet-Ros, and A. Cabellos-Aparicio, "Routenet: Leveraging graph neural networks for network modeling and optimization in sdn," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 10, pp. 2260–2270, 2020.

[40] Z. Xu, F. Y. Yan, R. Singh, J. T. Chiu, A. M. Rush, and M. Yu, "Teal: Learning-accelerated optimization of wan traffic engineering," in *Proceedings of the ACM SIGCOMM 2023 Conference*, 2023, pp. 378–393.

[41] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[42] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[43] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[44] K. Rusek, J. Suárez-Varela, P. Almasan, P. Barlet-Ros, and A. Cabellos-Aparicio, "Routenet: Leveraging graph neural networks for network modeling and optimization in sdn," *IEEE Journal on Selected Areas in Communications*, vol. 38, pp. 2260–2270, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:203642041

[45] M. F. Galmés, J. Paillissé, J. Suárez-Varela, K. Rusek, S. Xiao, X. Shi, X. Cheng, P. Barlet-Ros, and A. Cabellos-Aparicio, "Routenet-fermi: Network modeling with graph neural networks," *ArXiv*, vol. abs/2212.12070, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:255096226

[46] M. I. J. Zhang, M. Ye, S. M. I. Z. Guo, C.-Y. Yen, and F. I. H. J. Chao, "Cfr-rl: Traffic engineering with reinforcement learning in sdn," *IEEE Journal on Selected Areas in Communications*, vol. 38, pp. 2249–2259, 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:216553045

[47] B. Nougnanke, Y. Labit, M. Bruyère, U. Aïvodji, and S. Ferlin, "Ml-based performance modeling in sdn-enabled data center networks," *IEEE Transactions on Network and Service Management*, vol. 20, pp. 815–829, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:251506222

[48] A. Valadarsky, M. Schapira, D. Shahaf, and A. Tamar, "Learning to route," *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:21220060

[49] A. Valadarsky, M. Schapira, and D. Shahaf, "Learning to route with deep rl," 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:53486647

[50] T. A. Benson, A. Anand, A. Akella, and M. Zhang, "Microte: fine grained traffic engineering for data centers," in *Conference on Emerging Network Experiment and Technology*, 2011. [Online]. Available: https://api.semanticscholar.org/CorpusID:9919854

[51] W. Jiang, T. Lan, S. Ha, M. Chen, and M. Chiang, "Joint vm placement and routing for data center traffic engineering," *2012 Proceedings IEEE INFOCOM*, pp. 2876–2880, 2012. [Online]. Available: https://api.semanticscholar.org/CorpusID:871069