Towards Intelligent Resource Allocation in Highly-distributed Content Delivery Networks Using Graph Neural Networks

Juan Vanerio*†, Pedro Casas†, Stefan Schmid*‡
*University of Vienna, †AIT Austrian Institute of Technology, ‡TU Berlin

***Abstract*—We introduce GNN4Alloc, a learning-based framework for resource allocation in highly distributed Content Delivery Networks (CDNs). Focusing on the core challenges of content placement and routing, GNN4Alloc leverages Graph Neural Networks (GNNs) to enhance decision-making efficiency in dynamic and large-scale environments. Building on prior work that employs mathematical optimization and heuristic algorithms, we reformulate these problems using graph representation learning, leveraging the bipartite nature of content-to-node assignment and routing decisions in CDN resource allocation. The framework incorporates GNN-based modules – including neural algorithm executors and constrained optimization layers – to develop adaptive allocation policies that generalize across diverse network topologies and demand profiles. By doing so, GNN4Alloc aims to improve both the scalability and solution quality of content allocation strategies, contributing to the broader goal of advancing GNN-based control in distributed systems.**

I. Introduction

Driven by the ever-increasing traffic demand for content and low latencies, storage resources are being deployed closer to the end users. From information-centric networks to 5G wireless systems, caching popular content close to the network edge can alleviate performance bottlenecks and enhance the end-user experience [1]. This shift toward edge storage has intensified the need for efficient resource management in large-scale, distributed caching systems such as CDNs. As network traffic continues its exponential growth, CDN providers face mounting challenges in scaling storage and routing decisions to meet demand while maintaining performance and cost-efficiency [2]. While traditional caching mechanisms provide worst-case guarantees (e.g., Least-Recently-Used LRU), better techniques are needed to optimize performance on observed traffic [1]. Furthermore, existing approaches often struggle with scalability and adaptation to dynamic conditions [3].

Modern distributed caching systems face two core challenges: determining which content to place at which geographically distributed storage nodes, and routing user requests efficiently to minimize latency and bandwidth costs. Prior efforts such as Baiji [4] and Tero [3] have tackled these issues by formulating optimization objectives – under billing constraints or edge offloading scenarios – and proposing heuristic or algorithmic solutions. While these approaches have yielded valuable insights in terms of optimality conditions and algorithmic approaches, they often fall short in scalability and adaptability. Given the complexity and dynamic nature of the problem, machine learning methods, particularly Graph Neural Networks (GNNs), offer a compelling alternative by leveraging graph-structured representations to model complex allocation patterns [5], [6].

978-3-903176-74-4

The resource allocation problem in CDNs often exhibits a natural bipartite structure – linking content items with caching nodes (e.g., PoPs or edge devices) – making it particularly amenable to graph-based learning approaches. GNNs have recently shown strong potential in tackling complex network optimization tasks, and are increasingly applied to problems involving constrained optimization [7], [8], combinatorial search [7], and direct resource allocation [6], [9]. Particularly relevant is the neural execution paradigm [10], which enables GNNs to approximate algorithmic behavior, opening the door to scalable, adaptive solutions that can overcome the limitations of conventional heuristic and optimization-based methods.

The GNN4Alloc framework tackles the challenge of CDN resource allocation through a unified GNN-based approach that addresses both content placement and request routing. It leverages specialized GNN architectures tailored to bipartite graph structures inherent in the allocation problem, enabling efficient approximation of complex optimization objectives. By learning from data, GNN4Alloc develops adaptive strategies that generalize effectively across diverse network topologies and traffic demand patterns, offering scalable and robust performance in dynamic environments.

II. Problem Formulation

We model general CDN resource allocation problems using a heterogeneous bipartite graph representation that captures the elements relevant to both the domain allocation planning problem – e.g., Baiji, Figure 1 – and the traffic offloading problem – e.g., Tero, Figure 2. Baiji formulates the domain-to-PoP allocation problem as an offline optimization task with a nonlinear objective function based on the 95th percentile billing model used by service providers [4]. The problem's bipartite graph's nodes represent domain-area pairs and PoPs, while the edges indicate allowable allocations. The linear constraints involve bandwidth usage, admissible delays, and demand conservation. Due to the computational complexity of this allocation problem, Baiji employs multiple heuristics and a genetic algorithm to generate high-quality solutions [4].

