Advances in Designing Scalable GNNs: The Perspective of Graph Data Management

Ningyi Liao Siqiang Luo Xiaokui Xiao Reynold Cheng

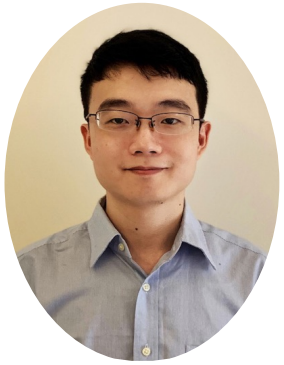
NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

NUS National University of Singapore

香 港 大 學 THE UNIVERSITY OF HONG KONG

2025 International Conference on Management of Data
Berlin, Germany – June 22, 2025

Tutorial Contributors / Presenters



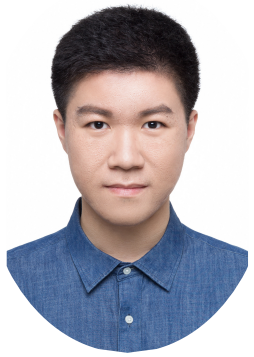
Siqiang Luo

Assistant Professor, NTU

siqiang.luo@ntu.edu.sg

https://siqiangluo.com

Graph algorithms/systems, KV systems, ML data management



Ningyi Liao

Ph.D Candidate, NTU

liao0090@e.ntu.edu.sg

https://nyliao.github.io

ML graph algorithms, Graph neural network

Reynold Cheng

Professor, HKU

ckcheng@cs.hku.hk

https://www.reynold.hku.hk

Data science, Big graph analytics, Uncertain data management

Xiaokui Xiao

Professor, NUS

xkxiao@nus.edu.sg

https://www.comp.nus.edu.sg/~xiaoxk/

Privacy-preserving data management, Large data analysis

9:00 – 10:30

Outline

25 min
9:00 – 9:25

Introduction: Application | Challenge | Opportunity
Siqiang Luo

25 min
9:25 – 9:50

Classical Approaches: Sampling | Partition | Decoupling
Siqiang Luo & Ningyi Liao

20 min
9:50 – 10:10

Hardware-aware Sampling
Ningyi Liao

20 min
10:10 – 10:30

Subgraph Extraction
Ningyi Liao

30 min
10:30 – 11:00

Q&A
Coffee Breaks

11:00 – 12:30

Outline

25 min
11:00 – 11:25
Node-wise Similarity
Siqiang Luo

20 min
11:25 – 11:45
Spectral Embedding
Ningyi Liao

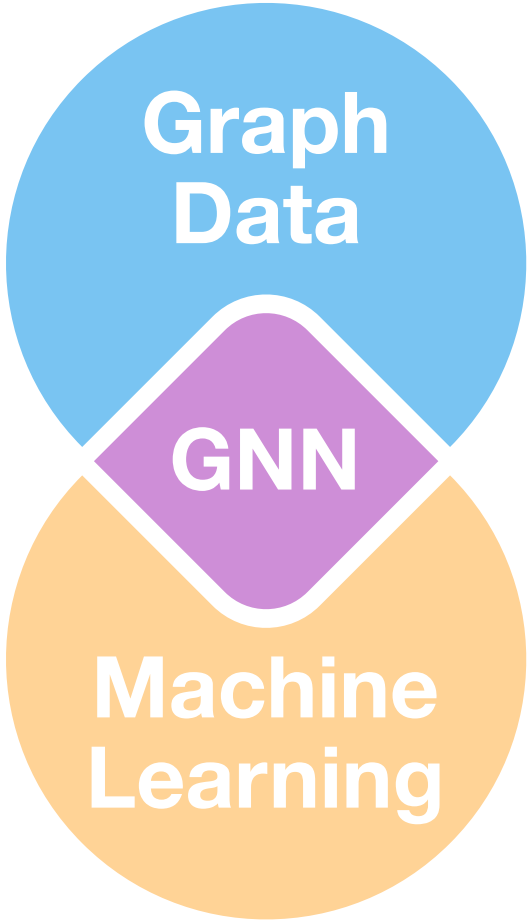
20 min
11:45 – 12:05
Graph Simplification
Ningyi Liao

25 min
12:05 – 12:30
Future Direction: Graph+LM | Specific Data | Graph System
Reynold Cheng

Q&A

12:30
Lunch

GNN: Current and Prospect

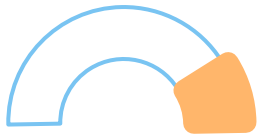


#1 WWW 2024

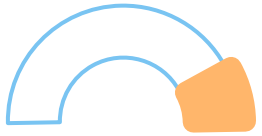
#4 NeurIPS 2025

#7 ICLR 2024

Ranking in keywords of accepted papers



7% VLDB 2024



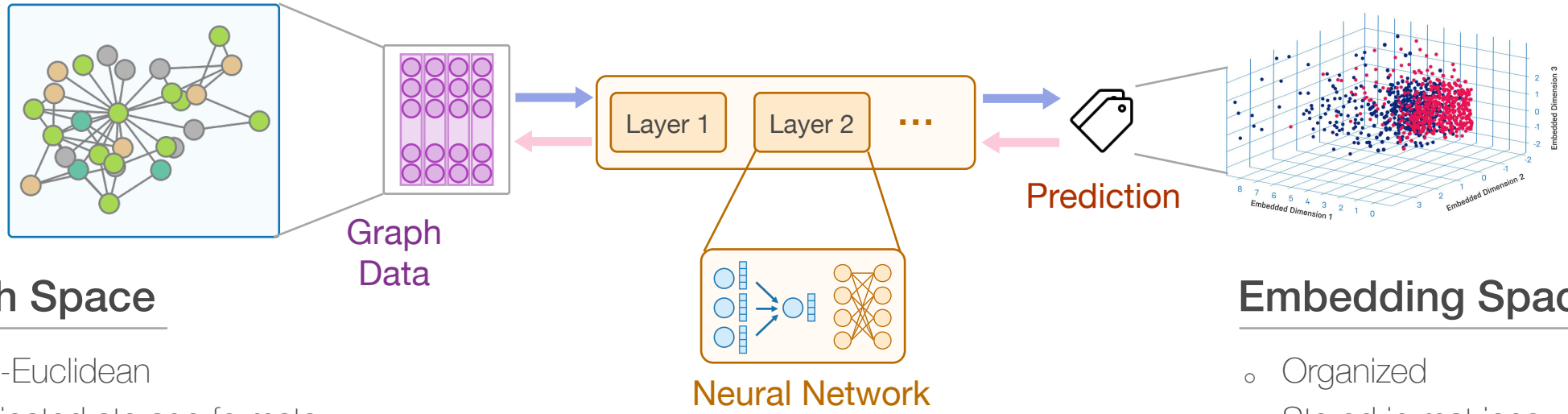
6% ICDE 2025

5% SIGMOD 2025

Percentage of topic in accepted papers

GNN: Learning on Graph Data

- Integrate graph computation and neural network architecture
- From semi-unstructured graph data to structured embeddings



Graph Space

- Non-Euclidean
- Dedicated storage formats

Embedding Space

- Organized
- Stored in matrices
- Suitable for GPU computation

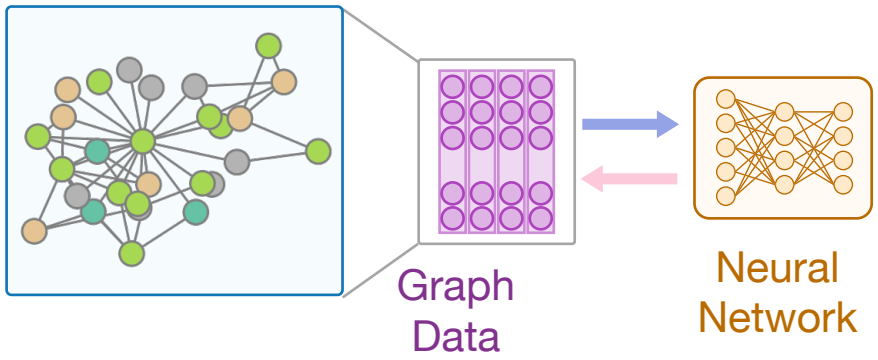
GNN: Learning on Graph Data

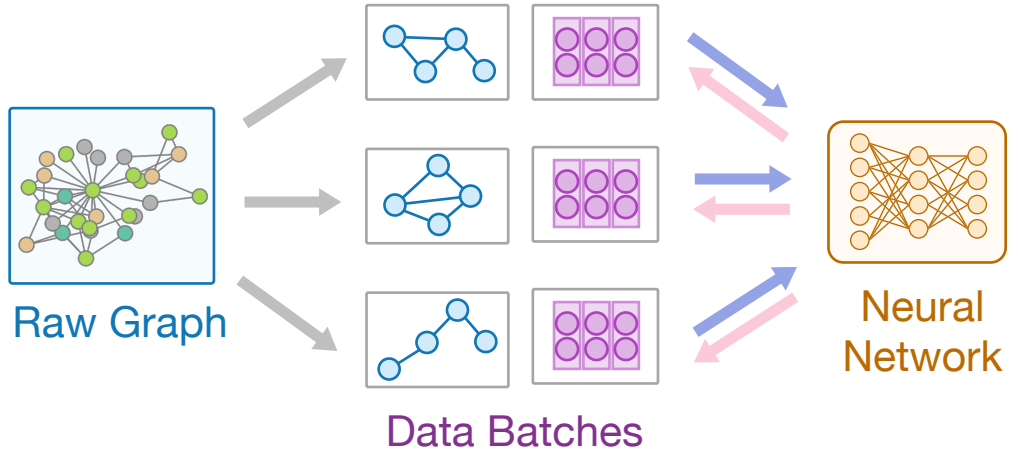
Full-batch (FB) Training

- NN learns graph data as a whole
- Preserve full graph information

Mini-batch (MB) Training

- NN learns part of graph each time
- Batch division affects performance

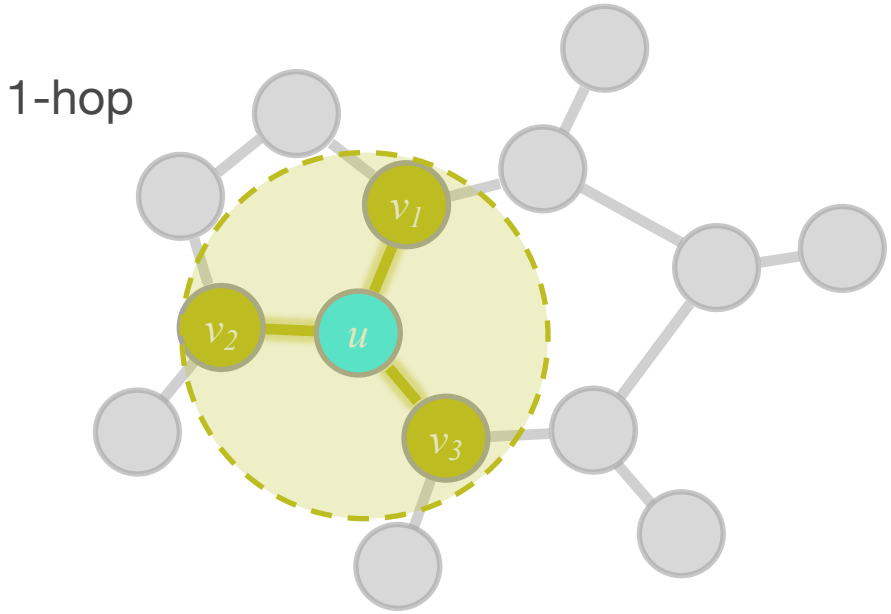


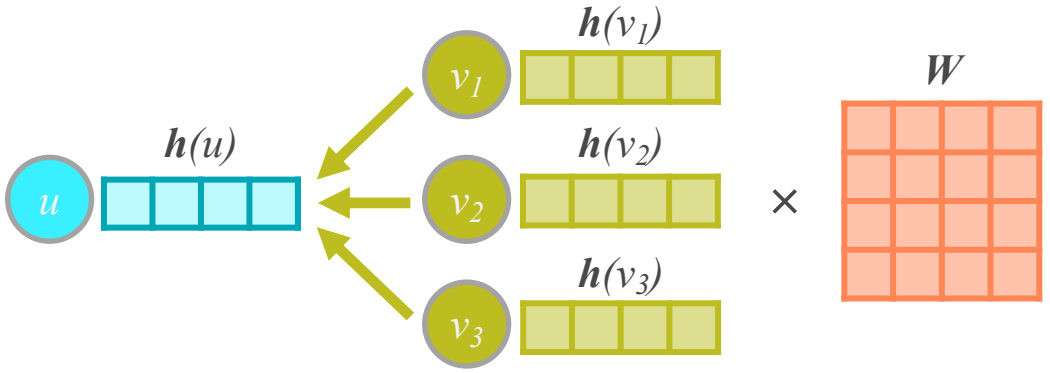


Forward Passing Backward Propagation

GNN: GCN [ICLR'17]

- Message Passing: aggregate neighbor information $N(u)$ by learnable weights $\boldsymbol{W}$ to each node as node representation $\boldsymbol{h}$



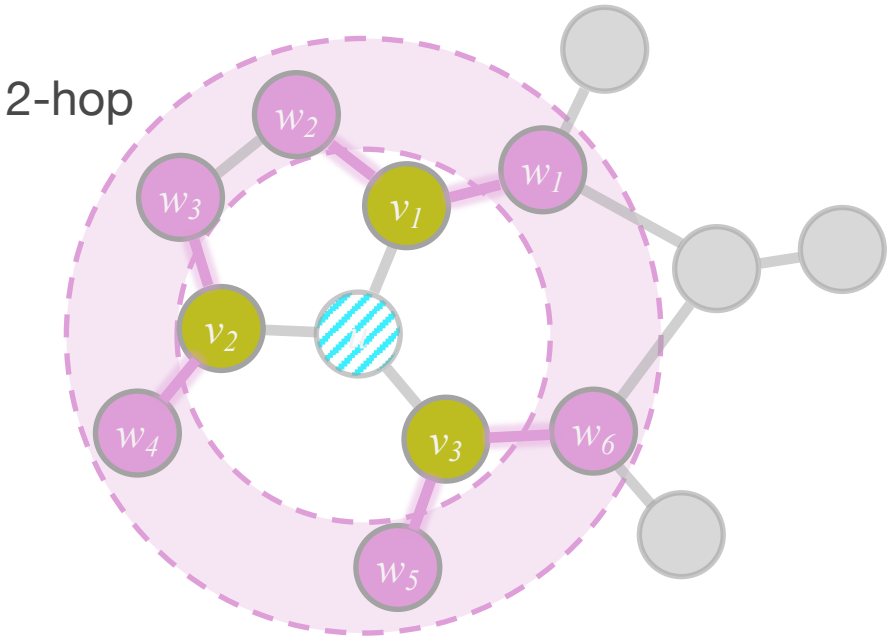


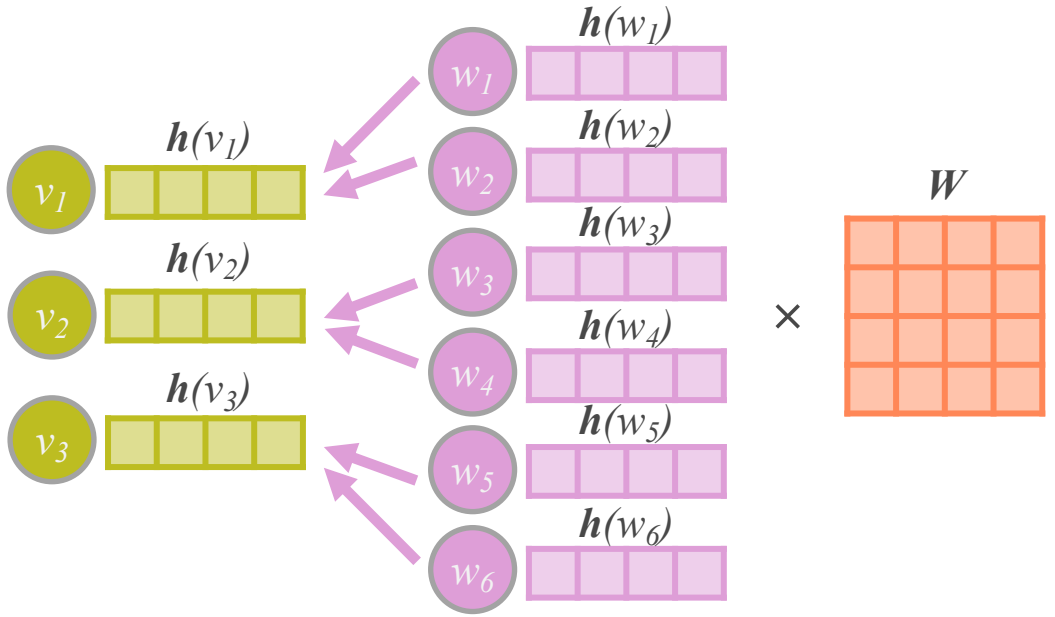
$$\boldsymbol{h}(u) = \sum_{v \in N(u)} \boldsymbol{h}(v) \cdot \boldsymbol{W}$$

T Kipf & M Welling. "Semi-supervised classification with graph convolutional networks". *ICLR'17*.

GNN: GCN [ICLR'17]

- GNN Layers: Stacking multi-hop graph convolutions





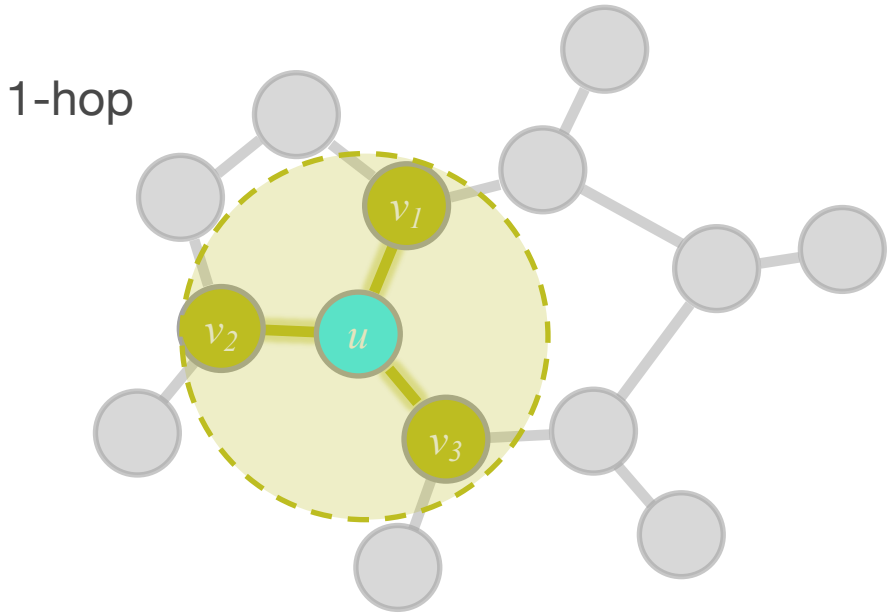
$$\boldsymbol{h}(v) = \sum_{w \in N(v)} \boldsymbol{h}(w) \cdot \boldsymbol{W}$$

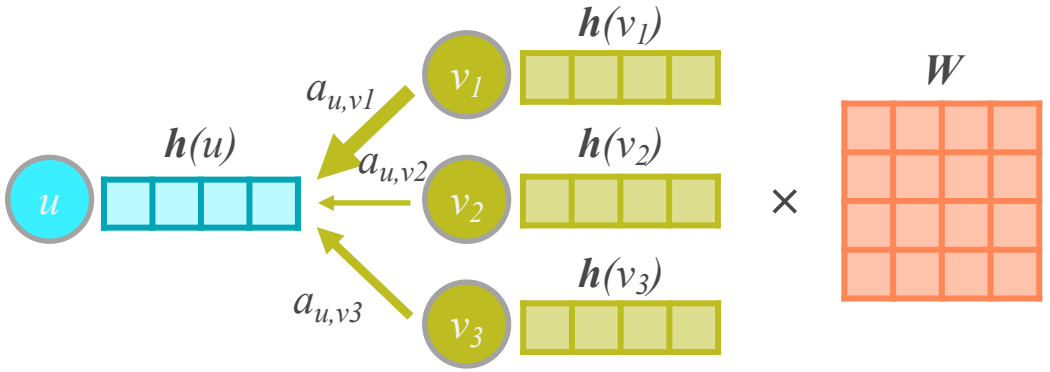
T Kipf & M Welling. "Semi-supervised classification with graph convolutional networks". *ICLR'17*.

GNN: GAT [ICLR'18]

- Graph Attention: learnable weights for edge aggregation

$$a_{u,v} = \frac{\exp(\text{attention}(\boldsymbol{h}(u), \boldsymbol{h}(v)))}{\sum_{v \in N(u)} \exp(\text{attention}(\boldsymbol{h}(u), \boldsymbol{h}(v)))}$$

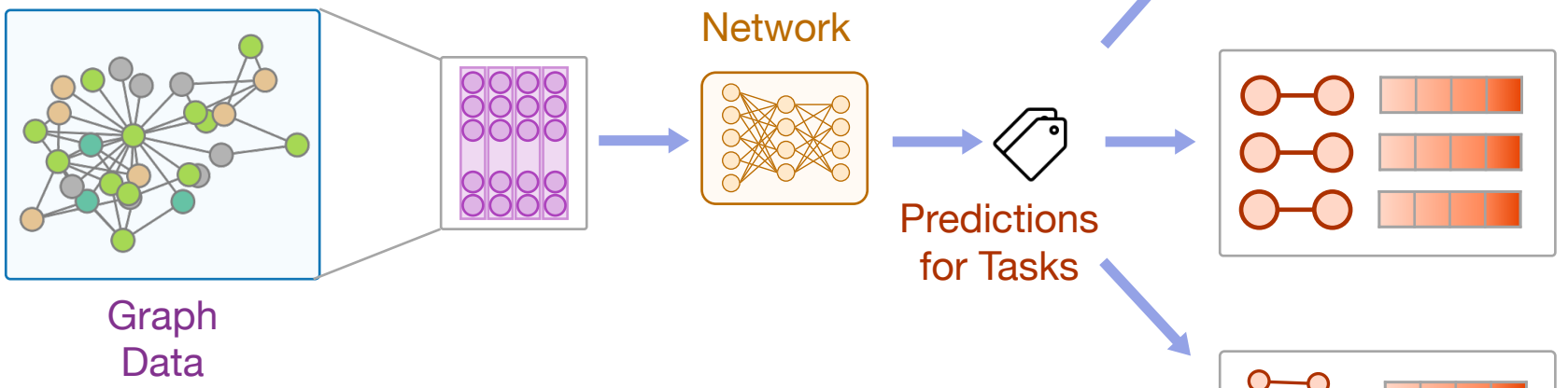




$$\boldsymbol{h}(u) = \sum_{v \in N(u)} a_{u,v} \cdot \boldsymbol{h}(v) \cdot \boldsymbol{W}$$

P Veličković et al. "Graph Attention Networks". *ICLR'18*.

GNN: Tasks and Applications



Node-level

individuals in the network

Edge-level

connections in the network

Graph-level

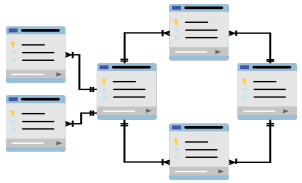
properties of the network

GNN: Tasks and Applications

Node-level

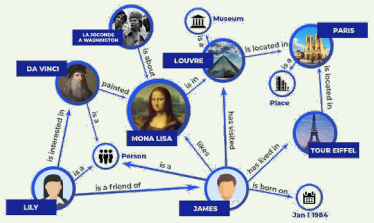
individuals in the network

User Profiling
E-commerce



Text-to-SQL
DB Schema

Friend Suggestion
Social Network



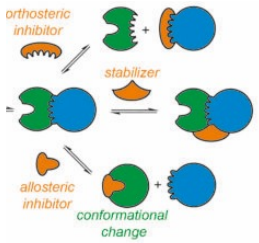
Knowledge Discovery
Knowledge Graph

Edge-level

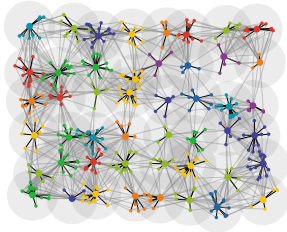
connections in the network

Graph-level

properties of the network



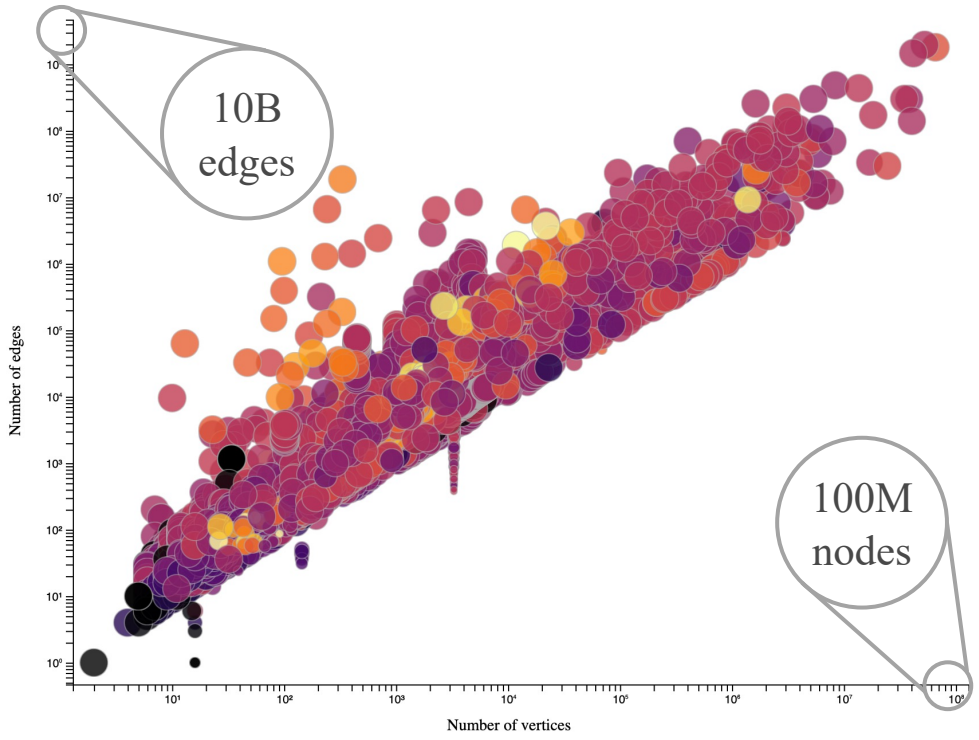
Drug Discovery
Protein Structure



Policy Optimization
Wireless Network

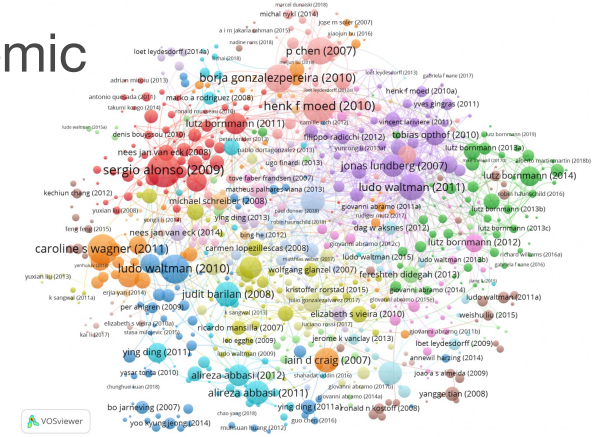
The Challenge: Graphs at Scale

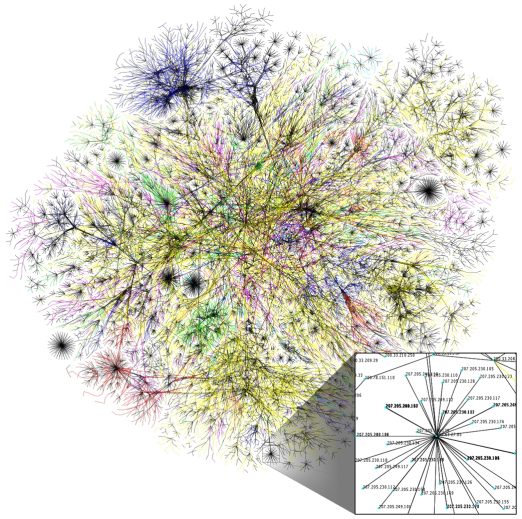
- Modern real-world graphs are on the scale of millions or billions



Scales of graph datasets

Microsoft Academic
240M nodes





Internet IP
4B nodes

T Peixoto, "The Netzschleuder network catalogue and repository", 2020. https://networks.skewed.de/

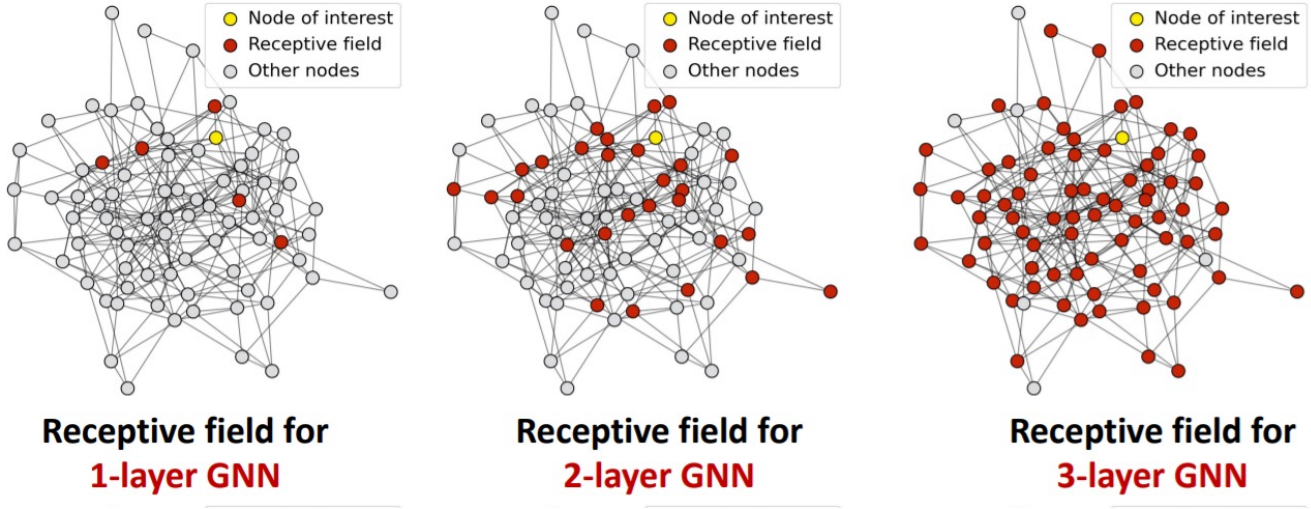
The Challenge: Graphs at Scale

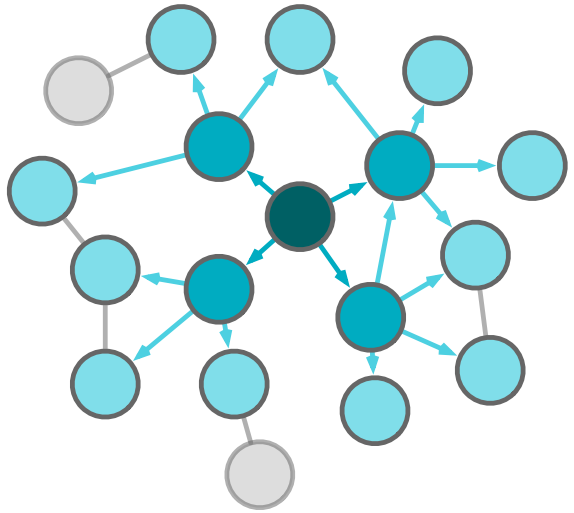
- Large scale graphs: diverse domains, diverse patterns

Name	Graph Size	Data Size	Domain
MAG	240M	200 GB	Heterogeneous academic graph
WikiKG	90M	160 GB	Wikidata knowledge graph
PCQM4Mv2	3M × 15	10 GB	Molecules in quantum chemistry interaction
Amazon	2M	8 GB	Product co-purchasing network
MD17	3M × 15	8 GB	Molecular dynamics trajectories
COCO-SP	120K × 500	12 GB	Segmentation of computer vision images
Snap-Patents	3M	20 GB	Patent citation network
Hyperlink Graph	4B	400 GB	Crawled webpage hyperlink

The Challenge: Neighbor Explosion

- Neighboring nodes intensively increases with network depth
- Exponential, related with edge size
- Lost of node identity



