# Autonomous Network Management in Multi-Domain 6G Networks based on Graph Neural Networks

Kaan Aykurt
*Chair of Communication Networks*
*Technical University of Munich*
Munich, Germany
kaan.aykurt@tum.de

Wolfgang Kellerer
*Chair of Communication Networks*
*Technical University of Munich*
Munich, Germany
wolfgang.kellerer@tum.de

***Abstract*—Sixth-generation (6G) networks propose integrating multiple networks and domains while improving network performance. Hence, today's networks are becoming increasingly larger and more complex. Traditional methods to manage networks are facing significant challenges as the topology sizes, traffic patterns, and network domains are changing.**

**This paper presents the state-of-the-art in literature for network management and proposes a research plan for an autonomous network management framework fueled by the Digital Twin (DT) paradigm. Unlike the existing methods such as Queuing Theory (QT) or network simulation studies, the proposed framework relies on state-of-the-art Graph Neural Networks (GNNs) for network performance analysis. We argue that seamless integration of networks while improving performance guarantees can be achieved via autonomous management of networks and present a research plan in this paper.**

***Index Terms*—network management, digital twins, multi-domain networks**

## I. Introduction

Over the past decades, the integration of technology into society is accelerating. The increasing popularity of augmented reality, virtual reality, and telepresence applications shape the requirements for the next generation of networking. While the older generations of network technology focus on enhancing human-to-human communications, the vision for the upcoming 6G technology imagines the future of networking as the integration of the digital and the human world [1]. The focus on human-machine collaboration paves the way for the development of novel use cases that will have distinct and more demanding requirements than the older generations of networks.

The novel forms of networks (e.g. Body Area Networks, Vehicular Networks, and Satellite Networks) pose significant challenges to existing network management paradigms. The individual networks need to fulfill high throughput, strict latency, and high availability constraints and function seamlessly across multiple domains to provide a better user experience in the proposed 6G architectures. The need to integrate multiple networks while maintaining strict performance guarantees (e.g. latency, bandwidth, and jitter) introduces a new challenge to network management. The increasing number of nodes, distinct network characteristics, topologies, and limited control over a third-party network render the existing network management solutions unsuitable for ensuring network guarantees in real-time.

The research on network management covers an extensive range of methods from manually configured networks to software defined networks for providing the best overall network performance. The current trend in the literature is shifting the focus from manual configurations to self-driving and autonomously managed networks. The objective of autonomous network management is to analyze the network and proactively apply better-suited network configurations when adapting to changes in network conditions. For this reason, the keys to successful network management are accurate network performance analysis and the ability to apply *what-if* analysis. In recent years, the advent of the DT paradigm [2] is gaining popularity for autonomous network management as an alternative method to traditional simulation studies in network analysis. A DT enables the mapping of physical systems to their virtual counterparts. The literature focuses on extracting the potential of DTs for network analysis [3]. Although various tools are actively being researched for network performance analysis, an end-to-end framework for autonomous management focusing on multi-domain networks is not yet existent.

This paper proposes a research plan to develop a Machine Learning (ML) based autonomous network management framework to shift human-centered network management solutions into machine-centered alternatives. We focus on extracting the potential of GNNs for the ML-based modeling of networks. The remainder of this paper is structured as follows: Sec. II provides an overview of network management concepts. Sec. III describes the state of the research in the literature. Sec. IV discusses the main challenges and proposes a path for the research forward. We finally conclude and discuss the next steps in Sec. V.

## II. Background

The ultimate goal of network management is to provide the best overall network performance (e.g. high throughput, low latency, and high availability). This section describes the traditional network performance analysis strategies and the state-of-the-art ML-aided methods to benchmark networks.

