Advances in Designing Scalable Graph Neural Networks: The Perspective of Graph Data Management

Ningyi Liao
Siqiang Luo
College of Computing and Data Science
Nanyang Technological University
Singapore
liao0090@e.ntu.edu.sg
siqiang.luo@ntu.edu.sg

Xiaokui Xiao
School of Computing
National University of Singapore
Singapore
xkxiao@nus.edu.sg

Reynold Cheng
School of Computing and Data Science
The University of Hong Kong
Hong Kong, China
ckcheng@cs.hku.hk

Abstract

Graph Neural Network (GNN) is a successful marriage of graph data management and deep learning, leading to notable improvements in learning quality over graphs. This advancement highly impacts graph-based applications in many areas, including computer vision, natural language processing, biology, medication, and social science. Despite the success, scaling up GNN models poses a formidable and long-lasting challenge, hindering the application to industrial-level graphs featuring millions or billions of nodes and edges. The rapid update of tasks and models requires continuous efforts in developing scalable GNN architectures. In specific, the scalability bottleneck of GNNs typically stem from graph-related computations, entailing more proficient processing and utilization of the unstructured graph data.

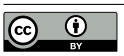
There has been a marked trend of incorporation between GNN and data management to tackle newly-emerged scalability challenges. This includes the utilization of graph algorithms such as Personalized PageRank (PPR) and subgraph discovery in GNN models, as well as exploring topics in graph domain including multi-scale representation and graph spectrum. This primer tutorial (3 hours) aims to provide a comprehensive overview of scalable GNN designs, highlighting the most recent and prominent models that focus on the scalability issue. We will also summarize the technical challenges and suggest potential future directions regarding the rapid developments in this field. We believe that this work can be used as one important reference for researchers looking to develop scalable GNN models.

CCS Concepts

• **Mathematics of computing** → **Graph algorithms**; • **Computing methodologies** → **Neural networks**; *Learning latent representations.*

Keywords

Graph Neural Networks; Graph Data Management; Scalability



SIGMOD-Companion '25, June 22–27, 2025, Berlin, Germany

ACM ISBN 979-8-4007-1564-8/2025/06
https://doi.org/10.1145/3722212.3725634

ACM Reference Format:
Ningyi Liao, Siqiang Luo, Xiaokui Xiao, and Reynold Cheng. 2025. Advances in Designing Scalable Graph Neural Networks: The Perspective of Graph Data Management. In *Companion of the 2025 International Conference on Management of Data (SIGMOD-Companion '25), June 22–27, 2025, Berlin, Germany*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3722212.3725634

1 Introduction

Recent years have witnessed the burgeoning of graph-based applications, such as viral marketing in social graphs, recommendation in e-commerce networks, drug discovery based on molecule networks, route planning in traffic road networks, as well as aiding large language model (LLM) deployment. In the trend of learning over complex graph data, graph neural networks emerge as a kind of neural network specialized in graph processing, and have demonstrated excellent learning performance in fundamental graph understanding tasks such as node classification and link prediction.

The canonical GNN design features the integration of graph processing and neural network learning, empowering the model with the capability to learn not only from the entity information of isolated nodes, but also from their relationships presented by the graph topology. This unique architecture brings promising model performance, but also results in critical *data scalability issues* with the graph-related operations identified as the bottleneck. The classical GNN model is known to be computationally expensive due to the consideration of the connection between data samples, where representing a single node relies on the representation of its neighbors. Such node dependency leads to graph-scale model computation, and iterating the network with multiple layers intensifies the scalability problem, which is recognized as neighborhood explosion. Recent development also raises new challenges for designing and applying GNNs to broader missions, demanding more comprehensive model ability while maintaining scalability. The emerging graph data and complicated tasks often leads to more sophisticated model designs, which, in turn, are more susceptible to scalability issues in graph computations.

However, there is an increasing popularity of performing learning tasks over large graphs with at least millions of nodes, such as Papers100M and Microsoft Academic Graph. Due to the huge data scale and relatively limited computation resources, especially GPU memory, training GNNs can easily lead to out-of-memory issues or impractical training times [39, 40, 57]. Moreover, looking ahead, there is a foreseeable increase in the size of data generated by graph-based online services like WeChat, Amazon, and Facebook.