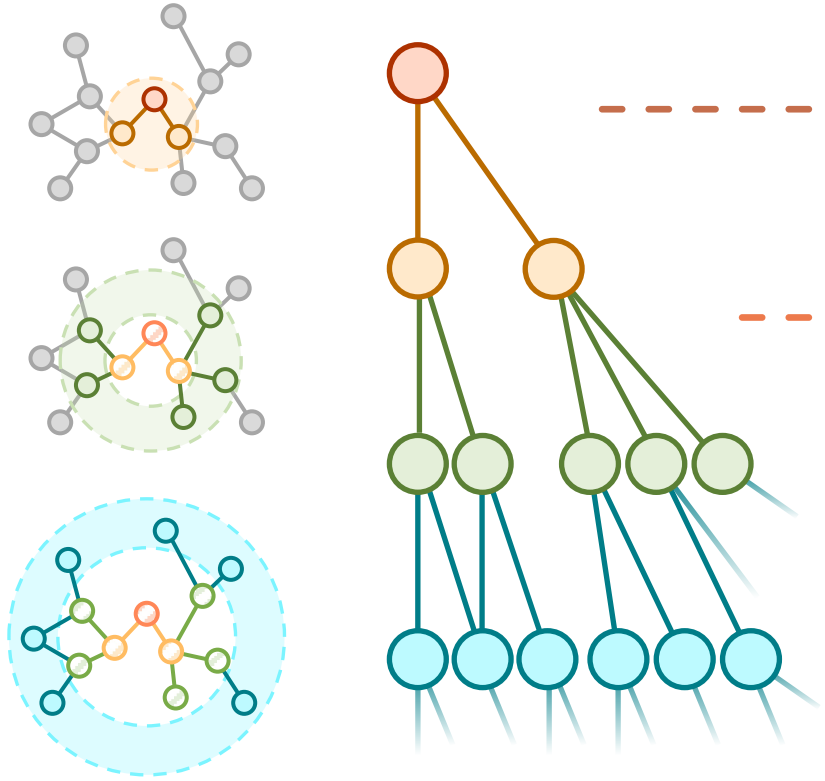


Time overhead

Expressiveness

The Challenge: Neighbor Explosion

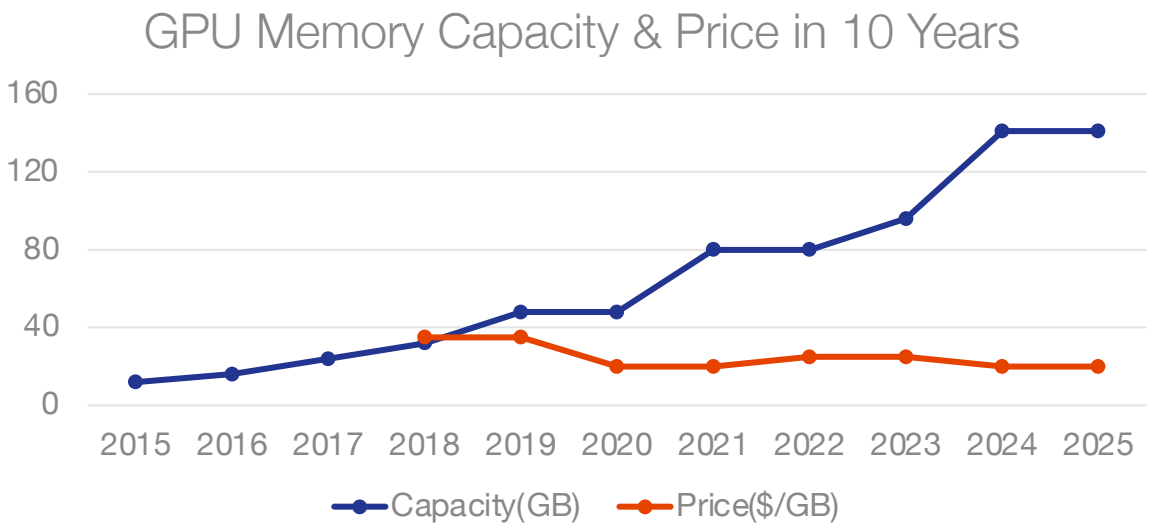
Case study: GCN computation graph

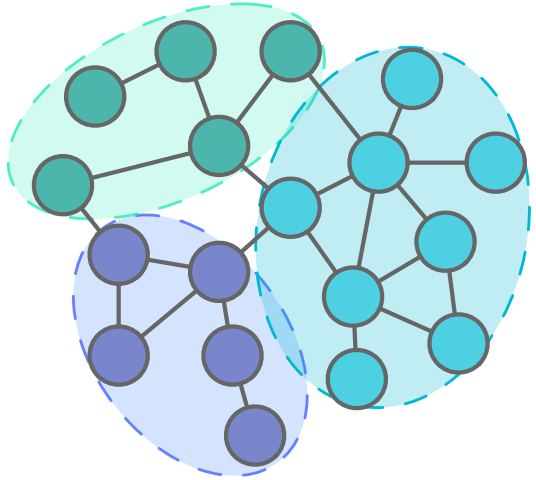


Network Layer	Neighborhood	Computation
1-layer	1-hop	$O(d)$
2-layer	2-hop	$O(d^2)$
3-layer	3-hop	$O(d^3)$
		...
L-layer	Upper bound:	$O(m)$

The Challenge: Limited Memory

- GPU memory is limited and costly
- Storing and processing graph data requires large memory space



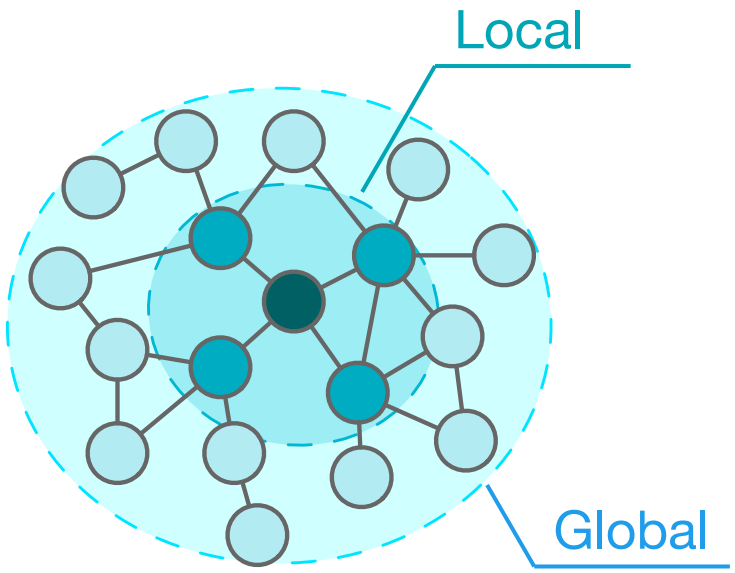


Memory overhead

Training scheme

The Challenge: Multi-scale

- Not only local nodes are important in prediction
- Need to capture global graph information efficiently
- Homophily: the inductive case, similar nodes cluster together
- Heterophily: the opposite case, connected nodes tend to be of dissimilar classes



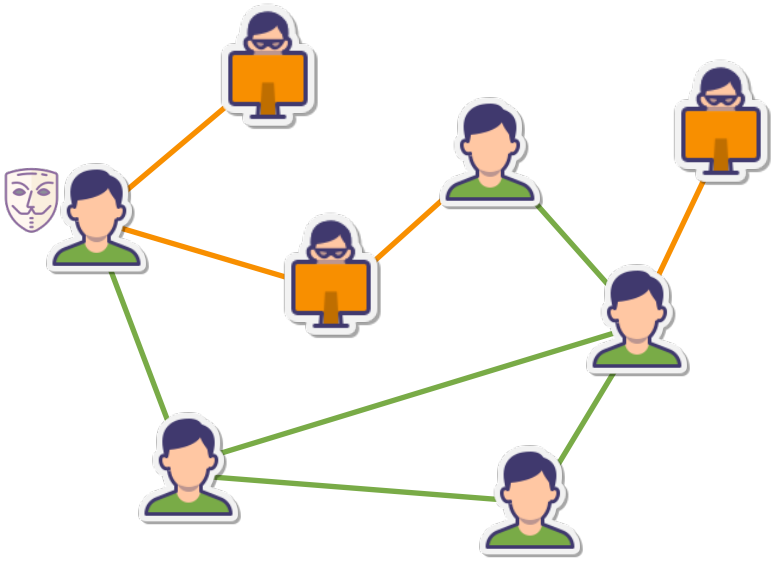
Time overhead

Expressiveness

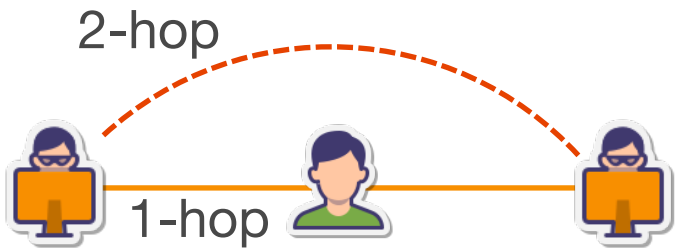
The Challenge: Multi-scale

Example: Heterophily in fraud detection

- **Graph**: financial transaction network with benign or fraud users
- **Task**: identify fraud users
- Fraudsters use benign accounts as spring-boards to reach other fraudsters

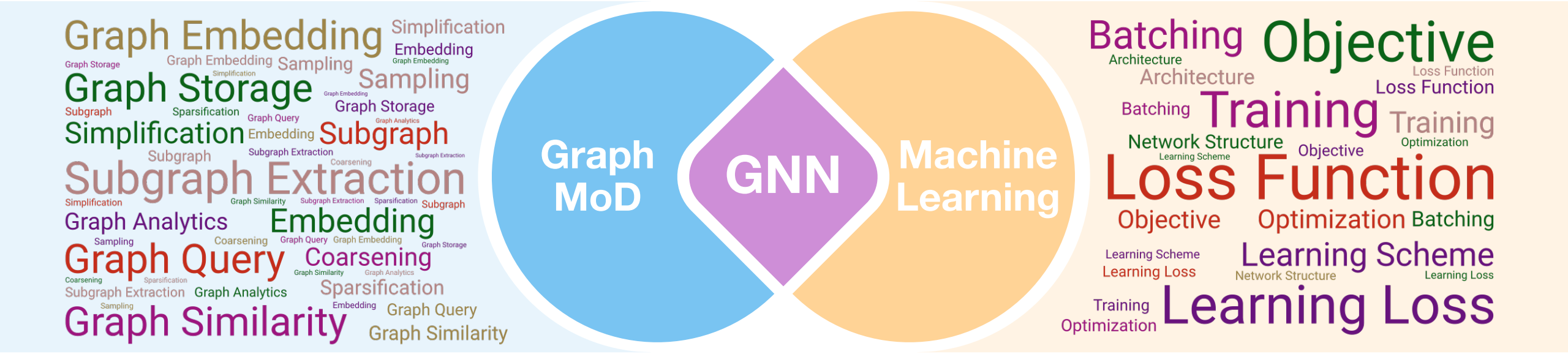


1-hop connection is not enough!



The Opportunity: Graph Data Management

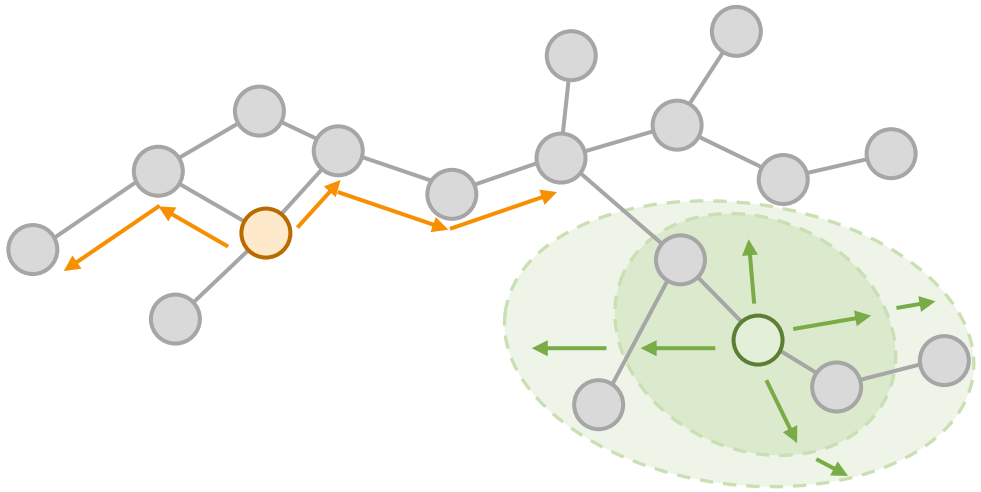
What can the DB community do for GNN challenges?



Preview: Neighbor Explosion

Solution from Node-wise Similarity: GBP [NeurIPS'20]

- Replace GNN computation by Personalized PageRank
- Reduce neighborhood by bidirectional search
- Decoupling CPU and GPU computation

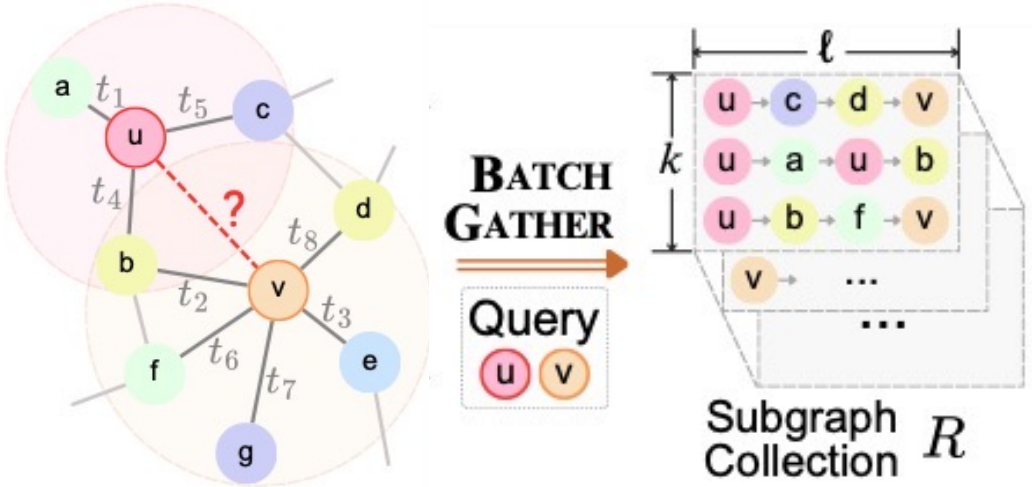


M Chen et al, "Scalable Graph Neural Networks Via Bidirectional Propagation", *NeurIPS'20*.

Preview: Limited Memory

Solution from Subgraph Extraction: GENTI [VLDB'24]

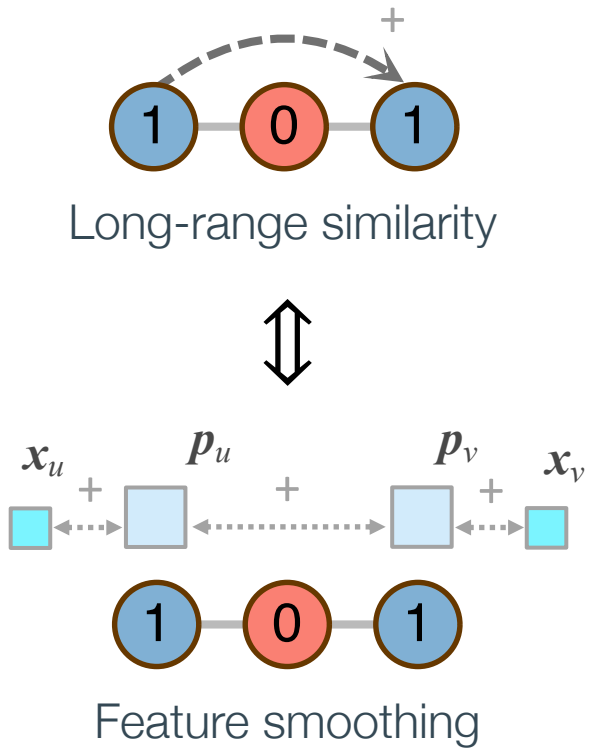
- Only load subgraph (part of the graph) for GNN computation
- Structured subgraph sampled by k-many random walks
- GPU-oriented data structure and batch computation



Preview: Multi-scale

Solution from Spectral Embedding: LD^2 [NeurIPS'23]

- Embed full-graph information in spectral domain
- Bridge spatial and spectral long-range graph operations
- Fast & scalable spectral embedding computation

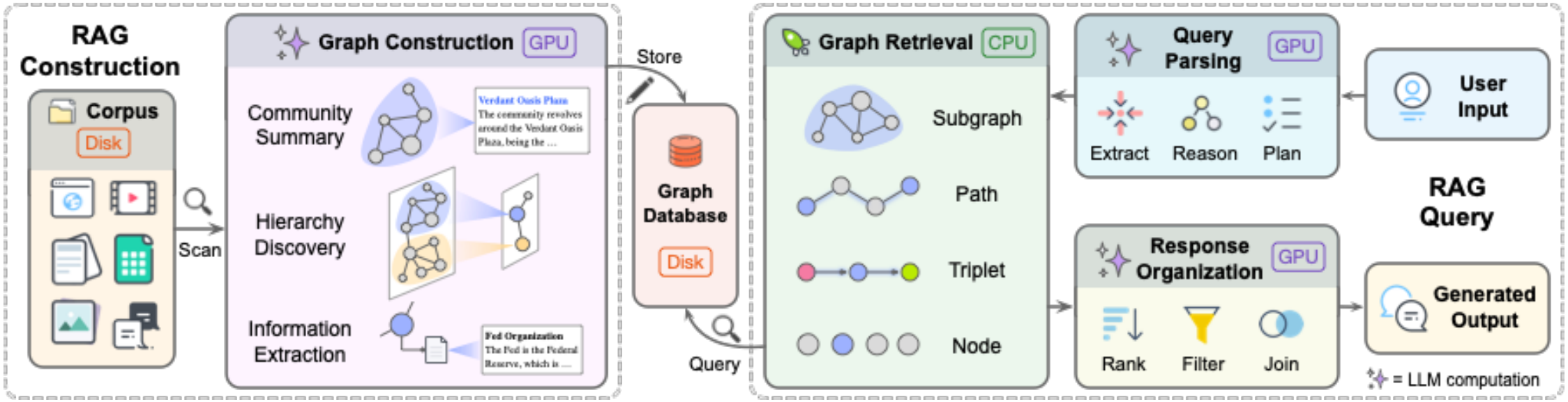


H Liu et al, "SIGMA: An Efficient Heterophilous Graph Neural Network with Fast Global Aggregation", ICDE'25.
N Liao et al, " LD2: Scalable Heterophilous Graph Neural Network with Decoupled Embedding", NeurIPS'23.

Preview: Scalable Graph+LM

Future Direction in Graph-based RAG

- Scalable graph embedding, indexing, and retrieval
- Integration between graph data & LLM



Unpublished figure

Classical Approaches for Scalable GNNs

Outline

Introduction

25 min
9:25 – 9:50

Classical Approaches for Scalable GNNs

Siqiang Luo & Ningyi Liao

Basic GNN: GCN (T Kipf & M Welling | ICLR'17)

Graph Sampling: GraphSAGE (W Hamilton et al | NeurIPS'17)

Graph Partition: AliGraph (R Zhu et al | VLDB'19)

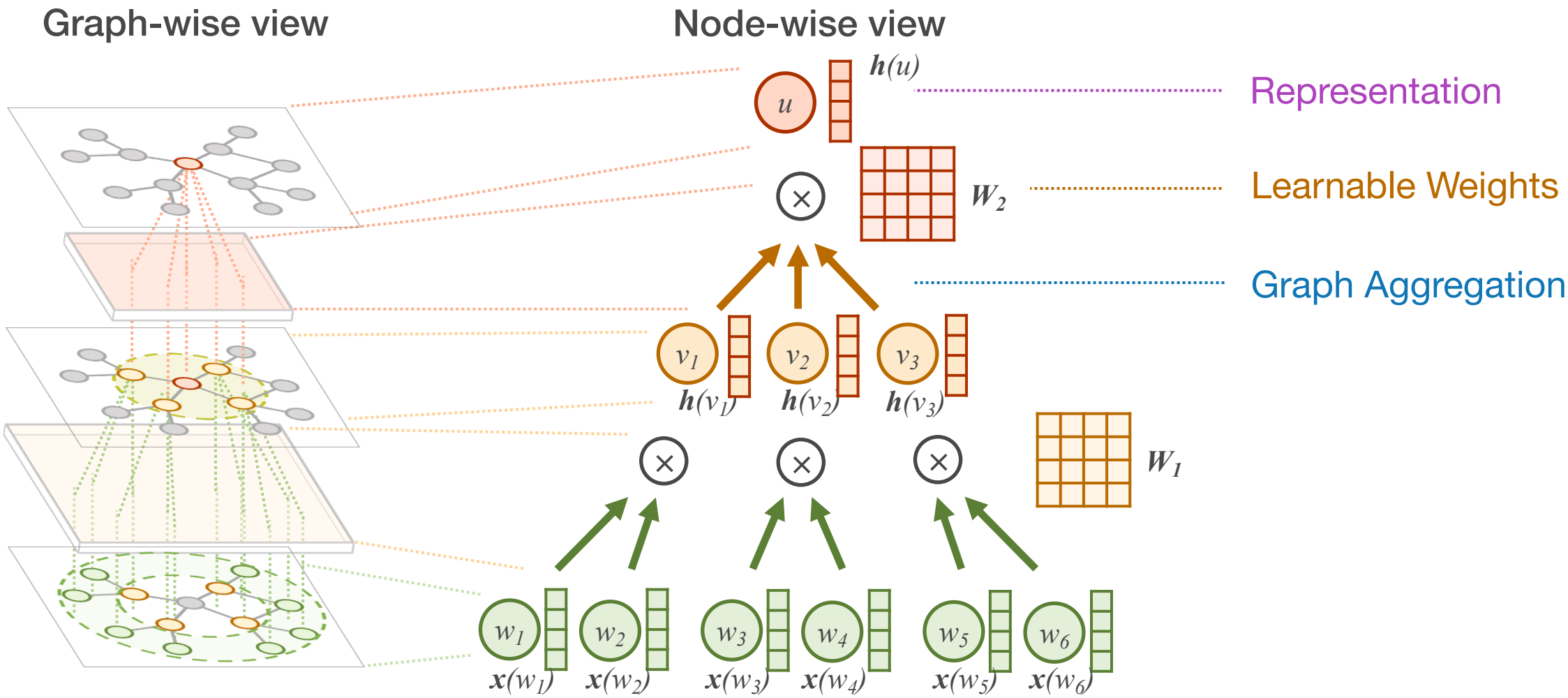
Decoupled Propagation: SGC (F Wu et al | ICLR'19)

***Application*:** Social Network Friend Recommendation

Hardware-aware Sampling

Subgraph Extraction

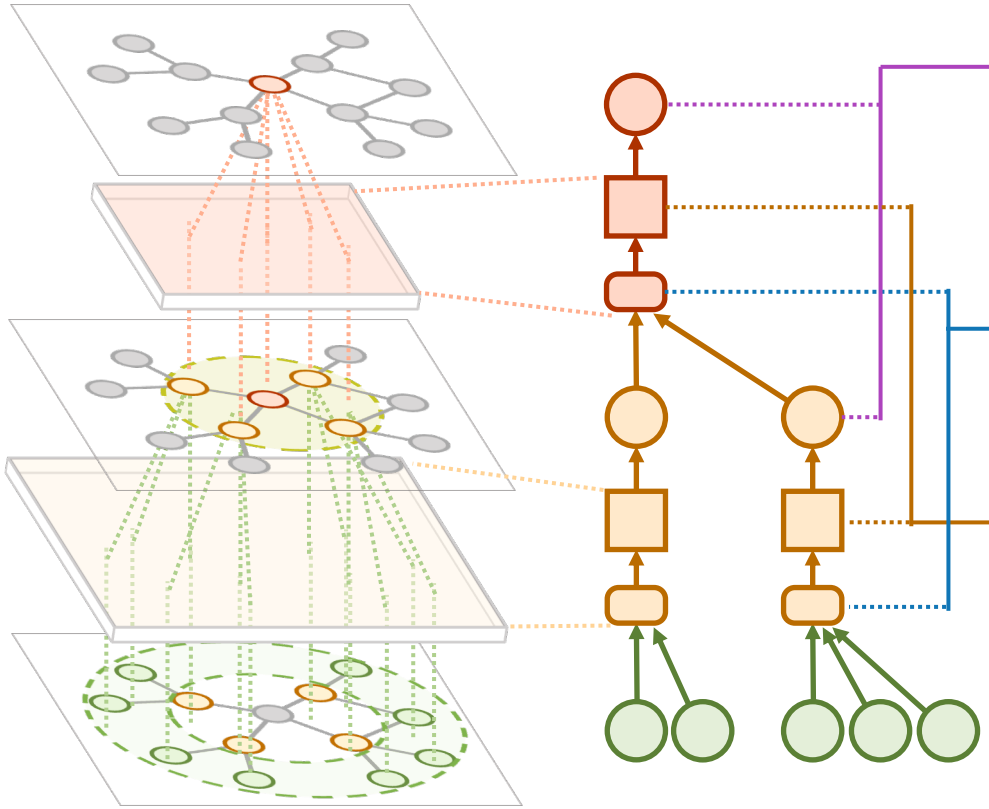
Basic GNN: GCN Scalability Analysis



T Kipf & M Welling. "Semi-supervised classification with graph convolutional networks". *ICLR'17*.

Basic GNN: GCN Scalability Analysis

Graph-wise view **Node-wise view** **Computation**



Representation

$n \times f$ Dense

$$H^{(l+1)} = \sigma\left(\tilde{A}H^{(l)}W^{(l)}\right)$$

Graph Aggregation

$n \times n$ Sparse

Learnable Weights

$f \times f$ Dense

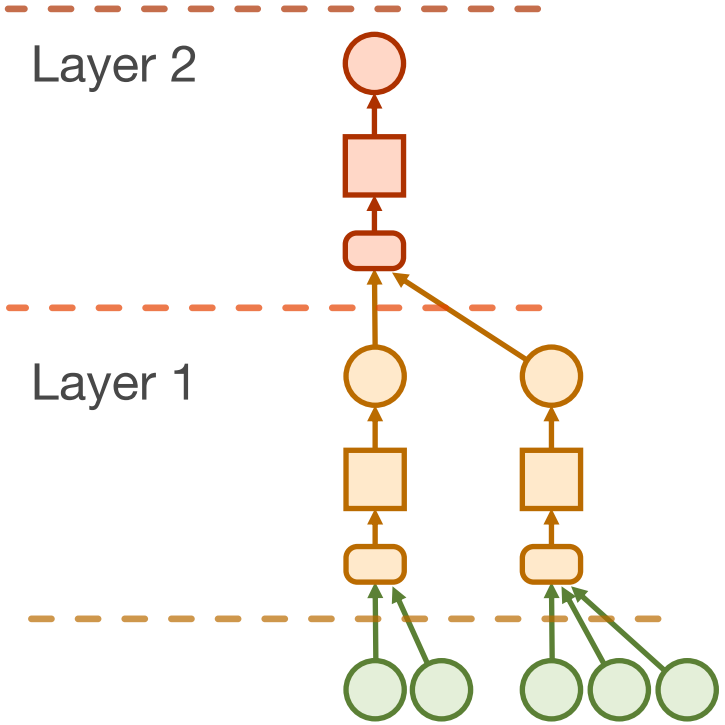
Node Attribute

Init: $H^{(0)} = X$

T Kipf & M Welling. "Semi-supervised classification with graph convolutional networks". *ICLR'17*.

Basic GNN: GCN Complexity

Node-wise view



Computation

$$\boldsymbol{H}^{(l+1)} = \sigma\left(\tilde{\boldsymbol{A}}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)}\right)$$

Graph Aggregation $O(mf)$

Sparse-dense Matrix Multiplication

Weight Transformation $O(nf^2)$

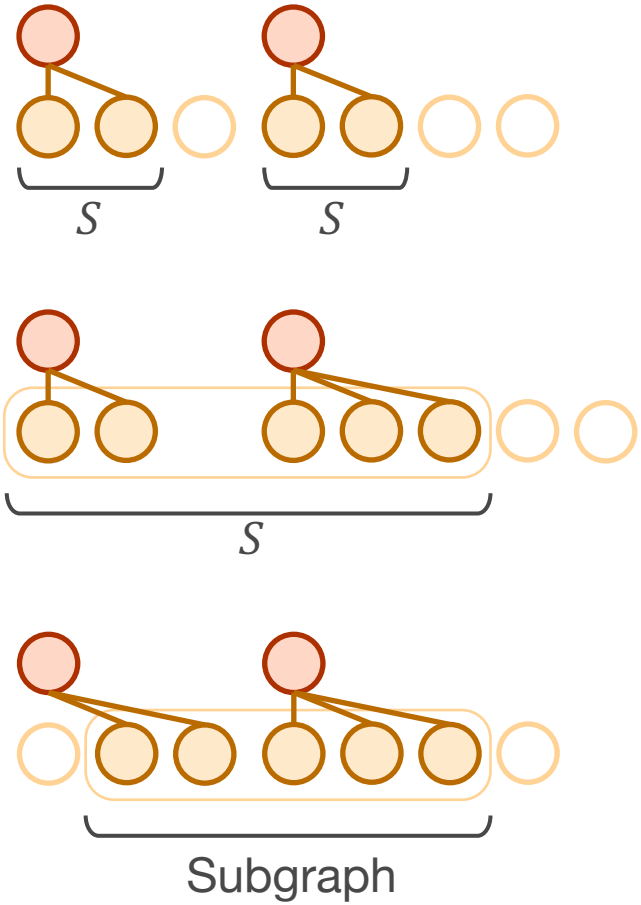
Dense-dense Matrix Multiplication

L-layers: $O(Lmf + Lnf^2)$

T Kipf & M Welling. "Semi-supervised classification with graph convolutional networks". *ICLR'17*.

Graph Sampling

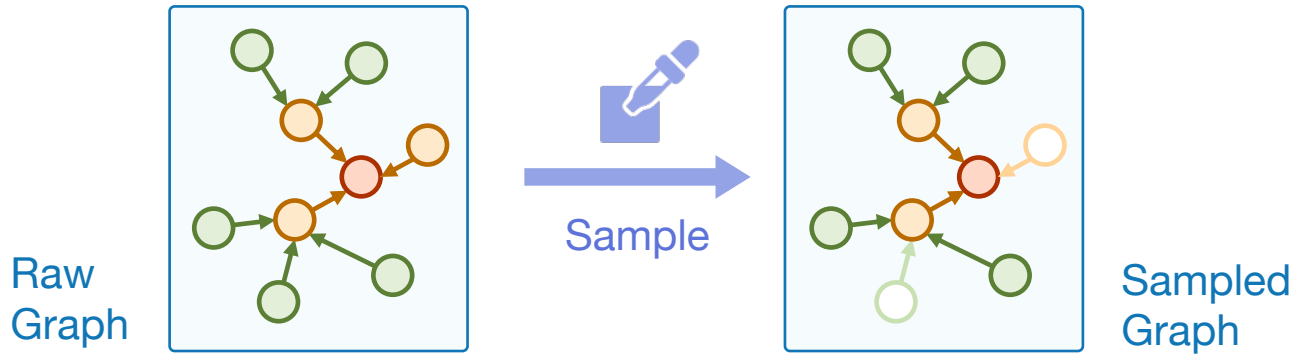
- Node-wise sampling: limit neighborhood size of each node
- Layer-wise sampling: limit total neighborhood size of each layer
- Subgraph-wise sampling: limit neighborhood within a subgraph

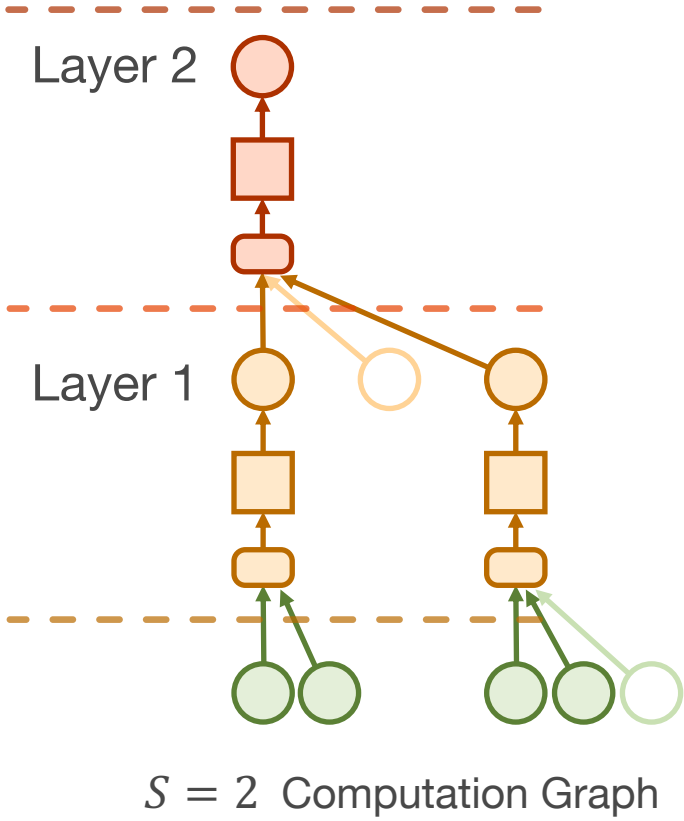


X Liu et al, "Sampling Methods for Efficient Training of Graph Convolutional Networks: A Survey", *J. Autom. Sinica 2022*.