Tero addresses the challenge of offloading CDN traffic onto massively distributed edge devices with limited storage and bandwidth capacities [3]. By leveraging content popularity forecasting, the system dynamically adapts to shifting demand

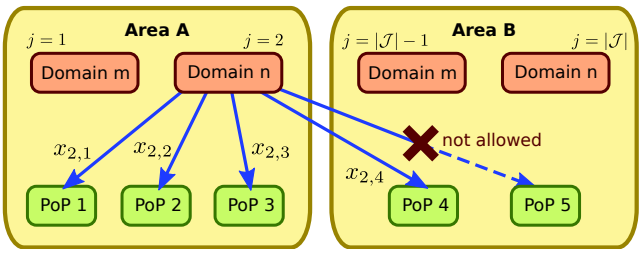

Fig. 1. Baiji domain demand pattern to PoP placement problem.

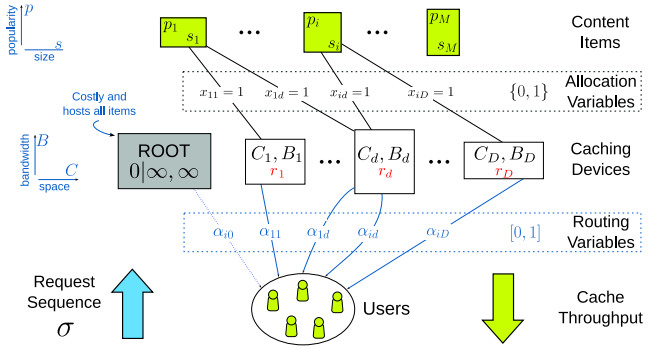

Fig. 2. Tero CDN traffic offloading to massively distributed edge devices.

patterns. The underlying optimization problem is formulated as a periodic, content allocation-driven throughput maximization task, subject to bandwidth and storage constraints. Structurally, the problem maps naturally onto a bipartite graph and lends itself to a Mixed Integer Linear Programming (MILP) formulation.

Formally, we define a graph $G = (V, E)$ that includes a set of nodes $V = V_c \cup V_d$ representing content or domains (V_c), and Points of Presence (PoPs) or distributed edge devices (V_d). The set of edges E captures allowable connections between these node types: $(c, d) \in E \implies c \in V_c \wedge d \in V_d$. The core of this representation is the bipartite structure between content/domain nodes V_c and caching nodes V_d. Node and edge attributes encode critical information such as system constraints (e.g., bandwidth, storage limits) and dynamic traffic patterns (i.e., demands for content).

GNN4Alloc tackles the optimization of content placement, determining which content should reside on which nodes, and of request routing, i.e., deciding how user requests are served [3], [4]. The framework supports diverse optimization objectives, such as minimizing billing costs across Points of Presence (PoPs), maximizing throughput at distributed edge nodes, enforcing delivery time through local quota constraints, and balancing cache hit rates throughout the network. While the specific objective function varies depending on the scenario, all formulations encode constraints directly into the problem graph, allowing the formulation of the framework to be applicable across tasks.

III. The GNN4Alloc Framework

The GNN4Alloc framework addresses resource allocation in distributed CDNs through a learning-based approach that integrates graph structure, algorithmic reasoning, and constraint handling. Its three core components include *Graph Encoding* – specialized message-passing operators that exploit the inherent bipartite structure between content/domains and caching nodes; *Neural Execution Module* – learns to mimic the execution of optimization algorithms (e.g., shortest path or bipartite matching) directly on the graph structure; *Constrained Optimization Layer* – ensures that resource allocation decisions respect real-world constraints, enabling learning and inference under bandwidth, storage, and demand limits.

A. Graph Encoding

As we said before. node and edge attributes encode critical information such as system constraints (e.g., bandwidth, storage limits) and dynamic traffic patterns (i.e., demands for content). GNNs utilize message-passing to learn rich representations of the CDN topology, content characteristics, and demand patterns. Message passing aggregates local information, allowing nodes to learn representations integrating their context, such as content properties and connectivity. The message passing can be formally represented as:

$$h_i^{(l+1)} = \phi^{(l)} \left(h_i^{(l)}, \bigoplus_{j \in N(i)} \psi^{(l)}(h_i^{(l)}, h_j^{(l)}, e_{ij}) \right) \quad (1)$$

where $h_i^{(l)}$ are node features, ϕ and ψ are learnable functions, $\bigoplus$ is a permutation-invariant aggregation, and e_{ij} are edge features. A central aspect is the design of specialized message-passing operators that exploit the inherent bipartite structure between content/domains and caching nodes.

B. Neural Execution Module

Inspired by Neural Execution of Graph Algorithms [10], this component learns to mimic the execution of optimization algorithms directly on the graph structure. Neural execution combines classical algorithms with the adaptability of learning, offering advantages for constrained optimization in CDNs by efficiently capturing algorithmic behavior within the GNN.