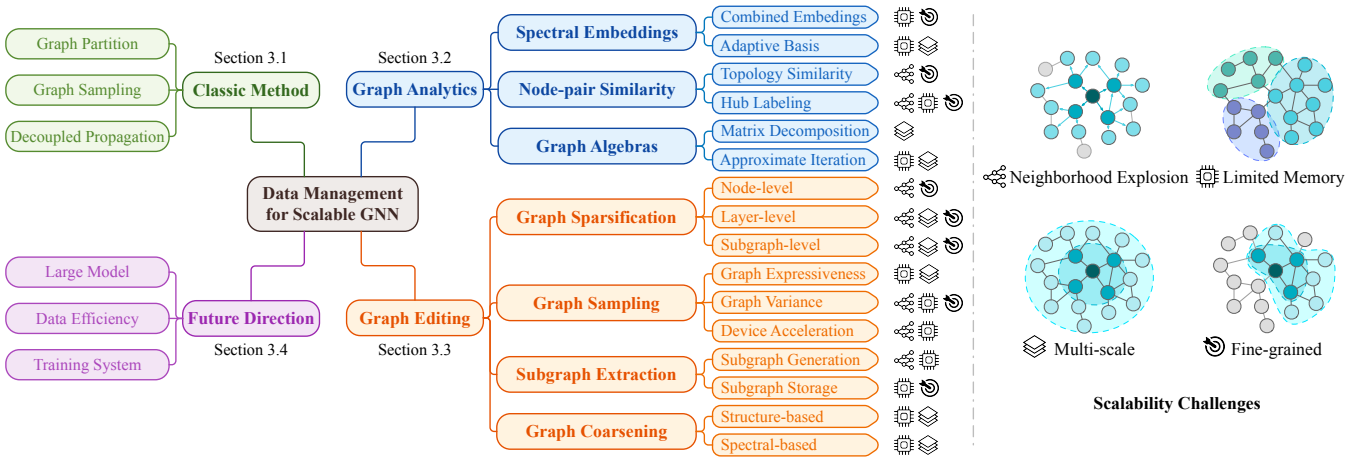


Figure 1: Overview of this tutorial. For each category of scalable GNN designs and models, we list corresponding scalability challenges they attempt to address.

To this end, integrating data management techniques into GNNs is considered a promising solution for addressing the scalability issue. Graph algorithms have been extensively studied by the data management community, advancing runtime performance, efficiency, and scalability. Pioneered by the classic APPNP [18], which substitutes the iterative graph convolution in GNN with approximate PPR computation to alleviate computational overhead, studies have been nourished by introducing a spectrum of ideas from the graph data management domain to remarkably address the graph processing bottleneck in GNN computation. For example, filtering algorithms from graph spectral theory extend GNN's capability in processing complex graph signals and inspire a family of model designs known as spectral GNNs. Canonical pairwise similarity metrics are also proved to be useful for querying important underlying patterns to aid model learning. Looking ahead, the emerging challenges and various application scenarios of GNNs present a broad range of opportunities for applying different graph data management techniques, as we attempt to convey in this tutorial.

Related recent tutorials. Due to the rapid growth in the realm of GNNs, a few tutorials have covered the fundamental concepts and specific topics of graph learning with GNNs. [51] introduces the basic concepts and common approaches in applying large-scale GNNs from the perspective of neural network design. [9] and [4] covers the dynamic and spectral variants of GNNs, respectively. [41] focuses on the system-level optimizations used for GNN training.

From the graph perspective, [8, 54] review subgraph-related tasks with respective focuses on large graphs and learning algorithms, while [35, 37] are for the graph query problem. [5, 17, 19, 50] cover the broader topic regarding the integration between data management and machine learning, i.e. DB4AI.

This tutorial particularly highlights the graph data management aspect in designing GNNs, presenting the latest advances of challenges and solutions for modern scalable GNNs. We identify and categorize different schemas of integrating the formulated graph analytics and editing techniques into the pipeline of GNN learning.

2 Target Audience and Length

Tutorial overview. In this tutorial, we aim to establish an intriguing view that the GNN scalability issue is highly related to techniques in *graph data management*. We provide a comprehensive review regarding recent utilization of graph data management techniques in scalable GNNs. We especially divide them into graph analytics and editing techniques with respective focuses on retrieving and editing the graph data, and categorize the relevant studies accordingly.

We plan the **3-hour tutorial** to be conducted based on the following sessions:

- **Section 3.1 Introduction of Scalable GNNs (40 mins)**
  - 3.1.1 Concepts and Applications of Scalable GNNs (5 mins)
  - 3.1.2 Classic Scalable GNNs (25 mins)
  - 3.1.3 Evaluation and Challenges of Scalable GNNs (5 mins)
  - 3.1.4 Opportunities for Graph Data Management (5 mins)
- **Section 3.2 Graph Analytics and Querying (45 mins)**
  - 3.2.1 Spectral Embeddings (15 mins)
  - 3.2.2 Node-pair Similarity (15 mins)
  - 3.2.3 Graph Algebras (15 mins)
- **Section 3.3 Graph Editing (60 mins)**
  - 3.3.1 Graph Sparsification (15 mins)
  - 3.3.2 Graph Sampling (15 mins)
  - 3.3.3 Subgraph Extraction (15 mins)
  - 3.3.4 Graph Coarsening (15 mins)
- **Section 3.4 Future Directions (20 mins)**
  - 3.4.1 Scalable Large Models with Graphs (10 mins)
  - 3.4.2 Learning Data Efficiency and Elasticity (5 mins)
  - 3.4.3 Scalable Training Schemes and Systems (5 mins)

Target audience. The tutorial is designed for researchers and practitioners who are with basic graph knowledge, and are interested in graph data management and GNN techniques. The topic of this tutorial intersects with the recent trend of DB for AI. By attending the tutorial, the audience is expected to learn about the prevailing methodologies in scalable GNNs, and the intersection of leveraging graph data management techniques to enhance GNN performance.

3 Tutorial Outline

3.1 Introduction of Scalable GNNs

3.1.1 Concepts and Applications of Scalable GNNs. In this part, we will briefly introduce the general concepts and typical tasks of GNNs. A canonical graph neural network operates in a message-passing manner, iteratively performing propagation based on graph topology and feature updates based on learnable weights. The introduction of graph data, however, becomes the scalability bottleneck due to the irregular data structure, which is relatively inefficient within the computation on specialized devices such as GPU. The *scalable GNN* is regarded as a family of model designs highlighting capability in processing large graphs, typically by reducing time and memory overhead during training and inference. Compared to GNNs focusing on efficacy, these designs prioritize enhancements in efficiency and scalability, usually orienting the management of graph data.

GNNs demonstrate unique capability in processing complicated graph patterns with typical graph understanding tasks such as node classification, link prediction, and graph regression. The enhanced scalability enables their application in real-world scale of data such as knowledge graph retrieval, social network analysis, e-commerce recommendation, and road network forecast.