Graph Sampling: GraphSAGE (NeurIPS'17)

- Sampling in graph aggregation: limit neighborhood size of each node to S
- Structured neighborhood: sampled representations are fixed-length
- Feasible for various aggregation schemes

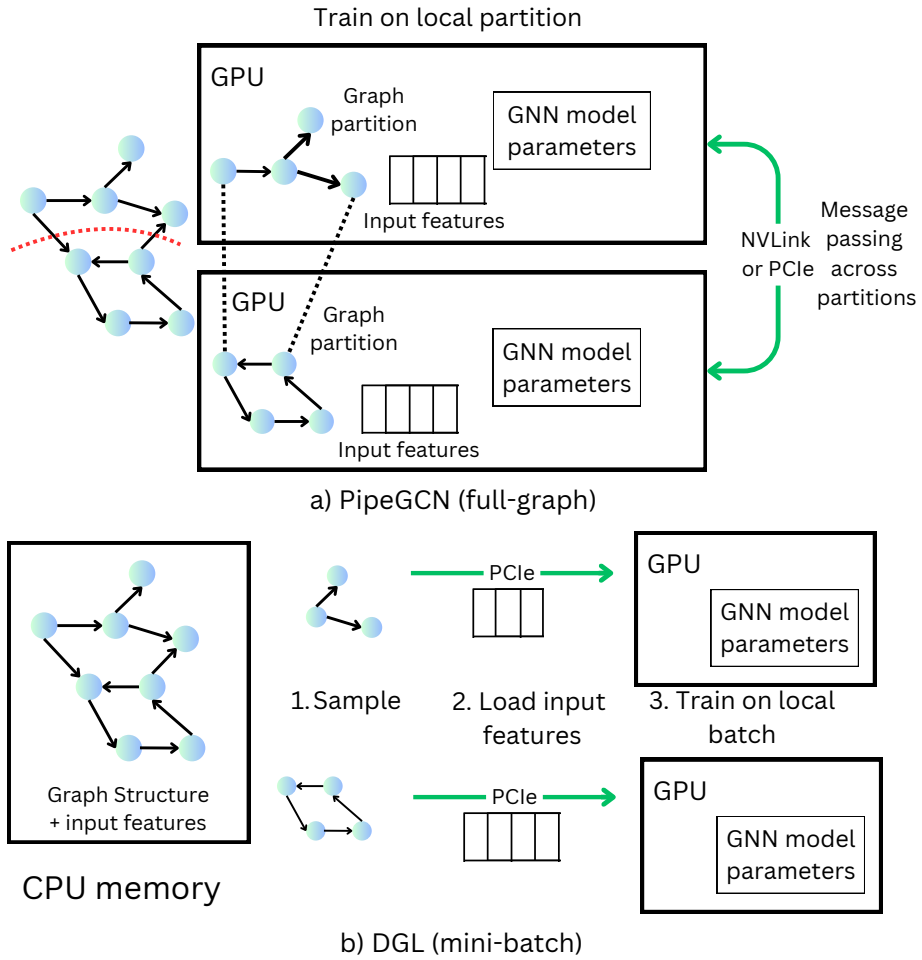




W Hamilton et al, "Inductive Representation Learning in Large Attributed Graphs", *NeurIPS'17.*

Graph Partition

- Objective: minimize data loss in subgraphs during training
- Commonly used in multi-host/ multi-GPU training
- MB: only load subgraph onto GPU

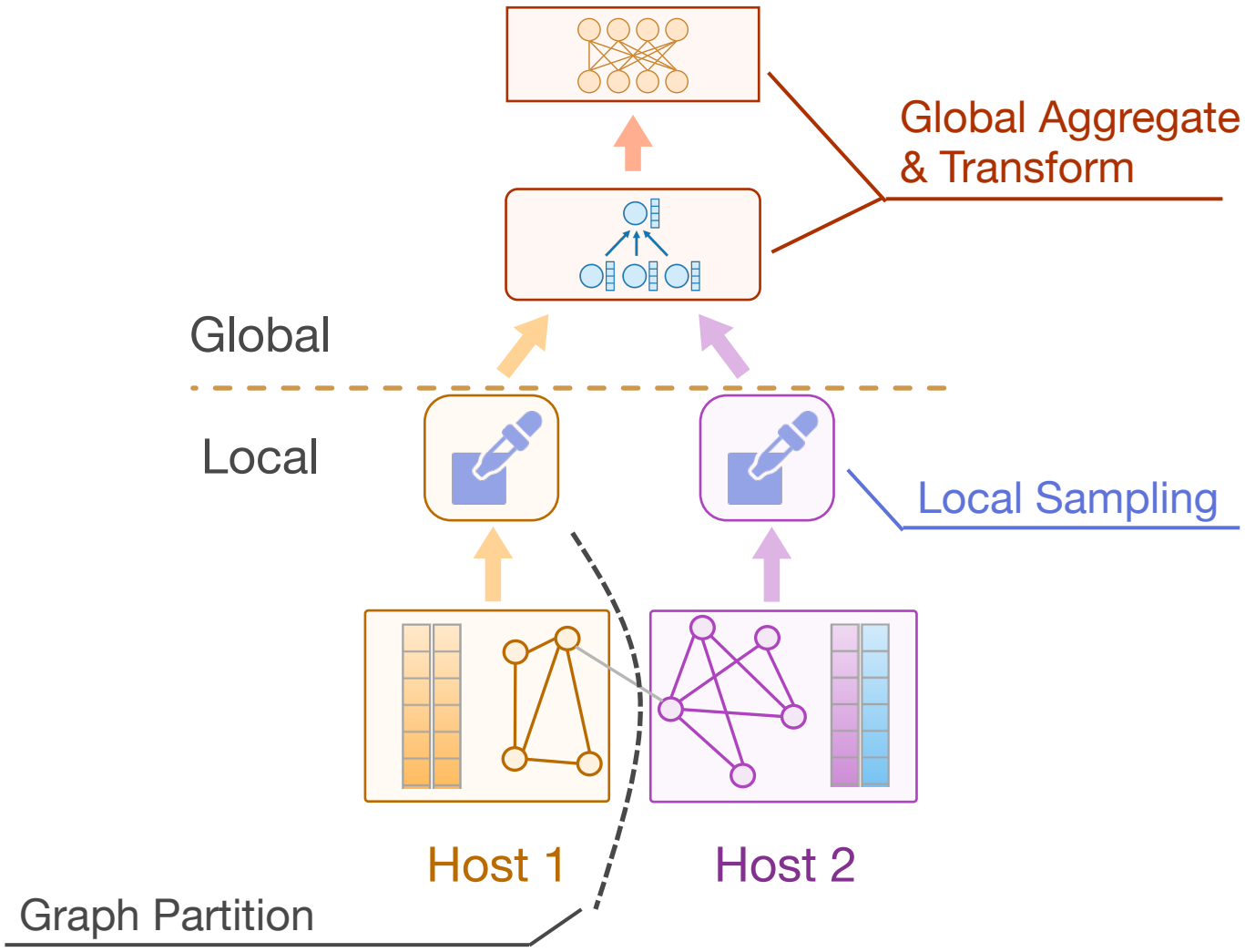


H Yuan et al, "Comprehensive Evaluation of GNN Training Systems: A Data Management Perspective", *VLDB'24.*

S Bajaj et al, "Graph Neural Network Training Systems: A Performance Comparison of Full-Graph and Mini-Batch", *VLDB'25*.

Graph Partition: AliGraph (VLDB'19)

- Available partitions:
  - METIS
  - Vertex/Edge cut
  - 2D partition
  - Streaming
- Local Sampling within subgraphs on each host



R Zhu et al, "Aligraph: A comprehensive graph neural network platform", *VLDB'19*.

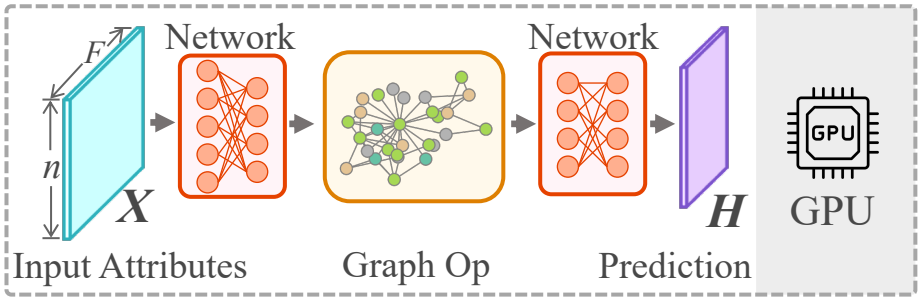
Decoupled Propagation

- Graph propagation is the computational bottleneck
- Graph computation → CPU
- Network learning → GPU
- MB: random sampling/ hashing

S Bajaj et al, "Graph Neural Network Training Systems: A Performance Comparison of Full-Graph and Mini-Batch", *VLDB'25.*

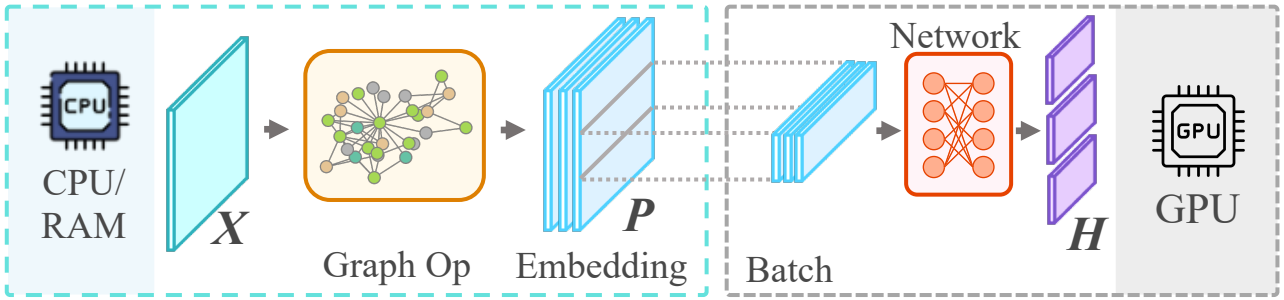
Full-batch

Full GPU Workload



Mini-batch

CPU+GPU Workload



N Liao et al, "A Comprehensive Benchmark on Spectral GNNs: The Impact on Efficiency, Memory, and Effectiveness", *SIGMOD'26*.

Decoupled Propagation: SGC (ICLR'19)

- Combine the iterative propagations in GCN
- Reduce GPU memory overhead: $O(m) \rightarrow O(s)$
- No change in time overhead: remain $O(Lm)$

GCN (iterative)

L Graph Propagation

$$\boldsymbol{H}^{(L)} = \sigma(\tilde{\boldsymbol{A}}\sigma(\tilde{\boldsymbol{A}}\cdots\tilde{\boldsymbol{A}}\boldsymbol{X}\boldsymbol{W}^{(1)}\cdots)\boldsymbol{W}^{(L-1)})$$

L Weight Transformation

SGC (decoupled)

L Graph Propagation

$$\boldsymbol{H}^{(0)} = \boldsymbol{P} = \tilde{\boldsymbol{A}}^{L} \cdot \boldsymbol{X}$$

CPU/RAM

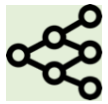
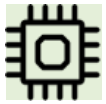
L Weight Transformation

$$\boldsymbol{H}^{(l+1)} = \sigma(\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)})$$

GPU

F Wu et al, "Simplifying Graph Convolutional Networks", *ICLR'19*.

Classical Approaches: Takeaways

- Graph computation is the bottleneck in GNN scalability
- Challenges:
  
  Neighbor Explosion
  
  Limited Memory
- Approaches:

  Graph Sampling
  - Reduce neighborhood aggregation

  Graph Partition
  - Batch training on subgraphs

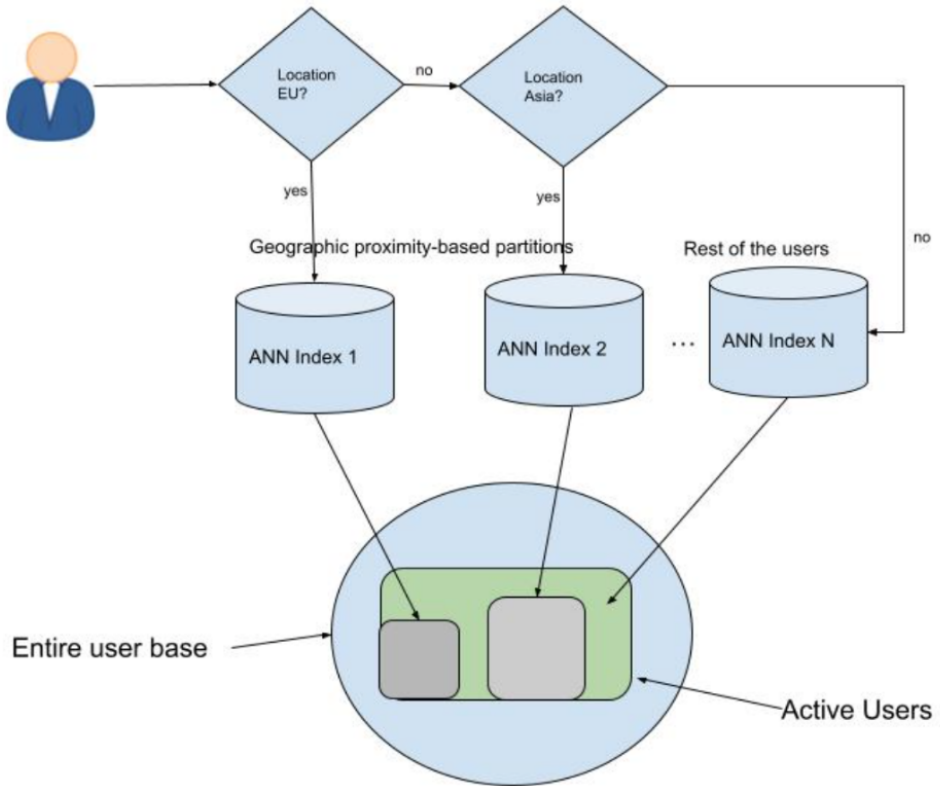
  Graph Partition
  - Reduce subgraph size

  Decoupled Propagation
  - Transfer computation to CPU

Classical Approaches: Applications

Social Network Friend Recommendation

- Candidates are partitioned: based on different retrieval criteria
- GraphSAGE sampling: fast neighbor lookup
- MB: sampling within partitions



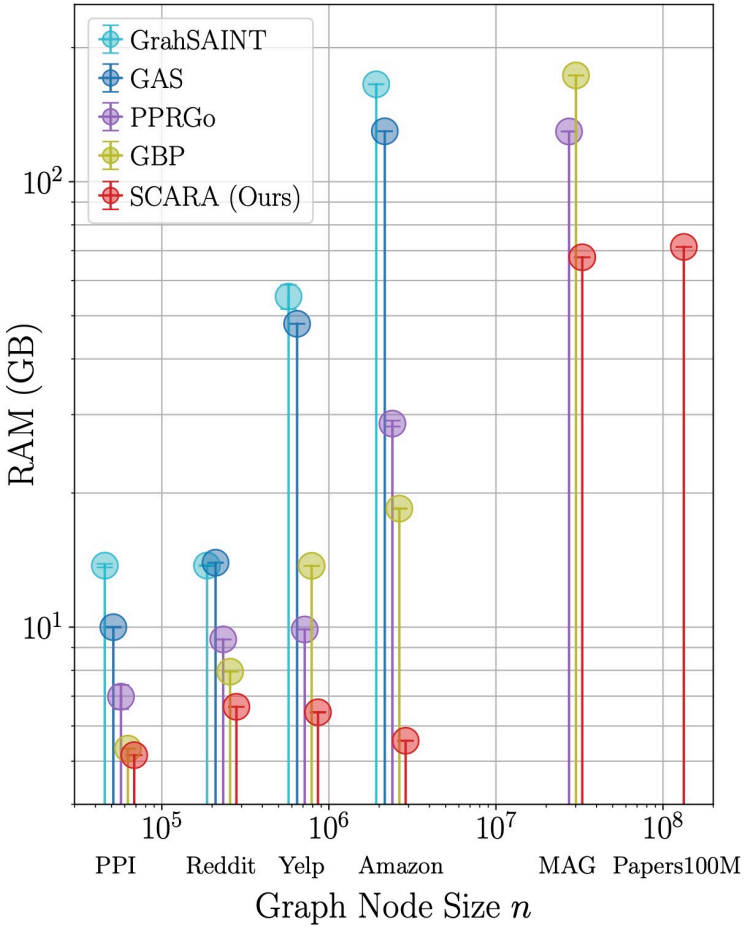
J Shi et al, "Embedding Based Retrieval in Friend Recommendation", *SIGIR'23*.

Classical Approaches: Evaluation

Memory Footprint

- **Sampling**: suffer from neighbor explosion for using more redundant memory
- **Decoupling**: better memory scalability for less additional memory overhead
- Possible OOM under limited memory

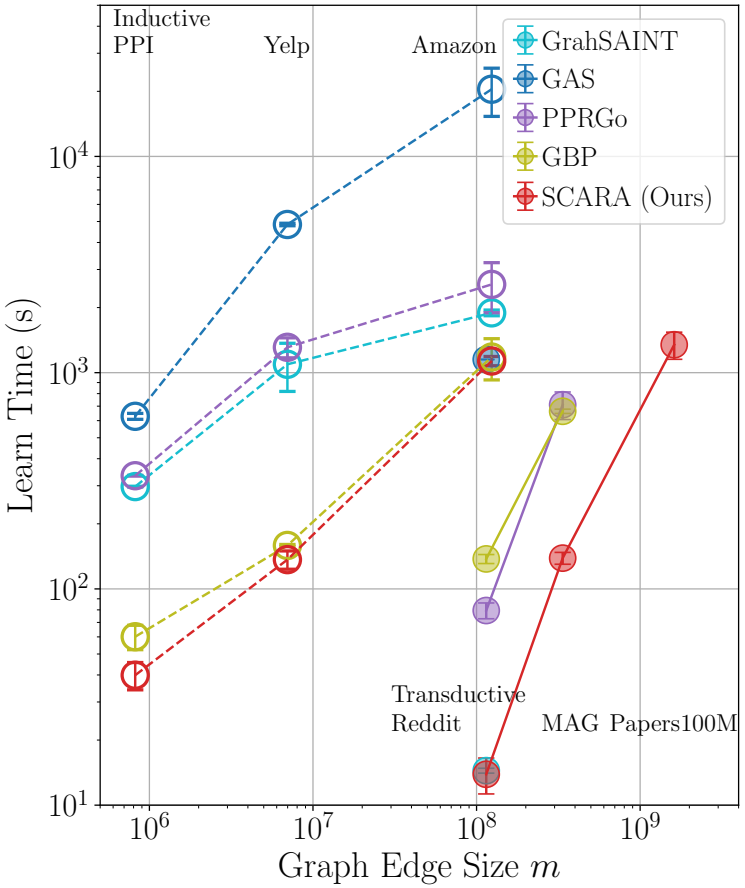
N Liao et al, "Scalable Decoupling Graph Neural Networks with Feature-Oriented Optimization", VLDBJ'23.



Classical Approaches: Evaluation

Time Efficiency

- **Sampling**: higher time overhead caused by device idleness and transmission
- **Decoupling**: speed mostly determined by graph computation algorithms



N Liao et al, "Scalable Decoupling Graph Neural Networks with Feature-Oriented Optimization", VLDBJ'23.

Hardware-aware Sampling

Outline

Introduction

Classical Approaches

20 min
9:50 – 10:10

Hardware-aware Sampling

Ningyi Liao

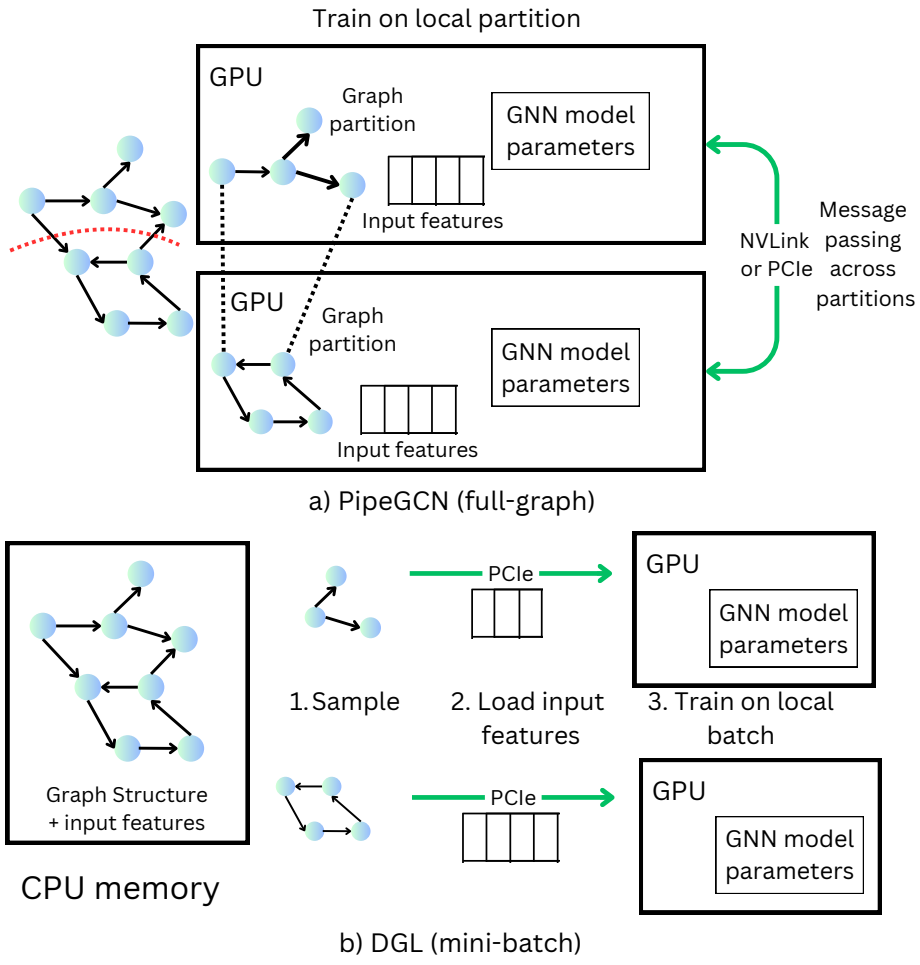
Transmission ADGNN [Z Song et al | SIGMOD'23]
GIDS [J Park et al | VLDB'24]
Computation NeutronOrch [X Ai et al | VLDB'24]
DAHA [Z Li et al | VLDB'24]

Subgraph Extraction

Transmission: ADGNN [SIGMOD'23]

Global Sampling

- Full graph view
- Precision guarantee
- Transmission overhead



Local Sampling

- Distributed storage & computation
- Information loss
- Depends on partition

Z Song et al, "ADGNN: Towards Scalable GNN Training with Aggregation-Difference Aware Sampling", *SIGMOD'23*.

Transmission: ADGNN [SIGMOD'23]

Balance Local & Global Sampling

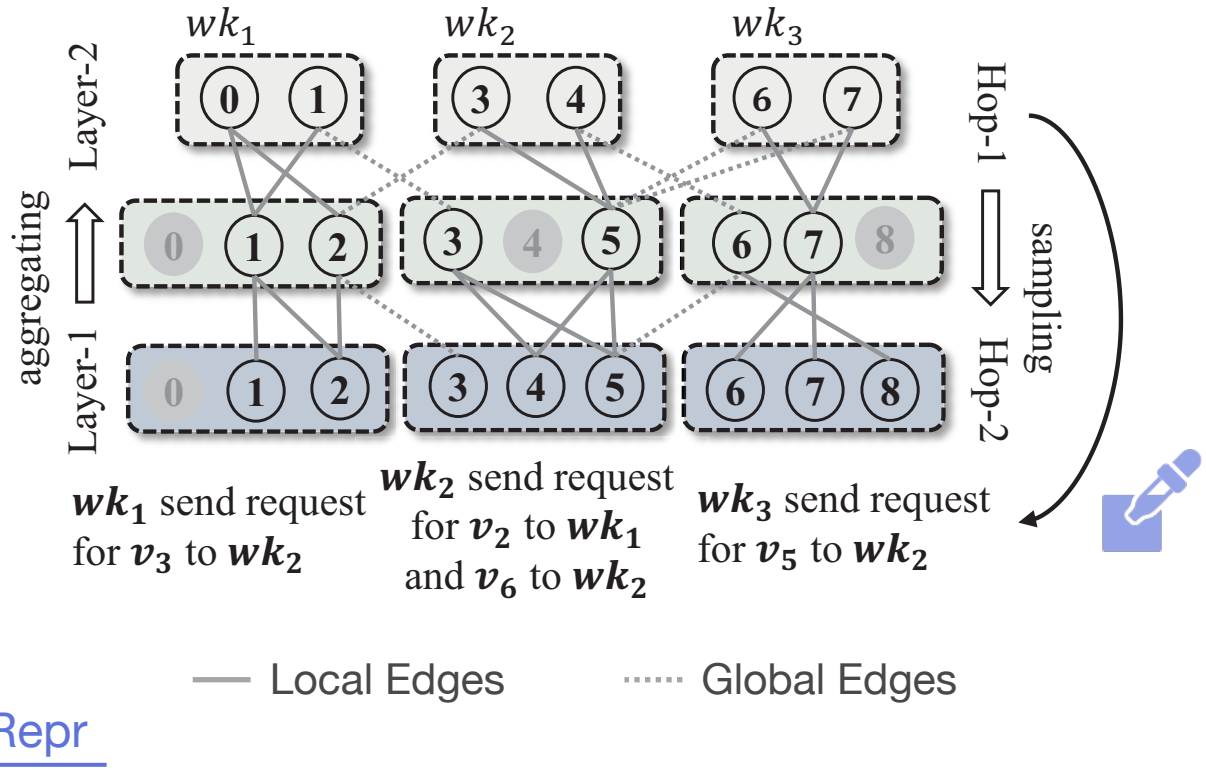
- Maintain a set of effective neighbors locally
- Only communicate for top remote nodes that reduce precision

$$AD^{t,\ell}_{\mathcal{S}(v;k)} = \| \frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} \boldsymbol{H}^{t,\ell}_u - \frac{1}{k} \sum_{u \in \mathcal{S}(v;k)} \boldsymbol{H}^{t,\ell}_u \|^2,$$

Precision

Sampled Repr

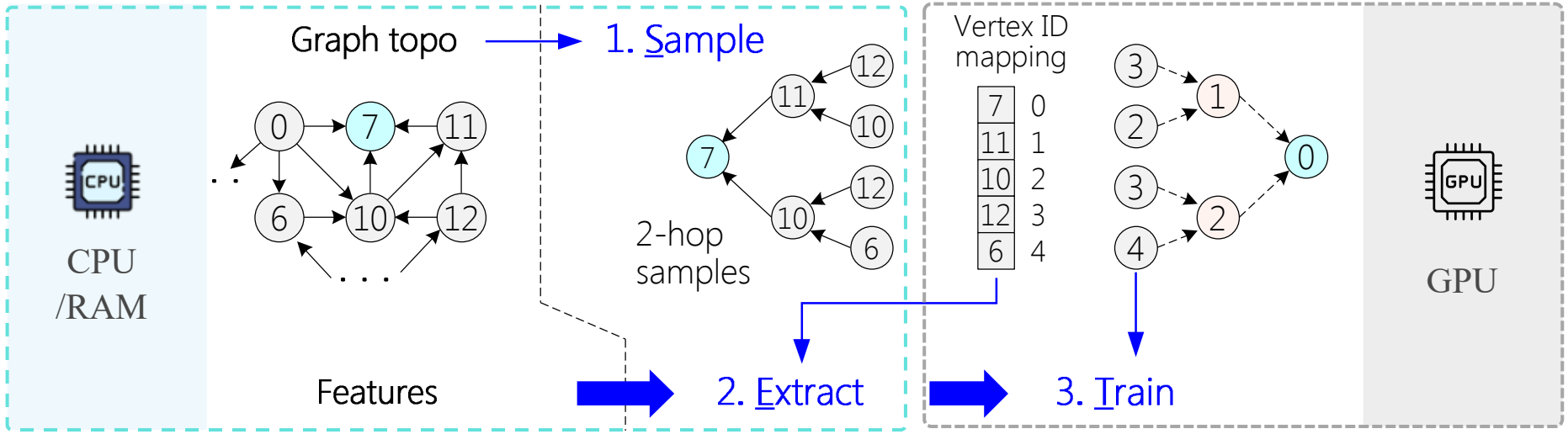
Sampled Repr



Transmission: GIDS [VLDB'24]

Graph Structure: Parallel Sampling

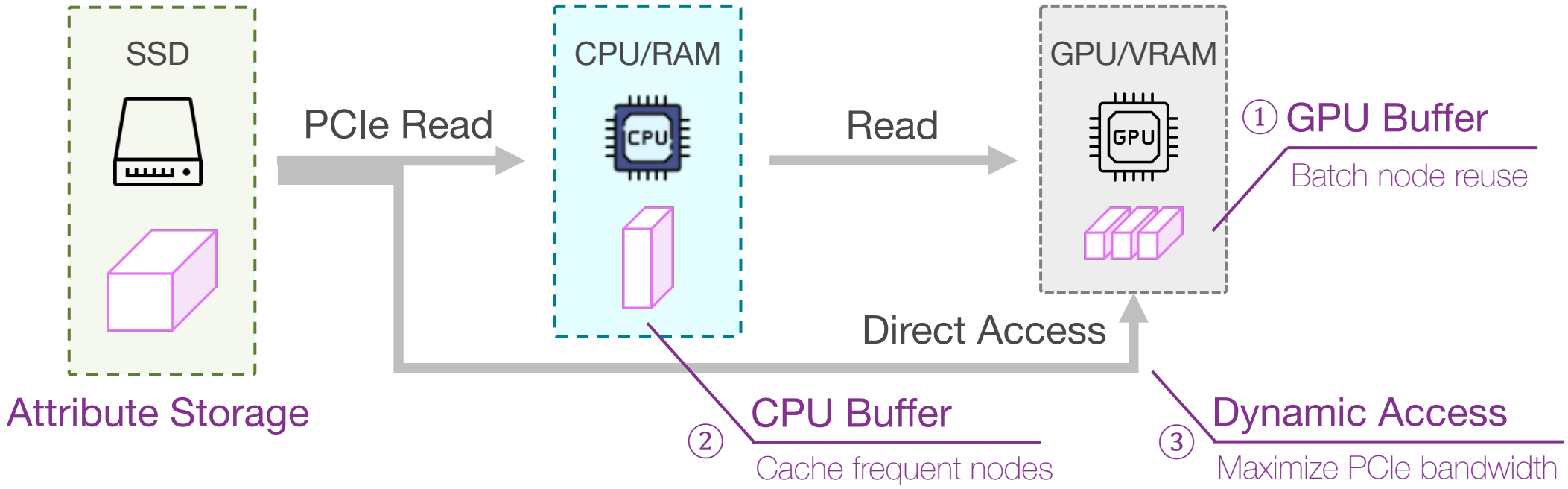
- Sampling can be performed independently from NN training
- Parallel sampling for batches on CPU graph storage



J Yang et al, "GNNLab: a factored system for sample-based GNN training over GPUs", *EuroSys'22.*
J Park et al, "Accelerating Sampling and Aggregation Operations in GNN Frameworks with GPU Initiated Direct Storage Accesses", *VLDB'24*.

Transmission: GIDS [VLDB'24]

Node Attribute: Hierarchical Buffer

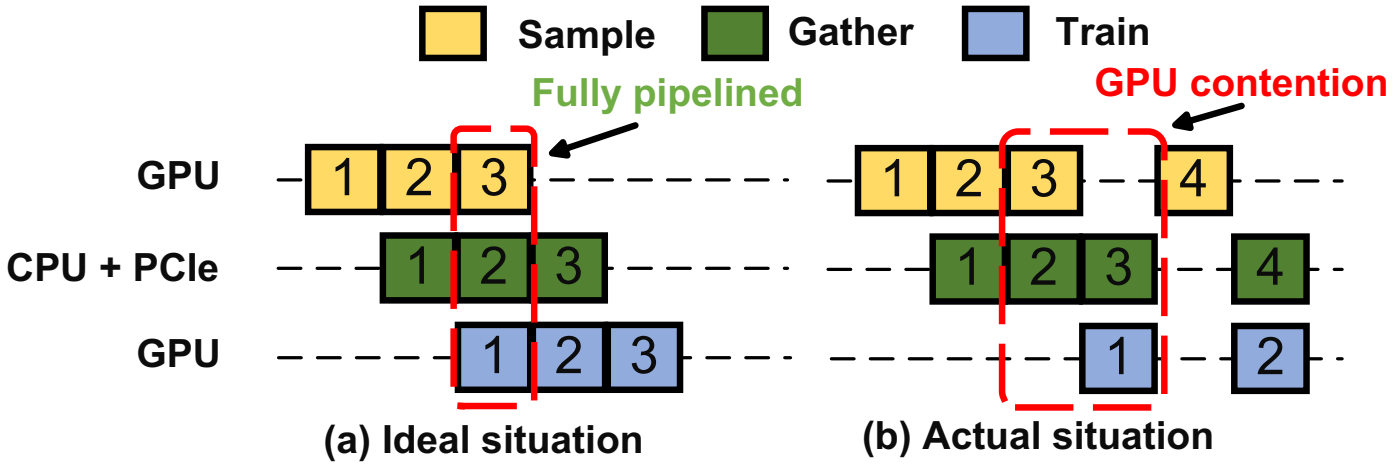


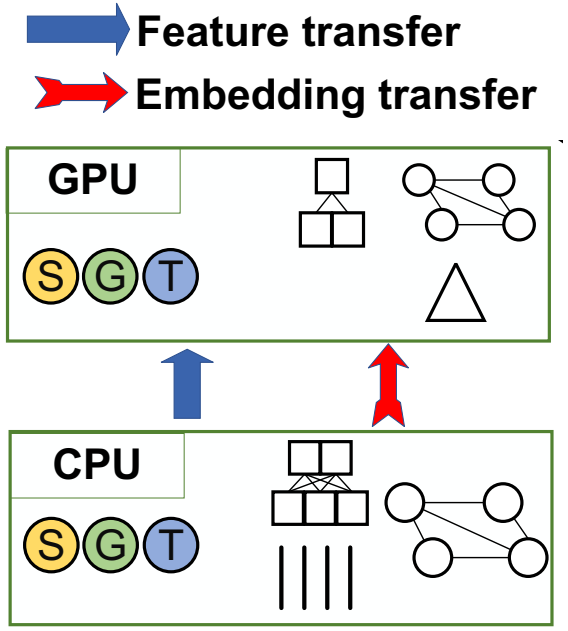
J Park et al, "Accelerating Sampling and Aggregation Operations in GNN Frameworks with GPU Initiated Direct Storage Accesses", *VLDB'24*.