### A. Traditional Network Analysis

Conventional network performance analysis approaches consist of theoretical analysis, simulation studies, and testbed measurements. Theoretical approaches (e.g. QT [4]) impose strict assumptions on network conditions for modeling. However, these assumptions generally do not hold in real networks and therefore lead to unrealistic performance benchmarks [5]. An alternative to the theoretical approach is computational modeling which is also referred to as network simulation studies in the literature. Such network simulators include a wide range of network elements and protocols in their software. The simulation approach is cheap and faster to implement in comparison to a real testbed setup. However, as the entities in the simulation are replicas of real objects, the implementation in software may differ from its real behavior. The simulation assumes that the hardware behaves as expected, but the research shows that the behavior of hardware can be different than the advertised behavior [6]. In addition to the behavioral differences, simulation is computationally expensive and network simulators cannot simulate complex scenarios in real-time, i.e. in the time necessary for network decisions. Despite its proven potential to accurately predict key network characteristics, the slow execution times render simulations an invalid candidate for DTs. The final method of conventional testbed measurements aims to build a small-scale prototype to measure network performance. The limitation of this approach is the cost and complexity of building a setup consisting of real hardware. Although it is essential to test the configurations in a prototype hardware setup, the ideal framework needs to monitor the real system to analyze the data and manage the network.

Among the existing implementations for network modeling and analysis, no feasible approach exists to provide accurate estimates of network characteristics such as the Flow Completion Time (FCT), delay, and jitter in real-time. Therefore, in order to analyze more complex scenarios, which are typical in 6G architectures, consisting of bigger topologies, more flows, and more packets, the literature has shifted its focus to ML-based alternative methods in network performance analysis. ML-based methods provide the opportunity for faster execution by their lightweight structures, and hence they open possibilities for building DTs of respective architectures.

### B. Digital Twins

A DT is defined as the mapping of a physical world into a digital space. Fig. 1 shows the typical depiction of a DT of a Clos topology-based Data Center Network (DCN) architecture. While the physical DCN is the collection of hardware and resources, a DT can be software that functions in real-time or even faster. The networking community's interest in DTs is growing enormously in the last few years [7]. The potential applications of a DT can help to troubleshoot network problems, detect anomalies and apply *what-if* analysis faster than real-time [3]. Hence, the operations, that are unfeasible in a physical network become viable with DTs.

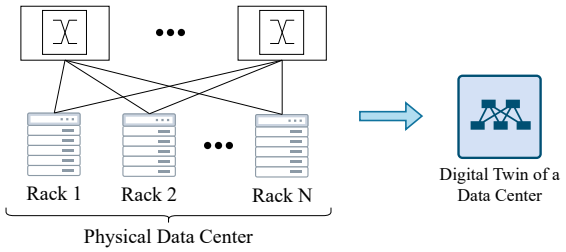


Fig. 1. Overview of a digital twin.

ML models are one of the possible components of a DT. ML applications can learn the behavior of a system after extensive training, and, in general, the execution time of such models is fast. Although early ML-based solutions were not successful in modeling network performance, recent development in GNN-based architectures is gaining popularity. Since the networks can essentially be modeled as graphs, the state-of-the-art network performance analysis implementations, such as RouteNet [8] and xNet [9] rely on GNN-based neural network architectures. Their performance and ability to be combined is the key enabler of the multi-domain autonomous network management framework which will be discussed in Sec. IV.

### C. Graph Neural Networks

In recent years, GNNs [10] are widely used for modeling data which can be expressed as graphs. While traditional neural networks assume a connection structure with a fixed-dimension input space (e.g. fully-connected neural networks, or convolutional neural networks), GNN architecture is determined dynamically based on the input graphs. This means that they have the potential to exploit graph-structured characteristics of the input data to model the relationships between the graph elements.

**Predictions:** A network manager's all-time dream is to sit back and watch the network, while the maintenance tasks are autonomously managed. The goal of network management is to include the human in the loop only for overseeing the actions the network management framework takes. GNN-based ML models create opportunities to predict network characteristics with high accuracy, and therefore serve as a powerful tool for network analysis and autonomous network management.