3.1.2 Classic Scalable GNNs. In this part, we will elaborate on the design goals and representative works regarding classic approaches to improving GNN scalability in more detail. A general goal of classic scalable GNNs is to reduce or shift the computational overhead of graph operations so that the critical GPU memory bottleneck can be addressed by performing mini-batch training. By examining the techniques used in handling graph data, we introduce the following approaches:

- **Graph Partition:** Due to constraint of GPU memory when loading large-scale graph data, a common model-agnostic solution is employing graph partition algorithms to divide the graph into smaller subgraphs. The approach is especially suitable for distributed learning, where subgraphs are allocated to multiple devices for training. Both classic partition techniques based on graph topology and those tailored for GNNs are utilized, and algorithmic goals include optimizing computational and communication overhead.
- **Graph Sampling:** Data sampling is one of the classical approaches for addressing the scalability issue in machine learning. Graph sampling implies randomly selecting specific graph nodes and edges according to certain metrics, and forming them as batches to learn during training iterations. Based on the scope of sample selection, strategies can be categorized into node-, layer-, and subgraph-level [32]. While the sampling strategy reduces computational overhead through learning, the iterative process ensures statistically similar learning outcomes.
- **Decoupled Graph Propagation:** As graph propagation and feature transformation entail different computational requirements, the idea of decoupled GNN emerges to separate them apart. The implication of decoupling strategy is that, messages generated through graph propagation can be disentangled from layer-by-layer updates and instead learned in an aggregated fashion. By this means, graph operations can be conducted with dedicated algorithmic and device optimizations, which addresses the scalability bottleneck while retaining model capability.

3.1.3 Evaluation and Challenges in Scalable GNNs. From the perspective of data management, there are different specific goals in the process of designing and applying GNNs to large-scale graphs, ranging from directly tackling the time and memory efficiency to pursuing better graph processing outcomes. Hence, beside comparing prediction accuracy, recent studies extend the evaluation of GNNs in different aspects. For instance, [7, 34, 36] offer comprehensive observation regarding large-scale GNN learning with a collection of acceleration techniques. [23] performs empirical assessment on the efficiency and fine-grained performance of decoupled models.

Corresponding to the design goals and empirical observations, the scalability issue of GNNs can be elaborated in different perspectives. We specifically identify the following challenges under the topic of scalable GNNs:

- **Neighborhood Explosion:** The intensive scale of neighborhoods in multi-layer model learning is a persistent issue hindering GNN time complexity and empirical performance. Various graph processing techniques have been proposed on the topic of how to conduct the graph-scale computation efficiently without losing graph information.
- **Limited Memory:** The realistic graphs also poses practical challenges in storing and maintaining the large amount of data, especially in the highly-constrained GPU memory. Therefore, an array of scalable GNNs explores efficient management of the data by reducing or shifting the memory overhead.
- **Multi-scale:** The graph property of heterophily is revealed to be dominant in tasks such as anomaly detection. In this case, nodes are connected with dissimilar neighbors, while conventional GNNs face difficulties due to their concentration on graph locality. Multi-scale GNNs mitigate the issue by supplementing non-local graph information. However, it comes into conflict with common scalable designs that tend to diminish global dependency. How to provide multi-scale ability to scalable GNN models is thus a challenging topic.
- **Fine-grained:** While scalable GNNs are designed to retrieve information from a large amount of data, it is uncovered that their prediction accuracy decreases on certain graph nodes. Fine-grained operations are useful for elevating attention on specific nodes or personalizing graph propagations. As the manipulation easily incurs additional time and memory overhead, it also entails novel scalable solutions to adapt these particular managements.

3.1.4 Opportunities for Graph Data Management. The classic scalable GNN approaches present various opportunities for integrating graph data management techniques into the GNN pipeline. For example, different graph partitioning algorithms can be employed to minimize and balance computation and communication; graph centrality metrics can be utilized to measure the importance of components for sampling; embedding algorithms are favored by the decoupled design for sufficiently representing the topology information. Based on the operations on graph data, we generally categorize promising graph data management techniques used for scalable GNNs into the following two types:

- **Graph Analytics and Querying:** These approaches does not directly change the graph data, but rather exploit the graph topology for augmenting GNN learning with enhanced efficiency and favorable effectiveness. Typically, this implies analyzing or querying graph information on graph, subgraph, or node level, and then using the results for model learning.
- **Graph Editing:** This family of techniques choose to modify the graph structure during the GNN learning pipeline for reducing the data scale, ranging from the traditional graph-based modifications such as sparsification and sampling, to GNN-tailored methods of coarsening and condensation.

Target. In this section, audience will be familiar with the basic concepts and pipelines of scalable GNNs, and will learn the recent advances regarding the scalability challenges and opportunities of applying graph data management techniques.

3.2 Graph Analytics and Querying

Since the canonical GNN design exhibits a resource-intensive architecture integrating graph and neural network computation, many studies seek to employ powerful graph processing techniques alternative to graph convolution to better retrieve and exploit graph data. Typically, by employing advanced graph data management algorithms, these techniques can shift the computational overhead from full-scale graph processing during iterative network training and enhance the efficiency of the overall GNN pipeline.

3.2.1 Spectral Embeddings. GNNs are known to be tightly relevant to spectral graph filtering, which offers a compact pipeline for applying graph transformations [4]. However, the complex graph patterns in different scenarios call for specific solutions to better adapt the spectral signals and generate embeddings. The following works mark the recent advances in embedding spectral information for scalable GNN learning.