Computation: NeutronOrch [VLDB'24]

Layer-based Computation Orchestrating

- CPU: 1st layer, from attribute to embedding
- GPU: higher layers, only process embeddings



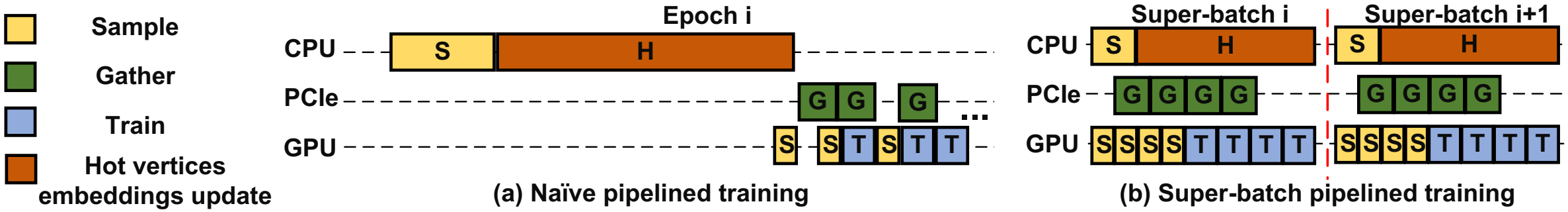


X Ai et al, " NeutronOrch: Rethinking Sample-Based GNN Training under CPU-GPU Heterogeneous Environments", *VLDB'24*.

Computation: NeutronOrch [VLDB'24]

Cross-batch Data Pipelining

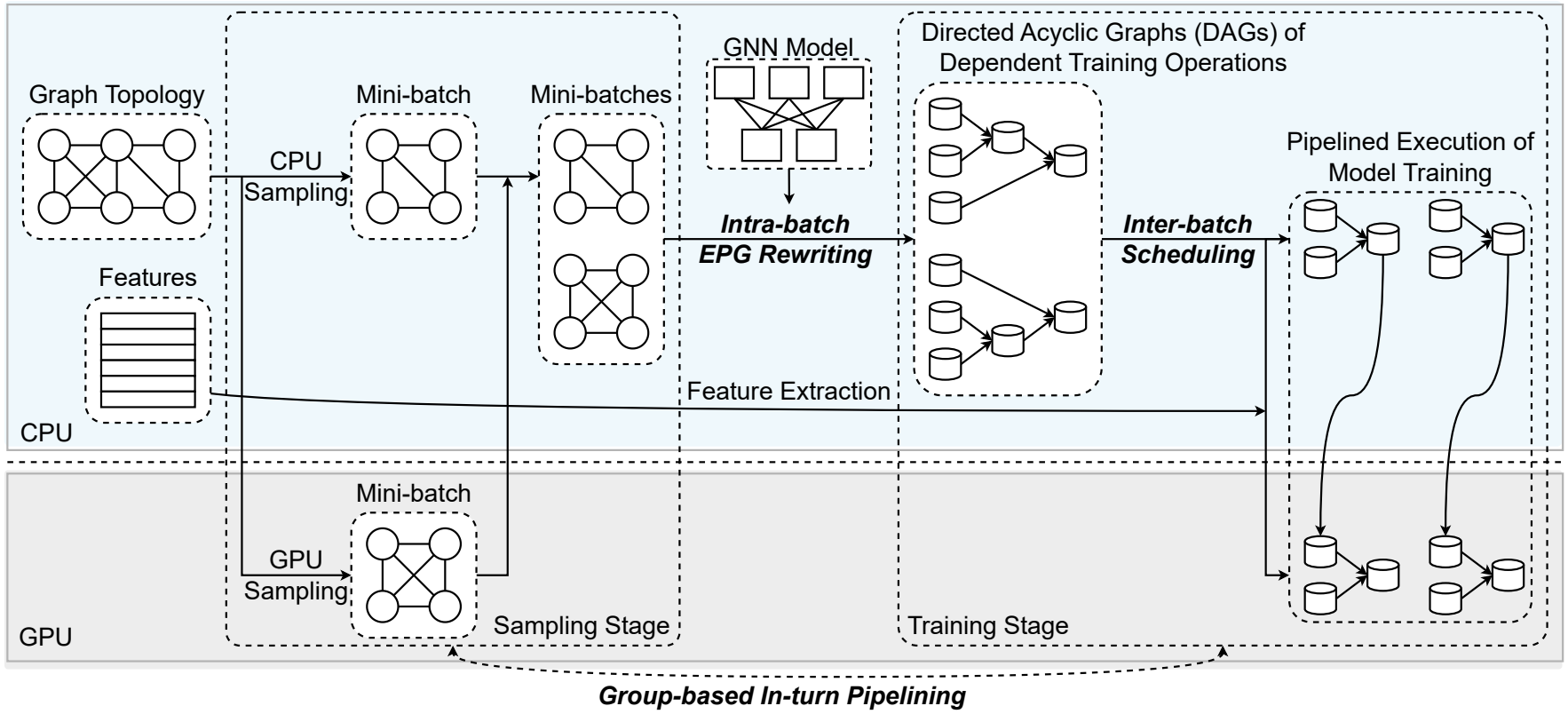
- Hotness-aware sampling across batches on CPU
- Constrain staleness within super-batch



X Ai et al, " NeutronOrch: Rethinking Sample-Based GNN Training under CPU-GPU Heterogeneous Environments", *VLDB'24*.

Computation: DAHA [VLDB'24]

- Profiling & scheduling between CPU and GPU computation



CPU

- Feature processing
- Weight transformation
- Major sampling

GPU

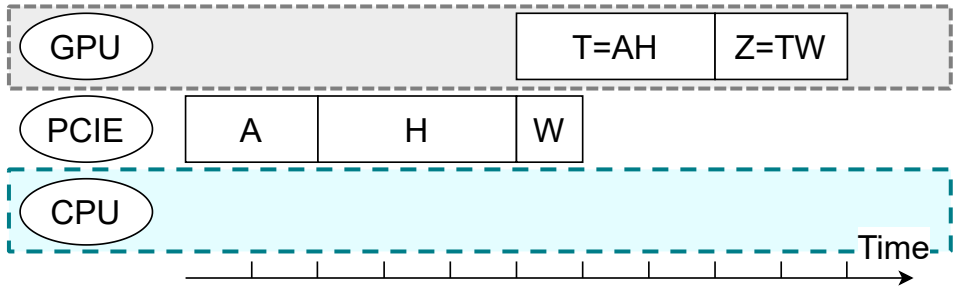
- Accelerate sampling
- Graph aggregation

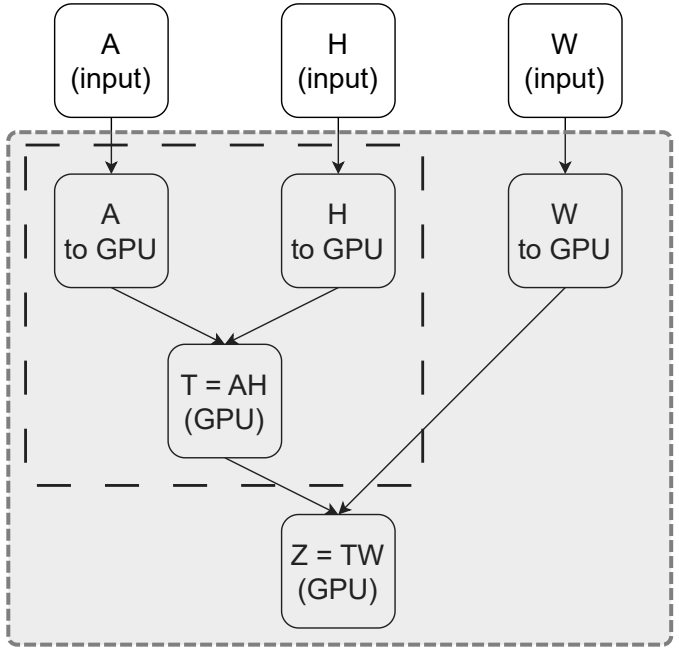
Z Li et al, "DAHA: Accelerating GNN Training with Data and Hardware Aware Execution Planning", *VLDB'24*.

Computation: DAHA [VLDB'24]

Computation 1: Pure GPU

- Fast feature processing (dense MM)
- Long data transfer



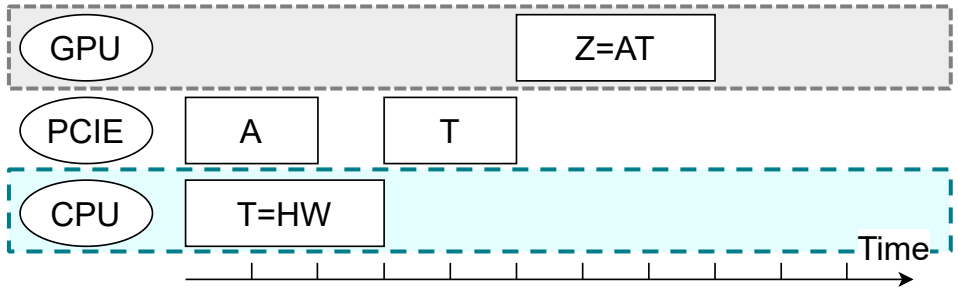


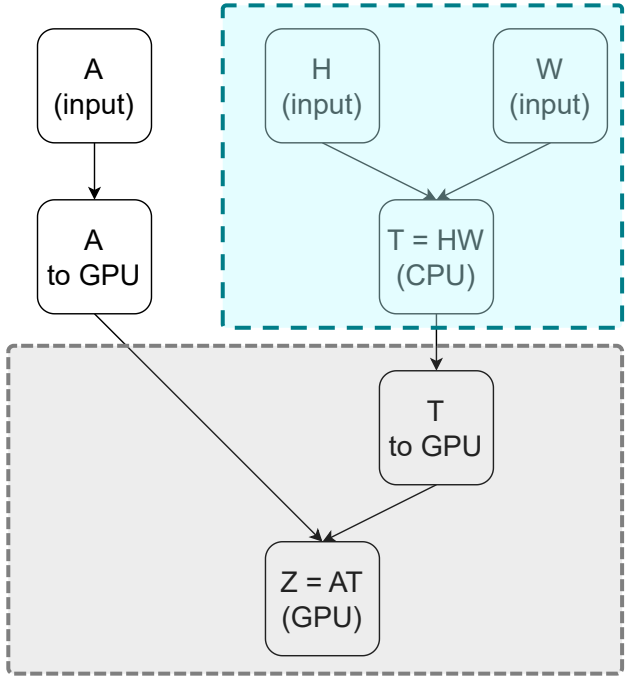
Z Li et al, "DAHA: Accelerating GNN Training with Data and Hardware Aware Execution Planning", *VLDB'24*.

Computation: DAHA [VLDB'24]

Computation 2: Hybrid GPU

- Shift computation workload to CPU
- Only transfer embedding with lower dimension





Z Li et al, "DAHA: Accelerating GNN Training with Data and Hardware Aware Execution Planning", *VLDB'24*.

Hardware-aware Sampling: Takeaways

Transmission

ADGNN [SIGMOD'23]

- Between computational devices
- Balance Local & Global Sampling

GIDS [VLDB'24]

- Between storage devices
- Hierarchical node attribute buffer

Computation

NeutronOrch [VLDB'24]

- Among GNN operations
- Layer- & batch-wise pipelining

DAHA [VLDB'24]

- Among data partitions
- Balance CPU & GPU computation

Subgraph Extraction

Outline

Introduction

Classical Approaches

Hardware-aware Sampling

20 min
10:10 – 10:30

Subgraph Extraction
Ningyi Liao

- **Partition** G3 [X Wan et al | SIGMOD'23]
- **Generation** SUREL [H Yin et al | VLDB'23] & GENTI [Z Yu et al | VLDB'24]
- **Slicing** TIGER (K Wang et al | VLDB'24)
- ***Application*:** Hardware Obfuscation Analysis

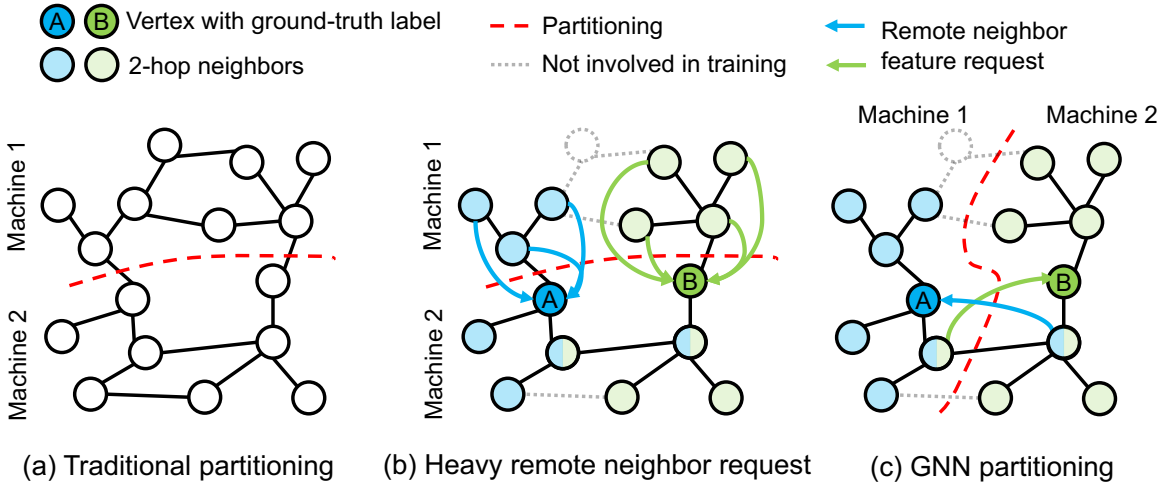
Subgraph Extraction: Challenges

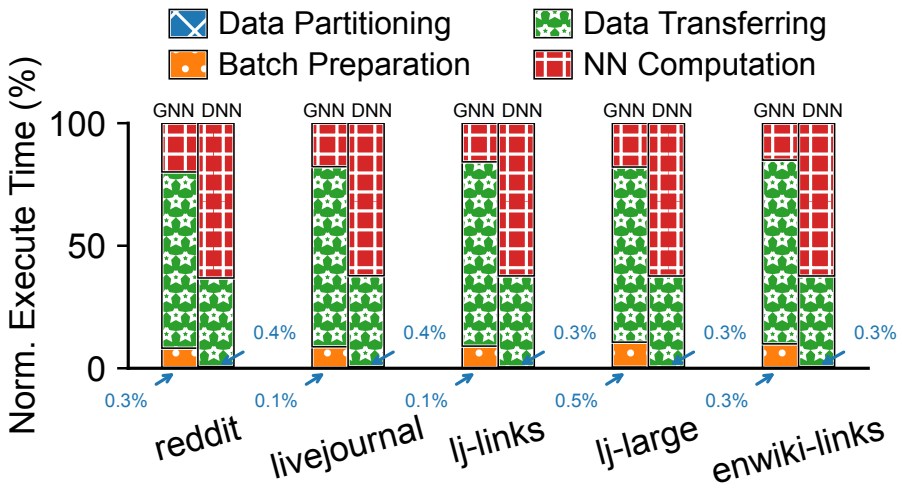
Effectiveness

- Topology-based partition may be not optimal

Efficiency

- Data transferring overhead is significant

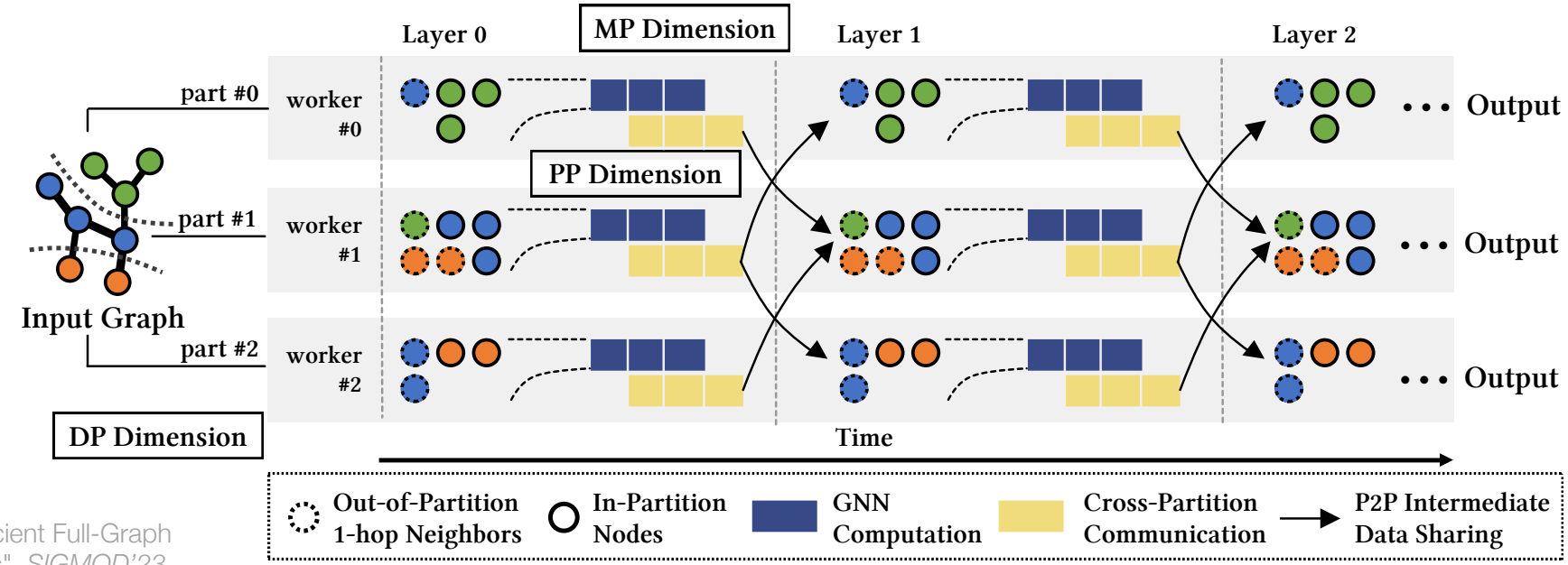




H Yuan et al, "Comprehensive Evaluation of GNN Training Systems: A Data Management Perspective", *VLDB'24.*

Partition: G3 [SIGMOD'23]

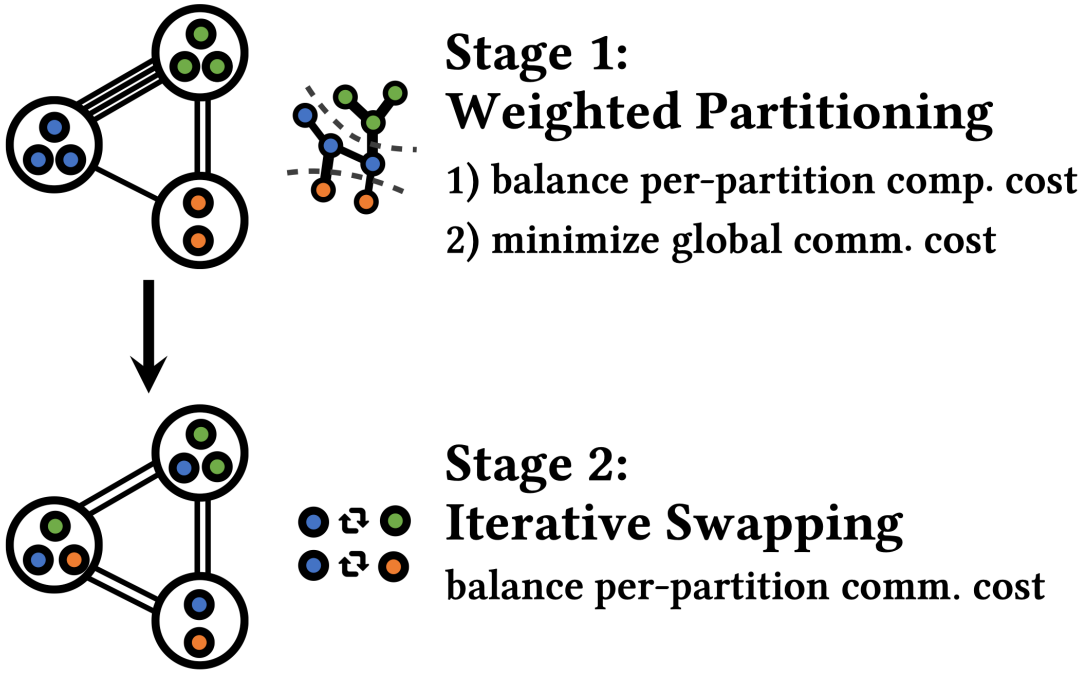
- Hybrid Parallelism
  - Data-parallel: graph partitions on workers
  - Model-parallel: full-graph training with P2P transmission
  - Pipeline-parallel: overlapped computation and transmission



X Wan et al, "Scalable and Efficient Full-Graph GNN Training for Large Graphs", *SIGMOD'23*.

Partition: G3 [SIGMOD'23]

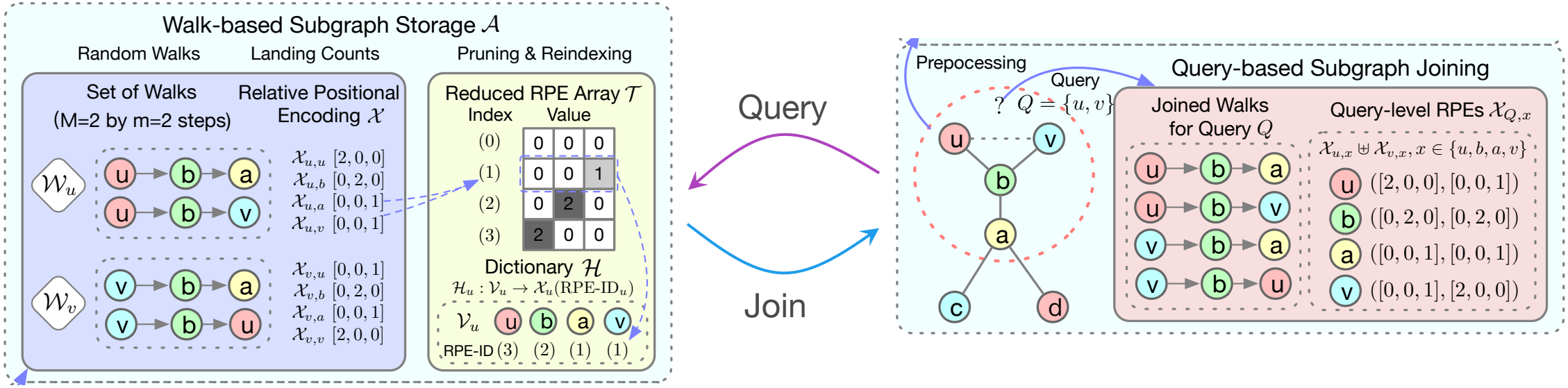
- 2-stage graph partition
  - **Stage 1**: unsupervised, minimize global communication by reducing cut edges
  - **Stage 2**: adaptive updated, balance empirical communication



X Wan et al, "Scalable and Efficient Full-Graph GNN Training for Large Graphs", *SIGMOD'23*.

Generation: SUREL [VLDB'23]

- Walk-based Subgraph Extraction
  - Sample multiple walks for each query node
  - Join walks to generate a subgraph

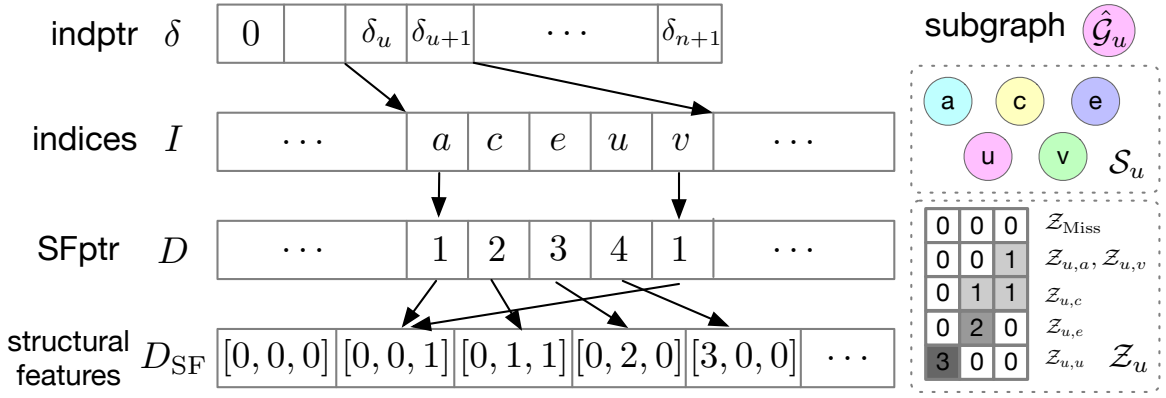


H Yin et al, "Algorithm and system co-design for efficient subgraph-based graph representation learning", *VLDB'22*.
H Yin et al, "SUREL+: Moving from Walks to Sets for Scalable Subgraph-Based Graph Representation Learning", *VLDB'23*.

Generation: SUREL [VLDB'23]

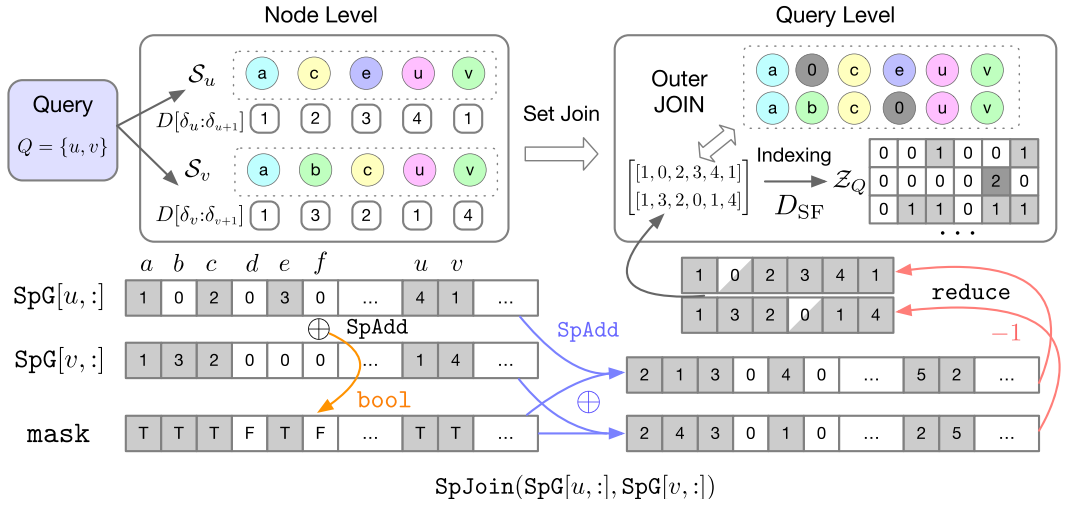
Walk-based Storage

- Fast neighbor access with pointer
- Reduce feature overlap



Sparse Join Operator

- Remove duplicate nodes
- Reduce sparse storage



H Yin et al, "Algorithm and system co-design for efficient subgraph-based graph representation learning", *VLDB'22*.
H Yin et al, "SUREL+: Moving from Walks to Sets for Scalable Subgraph-Based Graph Representation Learning", *VLDB'23*.

Generation: GENTI [VLDB'24]

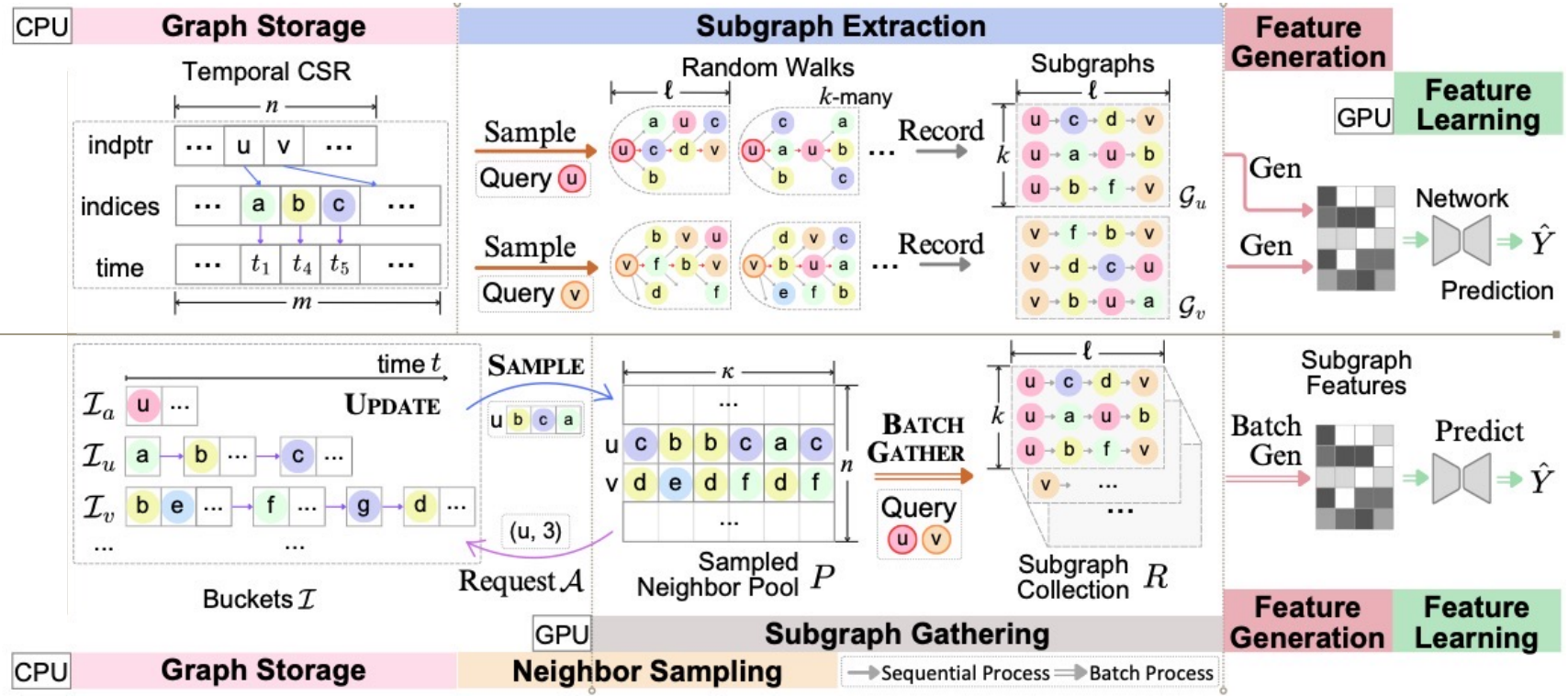
- Dynamic Subgraph Extraction: streaming of subgraph sampling & join

Standard Process

High CPU usage

GENTI

CPU-GPU pipeline

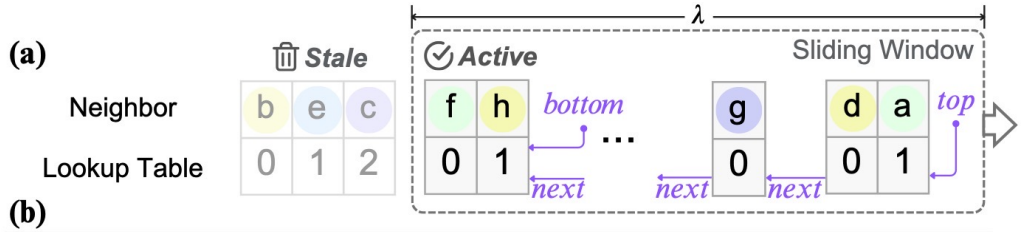


Z Yu et al, "GENTI: GPU-Powered Walk-Based Subgraph Extraction for Scalable Representation Learning on Dynamic Graphs", *VLDB'24*.