The algorithmic patterns are learned by training on solutions produced by exact solvers or heuristics applied to smaller problem instances, enabling the model to efficiently approximate complex allocation decisions at scale. The objective is not only to replicate the optimization strategies used in systems like Baiji and Tero, but also to enhance them through data-driven generalization. Figure 3 illustrates the architecture and operational flow of the neural execution module. The central concept that enables effective learning of algorithms with GNNs is *algorithmic alignment* – the degree to which a GNN's architecture and message-passing operations reflect the computational logic of the target algorithm. For example, in shortest path problems, the aggregation function ψ in Eq. 1 should implement a $\min$ operation, directly mirroring the

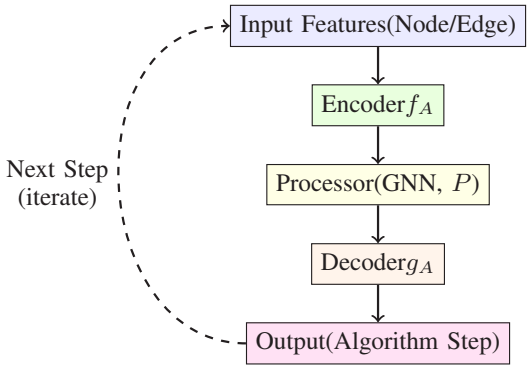

Fig. 3. Neural execution architecture. Input graph features are pipelined through encoder (f_A), graph neural network processor (P), and decoder (g_A) to produce one algorithm step output. The dashed arrow shows the iterative feedback loop enabling multi-step algorithm execution. [10]

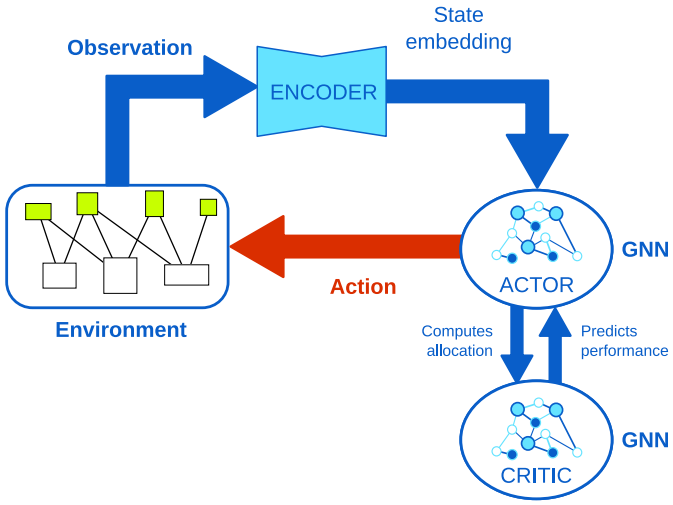

Fig. 4. Reinforcement Learning enhanced optimization. In this setup, the GNN is used to implement the Actor and Critic networks of an AAC architecture tailored for combinatorial optimization [12]

objective of the algorithm. When this structural alignment is achieved, the GNN can learn more efficiently, generalize better, and require less training data to reach high performance.

GNNs provide computational advantages over traditional CDN optimization methods. Message-passing operations scale linearly with network size, unlike the cubic complexity of MILP [7]. Furthermore, the GPU-friendly nature of these operations may enable parallelization-based gains. Finally, other GNN architectures [5], [7], [11] designed to solve specific combinatorial problems may also be used to further enhance the solution's space exploration capabilities.

C. Constrained Optimization Layer

Effectively managing real-world CDN constraints – such as bandwidth limits, storage capacity, and demand conservation – is essential for practical deployment. GNN4Alloc incorporates *differentiable optimization components* to address these constraints within the learning process. This integration enables the model to make allocation decisions that are not only efficient but also constraint-aware. To achieve this, constraints are modeled as penalty terms (soft-constraints) and embedded into a carefully designed surrogate loss function. This approach allows the model to balance objective performance with constraints during training, realizing feasible and adaptive solutions even in complex, large-scale environments:

$$\mathcal{L} = \underbrace{\mathcal{L}_{objective}}_{\mathcal{L}_{Content\_utility}} + \sum_{c} \lambda_c \cdot \underbrace{\max(0, v_c(X))}_{\text{bandwidth / storage / other}} \tag{2}$$

where $\mathcal{C}$ is the set of constraints, λ_c are learned constraint weights (potentially dynamic), $v_c(X)$ measures the violation of constraint c under allocation X, and X is the output of the GNN model representing allocation decisions. With this technique, constraints become learnable components in the optimization.