- **Combined Embeddings:** LD^2 [24] investigates the scalability issue of spectral filtering under heterophily. It adopts multiple filters and decoupled architecture to capture multi-scale information of heterophilous graphs. By this means, whole-graph information is sufficiently embedded, while the model still enjoys simple mini-batch training. The multi-scale spectral embeddings can also be extended to edge-wise learning [45–47].
- **Adaptive Basis:** UniFilter [15] devises a universal filter spanning across different heterophilous graph patterns. The filter design is found to effectively alleviate the common flaws of over-smoothing and over-squashing in GNN convolutions without compromising efficiency. AdaptKry [13] further designs an adaptive filter with provable controllability for diverse heterophily levels. Despite its spectral expressiveness, it can be achieved by a polynomial expression with favorable complexity.

3.2.2 Node-pair Similarity. Graph metrics representing node-pair relationships are useful in discovering underlying relevance in the graph topology, especially long-distance ones. Notably, these metrics showcase a practical pipeline for querying node-level information on demand instead of the full-graph manner.

- **Topology Similarity:** SIMGA [28] utilizes top-k SimRank to recognize and mitigate graph heterophily through structural similarity. It demonstrates that the metric is suffice in discovering and aggregating global similarity with a decoupled precomputation of sublinear overhead. DHGR [3] measures node-pair correlation by the cosine similarity of both topology and attributes, then employs a rewiring process to augment multi-scale edges and enhance performance under heterophily. Its design is feasible to subgraph-based batch training and hence maintains scalability.
- **Hub Labeling:** CFGNN [16] employs the hub labeling approach to discover underlying hierarchy in the graph topology and performs distinctive convolutions for core nodes in the hierarchy. DHIL-GT [27] explores the utilization of hub labeling for heterophilous node-pair rewiring and fast shortest path distance (SPD) bias querying in graph Transformer learning.

3.2.3 Graph Algebras. Instead of explicitly using graph information for network learning, implicit GNNs [12] replace the conventional GNN message-passing scheme by an algebraic expression involving the graph matrix. They acquire node representations by solving the equilibrium, thus capturing full-graph information in a single layer and bypassing the limited receptive field of general graph convolution. On top of their multi-scale nature, the following works aim to address various scalability issues

- **Matrix Decomposition:** EIGNN [31] introduces an efficient implicit calculation by considering a decoupled architecture. Its forward inference for the fixed-point equation can thus be directly acquired without iterative solvers, which enjoys better convergence and efficiency. It also employs eigendecomposition to simplify the large matrix computation.
- **Approximate Iteration:** MGNNI [30] looks into the multi-scale robustness and alleviates the sensitivity loss between distant nodes. It adopts a multi-hop graph adjacency in the aggregation equation, and hence directly expands the receptive field without occurring significant additional overhead in solving the implicit equation.
- **Graph Simplification:** SEIGNN [29] focuses on the training scalability in applying implicit GNNs to large graphs. To deploy mini-batch training, it introduces a graph coarsening approach that divides the graph into subgraphs while maintaining inter-subgraph propagation through linked coarse nodes. Batches are generated from the graph with additional coarse nodes.

Target. In this section, audience will be familiar with promising graph analytics and querying techniques used in scalable GNN variants. They will learn about different approaches of integrating graph data management techniques for augmenting graph learning and addressing the scalability challenges.

3.3 Graph Editing

Another array of studies mitigates the graph-scale bottleneck by applying specific procedures to edit the graph structure and reduce data size. The scalability is enhanced since the computation graph during learning is smaller. In the meantime, common GNN architectures can still be employed to leverage their capability. Nonetheless, how to design graph processing schemes to prevent information loss and performance degradation is always a key challenge for these models. In this section, we cover the mainstream graph editing techniques including graph sparsification, sampling, and condensation.

*3.3.1 **Graph Sparsification.*** Graph sparsification reduces the scale of data by removing edges in the graph based on certain criteria while preserving important properties such as node identity. The technique is especially useful for considering fine-grained properties by eliminating unimportant edges. On the effectiveness side, this is helpful to eliminate unwanted connections being harmful to the model prediction. Regarding efficiency, it also decrease the amount of operation in graph propagation.

- **Node-level:** SCARA [26] looks into the feature-wise similarity of the decoupled computation, which can be transformed into PPR calculation through re-normalization. This enables fine-grained node propagation and feature-oriented parallel computation, as well as scalability supported by layer-agnostic sublinear complexity. Unifews [25] formulates the layer-dependent propagation as spectral sparsification with approximation bounds of both iterative and decoupled architectures. The edge pruning scheme provides personalized maneuver while prevents additional computation overhead.
- **Layer-level:** NIGCN [14] achieves node- and layer-dependent propagation by controlling individual weight parameter during summation. It employs efficient neighbor sampling technique to approximate the decoupled embedding with linear complexity. ATP [20] discovers that the propagation performance is related with the node degree. It designs an augmented propagation by distinguishing nodes of high and low degrees. To invoke multi-scale representation, an additional encoding scheme based on embedding computations is utilized to represent node identity, local, and global information.
- **Subgraph-level:** GAMLP [56] establishes the attention mechanism to allocate node-wise importance in multi-scale embeddings. The learnable propagation and MLP network transformation are decoupled and respectively trained as an attempt to alleviate overhead of intermediate results. NAI [10] examines applying personalized design to various decoupled architectures. The propagation optimization acts as an external gated model for truncating the node-wise feature propagation. It further reduces the overhead of incorporating multi-scale feature embeddings through knowledge distillation.