**Generalization Capabilities:** One of the critical requirements of a successful network management framework is the ability to adapt and perform well under unknown circumstances. GNNs display promising abilities to generalize and perform well for unseen data, topology, and traffic characteristics [11]. Hence, this makes GNNs a good basis to realize DTs.

In addition to network characteristics, domain-specific information can also be modeled as nodes in graphs. Considering modern network topologies, in which multi-domain characteristics also need to be taken into account, the incorporation of domain-specific knowledge plays a key role in a successful network management framework. The remainder of this

work will focus on GNN-based implementations for network performance analysis and propose an autonomous network management framework based on GNN-aided DTs.

## III. Related Work

With the recent developments in ML, currently, state-of-the-art network analysis methods rely on ML-based approaches. A recent survey by Fadlullah et al. [12] discusses different ML-based methods used in various network analysis tasks such as traffic classification, FCT prediction, and latency analysis. In the literature, various approaches are introduced for applying ML models ranging from Support Vector Regression to model queuing, latency, and throughput [13] to Causal Bayesian Networks [14]. These generic approaches can be applied to a wide range of problems. However, just recently the literature focuses on extracting the potential of a graph-based data structure for networks.

GNNs were introduced by Gori et al. [15] with the goal of representing graph-structured data by applying message passing among the nodes of a graph until a fixed location is reached. Until now, GNNs were used in a range of applications from object localization to web page ranking. Recently, GNNs are utilized for network analysis as lightweight models which can be executed much faster than the traditional approaches for network analysis. Among the existing implementations, Deep-Q [16] focuses on predicting path delays by leveraging GNN capabilities. RouteNet [8] also focuses on extracting key network characteristics as well as generalizing them to unseen topologies during training. Most recently, xNet [9] focuses on replacing simulation studies for DCN topologies. The main limitations of existing approaches are that they rely on simulation-generated data and include paths, links, and flows as nodes in their graph. The former limitation leads to inaccurate consideration of hardware characteristics and puts a limit on the capability to come up with a DT that will be fed by real system data. The latter means that the larger and more complex topologies will be harder to model. These limitations make the existing GNN-based implementations unsuitable for large multi-domain network topologies.

One of the main enablers for digital twinning is fast runtime. With the fast execution time of GNNs, digital twinning is possible. Ferriol et al. [17] and Almasan et al. [3] discuss the potentials of GNNs for such DTs. In this work, we propose to come up with a GNN architecture, which is trained and fed by real hardware collected data for building an autonomous network management framework.

## IV. Research Methodology

The goal of the proposed research path is to create an autonomous network management framework that will be used for network performance analysis before deployment to a real system. This section introduces the proposed network management framework, discusses open challenges, and establishes the initial testbed for measurement studies.

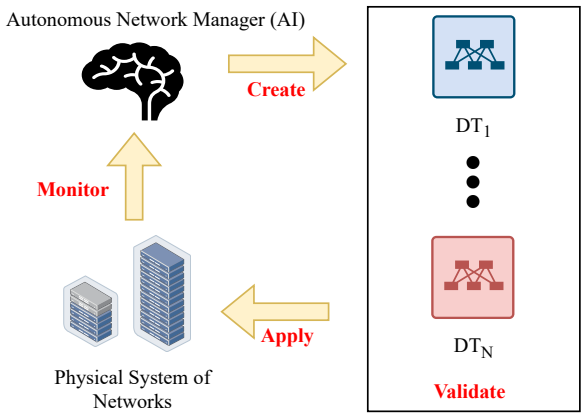


Fig. 2. Proposed autonomous network management framework: AI-aided network manager monitors the network conditions with real-time data collected from the hardware. Upon analyzing the data, it creates a set of candidate configurations. These configurations are tested and validated in a network of DTs. Finally, the selected configuration is applied to the physical system.