Generation: GENTI [VLDB'24]

Time-based Storage

- Fast sampling across buckets
- Constrain time by sliding window

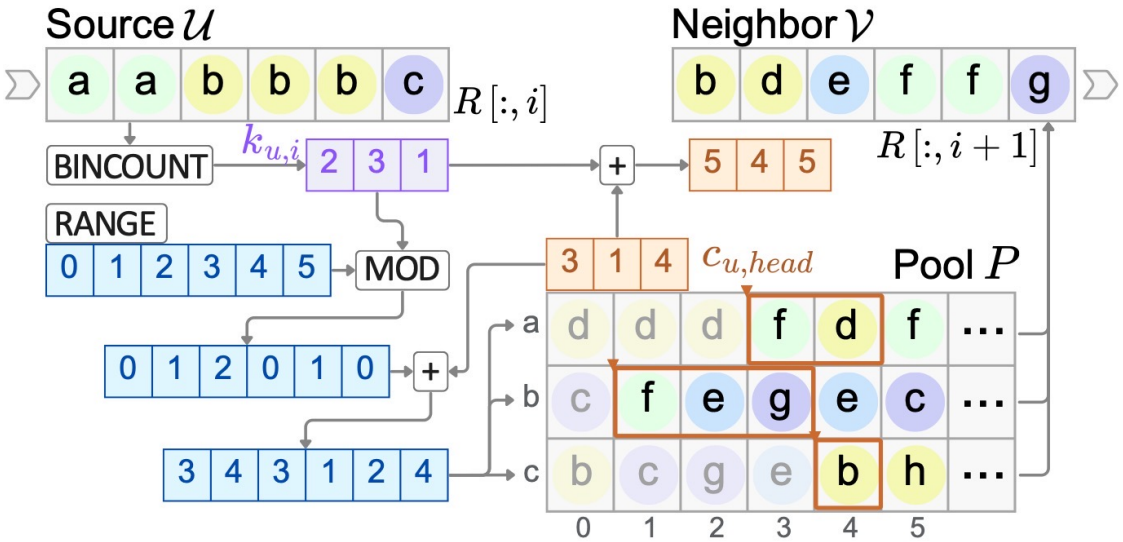


(b)

Bucket	$\mathcal{I}_{u,0}$	$\mathcal{I}_{u,1}$	⋯	$\mathcal{I}_{u,r-1}$	$\mathcal{I}_{u,r}$
Timestamp	$[0, \alpha)$	$[\alpha, 2\alpha)$	⋯	$[(r-1)\alpha, r\alpha)$	$[r\alpha, (r+1)\alpha)$
Sample Weight	$[0, e^1)$	$[e^1, e^2)$	⋯	$[e^{r-1}, e^r)$	$[e^r, e^{r+1})$
Random Number	$\varnothing$	$\{z_4\}$	⋯	$\{z_3\}$	$\{z_1, z_2\}$
Fetch Amount	0	1	⋯	1	2
Sampled Neighbor	$\varnothing$	f	⋯	g	a d

Batch Join GPU Operator

- Dense operations on GPU
- Complexity: reduce from L to $\sqrt{L}$



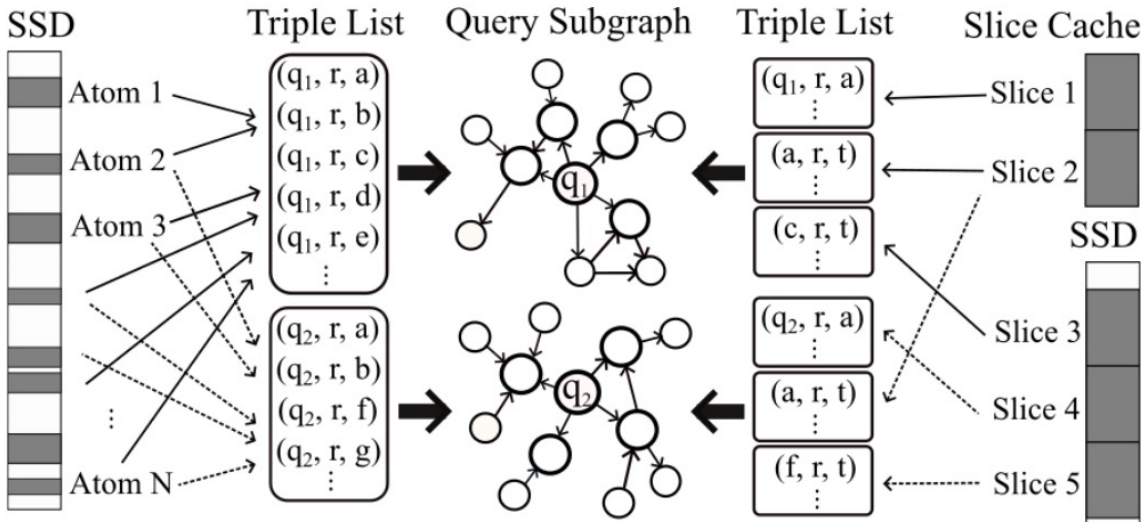
Z Yu et al, "GENTI: GPU-Powered Walk-Based Subgraph Extraction for Scalable Representation Learning on Dynamic Graphs", *VLDB'24*.

Slicing: TIGER [VLDB'24]

- Subgraph extraction on heterogeneous graph: based on edge triples
- Bottleneck: element lookup on disk
- Slicing: store elements of same triples together ← NP-Hard

Simple Extraction

- Random disk access



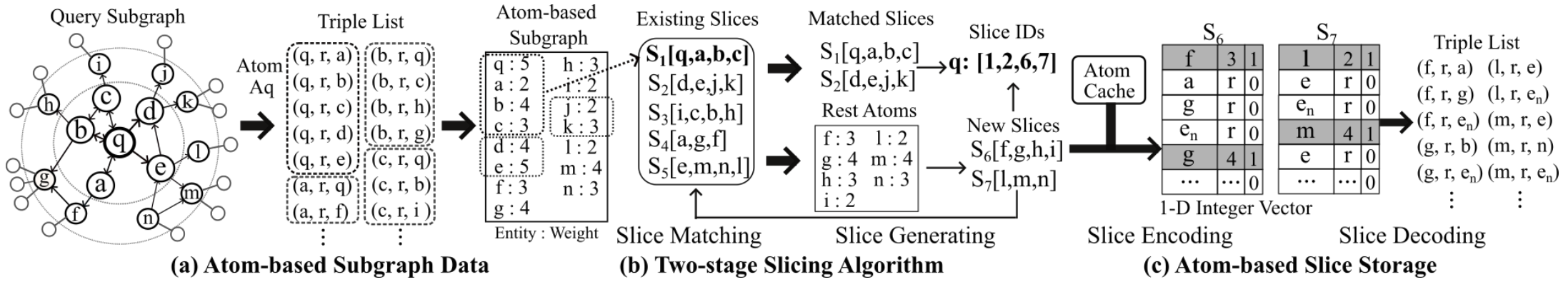
Slice-based Extraction

- Sequential disk access
- Slicing overhead

K Wang et al, "TIGER: Training Inductive Graph Neural Network for Large-Scale Knowledge Graph Reasoning", *VLDB'24*.

Slicing: TIGER [VLDB'24]

- **Slicing**: match-then-generate to reduce overlap storage
- **Caching**: recently accessed elements in memory



K Wang et al, "TIGER: Training Inductive Graph Neural Network for Large-Scale Knowledge Graph Reasoning", *VLDB'24*.

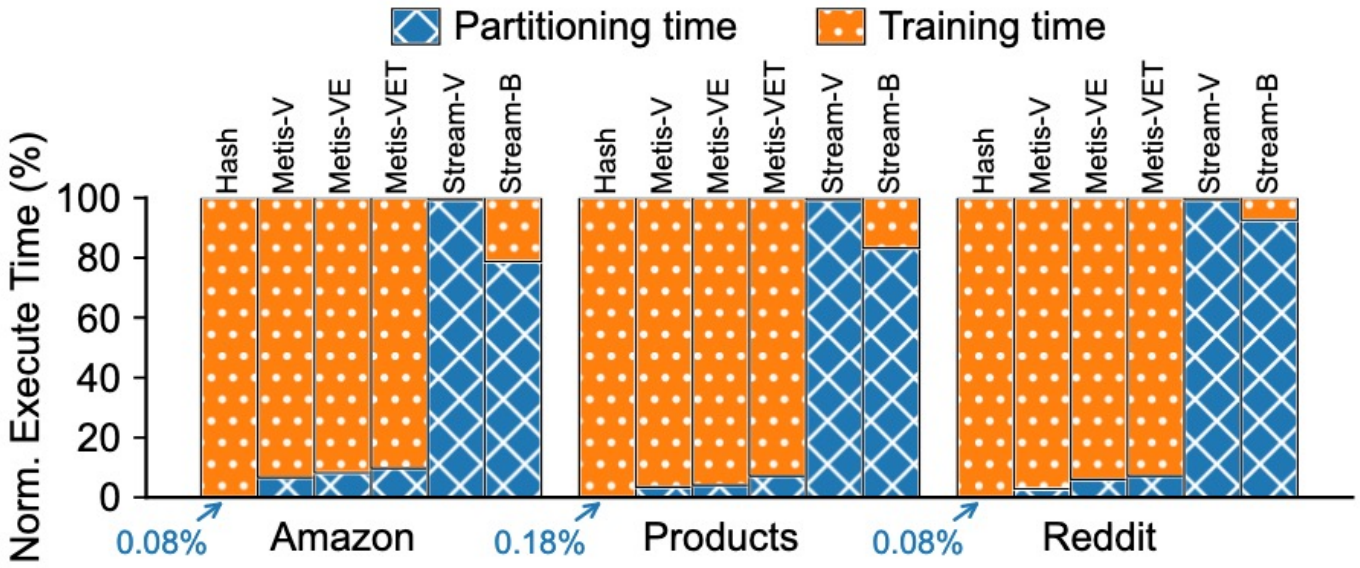
Subgraph Extraction: Takeaways

Partition	Generation	Slicing
G3 [SIGMOD'23]	SUREL [VLDB'23] GENTI [VLDB'24]	TIGER [VLDB'24]
◦ 3 Parallel enhancements ◦ Communication overhead persist under full-graph training	◦ Generated subgraphs may be more expressive ◦ Generation and gathering overhead	◦ On-disk subgraph extraction ◦ 2-stage slicing and caching for batter local access

Subgraph Extraction: Evaluation

Model Operations

- **Hash**: constantly fastest & insignificant
- **Metis family**: fast but less scalable
- **Streaming family**: slow & less scalable

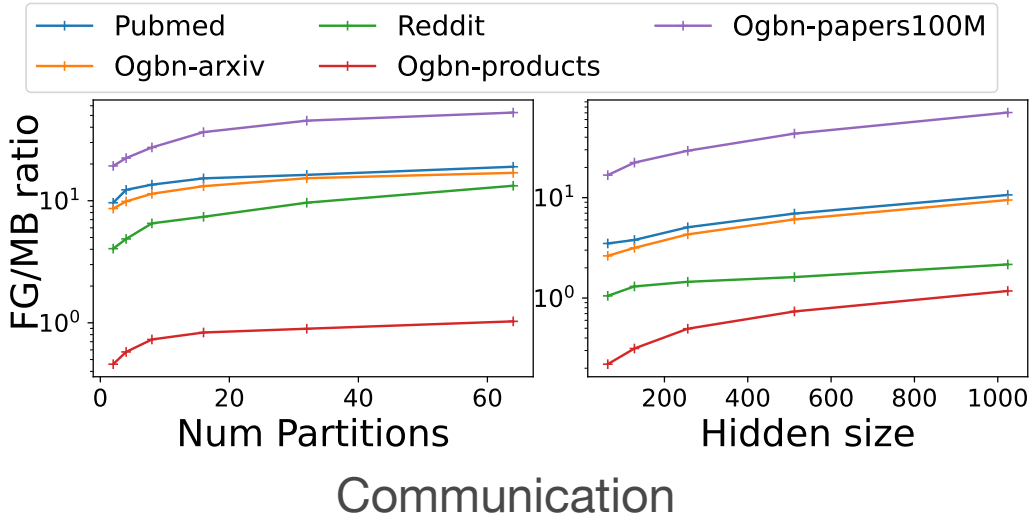


H Yuan et al, "Comprehensive Evaluation of GNN Training Systems: A Data Management Perspective", *VLDB'24.*

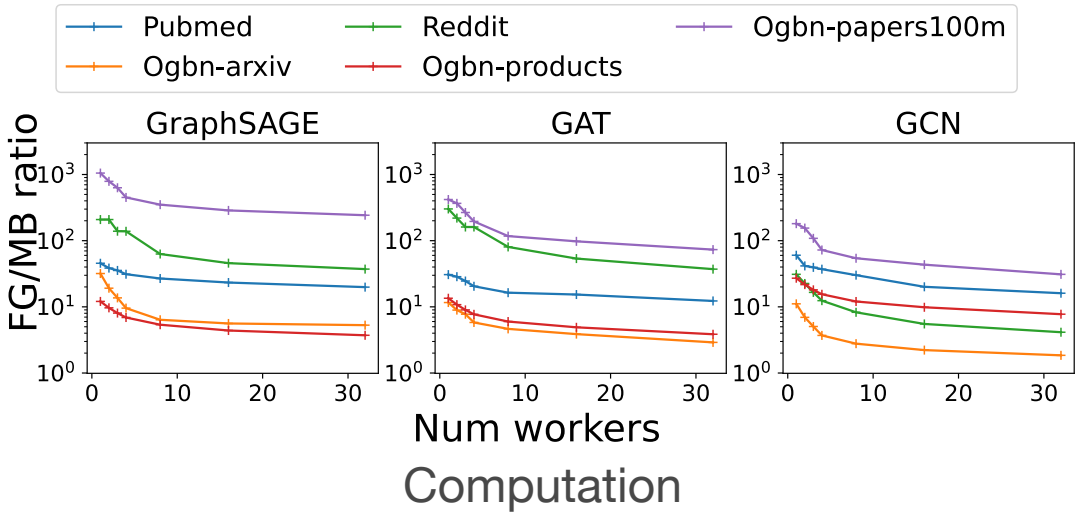
Subgraph Extraction: Evaluation

Training Schemes

- Mini-batch training is more scalable than full-graph training
- Mini-batch + local sampling is effective in reducing computation



Communication



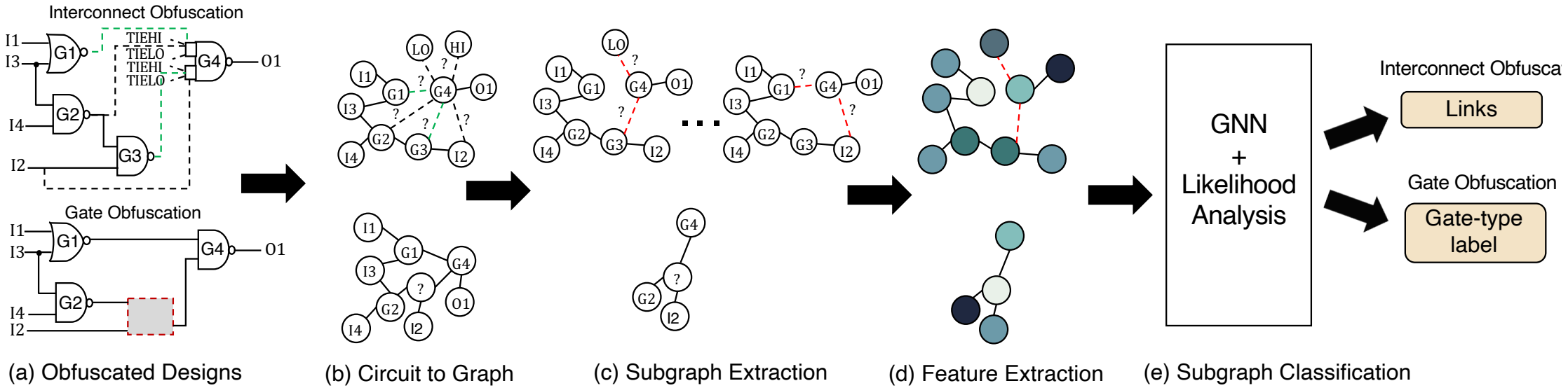
Computation

S Bajaj et al, "Graph Neural Network Training Systems: A Performance Comparison of Full-Graph and Mini-Batch", *VLDB'25*.

Subgraph Extraction: Applications

Hardware Obfuscation Analysis

- **Graph**: integrated circuits logic gates
- **Task**: identify obfuscated gates & interconnections



L Mankali et al, "Titan: Security Analysis of Large-Scale Hardware Obfuscation Using Graph Neural Networks", *TIFS'22*.

Outline

30 min	**Q&A**
10:30 – 11:00	Coffee Breaks
25 min	**Node-wise Similarity**
11:00 – 11:25	Siqiang Luo
20 min	**Spectral Embedding**
11:25 – 11:45	Ningyi Liao
20 min	**Graph Simplification**
11:45 – 12:05	Ningyi Liao
25 min	**Future Direction:** Graph+LM | Specific Data | Graph System
12:05 – 12:30	Reynold Cheng

Node-wise Similarity

Outline

25 min
11:00 – 11:25

Node-wise Similarity

Siqiang Luo

Personalized PageRank GBP [Chen et al | NeurIPS'20] & SCARA [Liao et al | VLDB'22]

Pair-wise Similarity SIGMA [H Liu et al | ICDE'25]

Hub Labeling CFGNN [Preprint] | HubGT [Preprint]

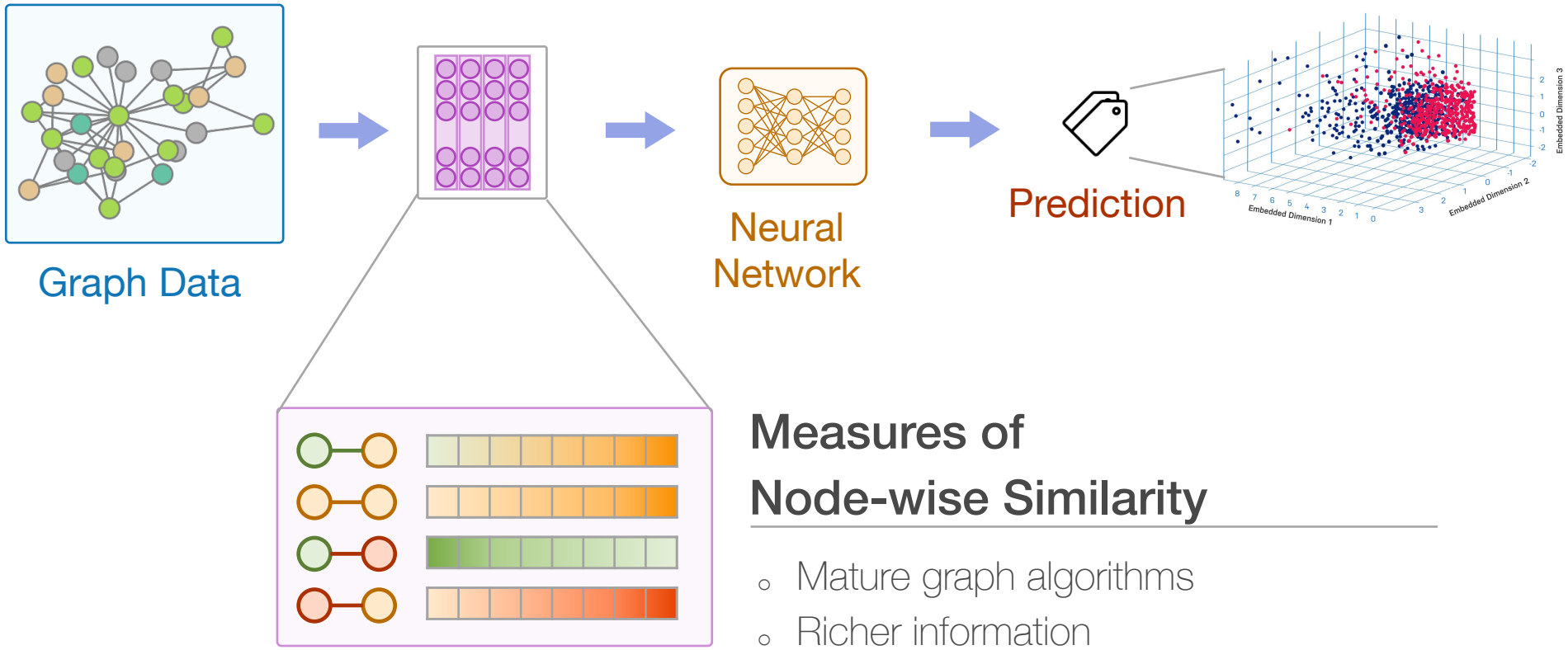
***Application*:** Recommendation Popularity

Spectral Embedding

Graph Simplification

Future Direction

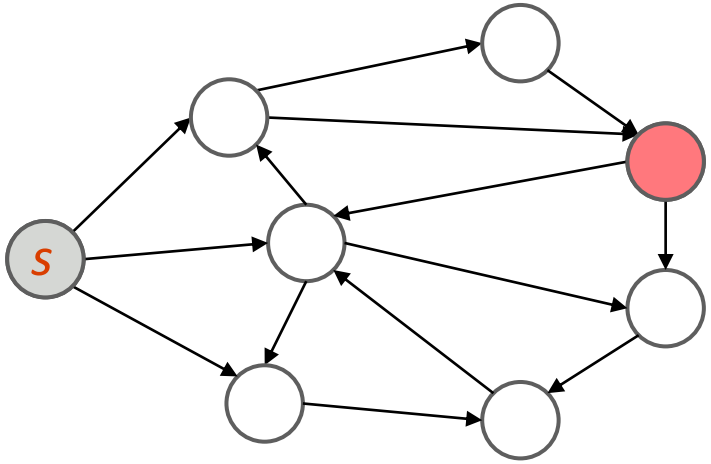
Node Similarity: Schema

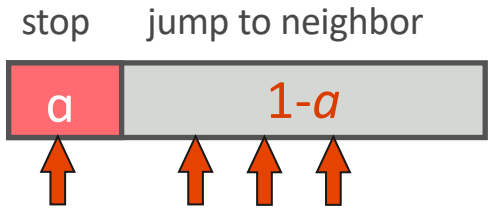


Measures of Node-wise Similarity

- Mature graph algorithms
- Richer information

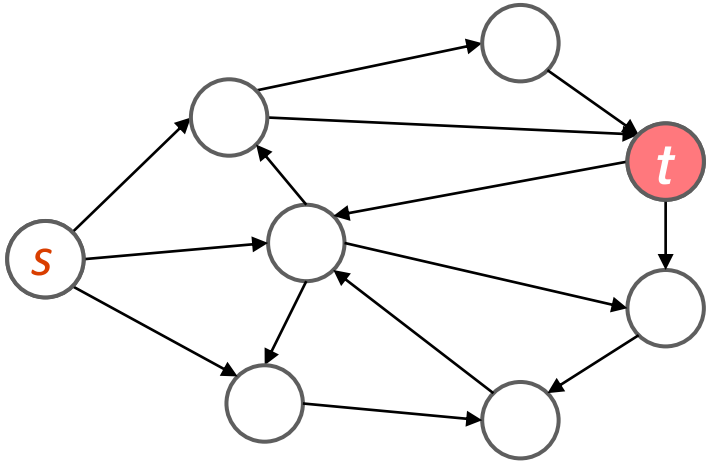
Personalized PageRank (PPR)

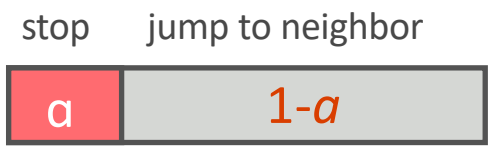




- A *α-decay* random walk:
  - Stop with *α* probability at each step
  - If not stop, randomly jumps to one of its out-neighbors

Personalized PageRank (PPR)

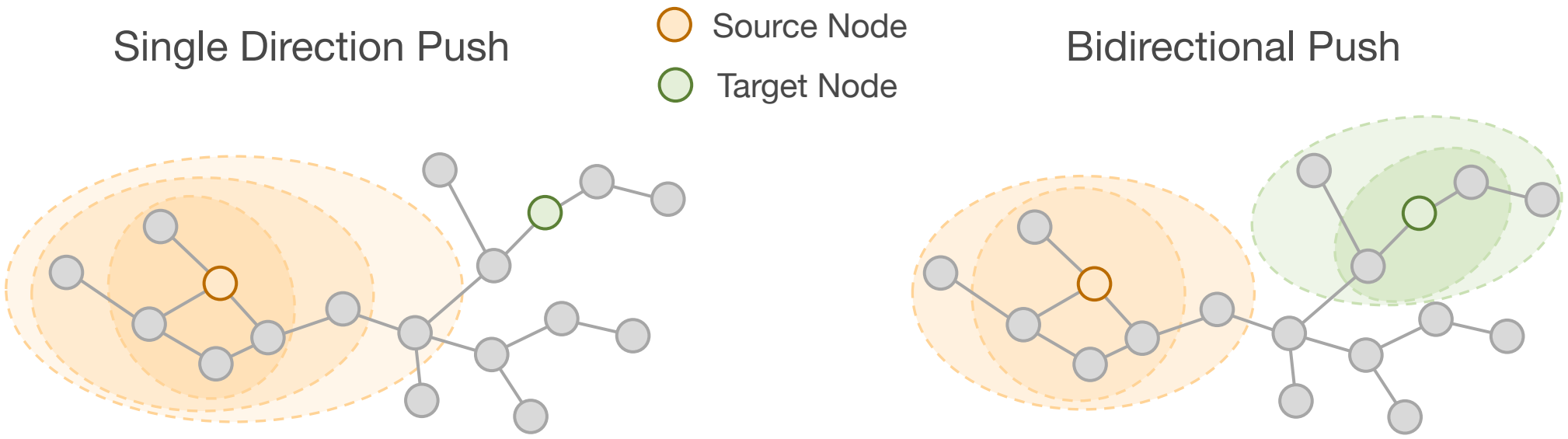




- Random walk interpretation of PPR(s, t):
  - probability that a random walk from s stops at t

Personalized PageRank (PPR)

- Bidirectional push for faster node-wise computation



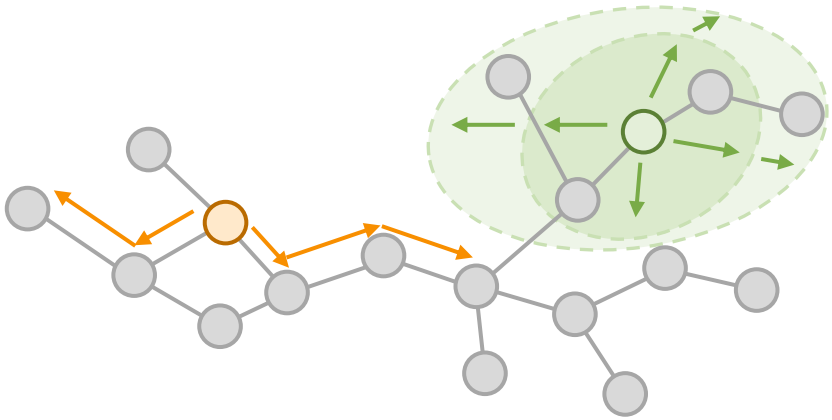
Computation: $\sim m$ $\sim\sqrt{m}$

P Lofgren et al, "Personalized PageRank Estimation and Search: A Bidirectional Approach", *WSDM'16*.

PPR: GBP [NeurIPS'20]

- Bidirectional propagation
  - **Forward**: Random Walk from ego nodes
  - **Backward**: Reverse Push from feature vectors
- Decoupled propagation and network
- Reduced propagation complexity

$$O\left(L\sqrt{Lm\log(Ln)}/\varepsilon\right)$$



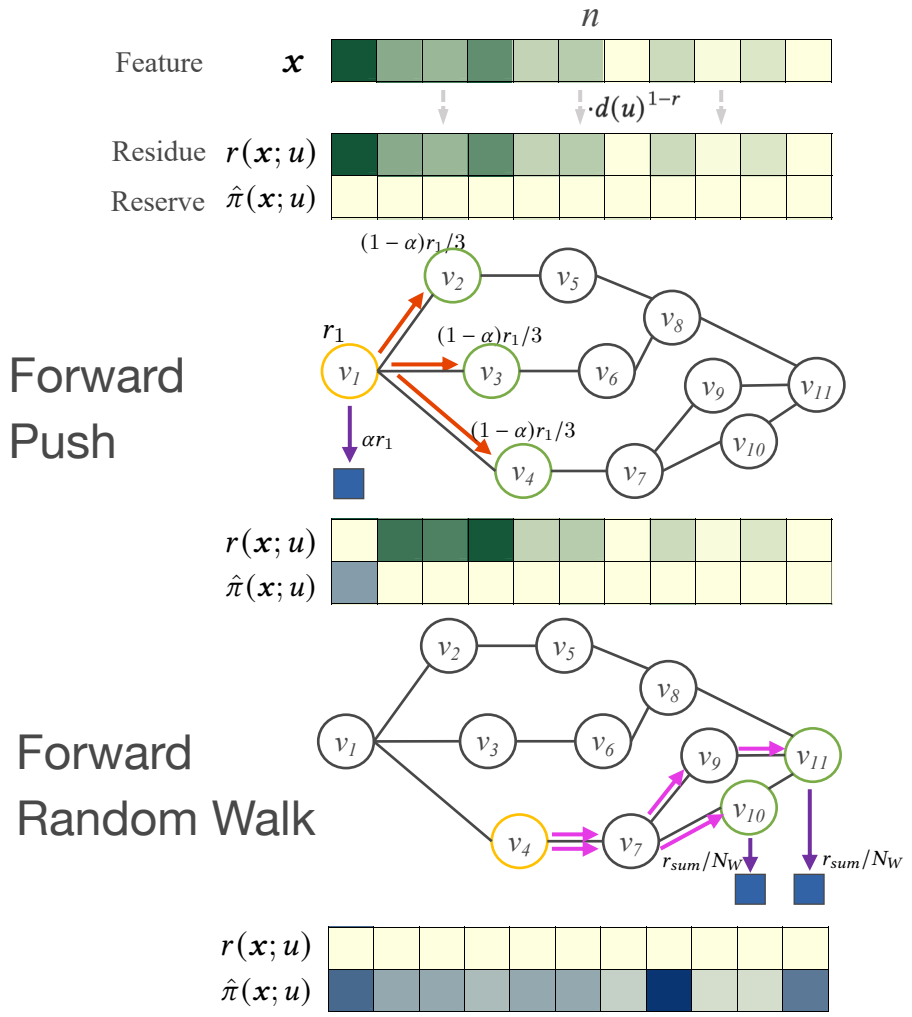
Forward: Random Walk

Backward: Reverse Push

PPR: SCARA [VLDB'22]

- Feature-oriented: start from each node attribute vector
- Pure forward scheme
  - **Insignificant push**: control by threshold
  - **Significant push**: compensate by random walk
- Reduced propagation complexity

$$O\left(\sqrt{m\log(n)}/\lambda\right)$$

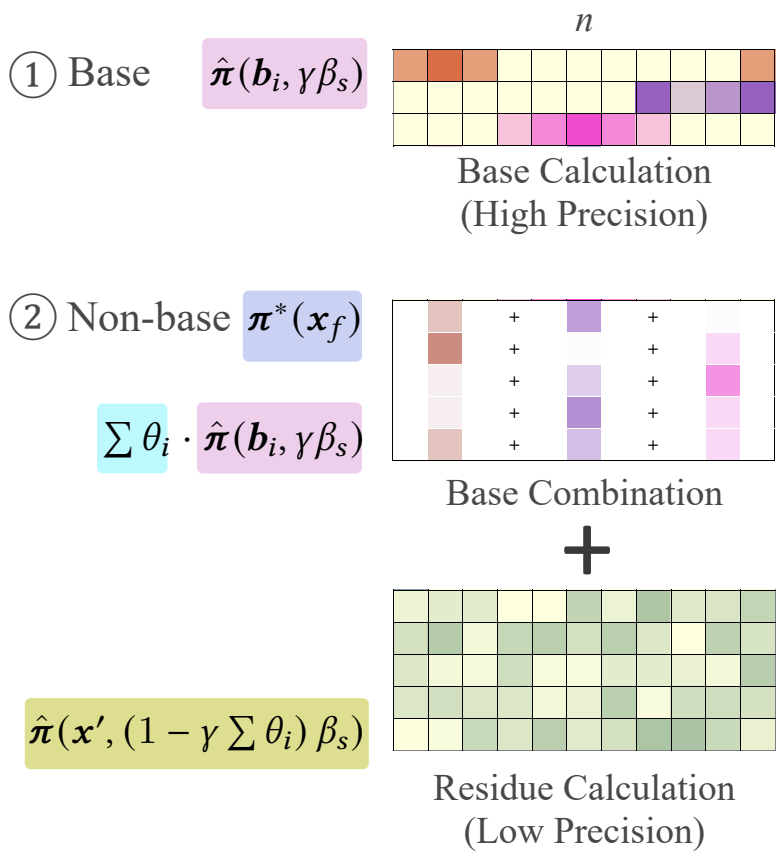


N Liao et al, "SCARA: Scalable Graph Neural Networks with Feature-Oriented Optimization", *VLDB'22*.
N Liao et al, "Scalable Decoupling Graph Neural Networks with Feature-Oriented Optimization", *VLDBJ'23*.