*3.3.2 **Graph Sampling.*** Sampling remains a prolific topic in scalable GNNs thanks to its simplicity in reducing graph size by varied graph-related techniques while assuring model capability. Built on preceding algorithms, the following works target drawbacks of existing methods and tackle new challenges including sampling expressiveness, procedural overhead, and theoretical guarantee.

- **Graph Expressiveness:** ADGNN [43] proposes a set of strategies to computation and communication cost in distributed scenarios by defining corresponding node importance. Theoretical derivations are given to bound the aggregation difference between sampled and full topology. PyGNN [11] alternatively considers subgraphs with specific frequency ranges and conducts distinctive learning in spectral domain. Signals are then merged to form a dedicated and multi-scale representation of the graph.
- **Graph Variance:** LABOR [2] considers the stability and optimality in complex scenarios such as multi-layer and nonlinear layer-level sampling. It takes the advantage of node-dependent neighbor sampling, which restrains variance while requiring less samples. HDSGNN [21] aims to minimize the variance caused by sampling to ensure training convergence and effectiveness. It interpolates graph sampling into an optimization process, where the cached sampling results are included to generate the incremental graph components. LMC [42] enhances the ability of subgraph-level sampling by extended gradient computation and error compensation.
- **Device Acceleration:** GIDS [1] specifically accelerates the CPU-GPU loading process for GNN sampling and aggregation. NeutronOrch [38] actively balance the workload of CPU-based sampling and GPU-based training, as an attempt to effectively leverages the computation and memory resources. DAHA [22] simultaneously utilizes CPU and GPU for sampling and training, exploiting a cost model for adaptive execution planning.

*3.3.3 **Subgraph Extraction.*** Extracting subgraphs as one-time sparsification or iterative sampling can be costly in scalable GNN pipelines. Moreover, representative subgraphs benefit fine-grained GNN expressiveness since they introduce stronger local topology. Hence, there are studies dedicated to optimizing the process of efficiently managing subgraphs for downstream GNN learning.

- **Subgraph Generation:** G3 [44] investigates the subgraph generating and transferring behavior in distributed settings. Its sampling phase in the pipeline is able to seamlessly extract and distribute the subgraphs without occupying excessive GPU memory. TIGER [48] progressively gathers required triples by similarity matching on heterogeneous knowledge graphs.
- **Subgraph Storage:** SUREL [53] and its following work [52] employs algorithm and system co-design as a holistic framework, including extracting subgraphs by sampling and storing subgraphs as sparse representation. GENTI [55] further designs data structure specialized for streaming graph data, alleviating the blockage in GPU training.

*3.3.4 **Graph Coarsening.*** Coarsening describes another simplification approach that reduces the graph size by contracting nodes into subsets. It hence results in a different graph with its own node and edge sets, while still sharing similarities with the original graph. The scale of the coarse graph is usually significantly smaller, so that the GNN model can learn on the coarse graph with reduced time and memory overhead.

- **Structure-based:** GDEM [33] explores the graph spectrum in generating and training condensed graphs. It transforms the eigenbasis matching objective into an iterative constrained optimization process and ensures GNNs learns the approximate spectrum from the synthetic graph. ConvMatch [6] further approximates the process of generating supernodes through bounded node-pair representations, which enhances scalability on larger original graphs.
- **Spectral-based:** GC-SNTK [49] examines the efficiency and scalability issue in the bi-level optimization of structural condensation, alternatively formulating it as a kernel ridge regression task. The computational complexity is reduced thanks to less iterations in training the model.

Target. In this section, audience will be familiar with advances of the graph editing techniques in scalable GNNs. They will learn about different schemes of efficiently managing graph data to reduce graph scales and enable scalable computation.

3.4 Future Directions

3.4.1 Scalable Large Models with Graphs. With the recent advances in graph learning, a compelling task is to apply scalable graph data management techniques to a broader range of models. **Large language models (LLMs)** have been integrated into the graph learning pipeline for their power in task understanding and generalization. However, it comes at the price of efficiency and scalability, as the expense of LLM intensifies exponentially with the graph scale. We show that there are a number of opportunities of graph data management towards more scalable large graph models. For example, LLM on graph tasks requires informative and efficient embedding of the graph data by GNN or other graph embedding models, which is also a two-phase decoupled process and can be enhanced by graph analytics methods. The **graph with retrieval-augmented generation (GraphRAG)** pipeline operates knowledge graphs to provides semantic information in LLM inference. However, its dependence on community detection and querying algorithms becomes the critical efficiency bottleneck for deploying the technique at scale and therefore calls for calls for more proficient enhancements.

Graph Transformer is also an emerging GNN architecture that learns graph topology as sequence, which differs from convolution-based models and has become the backbone model for many graph-relevant large models. Due to the architectural difference, these models exhibit dissimilar graph scalability issues and requires dedicated optimizations. Graph data management techniques for efficiently querying and representing sequential information are hence particularly suitable to be incorporated.

3.4.2 Learning Data Efficiency and Elasticity. While canonical GNNs assume simple and labeled graphs to perform learning, the realistic conditions are more complicated and put forward new requirements for designing proper scalable GNNs. One of the issues is **insufficient labels**, that it is common for available data to lack ground-truth labels in real-world graph learning applications, especially for large-scale graphs. Certain graph processing and model embedding strategies are uniquely useful in these challenging scenarios. For example, in self-supervised tasks, scalable graph computation for contrastive learning is promising to offer efficient graph information retrieval. It is also beneficial to explore other integrations of scalable GNNs, such as few-shot learning for semi-supervised settings.