D. Solution Approaches

GNN4Alloc learns to imitate the behavior of algorithm execution through demonstration, multi-task learning, and algorithmic alignment [10]. Training data is drawn from solutions generated by the target algorithm on small instances. The problem is proposed as a node or edge classification problem.

For the 95th percentile billing optimization [4], we explore how GNNs can learn the concept of "affinity" between traffic patterns, a key insight that drives effective allocation under this billing model. The goal is to identify complementary traffic patterns that utilize the same 5% "free" time slots.

Alternatively, to handle the dynamic aspects of traffic offloading from Tero, we may also integrate reinforcement learning techniques with GNN4Alloc. The problem is then formulated as a Markov Decision Process, where the states represent the current demands and content distribution, the actions correspond to routing and caching decisions, and the reward signal moves in opposition to the loss function of Eq. 2.

The actor network is trained to optimize performance metrics through interactions with a simulated environment. Internally, a GNN reflects the perceived system state. Then, a trainable Critic network estimates the value of the computed actions. Both networks operate in tandem to identify the best possible action for each state. Figure 4 illustrates the concept.

E. Next Steps

GNN4Alloc is currently an ongoing implementation, with the initial goal of benchmarking its performance against established systems such as Baiji and Tero. The framework aims to deliver several key contributions: leveraging GNNs to enable faster decision-making compared to traditional optimization methods, ensuring scalability through parallel message passing, and integrating neural execution with differentiable constraint handling to balance performance objectives with system limits. Its design targets the challenges of highly distributed and heterogeneous CDN topologies, including edge and mobile nodes, by learning generalizable allocation policies that adapt across diverse network structures.

ACKNOWLEDGMENT

This work has been supported by the FWF Austrian Science Fund, Project reference I-6653, as part of the EU CHIST-ERA-2022-SPiDDS-02 project *GRAPHS4SEC*.

REFERENCES

[1] G. S. Paschos, G. Iosi̇̈fidis, M. Tao, D. Towsley, and G. Caire, "The role of caching in future communication systems and networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 6, pp. 1111–1125, 2018.

[2] R. Buyya, M. Pathan, and A. Vakali, *Content delivery networks*. Springer Science & Business Media, 2008, vol. 9.

[3] J. Vanerio, L. Hügerich, and S. Schmid, "Tero: Offloading cdn traffic to massively distributed devices," in *Proceedings of the 25th International Conference on Distributed Computing and Networking*, ser. ICDCN '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 186–198.

[4] J. Vanerio, H. Liu, Q. Zhang, and S. Schmid, "Baiji: Domain planning for cdns under the 95th percentile billing model," in *2024 IFIP Networking Conference (IFIP Networking)*, 2024, pp. 1–9.

[5] S. T. Thomdapu, P. Katiyar, and K. Rajawat, "Dynamic cache management in content delivery networks," *Computer Networks*, vol. 187, p. 107822, 2021.

[6] M. Randall, P. Belzarena, F. Larroca, and P. Casas, "Grows: improving decentralized resource allocation in wireless networks through graph neural networks," in *Proceedings of the 1st International Workshop on Graph Neural Networking*, December 2022, pp. 24–29.

[7] Q. Cappart, D. Chételat, E. B. Khalil, A. Lodi, C. Morris, and P. Veličković, "Combinatorial optimization and reasoning with graph neural networks," *Journal of Machine Learning Research*, vol. 24, no. 130, pp. 1–61, 2023.

[8] P. Almasan, J. Suárez-Varela, K. Rusek, P. Barlet-Ros, and A. Cabellos-Aparicio, "Deep reinforcement learning meets graph neural networks: exploring a routing optimization use case," *Computer Communications*, vol. 196, pp. 184–194, 2022.

[9] P. Cheng, G. Chen, and Z. Han, "Graph neural networks based resource allocation in heterogeneous wireless networks," in *Proceedings of the 7th International Conference on Intelligent Information Processing*, ser. ICIIP '22. New York, NY, USA: Association for Computing Machinery, 2023.

[10] P. Veličković, R. Ying, M. Padovano, R. Hadsell, and C. Blundell, "Neural execution of graph algorithms," *arXiv preprint arXiv:1910.10593*, 2019.

[11] C.-H. Pan, Y. Qu, Y. Yao, and M.-J.-S. Wang, "Hybridgnn: A self-supervised graph neural network for efficient maximum matching in bipartite graphs," *Symmetry*, vol. 16, no. 12, 2024.

[12] E. M. Sánchez, J. B. Clempner, and A. S. Poznyak, "A priori-knowledge/actor-critic reinforcement learning architecture for computing the mean–variance customer portfolio: The case of bank marketing campaigns," *Engineering Applications of Artificial Intelligence*, vol. 46, pp. 82–92, 2015.