PPR: SCARA [VLDB'22]

- **Conventional PPR**: each node possesses a scalar value
- **PPR in GNN**: each node possesses a vector (feature vector)
- There are overlaps among feature dimensions

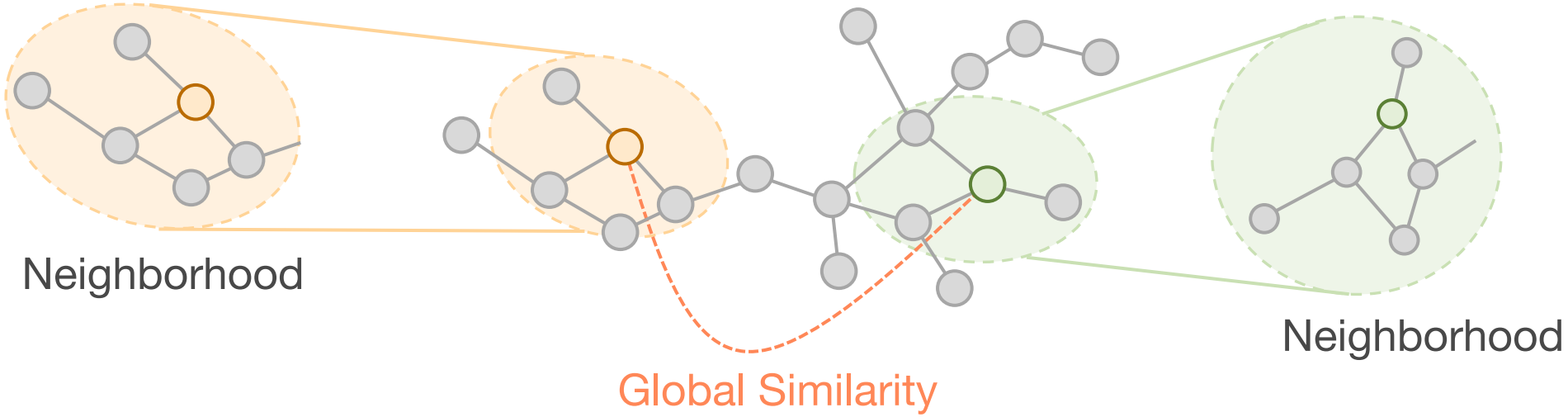
We can reuse PPR calculations among feature dimensions!



N Liao et al, "SCARA: Scalable Graph Neural Networks with Feature-Oriented Optimization", *VLDB'22*.
N Liao et al, "Scalable Decoupling Graph Neural Networks with Feature-Oriented Optimization", *VLDBJ'23*.

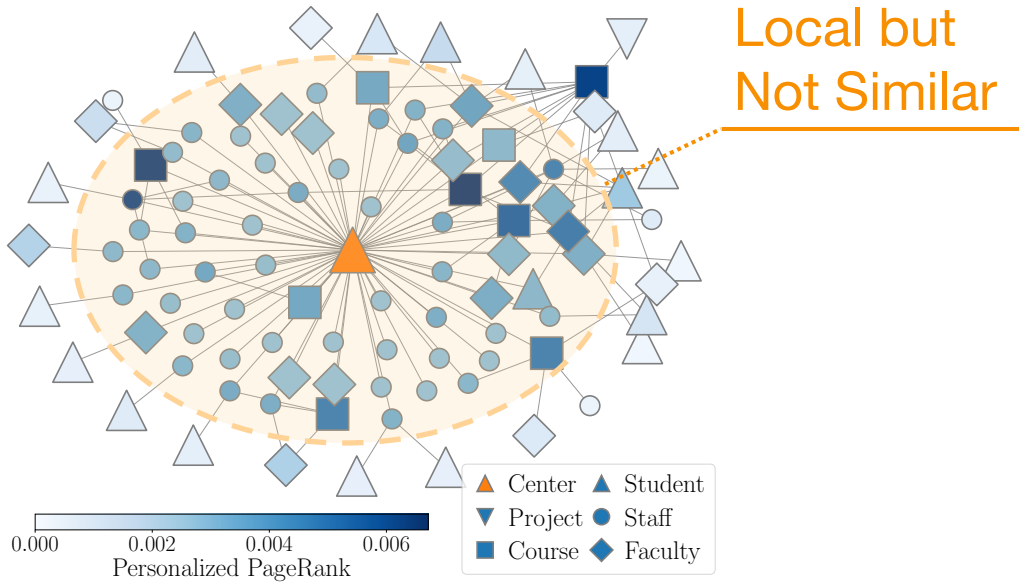
Pair-wise Similarity

- SimRank: a measure of node similarity based on neighborhood topology
- Global Similarity: distant node pair sharing the similar topology

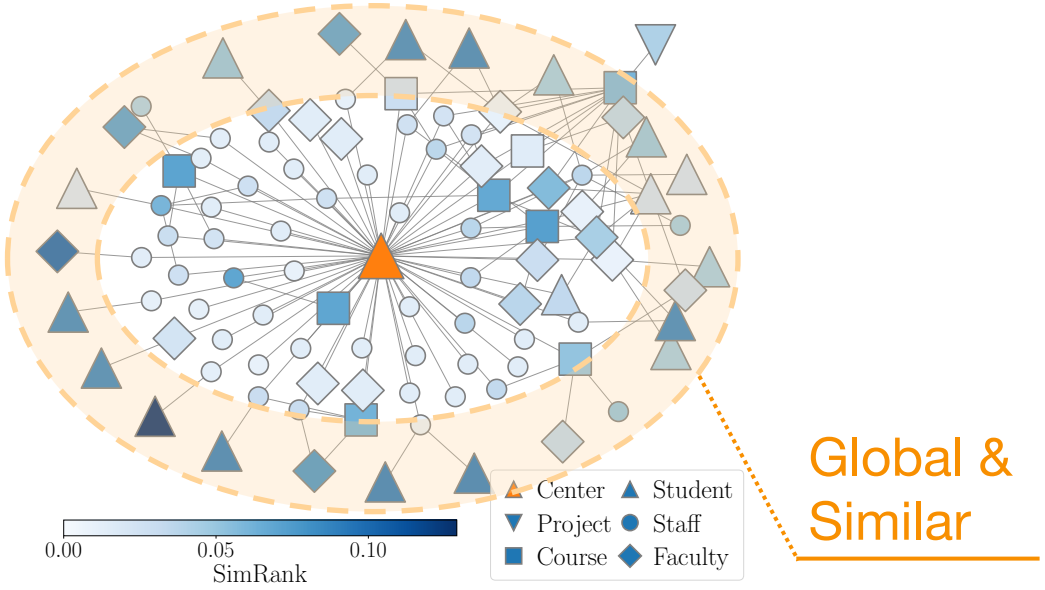


Pair-wise Similarity: SIGMA [ICDE'25]

- Global similarity: address heterophily by graph metric



PPR



SimRank

H Liu et al, "SIGMA: An Efficient Heterophilous Graph Neural Network with Fast Global Aggregation", *ICDE'25*.

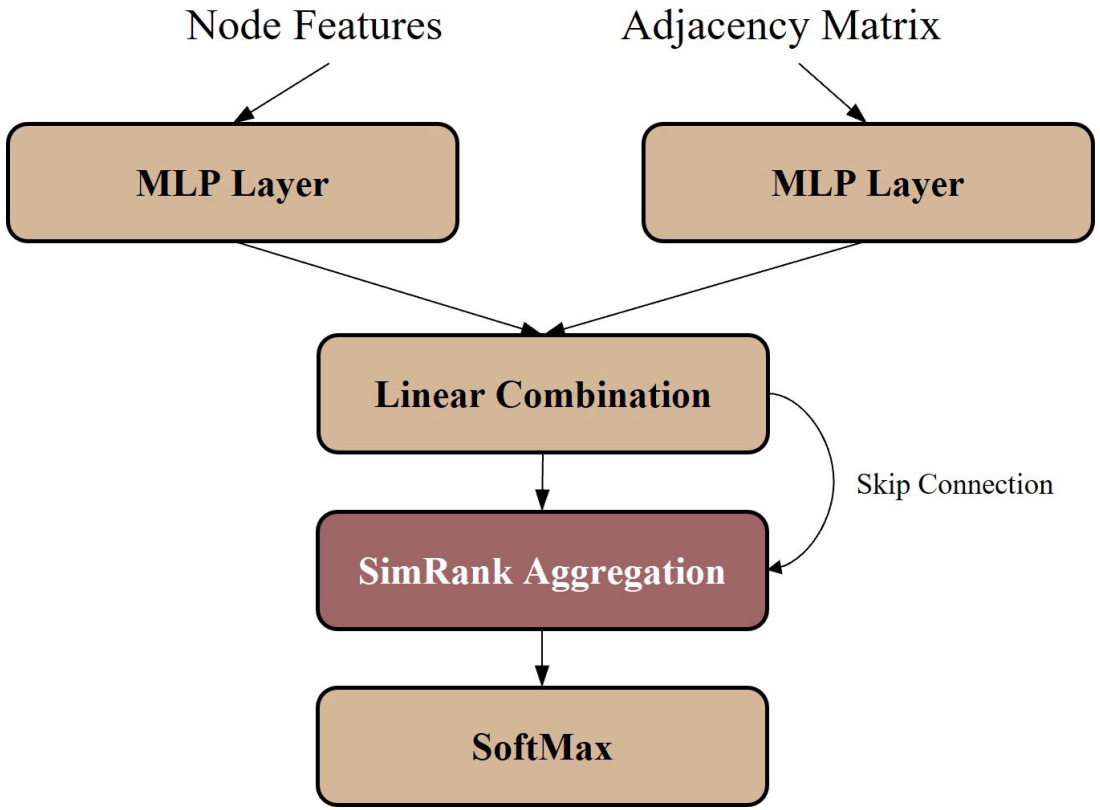
Pair-wise Similarity: SIGMA [ICDE'25]

- Combining attribute and topological representations:

$$\mathbf{H} = \mathrm{MLP}_H\left(\delta \cdot \mathbf{H_X} + (1-\delta) \cdot \mathbf{H_A}\right).$$

- Combining local and global representations:

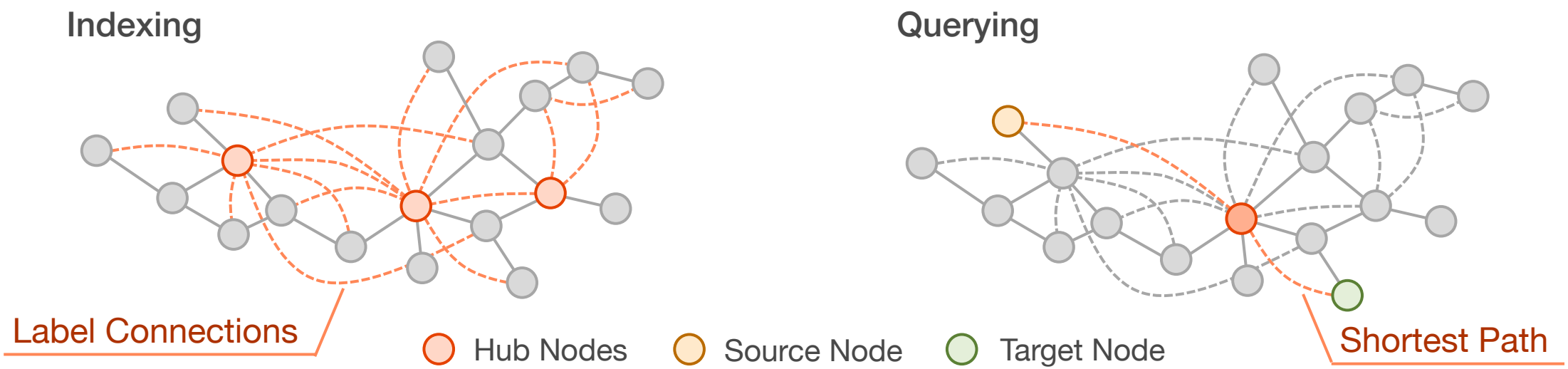
$$\mathtt{AGG}: \ \widehat{\mathbf{Z}}_u = \sum_{v \in V} \mathbf{S}(u,v) \cdot \mathbf{H}_v,$$
$$\mathtt{UPD}: \ \mathbf{Z_u} = (1-\alpha) \cdot \widehat{\mathbf{Z}}_u + \alpha \cdot \mathbf{H}_u,$$



H Liu et al, "SIGMA: An Efficient Heterophilous Graph Neural Network with Fast Global Aggregation", *ICDE'25*.

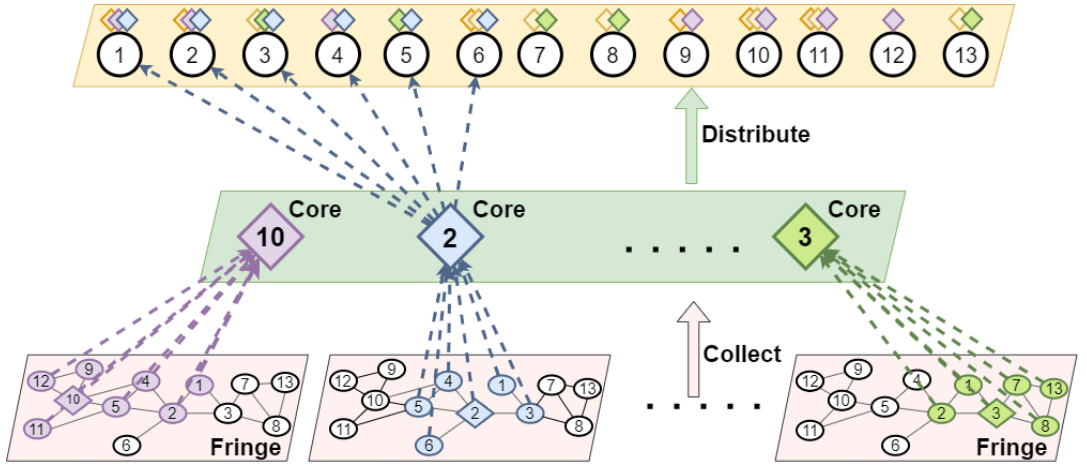
Hub Labeling

- Hub Labeling: adding new links to selected hub nodes, so that distant node pairs can be easily reached through hubs
- Shortest Path: reachability and distance can be computed through labels



Hub Labeling: CFGNN [Preprint]

- Hub labeling: identify hub nodes as the center of node clusters
- Distinctive message-passing:
  - Fringe nodes -> Hub nodes: collect
  - Hub nodes -> Fringe nodes: distribute



P Ip et al, "Bridging Indexing Structure and Graph Learning: Expressive and Scalable Graph Neural Network via Core-Fringe", *2023*.

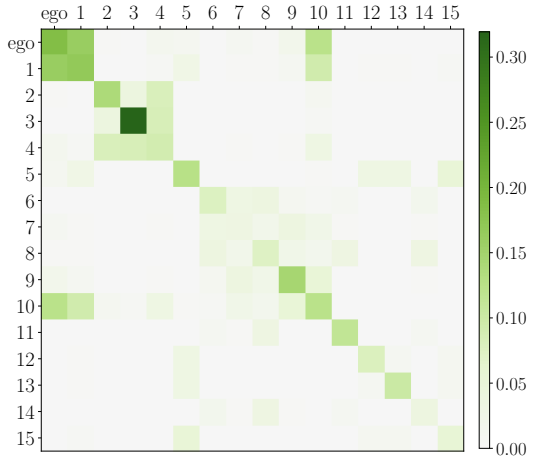
Hub Labeling: HubGT [Preprint]

- Hub Labeling: compute node-pair Shortest Path Distance (SPD)
- Positional Encoding: emphasize important pairs in Graph Transformer

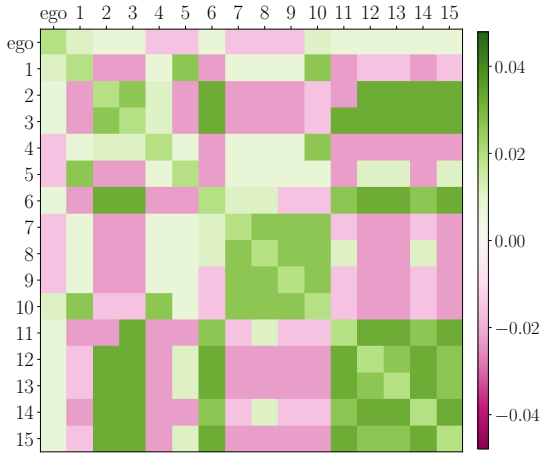
$$\tilde{\boldsymbol{H}}_i = \text{softmax}\left(\frac{\boldsymbol{Q}_i \boldsymbol{K}_i^\top}{\sqrt{F_K}} + \boldsymbol{P}\right)\boldsymbol{V}_i,$$

- Complexity: $O(n^2)$ (full graph)

  $O(s^2)$ (within subgraph)



Adjacency



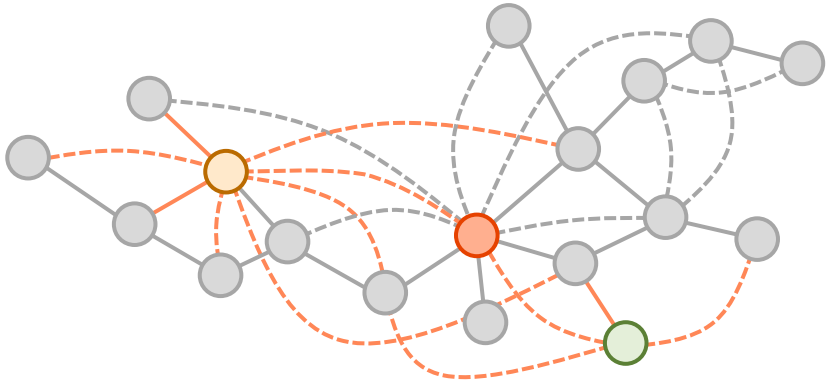
SPD Encoding

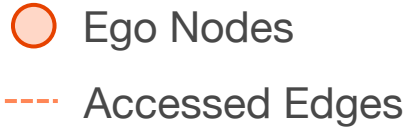
N Liao et al, "HubGT: Fast Graph Transformer with Decoupled Hierarchy Labeling", *arXiv:2412.04738, 2024*.

Hub Labeling: HubGT [Preprint]

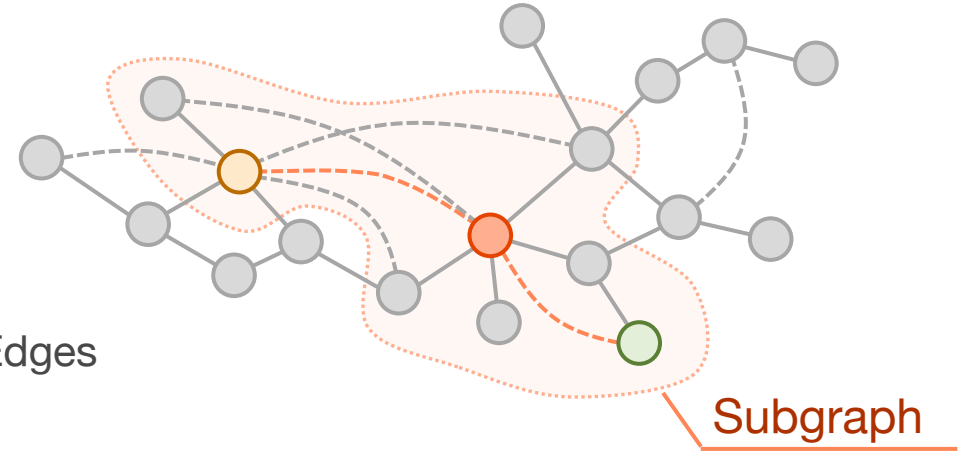
- Subgraph SPD: fast querying any node pairs within subgraph of a node

Conventional SPD





Subgraph SPD



	Conventional SPD	Subgraph SPD
Query Time	$O(s)$	$O(1)$
Index Size	$O(ns^2)$	$O(ns^2)$

N Liao et al, "HubGT: Fast Graph Transformer with Decoupled Hierarchy Labeling", *arXiv:2412.04738, 2024*.

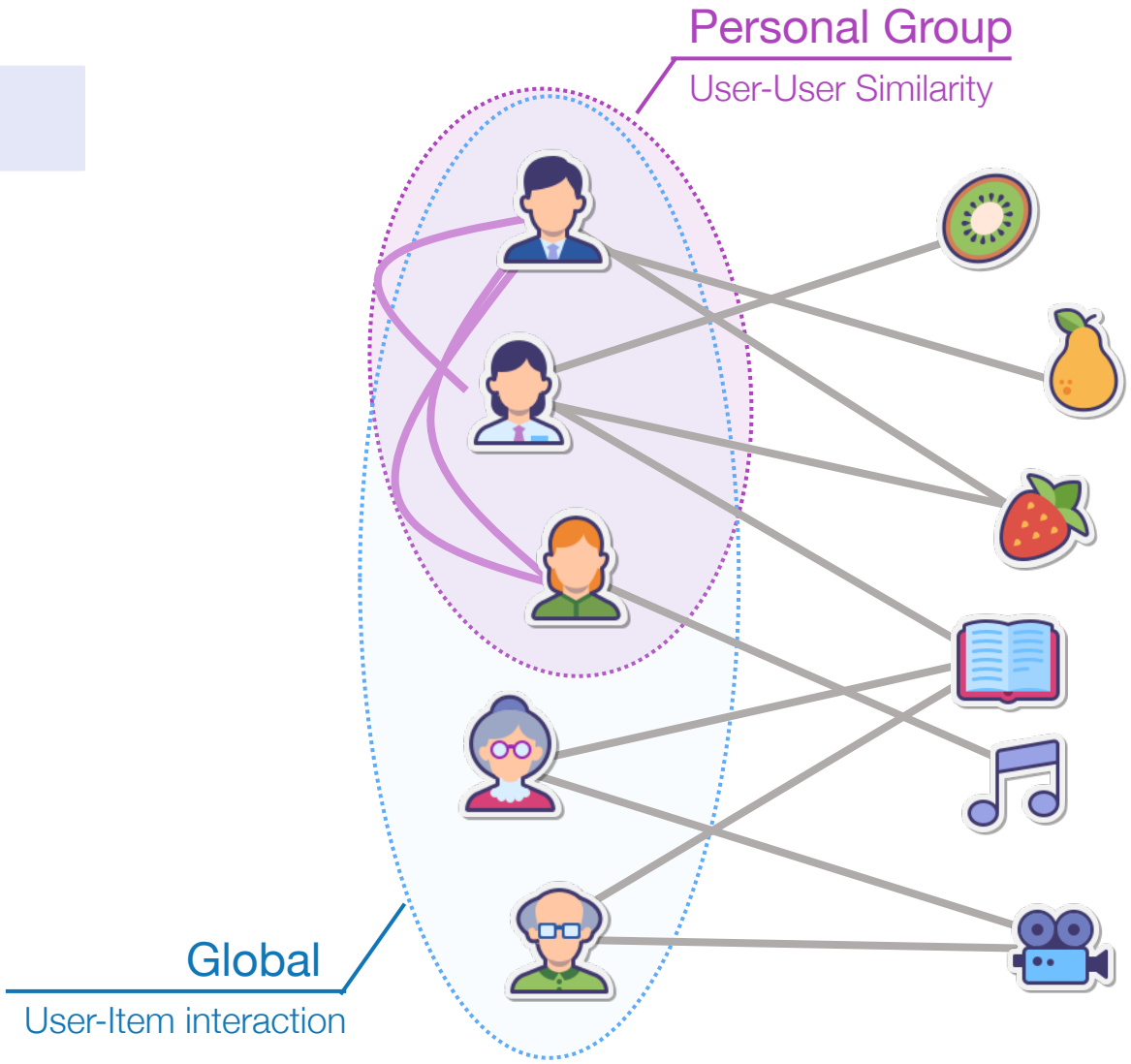
Node Similarity: Takeaways

PPR	SimRank	Hub Labeling
GBP [NeurIPS'20] SCARA [VLDB'22]	SIGMA [ICDE'25]	CFGNN [Preprint] HubGT [Preprint]
◦ Similar to graph convolution with faster computation ◦ Scalar ⟶ feature vector	◦ Global similarity between distant nodes ◦ Fast push computation	◦ Rewiring with core hub nodes ◦ Applicable to MPNN & GT

Node Similarity: Applications

Debiasing Recommendation

- **Graph**: user-item interaction
- **Task**: unbiased recommendation
- **Global popularity**: biased to top popular items
- **Personal popularity**: based on user similarity



W Ning et al, "Debiasing Recommendation with Personal Popularity", *WWW'24*.

Spectral Embedding

Outline

20 min
11:25 – 11:45

Node-wise Similarity

Spectral Embedding

Ningyi Liao

Channel Combination LD^2 [Liao et al | NeurIPS'23]

TER [Wang et al | KDD'23] & S3GCL [Wan et al | ICML'24]

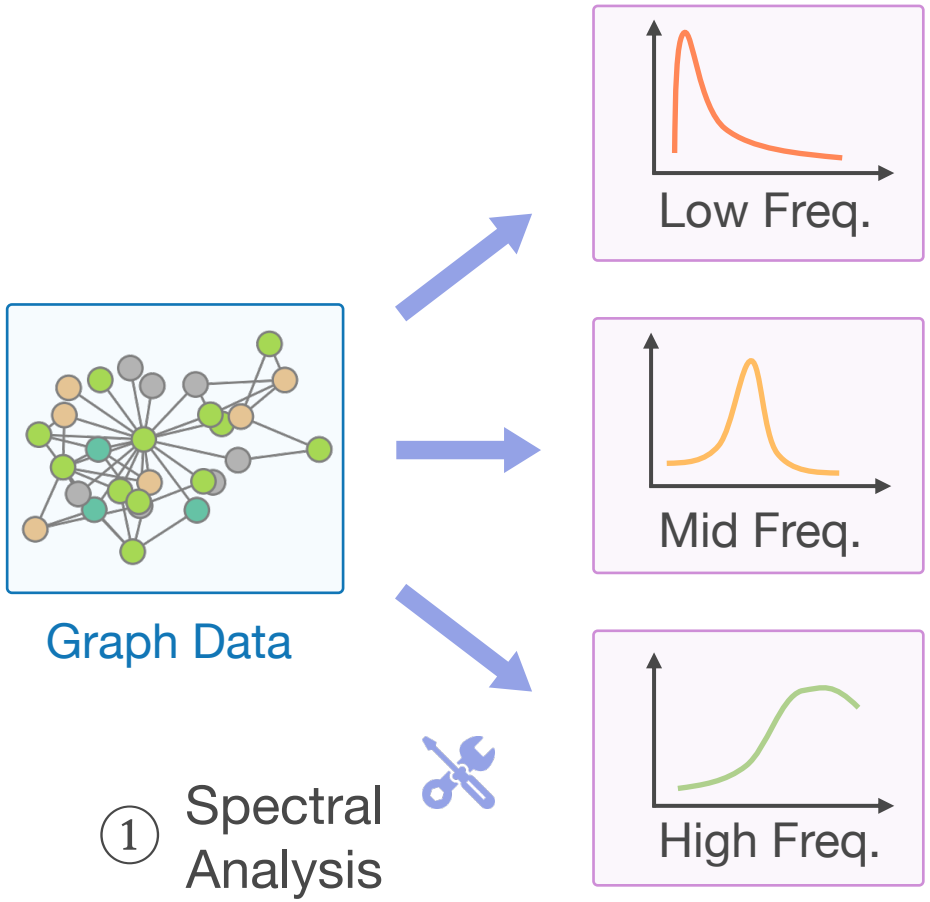
Adaptive Selection UniFilter & AdaptKry [Hunag et al | ICML & WWW'24]

***Application*: Image Filtering**

Graph Simplification

Future Direction

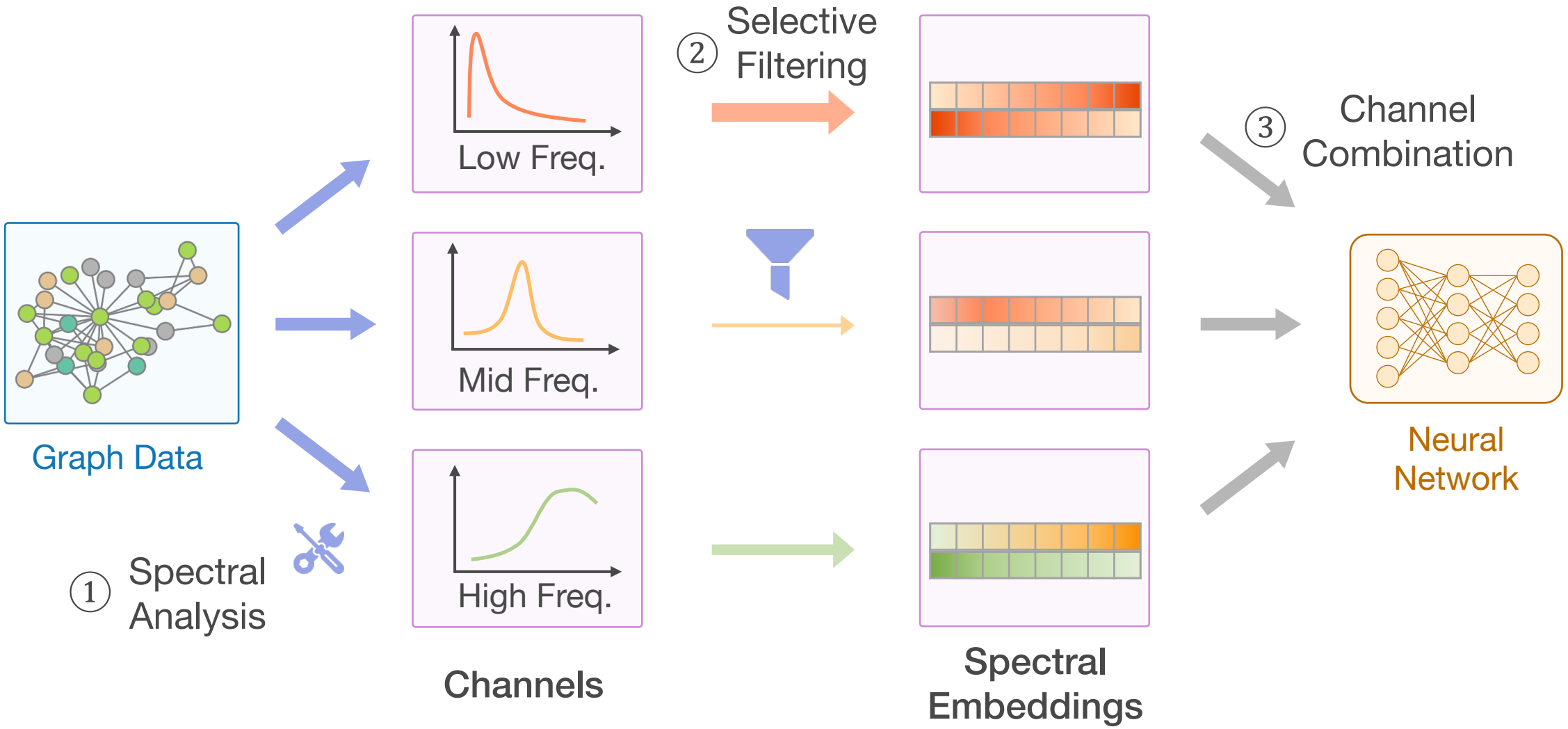
Spectral Embedding: Schema



Spectral Analysis

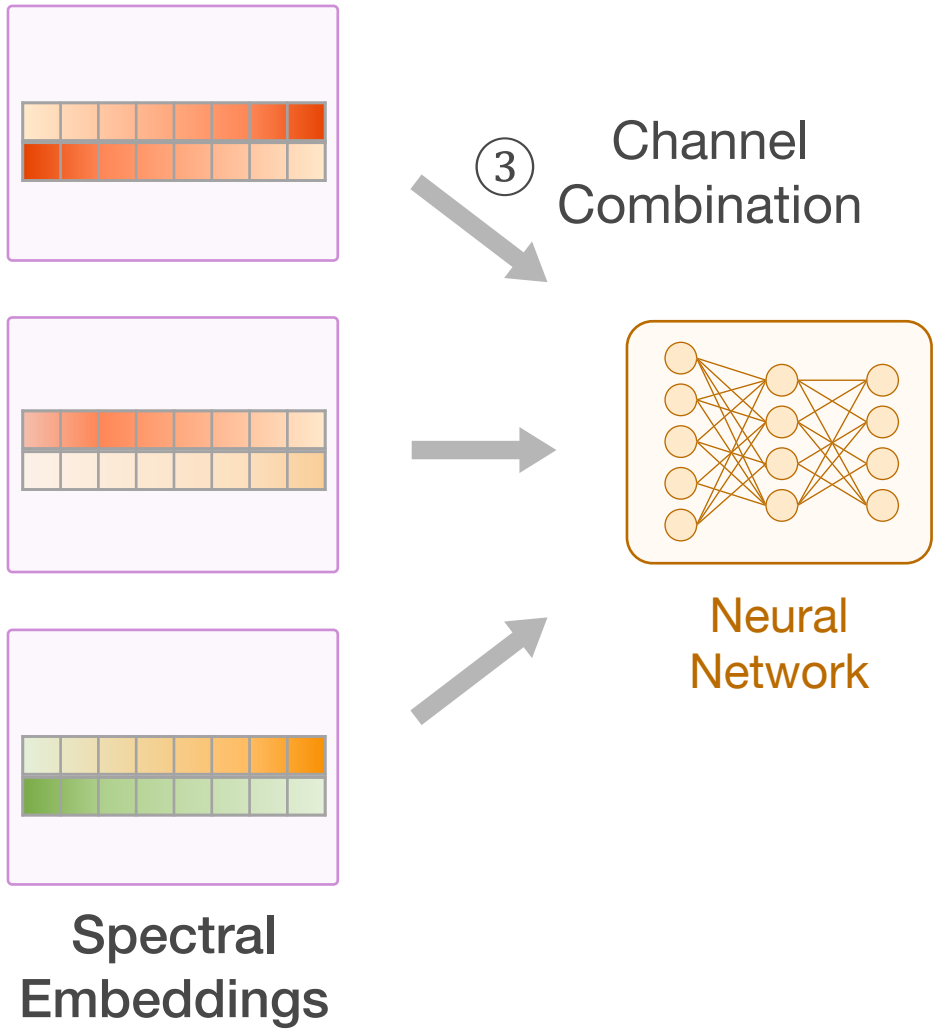
- Comprehensive graph information
- Explainable & controllable
- Separated graph computation

Spectral Embedding: Schema



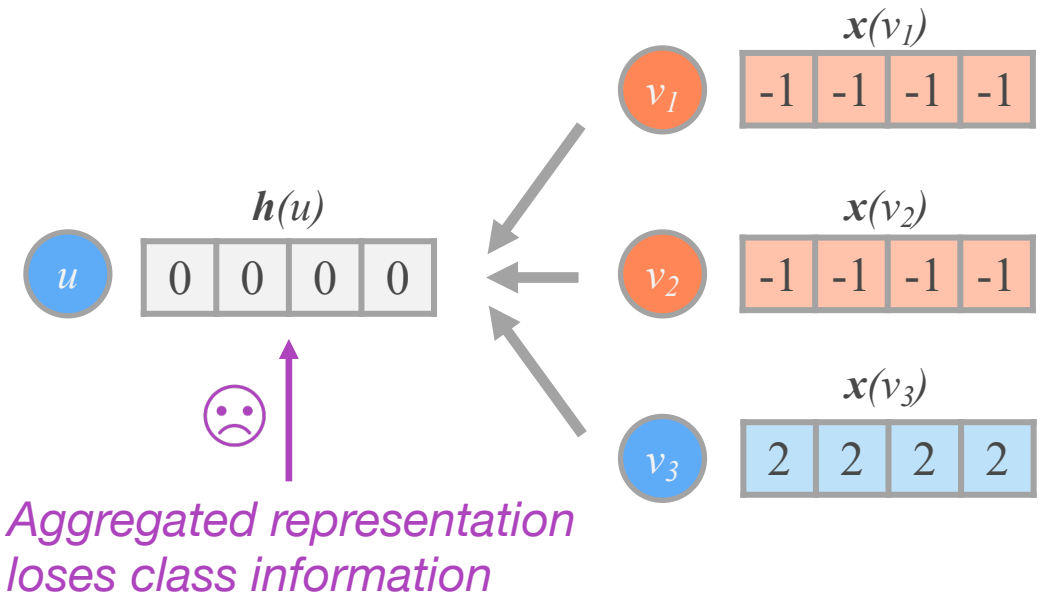
Channel Combination

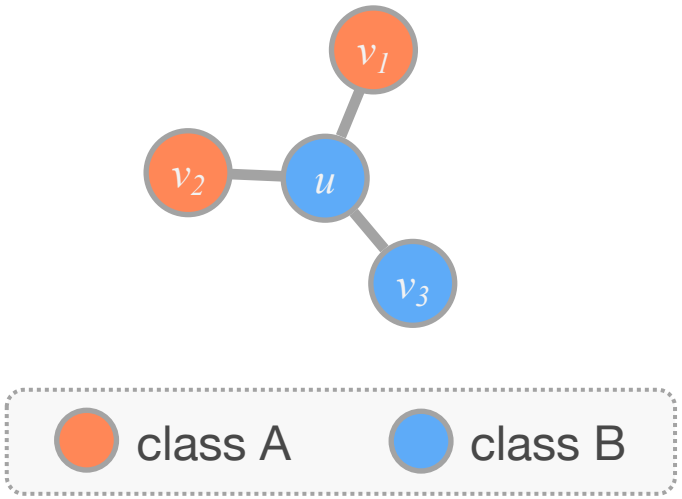
How to select & combine useful channels?