Dynamic graphs, characterized by changes in topology, pose a challenge for many GNN designs due to the additional temporal dimension. While a number of practical techniques have been proposed for handling sequential data in both graph management and neural network regimes, it deserves further investigation of how these algorithms can integrate and accommodate the paradigm of scalable GNNs.

3.4.3 Scalable Training Schemes and Systems. For a wide range of emerging scalable GNNs, enhancements at the system level are largely under-explored. **Device-specific** optimization is useful for allocating proper workload to particular devices such as GPU and TPU according to the model design, which heavily impacts the realistic execution performance. **Distributed training** also features a large scope of learning systems harnessing data and model parallelism, where efforts are demanded for migrating scalable data processing and network designs into these environments.

Target. In this section, the audience will receive an overview of the promising directions and open questions regarding the topic of scalable graph learning and the application of graph data management techniques from model, data, and system levels. The tutorial will conclude with a summary of the approaches on the integration of graph data management techniques and scalable GNNs covered in this talk.

4 Presenters

- **Reynold Cheng** (Website: https://www.reynold.hku.hk/) is a Professor in the University of Hong Kong (HKU). He is also the Head of the Computer Science Division in the HKU School of Computing and Data Science. His research interests are in data science, big graph analytics and uncertain data management. He received his BEng in 1998, and MPhil in 2000 from HKU. He then obtained his MSc and PhD degrees from Department of Computer Science of Purdue University in 2003 and 2005.
- **Xiaokui Xiao** (Website: https://www.comp.nus.edu.sg/~xiaoxk/) is a professor at the School of Computing, National University of Singapore. His research focuses on data management, especially on data privacy and algorithms for large data. He was a recipient of the VLDB 2021 Best Research Paper Award and the 2022 ACM SIGMOD Research Highlight Award. He is an IEEE fellow, an ACM distinguished member, and a trustee of the VLDB Endowment.
- **Siqiang Luo** (Website: https://siqiangluo.com/) is a Nanyang Assistant Professor at the College of Computing and Data Science, Nanyang Technological University (NTU), Singapore. His research interest lies in graph data management, such as GNNs, PageRanks and community search, as well as key-value data management, such as NoSQL key-value stores and learned indexes. His research has been regularly published in top venues such as SIGMOD, PVLDB and ICDE.
- **Ningyi Liao** (Website: https://nyliao.github.io/) is currently a Ph.D candidate and part-time tutor at the College of Computing and Data Science, NTU, Singapore. His research interest lies in scalable graph neural networks, neural network optimization, and graph algorithms. He has more than three years of experience working on graph neural networks.

Acknowledgments

This research is supported by NTU-NAP startup grant (022029-00001) and MOE AcRF Tier-2 Grant (T2EP20122-0003). Ningyi Liao is supported by the Joint NTU-WeBank Research Centre on FinTech, Nanyang Technological University, Singapore. Reynold Cheng was supported by the Hong Kong Jockey Club Charities Trust (Project 260920140), the University of Hong Kong (Project 2409100399), the HKU Outstanding Research Student Supervisor Award 2022-23, and the HKU Faculty Exchange Award 2024 (Faculty of Engineering).

References

[1] Xin Ai, Qiange Wang, Chunyu Cao, Yanfeng Zhang, Chaoyi Chen, Hao Yuan, Yu Gu, and Ge Yu. 2024. NeutronOrch: Rethinking Sample-Based GNN Training under CPU-GPU Heterogeneous Environments. In *PVLDB*, Vol. 17.

[2] Muhammed Fatih Balin and Ümit Çatalyürek. 2023. Layer-Neighbor Sampling — Defusing Neighborhood Explosion in GNNs. In *NeurIPS*.

[3] Wendong Bi, Lun Du, Qiang Fu, Yanlin Wang, Shi Han, and Dongmei Zhang. 2024. Make Heterophilic Graphs Better Fit GNN: A Graph Rewiring Approach. *TKDE* 36, 12.

[4] Zhiqian Chen, Lei Zhang, and Liang Zhao. 2024. Unifying Spectral and Spatial Graph Neural Networks. In *CIKM*.

[5] Gao Cong, Jingyi Yang, and Yue Zhao. 2024. Machine Learning for Databases: Foundations, Paradigms, and Open problems. In *SIGMOD*.

[6] Charles Dickens, Edward Huang, Aishwarya Reganti, Jiong Zhu, Karthik Subbian, and Danai Koutra. 2024. Graph Coarsening via Convolution Matching for Scalable Graph Neural Network Training. In *WWW*.

[7] Keyu Duan, Zirui Liu, Peihao Wang, Wenqing Zheng, Kaixiong Zhou, Tianlong Chen, Xia Hu, and Zhangyang Wang. 2022. A comprehensive study on large-scale graph training: Benchmarking and rethinking. *NeurIPS*.

[8] Yixiang Fang, Wensheng Luo, and Chenhao Ma. 2022. Densest subgraph discovery on large graphs: applications, challenges, and techniques. In *PVLDB*, Vol. 15.

[9] Dongqi Fu, Zhe Xu, Hanghang Tong, and Jingrui He. 2023. Natural and Artificial Dynamics in GNNs: A Tutorial. In *WSDM*.

[10] Xinyi Gao, Wentao Zhang, Junliang Yu, Yingxia Shao, Quoc Viet Hung Nguyen, Bin Cui, and Hongzhi Yin. 2024. Accelerating Scalable Graph Neural Network Inference with Node-Adaptive Propagation. In *ICDE*.