### A. Framework

Fig. 2 illustrates the proposed autonomous network management framework. The system is a self-driving framework, in which the configurations are created, validated, and finally applied to the real system in an autonomous manner. The initial step in the framework is the creation of possible candidate configurations, which can proactively improve the network state. The possible set of configurations is created by the Artificial Intelligence (AI) based network manager through monitoring of the physical system. These created configurations are then tested in a network of DTs. We define the network of DTs as the collection of individually twinned networks. This approach makes it feasible to combine distinct twins, such as twins of different network domains, and come up with a functioning validation platform. The final step is the deployment of a configuration in order to be applied to the hardware.

Currently, most of the research focuses on fixing network conditions reactively upon the detection of an anomaly by a set of rules that are pre-determined via the network manager. Therefore, a completely self-monitoring and self-driving network does not exist in the literature. To close this gap, our research will focus on actively monitoring network status via real-time data collection in a hardware setup. With the information obtained through real-time data collection, a pre-trained GNN-based model will be executed to predict the network condition in the future. The ability to execute such a model fast is the key enabler to proactively adapt to changes in network conditions. Future extensions of the framework will consider the addition of DTs of other network domains and inter-domain characteristics.

### B. Challenges

Effective network monitoring is a necessity for reacting to changes in network conditions. The real system must be able to monitor the network conditions appropriately, in order to feed

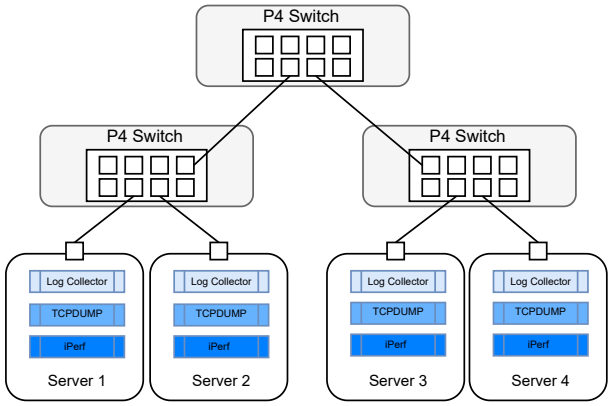


Fig. 3. Overview of the proposed testbed.

the DT correctly. As the planned DTs are implementations of GNNs, correct data collection for successful execution is the key to accurate network modeling. Our research until now shows that data collection in the data plane via P4-enabled devices has the potential to collect meaningful data during runtime, but the limitations need to be explored further.

Modern networks are becoming more and more complex. As the topology sizes are getting bigger and distinct networks are interacting with each other, it is not easy to model the interaction of networks. To this end, we propose to individually twin the network elements to model their behavior across each other. However, due to the limited ability to model the interaction in hardware, the DT needs to have generalization capabilities so that it functions well for unseen topologies.

### C. Testbed

A functioning DT requires the monitoring and collection of accurate data in real hardware. Fig. 3 shows the testbed for the initial measurement study. The formulation of the testbed is to build a simple leaf-spine topology. The testbed consists of four servers and three P4 switches. All the links have $10\,\mathrm{Gbps}$ capacities. Each of the servers is equipped with an iPerf tool for traffic generation. Additionally, we install *tcpdump* and log collection to servers for traffic collection. Log and traffic collection in the endpoints exist merely due to validation reasons for the experiment and they are not planned to be a part of the network management paradigm. Initially, the goal of the system is to collect throughput data and system queue statistics in the data plane. Data collection in the data plane will enable a centralized and fast data collection which can be fed into the GNN-based model for training and execution. P4 programming language enables the successful and fast collection of data centrally without relying on external mechanisms. For this reason, with P4 switches, we have shown that we can append rate statistics to packets in the data plane at line rate. The later stages of research will focus on training a custom-built GNN module with the data collected in hardware. The same model will be the key element of the proactive autonomous network management framework.

## V. Conclusion

This paper presents a research plan for an autonomous network management framework. Based on our initial findings, we believe that data plane applications provide a wide range of possibilities for data collection. We envision that our framework will be fueled by the collected data and result in a proactive autonomous network management framework.