Channel Combination: LD^2 [NeurIPS'23]

- Heterophily: 1-hop message passing fails





N Liao et al, " LD²: Scalable Heterophilous Graph Neural Network with Decoupled Embedding", *NeurIPS'23*.

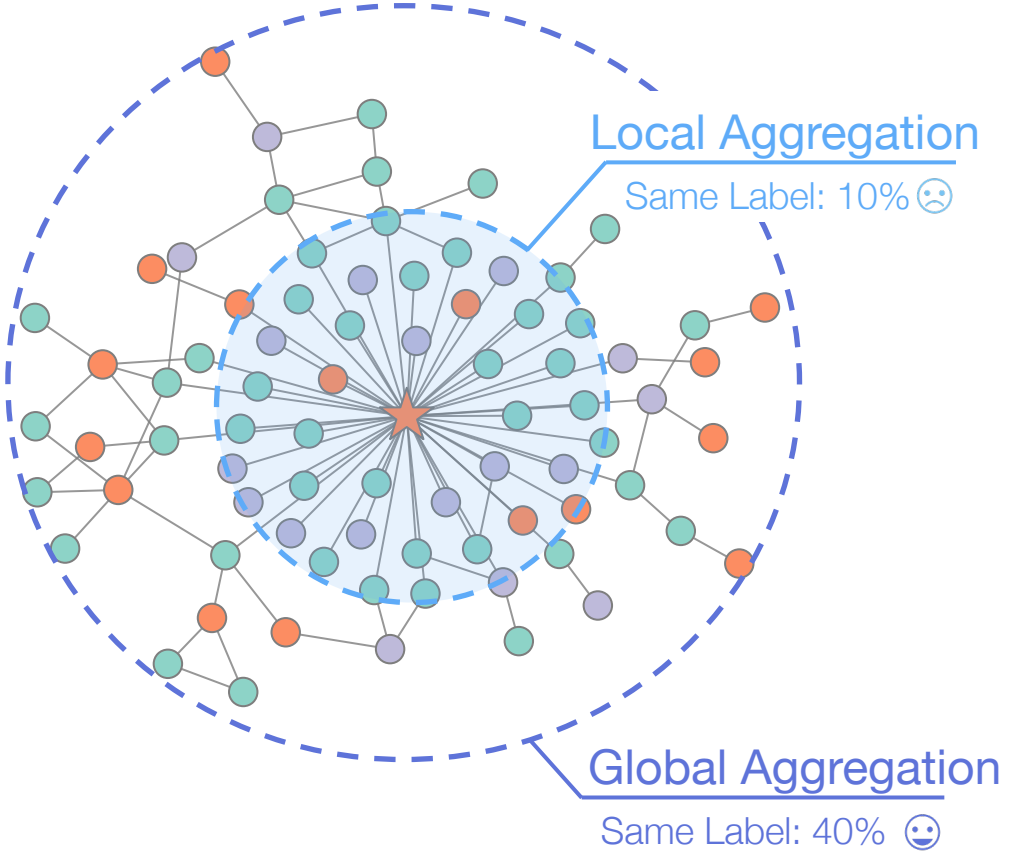
Channel Combination: LD^2 [NeurIPS'23]

- Heterophily: specifically focus 2-hop connections

2-hop Adjacency: $\boldsymbol{P}_A = \phi\left(\boldsymbol{A}^{2L} \cdot \boldsymbol{\mathcal{N}}\right)$

Constant Feature: $\boldsymbol{P}_{X,L2} = \sum_{l=1}^{L} \bar{\boldsymbol{A}}^{2l} \cdot \boldsymbol{X}$

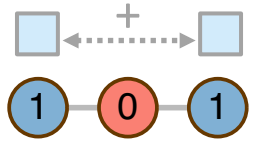
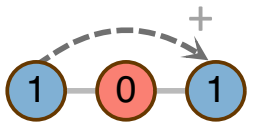
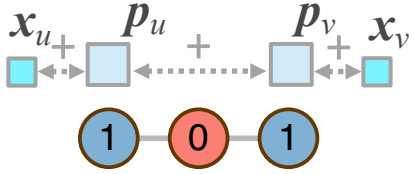
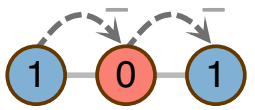
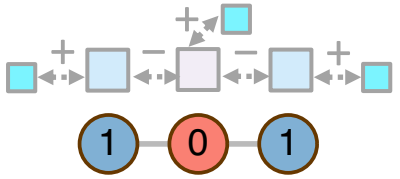
Inverse Feature: $\boldsymbol{P}_{X,H} = \sum_{l=1}^{L} \tilde{\boldsymbol{L}}^{l} \cdot \boldsymbol{X}$



N Liao et al, " LD²: Scalable Heterophilous Graph Neural Network with Decoupled Embedding", *NeurIPS'23*.

Spectral Embedding

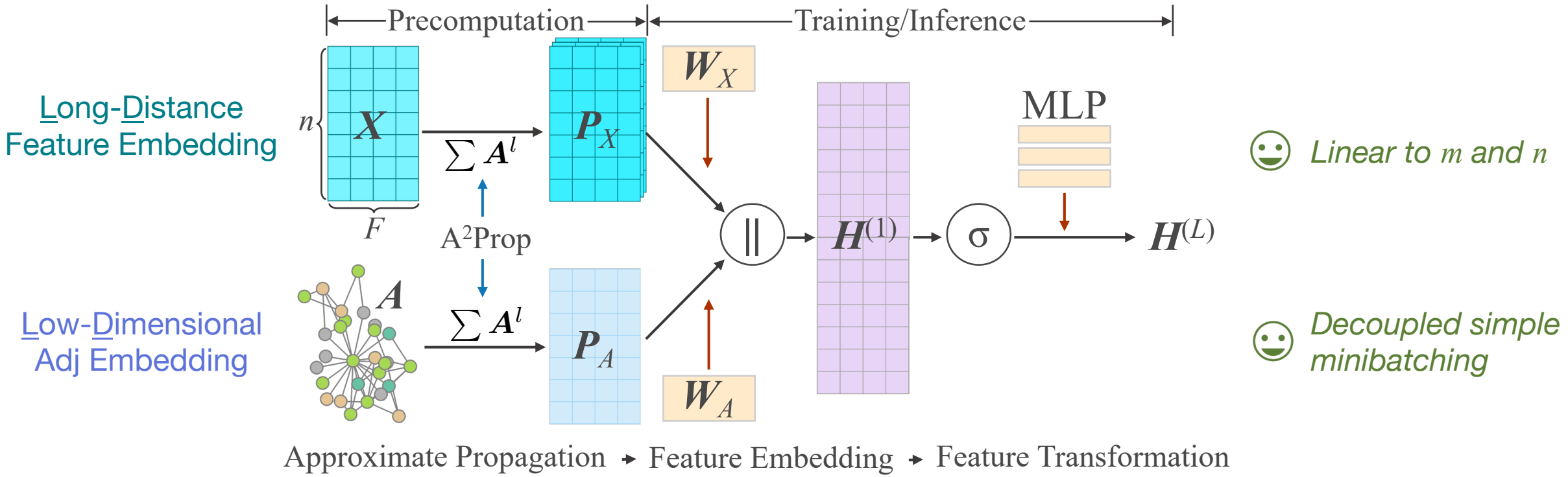
Channel Combination: LD^2 [NeurIPS'23]

		Spatial	Spectral
2-hop Adjacency:	$\boldsymbol{P}_A = \phi\left(\boldsymbol{A}^{2L} \cdot \boldsymbol{\mathcal{N}}\right)$	$\boldsymbol{P}_A \cdot \boldsymbol{P}_A^{\top} \approx \boldsymbol{A}^2$ Low-rank approximation	 Structure smoothing
Constant Feature:	$\boldsymbol{P}_{X,L2} = \sum_{l=1}^{L} \bar{\boldsymbol{A}}^{2l} \cdot \boldsymbol{X}$	 Long-range similarity	 Feature smoothing
Inverse Feature:	$\boldsymbol{P}_{X,H} = \sum_{l=1}^{L} \tilde{\boldsymbol{L}}^{l} \cdot \boldsymbol{X}$	 Fine-grained details	 Structure difference

N Liao et al, " LD²: Scalable Heterophilous Graph Neural Network with Decoupled Embedding", *NeurIPS'23*.

Channel Combination: LD^2 [NeurIPS'23]

- Decoupled spectral embedding



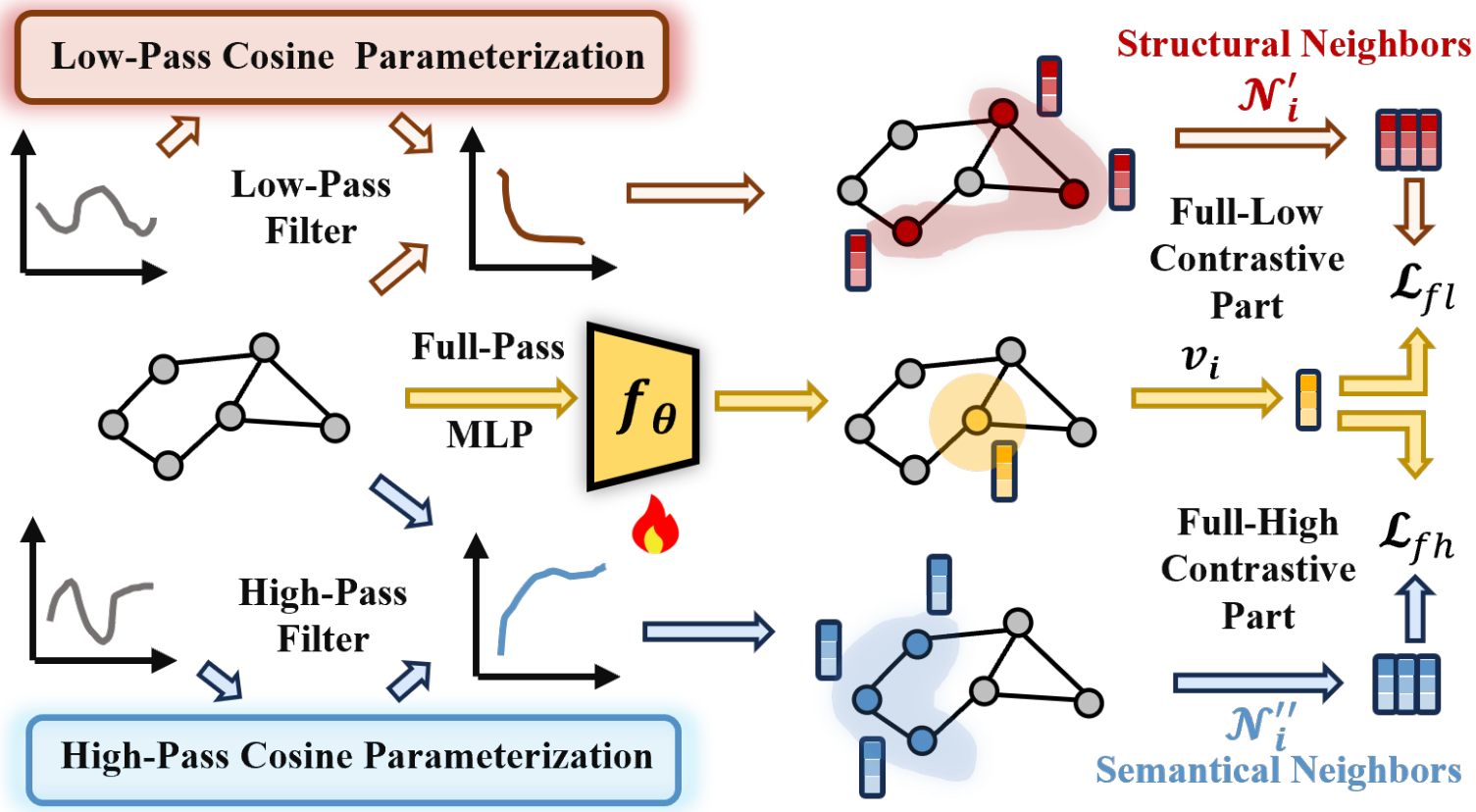
☺ *Linear to m and n*

☺ *Decoupled simple minibatching*

N Liao et al, " LD²: Scalable Heterophilous Graph Neural Network with Decoupled Embedding", *NeurIPS'23*.

Channel Combination: S3GCL [ICML'24]

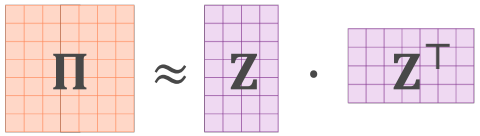
- Contrastive learning: difference between low and high frequency signals



G Wan et al, " S3GCL: Spectral, Swift, Spatial Graph Contrastive Learning", *ICML'24*.

Channel Combination: TER [KDD'23]

- Representation based on edge-wise proximity:



$$\min_{\mathbf{Z}\in\mathbb{R}^{m\times k}} \|\mathbf{\Pi} - \mathbf{Z}\mathbf{Z}^\top\|_F^2. \qquad \mathbf{\Pi} = \sum_{t=0}^{\infty} w(t)\cdot \mathbf{P}^t$$

Representation $m\times k$ — Proximity (⚠ Dense) — Edge Transition $m\times m$

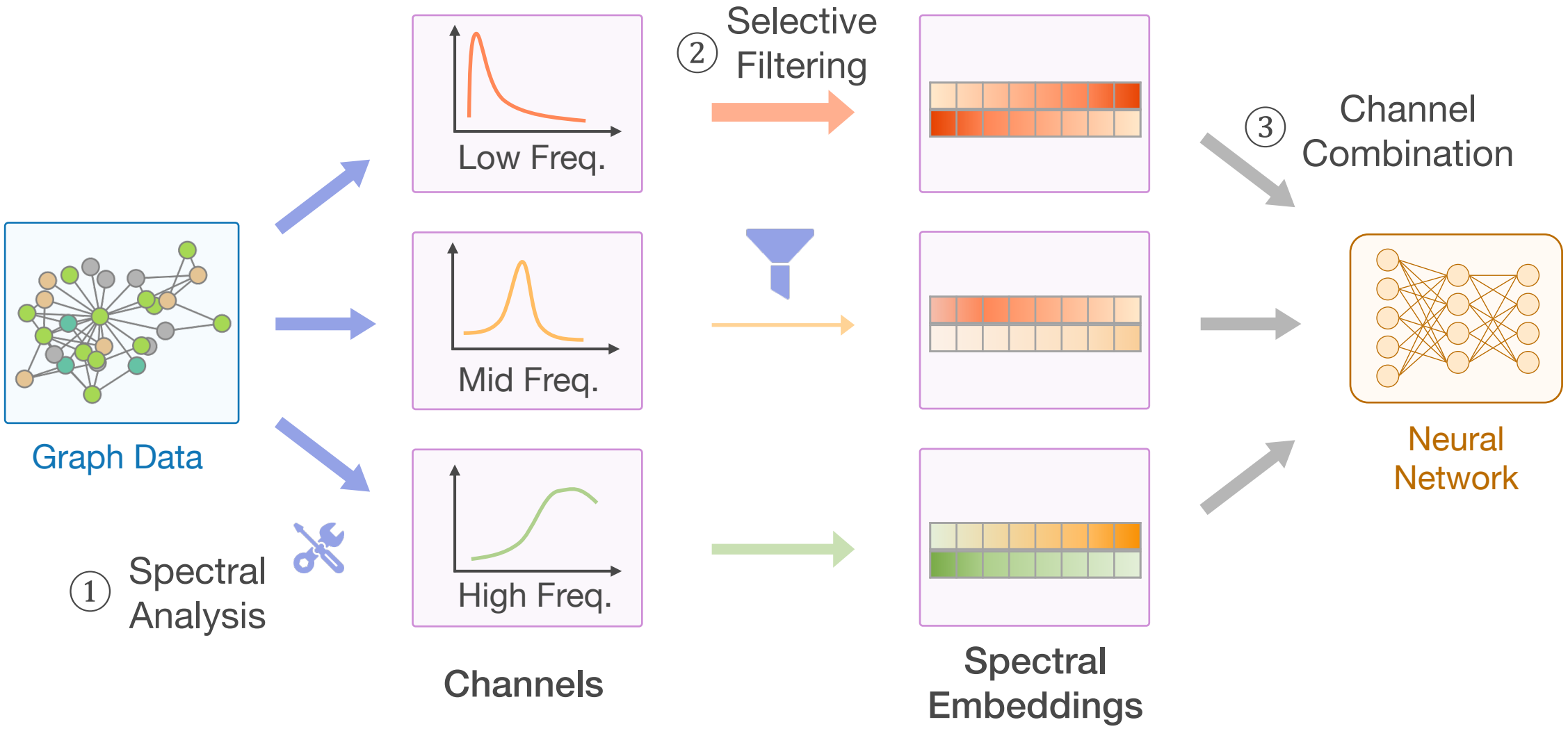
- Eigen-decomposition:

$$\mathbf{Z} = \mathbf{U}\sqrt{\sum_{t=0}^{\infty} w(t)\mathbf{\Lambda}^t} \qquad \boldsymbol{P} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^\top$$

$$\mathbf{\Pi} = \mathbf{U}\cdot\left(\sum_{t=0}^{\infty} w(t)\mathbf{\Lambda}^t\right)\cdot\mathbf{U}^\top$$

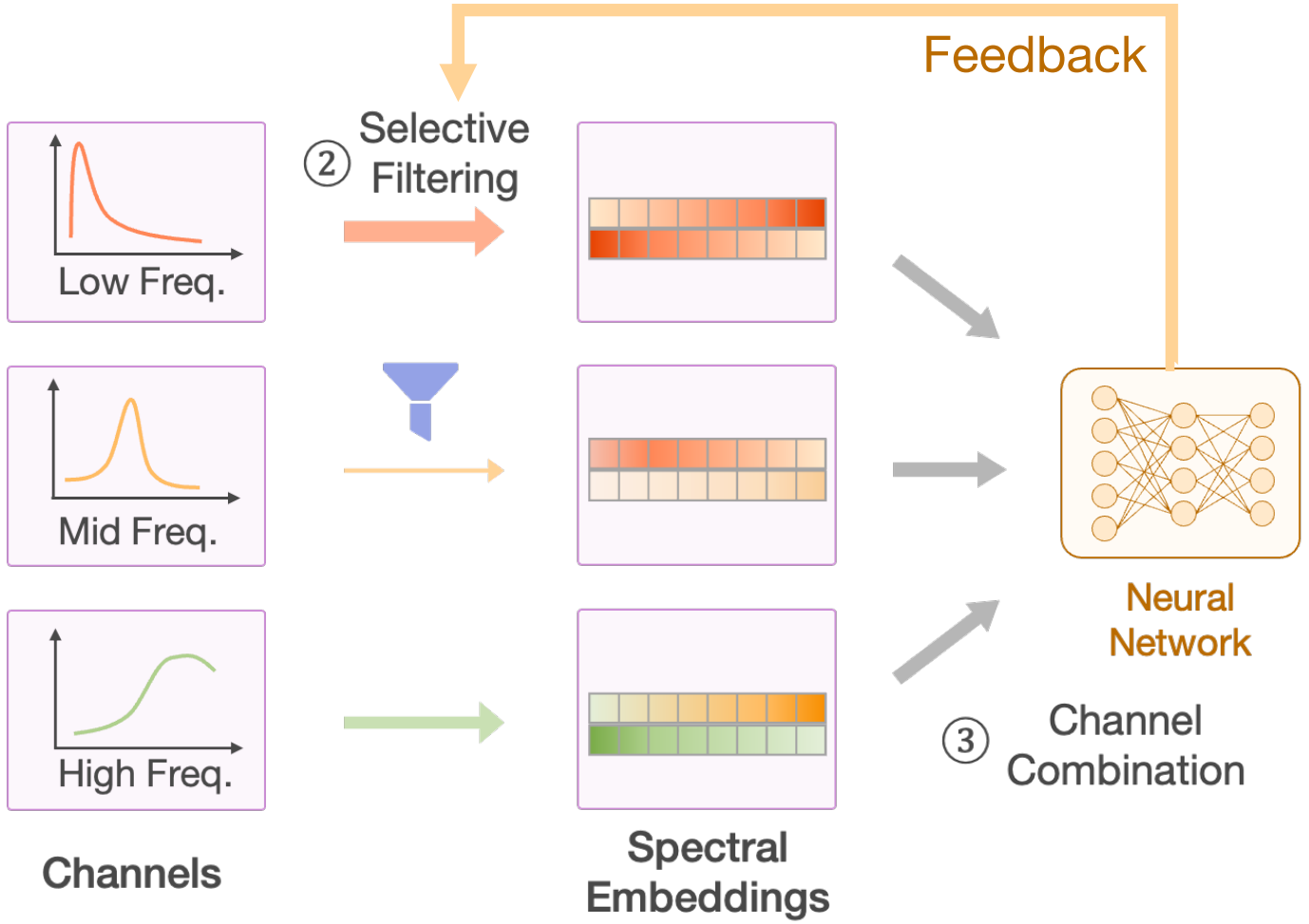
H Wang et al, " Efficient and Effective Edge-wise Graph Representation Learning", *KDD'23*.

Adaptive Selection



Adaptive Selection

How to filter based on learning feedback?



Adaptive Selection: UniFilter [ICML'24]

- Spatial computation

$$P = \sum_{l=0}^{L} W_l (A^l X + u_l)$$

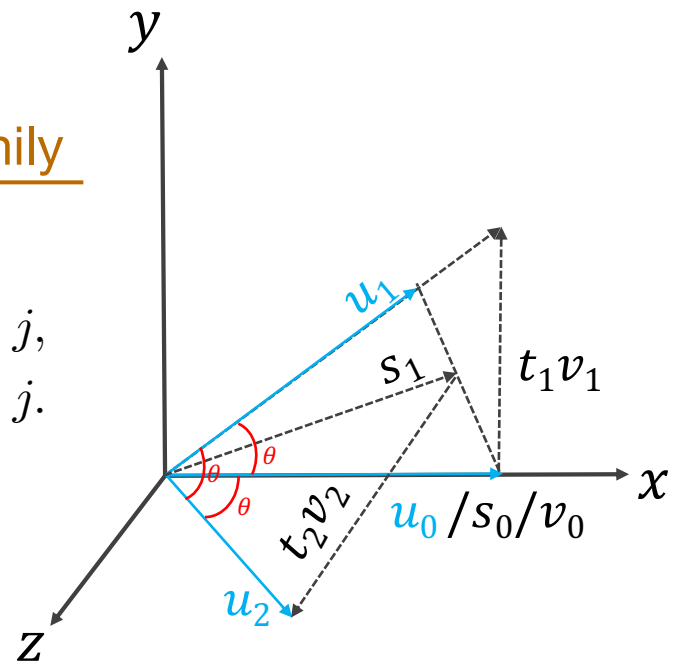
Heterophily Bias

Unique for each hop

Learnable Weight

$$\mathbf{u}_i \cdot \mathbf{u}_j = \begin{cases} \cos(\frac{(1-h)\pi}{2}) & if\ i \neq j, \\ 1 & if\ i = j. \end{cases}$$

Graph Heterophily

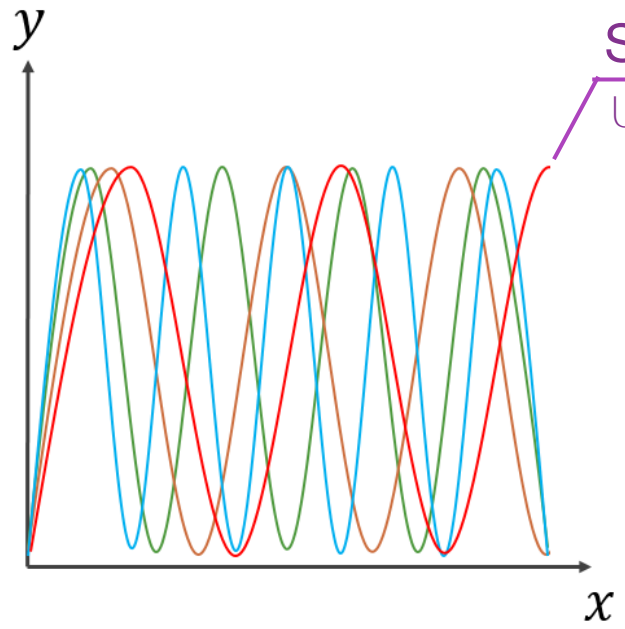


Spatial Domain (Embedding Space)

K Huang et al, " How Universal Polynomial Bases Enhance Spectral Graph Neural Networks: Heterophily, Over-smoothing, and Over-squashing", *ICML'24*.

Adaptive Selection: UniFilter [ICML'24]

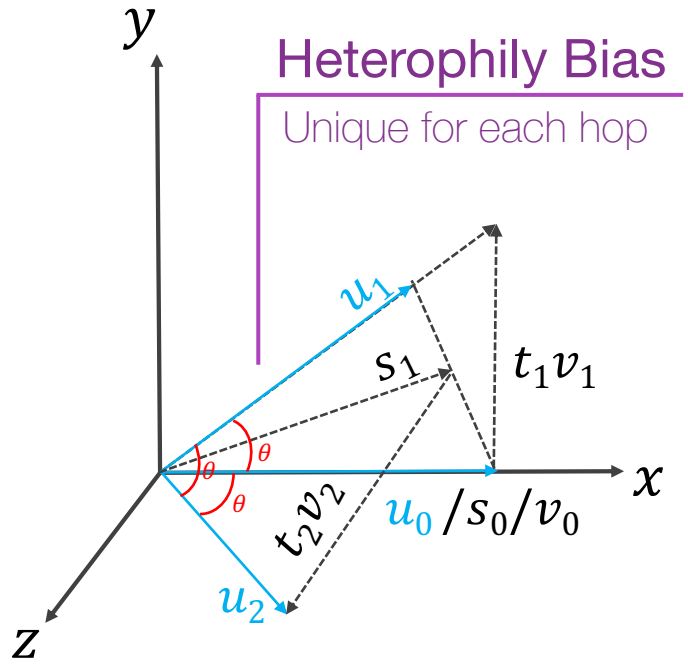
- Spectral explanation



Spectral Bases

Unique frequency

Monotonic relationship of spectral bases



Heterophily Bias

Unique for each hop

Spectral Domain (Frequency Space)

Spatial Domain (Embedding Space)

K Huang et al, " How Universal Polynomial Bases Enhance Spectral Graph Neural Networks: Heterophily, Over-smoothing, and Over-squashing", *ICML'24*.

Adaptive Selection: AdaptKry [WWW'24]

- Understanding graph propagation as heat diffusion:

Laplacian

$$A = I - L$$

$$H_{t+1} = (I - L)H_t \quad \rightarrow \quad \frac{\mathrm{d}H_t}{\mathrm{d}t} = -LH_t \quad \tau \rightarrow 0$$

$$\begin{aligned} \mathbf{H}_{t+\tau} &= \lim_{\tau \to 0^+} \mathbf{H}_t - \tau \mathcal{L} \mathbf{H}_t \\ &= \lim_{\tau \to 0^+} \mathbf{H}_t (\mathbf{I} - \tau \mathcal{L}) \\ &= \lim_{\tau \to 0^+} \mathbf{H}_t ((1-\tau)\mathbf{I} + \tau \mathbf{D}^{-\frac{1}{2}} \mathbf{A} \mathbf{D}^{-\frac{1}{2}}) \end{aligned}$$

- **Krylov Subspace**: fixed propagation

$$\mathcal{K}_K(A, \mathrm{v}) = \{\mathrm{v}, A\mathrm{v}, A^2\mathrm{v}, \ldots, A^{K-1}\mathrm{v}\}.$$

K Huang et al, " Optimizing Polynomial Graph Filters: A Novel Adaptive Krylov Subspace Approach", *WWW'24*.

Adaptive Selection: AdaptKry [WWW'24]

- Renormalize propagation:

$$P = (1-\tau)\mathrm{I} + \tau D^{-1/2} A D^{-1/2}$$

$$P_\tau = (\tau D + (1-\tau)\mathrm{I})^{-1/2} (\tau A + (1-\tau)\mathrm{I}) (\tau D + (1-\tau)\mathrm{I})^{-1/2}$$

$$P_\tau = D_\tau^{-\frac{1}{2}} A_\tau D_\tau^{-\frac{1}{2}}$$

- **Adaptive Krylov Subspace**: control graph message

$$\mathcal{K}_K(P_\tau, \mathrm{x}) = \{\mathrm{x}, P_\tau \mathrm{x}, P_\tau^2 \mathrm{x}, \ldots, P_\tau^{K-1} \mathrm{x}\}$$

K Huang et al, " Optimizing Polynomial Graph Filters: A Novel Adaptive Krylov Subspace Approach", *WWW'24*.