[11] Haoyu Geng, Chao Chen, Yixuan He, Gang Zeng, Zhaobing Han, Hua Chai, and Junchi Yan. 2023. Pyramid Graph Neural Network: A Graph Sampling and Filtering Approach for Multi-scale Disentangled Representations. In *KDD*.

[12] Fangda Gu, Heng Chang, Wenwu Zhu, Somayeh Sojoudi, and Laurent El Ghaoui. 2020. Implicit Graph Neural Networks. In *NeurIPS*, Vol. 33.

[13] Keke Huang, Wencai Cao, Hoang Ta, Xiaokui Xiao, and Pietro Liò. 2024. Optimizing Polynomial Graph Filters: A Novel Adaptive Krylov Subspace Approach. In *WWW*.

[14] Keke Huang, Jing Tang, Juncheng Liu, Renchi Yang, and Xiaokui Xiao. 2023. Node-Wise Diffusion for Scalable Graph Learning. In *WWW*. ACM.

[15] Keke Huang, Yu Guang Wang, Ming Li, and and Pietro Liò. 2024. How Universal Polynomial Bases Enhance Spectral Graph Neural Networks: Heterophily, Over-smoothing, and Over-squashing. In *ICML*, Vol. 235. PMLR.

[16] Pak Lon Ip, Shenghui Zhang, Xuekai Wei, Tsz Nam Chan, and Leong Hou U. 2023. Bridging Indexing Structure and Graph Learning: Expressive and Scalable Graph Neural Network via Core-Fringe. *OpenReview* j56A1HUTQS.

[17] Alekh Jindal and Matteo Interlandi. 2021. Machine learning for cloud data systems: the progress so far and the path forward. In *PVLDB*, Vol. 14.

[18] Johannes Klicpera, Aleksandar Bojchevski, and Stephan Günnemann. 2019. Predict then propagate: Graph neural networks meet personalized PageRank. *ICLR*.

[19] Guoliang Li, Xuanhe Zhou, and Lei Cao. 2021. AI Meets Database: AI4DB and DB4AI. In *SIGMOD*.

[20] Xunkai Li, Jingyuan Ma, Zhengyu Wu, Daohan Su, Wentao Zhang, Rong-Hua Li, and Guoren Wang. 2024. Rethinking Node-Wise Propagation for Large-Scale Graph Learning. In *WWW*.

[21] Yang Li, Bingbing Xu, Fei Sun, Qi Cao, Yige Yuan, Huawei Shen, and Xueqi Cheng. 2024. History Driven Sampling for Scalable Graph Neural Networks. In *DASFAA*.

[22] Zhiyuan Li, Xun Jian, Yue Wang, Yingxia Shao, and Lei Chen. 2024. DAHA: Accelerating GNN Training with Data and Hardware Aware Execution Planning. In *PVLDB*, Vol. 17.

[23] Ningyi Liao, Haoyu Liu, Zulun Zhu, Siqiang Luo, and Laks V.S. Lakshmanan. 2024. Benchmarking Spectral Graph Neural Networks: A Comprehensive Study on Effectiveness and Efficiency. arXiv:2406.09675

[24] Ningyi Liao, Siqiang Luo, Xiang Li, and Jieming Shi. 2023. LD2: Scalable Heterophilous Graph Neural Network with Decoupled Embedding. In *NeurIPS*, Vol. 36.

[25] Ningyi Liao, Siqiang Luo, and Zihao Yu. 2024. Unifews: Unified Entry-Wise Sparsification for Efficient Graph Neural Network. arXiv:2403.13268

[26] Ningyi Liao, Dingheng Mo, Siqiang Luo, Xiang Li, and Pengcheng Yin. 2023. Scalable Decoupling Graph Neural Networks with Feature-Oriented Optimization. *PVLDB J.* 33, 17 pages.

[27] Ningyi Liao, Zihao Yu, and Siqiang Luo. 2024. DHIL-GT: Scalable Graph Transformer with Decoupled Hierarchy Labeling. arXiv:2412.04738

[28] Haoyu Liu, Ningyi Liao, and Siqiang Luo. 2023. SIMGA: A Simple and Effective Heterophilous Graph Neural Network with Efficient Global Aggregation. arXiv:2305.09958

[29] Juncheng Liu, Bryan Hooi, Kenji Kawaguchi, Yiwei Wang, Chaosheng Dong, and Xiaokui Xiao. 2024. Scalable and Effective Implicit Graph Neural Networks on Large Graphs. In *ICLR*.

[30] Juncheng Liu, Bryan Hooi, Kenji Kawaguchi, and Xiaokui Xiao. 2022. MGNNI: Multiscale Graph Neural Networks with Implicit Layers. In *NeurIPS*.

[31] Juncheng Liu, Kenji Kawaguchi, Bryan Hooi, Yiwei Wang, and Xiaokui Xiao. 2021. EIGNN: Efficient Infinite-Depth Graph Neural Networks. In *NeurIPS*, Vol. 34. Curran Associates, Inc.

[32] Xin Liu, Mingyu Yan, Lei Deng, Guoqi Li, Xiaochun Ye, and Dongrui Fan. 2022. Sampling Methods for Efficient Training of Graph Convolutional Networks: A Survey. *Journal of Automatica Sinica* 9, 2.

[33] Yang Liu, Deyu Bo, and Chuan Shi. 2024. Graph Distillation with Eigenbasis Matching. In *ICML*, Vol. 235. PMLR.