## Acknowledgments

We acknowledge the financial support by the Federal Ministry of Education and Research of Germany (BMBF) in the program of "Souverän. Digital. Vernetzt." joint project 6G-life, project identification number 16KISK002.

## References

[1] H. Viswanathan and P. E. Mogensen, "Communications in the 6g era," *IEEE Access*, vol. 8, pp. 57 063–57 074, 2020.

[2] F. Tao, H. Zhang, A. Liu, and A. Y. Nee, "Digital twin in industry: State-of-the-art," *IEEE Transactions on industrial informatics*, vol. 15, no. 4, pp. 2405–2415, 2018.

[3] P. Almasan, M. Ferriol-Galmés, J. Paillisse, J. Suárez-Varela, D. Perino, D. López, A. A. P. Perales, P. Harvey, L. Ciavaglia, L. Wong *et al.*, "Digital twin network: Opportunities and challenges," *arXiv preprint arXiv:2201.01144*, 2022.

[4] G. Giambene, *Queuing theory and telecommunications*. Springer, 2014, vol. 585.

[5] Z. Xu, J. Tang, J. Meng, W. Zhang, Y. Wang, C. H. Liu, and D. Yang, "Experience-driven networking: A deep reinforcement learning based approach," in *IEEE INFOCOM 2018*. IEEE, 2018, pp. 1871–1879.

[6] A. Van Bemten, N. Đerić, A. Varasteh, A. Blenk, S. Schmid, and W. Kellerer, "Empirical predictability study of sdn switches," in *2019 ACM/IEEE ANCS*. IEEE, 2019, pp. 1–13.

[7] Y. Wu, K. Zhang, and Y. Zhang, "Digital twin networks: A survey," *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 13 789–13 804, 2021.

[8] M. Ferriol-Galmés, K. Rusek, J. Suárez-Varela, S. Xiao, X. Shi, X. Cheng, B. Wu, P. Barlet-Ros, and A. Cabellos-Aparicio, "Routenet-erlang: a graph neural network for network performance evaluation," in *IEEE INFOCOM 2022*. IEEE, 2022, pp. 2018–2027.

[9] M. Wang, L. Hui, Y. Cui, R. Liang, and Z. Liu, "xnet: Improving expressiveness and granularity for network modeling with graph neural networks," in *IEEE INFOCOM 2022*. IEEE, 2022, pp. 2028–2037.

[10] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.

[11] J. Suárez-Varela, S. Carol-Bosch, K. Rusek, P. Almasan, M. Arias, P. Barlet-Ros, and A. Cabellos-Aparicio, "Challenging the generalization capabilities of graph neural networks for network modeling," in *Proc. ACM SIGCOMM Conference Posters and Demos*, 2019, pp. 114–115.

[12] Z. M. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2432–2455, 2017.

[13] M. Mirza, J. Sommers, P. Barford, and X. Zhu, "A machine learning approach to tcp throughput prediction," *ACM SIGMETRICS Performance Evaluation Review*, vol. 35, no. 1, pp. 97–108, 2007.

[14] M. B. Tariq, K. Bhandankar, V. Valancius, A. Zeitoun, N. Feamster, and M. Ammar, "Answering "what-if" deployment and configuration questions with wise: Techniques and deployment experience," *IEEE/ACM Transactions on Networking*, vol. 21, no. 1, pp. 1–13, 2013.

[15] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *Proc. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, vol. 2. IEEE, 2005, pp. 729–734.

[16] S. Xiao, D. He, and Z. Gong, "Deep-q: Traffic-driven qos inference using deep generative network," in *Proc. 2018 Workshop on Network Meets AI & ML*, 2018, pp. 67–73.

[17] M. Ferriol-Galmés, J. Suárez-Varela, J. Paillissé, X. Shi, S. Xiao, X. Cheng, P. Barlet-Ros, and A. Cabellos-Aparicio, "Building a digital twin for network optimization using graph neural networks," *Computer Networks*, vol. 217, p. 109329, 2022.