[34] Lu Ma, Zeang Sheng, Xunkai Li, Xinyi Gao, Zhezheng Hao, Ling Yang, Wentao Zhang, and Bin Cui. 2024. Acceleration Algorithms in GNNs: A Survey. arXiv:2405.04114

[35] Ioana Manolescu and Madhulika Mohanty. 2023. Full-Power Graph Querying: State of the Art and Challenges. In *PVLDB*, Vol. 16.

[36] Nikolai Merkel, Pierre Toussing, Ruben Mayer, and Hans-Arno Jacobsen. 2024. Can Graph Reordering Speed Up Graph Neural Network Training? An Experimental Study. In *PVLDB*, Vol. 18.

[37] Amine Mhedhbi and Semih Salihoğlu. 2022. Modern Techniques for Querying Graph-Structured Relations: Foundations, System Implementations, and Open Challenges. In *PVLDB*, Vol. 15.

[38] Jeongmin Brian Park, Vikram Sharma Mailthody, Zaid Qureshi, and Wen-mei Hwu. 2024. Accelerating Sampling and Aggregation Operations in GNN Frameworks with GPU Initiated Direct Storage Accesses. In *PVLDB*, Vol. 17.

[39] Yeonhong Park, Sunhong Min, and Jae W. Lee. 2022. Ginex: SSD-Enabled Billion-Scale Graph Neural Network Training on a Single Machine via Provably Optimal in-Memory Caching. In *PVLDB*, Vol. 15.

[40] Jingshu Peng, Zhao Chen, Yingxia Shao, Yanyan Shen, Lei Chen, and Jiannong Cao. 2022. SANCUS: Staleness-Aware Communication-Avoiding Full-Graph Decentralized Training in Large-Scale Graph Neural Networks. In *PVLDB*, Vol. 15.

[41] Yanyan Shen, Lei Chen, Jingzhi Fang, Xin Zhang, Shihong Gao, and Hongbo Yin. 2024. Efficient Training of Graph Neural Networks on Large Graphs. In *PVLDB*, Vol. 17.

[42] Zhihao Shi, Xize Liang, and Jie Wang. 2023. LMC: Fast Training of GNNs via Subgraph Sampling with Provable Convergence. In *ICLR*.

[43] Zhen Song, Yu Gu, Tianyi Li, Qing Sun, Yanfeng Zhang, Christian S. Jensen, and Ge Yu. 2023. ADGNN: Towards Scalable GNN Training with Aggregation-Difference Aware Sampling. *SIGMOD* 1, 4.

[44] Xinchen Wan, Kaiqiang Xu, Xudong Liao, Yilun Jin, Kai Chen, and Xin Jin. 2023. Scalable and Efficient Full-Graph GNN Training for Large Graphs. *SIGMOD* 1, 2.

[45] Hewen Wang, Renchi Yang, Keke Huang, and Xiaokui Xiao. 2023. Efficient and Effective Edge-wise Graph Representation Learning. In *KDD*.

[46] Kai Wang, Dan Lin, and Siqiang Luo. 2025. Graph Percolation Embeddings for Efficient Knowledge Graph Inductive Reasoning. *IEEE Transactions on Knowledge and Data Engineering* 37, 3.

[47] Kai Wang and Siqiang Luo. 2024. Towards Graph Foundation Models: The Perspective of Zero-shot Reasoning on Knowledge Graphs. arXiv:2410.12609

[48] Kai Wang, Yuwei Xu, and Siqiang Luo. 2024. TIGER: Training Inductive Graph Neural Network for Large-Scale Knowledge Graph Reasoning. In *PVLDB*, Vol. 17.

[49] Lin Wang, Wenqi Fan, Jiatong Li, Yao Ma, and Qing Li. 2024. Fast Graph Condensation with Structure-based Neural Tangent Kernel. In *WWW*.

[50] Abdul Wasay, Subarna Chatterjee, and Stratos Idreos. 2021. Deep Learning: Systems and Responsibility. In *SIGMOD*.

[51] Rui Xue, Haoyu Han, Tong Zhao, Neil Shah, Jiliang Tang, and Xiaorui Liu. 2023. Large-Scale Graph Neural Networks: The Past and New Frontiers. In *KDD*.

[52] Haoteng Yin, Muhan Zhang, Jianguo Wang, and Pan Li. 2023. SUREL+: Moving from Walks to Sets for Scalable Subgraph-Based Graph Representation Learning. In *PVLDB*, Vol. 16.

[53] Haoteng Yin, Muhan Zhang, Yanbang Wang, Jianguo Wang, and Pan Li. 2022. Algorithm and system co-design for efficient subgraph-based graph representation learning. In *PVLDB*, Vol. 15.

[54] Kai Siong Yow, Ningyi Liao, Siqiang Luo, and Reynold Cheng. 2023. Machine Learning for Subgraph Extraction: Methods, Applications and Challenges. In *PVLDB*, Vol. 16.

[55] Zihao Yu, Ningyi Liao, and Siqiang Luo. 2024. GENTI: GPU-Powered Walk-Based Subgraph Extraction for Scalable Representation Learning on Dynamic Graphs. In *PVLDB*, Vol. 17.

[56] Wentao Zhang, Ziqi Yin, Zeang Sheng, Yang Li, Wen Ouyang, Xiaosen Li, Yangyu Tao, Zhi Yang, and Bin Cui. 2022. Graph Attention Multi-Layer Perceptron. In *KDD*. ACM.

[57] Chenguang Zheng, Hongzhi Chen, Yuxuan Cheng, Zhezheng Song, Yifan Wu, Changji Li, James Cheng, Hao Yang, and Shuai Zhang. 2022. ByteGNN: Efficient Graph Neural Network Training at Large Scale. In *PVLDB*, Vol. 15.