Spectral Embedding: Takeaways

Channel Combination

LD^2 [NeurIPS'23]

- 2-hop MP for global information
- Link spatial & spectral computation

TER [KDD'23]

S3GCL [ICML'24]

- Extending spectral embedding to edge-level task and contrastive learning

Adaptive Selection

UniFilter [ICML'24]

- Link spatial & spectral computation
- Additional bias for hop-wise heterophily

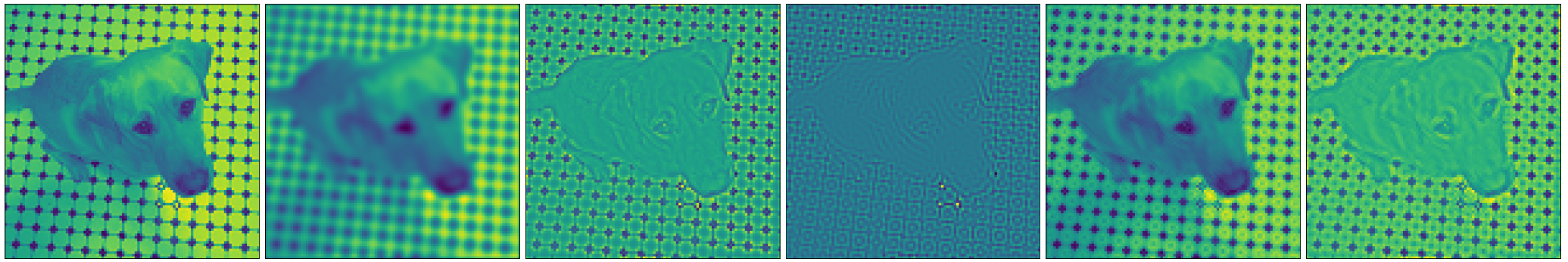
AdaptKry [WWW'24]

- MP as heat diffusion
- Adjust graph normalization

Spectral Embedding: Applications

Image Filtering

- Image as graph: better depict unstructured relationships, e.g. contextual connection of segmentations



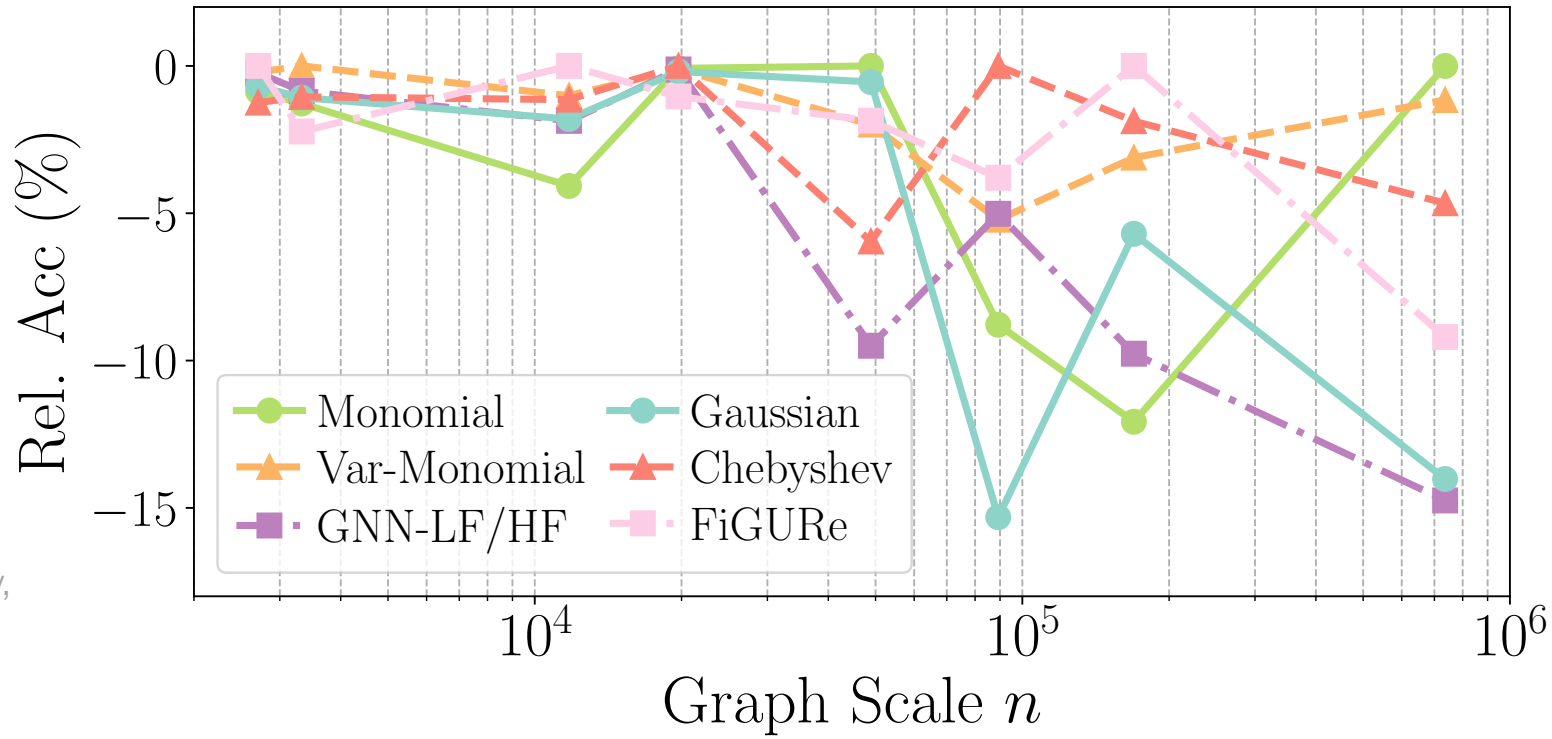
(a) Original (b) Low-pass (c) High-pass (d) Band-pass (e) Band-rejection (f) Comb

M He et al, " BernNet: Learning Arbitrary Graph Spectral Filters via Bernstein Approximation", *NeurIPS'21*.

Spectral Embedding: Evaluation

Graph Propagation as Spectral Filter

- Global view is important for large-scale prediction
- Simple, invariable filters can also capture global view



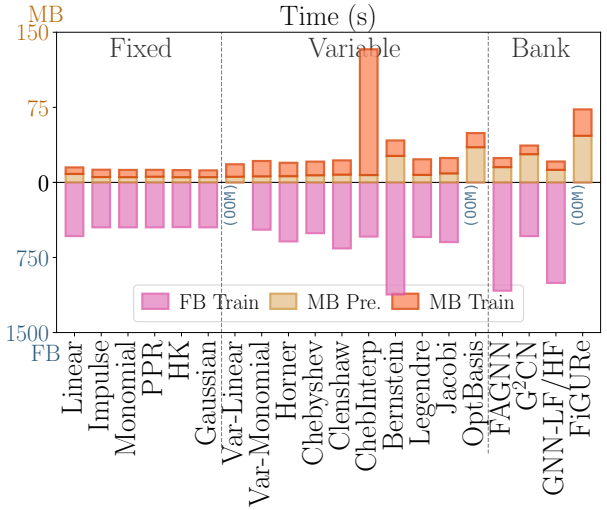
N Liao et al, "A Comprehensive Benchmark on Spectral GNNs: The Impact on Efficiency, Memory, and Effectiveness", SIGMOD'26.

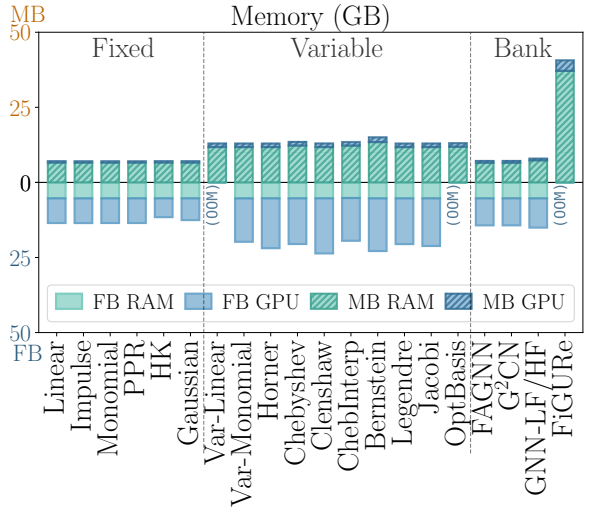
Spectral Embedding: Evaluation

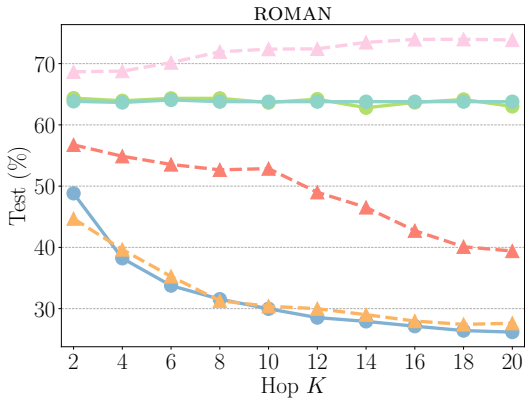
Graph Propagation as Spectral Filter

- Comprehensive evaluation on: efficiency, memory footprint, propagation hops of graph filters, …
- Insights on finding fast and effective graph filters

N Liao et al, "A Comprehensive Benchmark on Spectral GNNs: The Impact on Efficiency, Memory, and Effectiveness", SIGMOD'26.



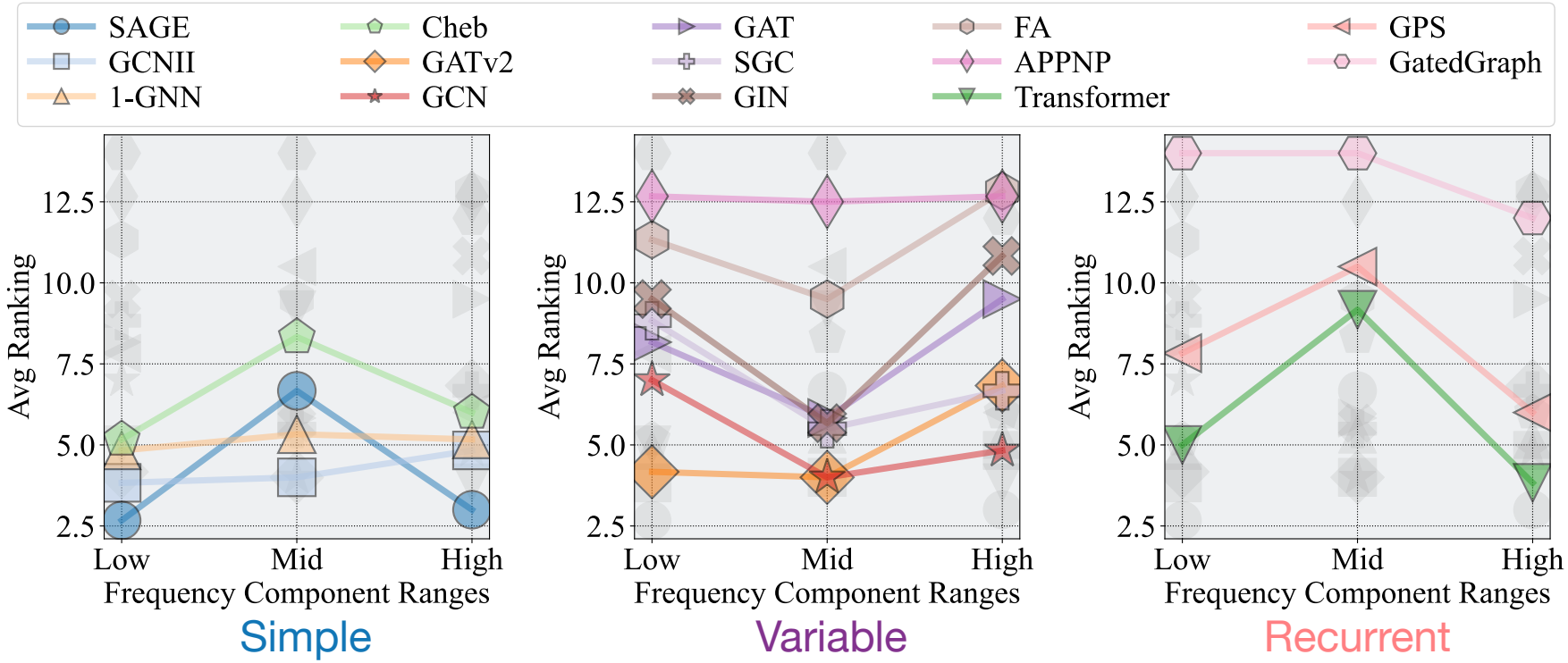




Spectral Embedding: Evaluation

Weight Transformation as Spectral Filter

- Weight transformation also affects spectral filtering



Y Dong et al, "Graph Neural Networks Are More Than Filters", *ICLR'25*.

Advances in Designing Scalable GNNs

Graph Simplification

Outline

Node-wise Similarity

Spectral Embedding

20 min
11:45 – 12:05

Graph Simplification
Ningyi Liao

Layer-wise NIGCN [K Huang et al | WWW'23]

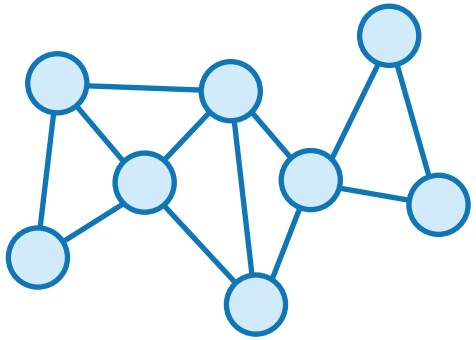
Pair-wise ATP [X Li et al | WWW'24]

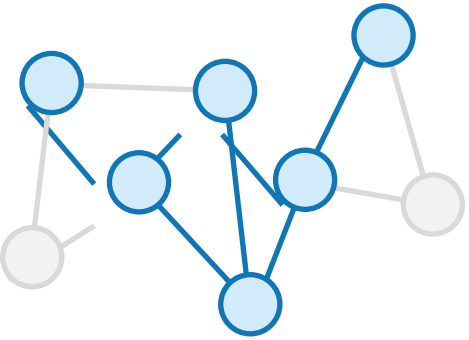
Entry-wise Unifiews [N Liao et al | ICML'25]

Future Direction

Graph Simplification: Strategies

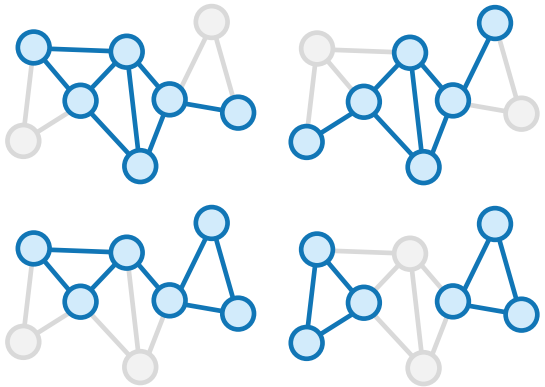
Raw Graph





Sparsification

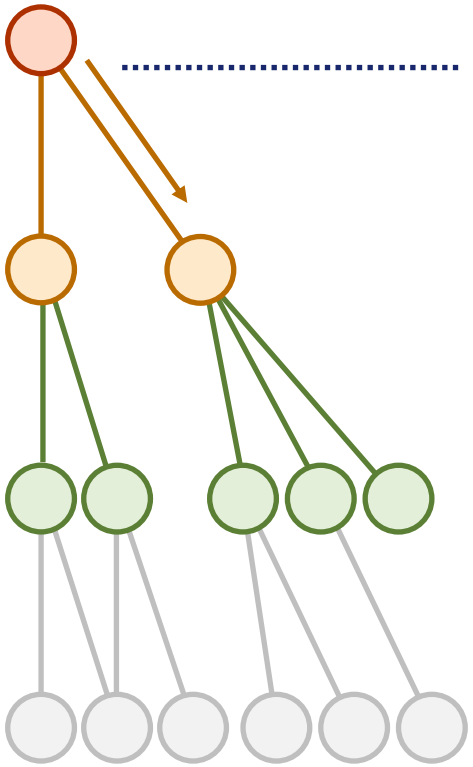
- Static graphs structure during learning
- Dedicated algorithms



Sampling

- Different graphs through training iterations
- Repetitive sampling overhead

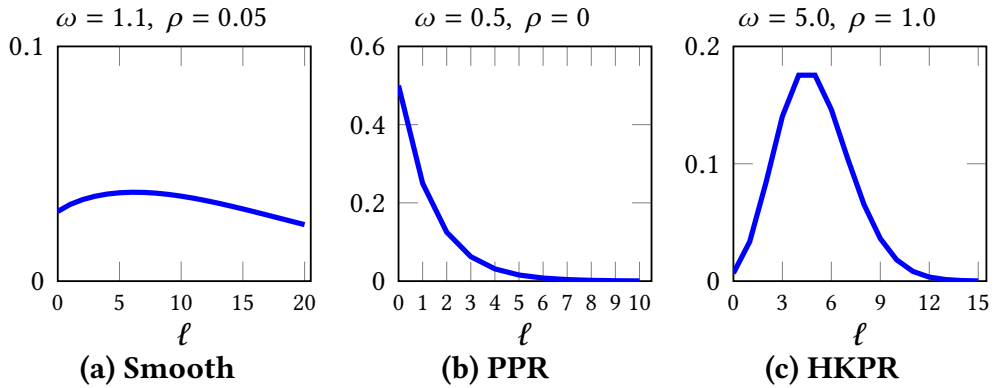
Layer-wise Sparsification: NIGCN [WWW'23]



General Heat Diffusion

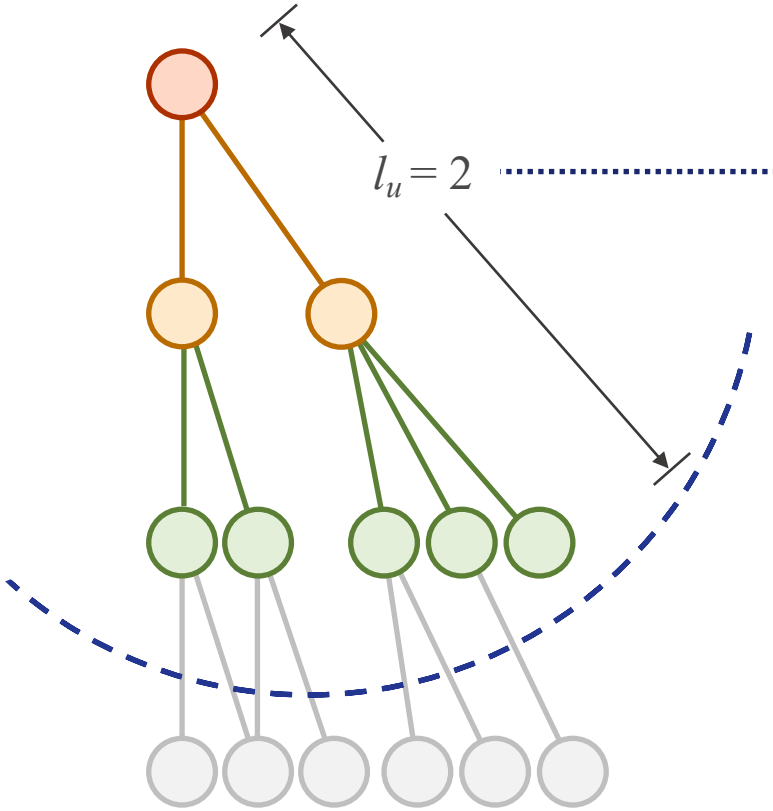
$$U(\omega, \rho, \ell) = \frac{\omega^{\ell}}{(\ell!)^{\rho} \cdot C}$$

- Coverage: smooth (GCN), PPR (GBP), HKPR (AGP)
- Applicable to decoupled architecture



K Huang et al, "Node-Wise Diffusion for Scalable Graph Learning", *WWW'23*.

Layer-wise Sparsification: NIGCN [WWW'23]



Node-wise Diffusion Distance

$$\ell_u := \left\lceil \log_\lambda \frac{\tau\sqrt{d_{\min} d_u}}{2m} \right\rceil$$

- Bounds computation complexity & approximation precision
- Nested in graph propagation

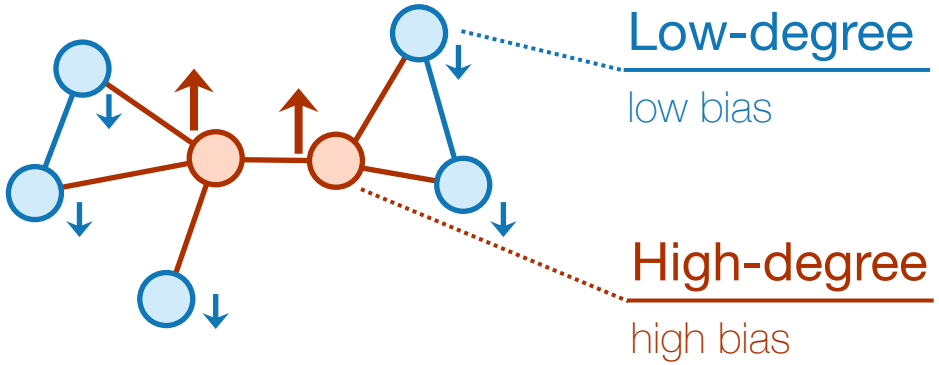
K Huang et al, "Node-Wise Diffusion for Scalable Graph Learning", *WWW'23*.

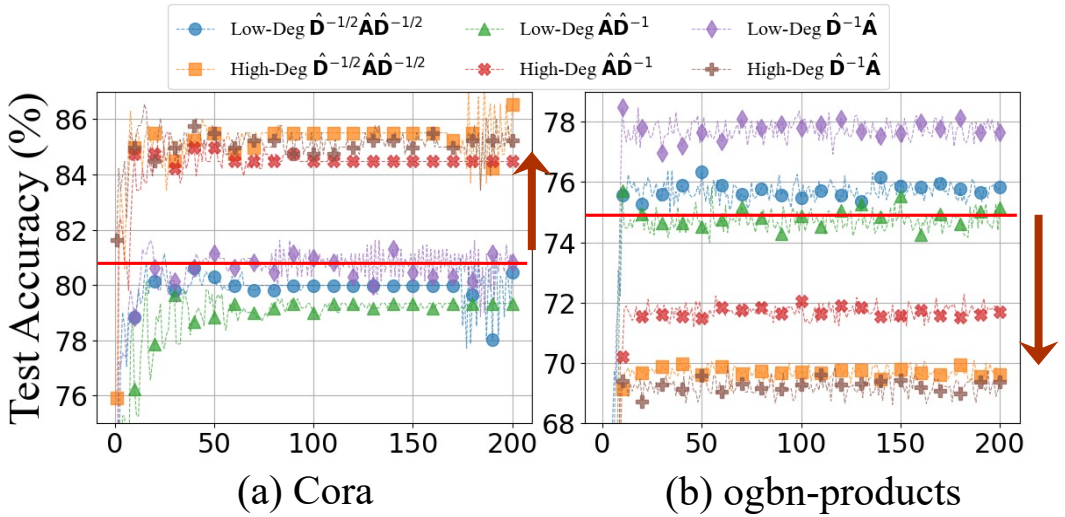
Pair-wise Sparsification: ATP [WWW'24]

Conventional Graph Propagation

$$\mathbf{\Pi} = \sum_{i=0}^{l} w_i \cdot \left(\hat{\mathbf{D}}^{r-1}\hat{\mathbf{A}}\hat{\mathbf{D}}^{-r}\right)^i,$$

- Unified normalization for all nodes
- High-degree nodes produce biased accuracy





X Li et al, "Rethinking Node-wise Propagation for Large-scale Graph Learning", *WWW'24*.

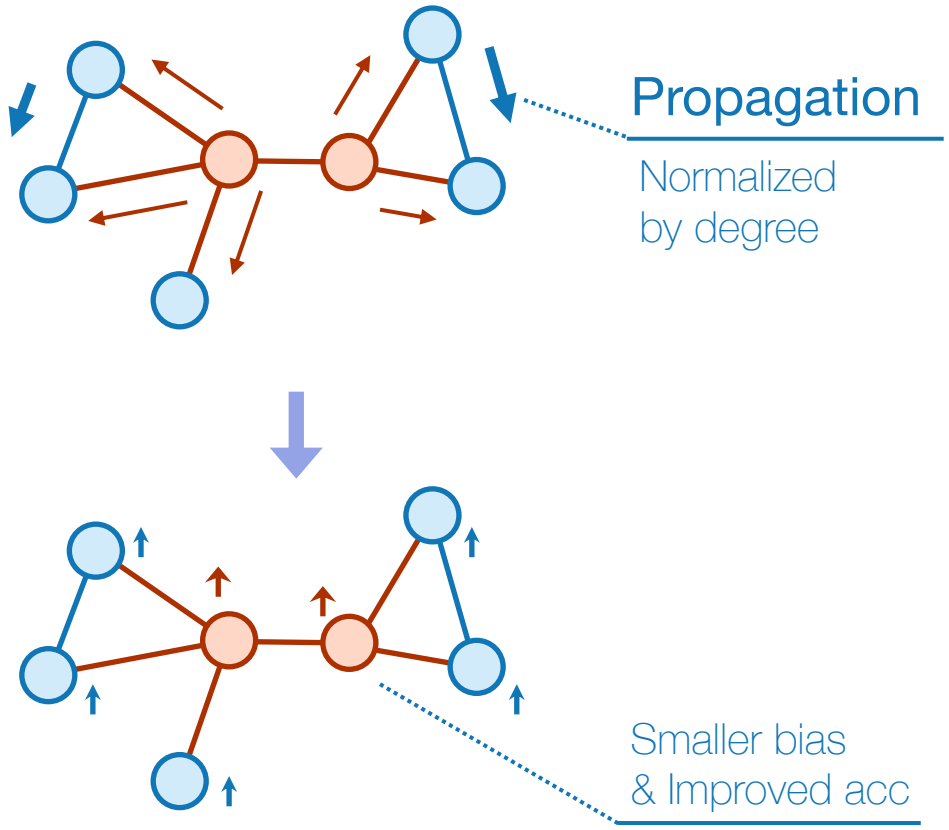
Pair-wise Sparsification: ATP [WWW'24]

Node-wise Propagation

$$\tilde{\mathbf{\Pi}} = \sum_{i=0}^{l} w_i \cdot \left([\hat{\mathbf{D}}]^{\tilde{\mathbf{R}}-1} [\hat{\mathbf{A}}] [\hat{\mathbf{D}}]^{-\tilde{\mathbf{R}}} \right)^i$$

$$\mathbf{R}_0 = \mathrm{Diag}\left(\frac{[d]_1}{n-1}, \ldots, \frac{[d]_n}{n-1} \right).$$

- Specific normalization for each node
- Criteria based on degree, eigenvector, and connectivity



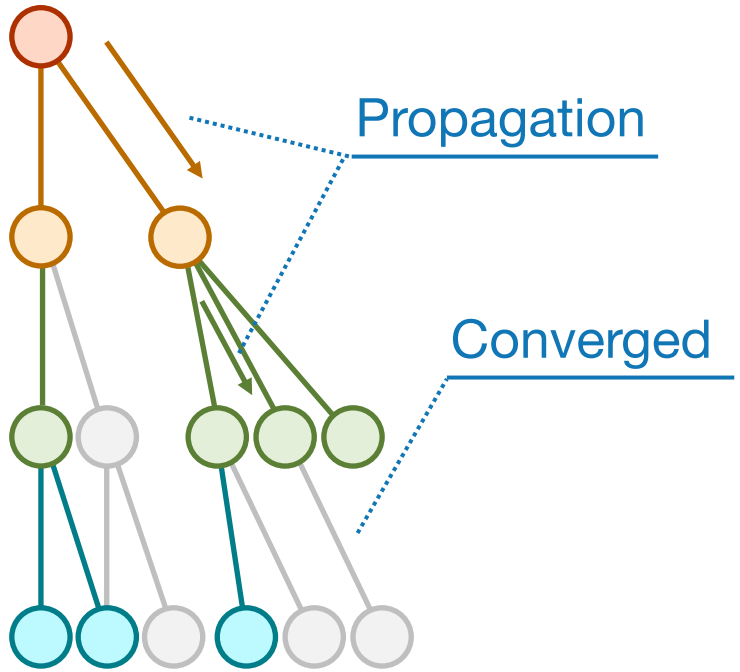
X Li et al, "Rethinking Node-wise Propagation for Large-scale Graph Learning", *WWW'24*.

Pair-wise Sparsification: ATP [WWW'24]

Pair-wise Propagation Distance

$$\left|\left(\mathbf{P}^k e_i\right)_j - \tilde{\pi}_j\right| \leq \sqrt{\frac{\tilde{d}_j}{\tilde{d}_i}}\lambda_2^k,$$

- Precompute the converged value
- Threshold propagations closed to converged values



X Li et al, "Rethinking Node-wise Propagation for Large-scale Graph Learning", *WWW'24*.

Sparsification: Unifews [ICML'25]

Entry-wise Sparsification

$$\boldsymbol{p}_{(l+1)}[u] = \sum_{v \in \mathcal{N}(u)} \boldsymbol{T}_{(l)}[u,v] \cdot \boldsymbol{p}_{(l)}[v]$$

Embedding

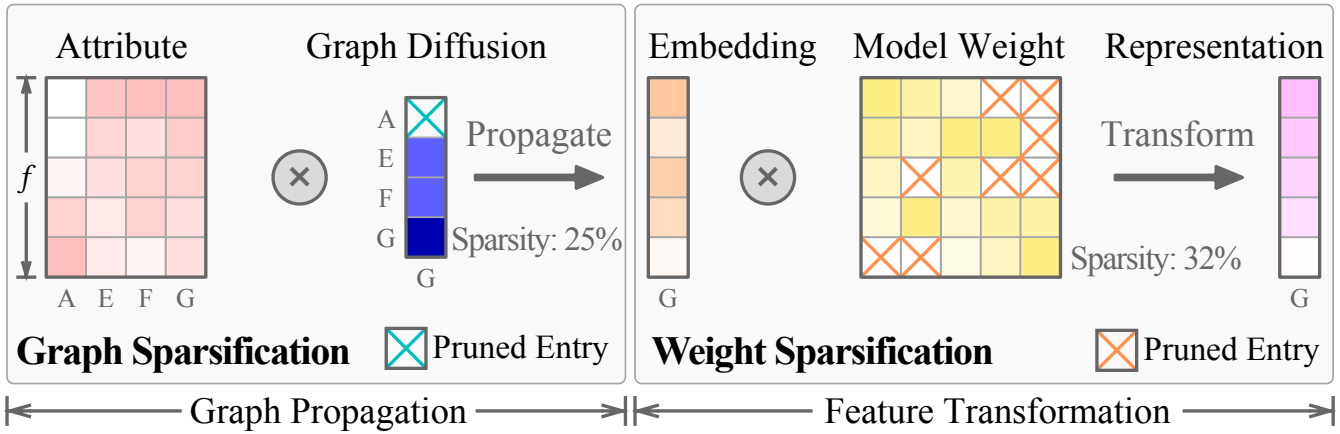
Propagation

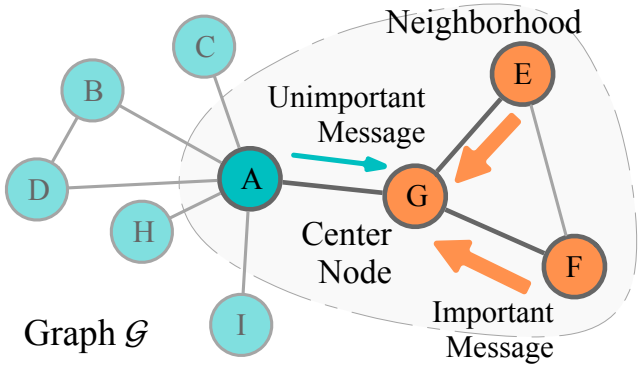
Subject to prune

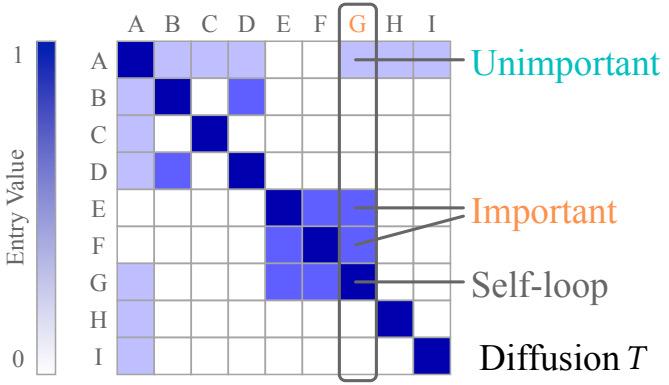
$$\boldsymbol{H}_{(l+1)}[:,i] = \sum_{j=1}^{f} \boldsymbol{W}_{(l)}[j,i] \cdot \boldsymbol{P}_{(l)}[:,j]$$

Weight

Subject to prune







N Liao et al, " Unifews: You Need Fewer Operations for Efficient Graph Neural Networks", *ICML'25*.

Sparsification: Unifews [ICML'25]

Sparsification ⇔ Approximate Graph Smoothing

GNN Graph Smoothing:

$$\boldsymbol{p}^* = \arg\min_{\boldsymbol{p}} \mathcal{L}, \quad \mathcal{L} = \|\boldsymbol{p} - \boldsymbol{x}\|^2 + c \cdot \boldsymbol{p}^\top \boldsymbol{L} \boldsymbol{p},$$

Embedding · Attribute · Graph

Spectral Approximation:

$$\|\hat{\boldsymbol{p}}^* - \boldsymbol{p}^*\| \le c\epsilon \|\boldsymbol{p}^*\|.$$

Approx Bound

Error Guarantee:

(general) ⟹

$$q_a \delta_a \le \epsilon \|\boldsymbol{p}\|$$

Sparsity · Prune Threshold

(scale-free graph) ⟹

$$\epsilon = O\left(\eta_a (1-\eta_a)^{\frac{1}{1-\alpha}}\right)$$

Graph Distribution

N Liao et al, " Unifews: You Need Fewer Operations for Efficient Graph Neural Networks", *ICML'25*.

Sparsification: Unifews [ICML'25]

Layer-progressive Sparsification

Graph Sparsification:

$$\|\boldsymbol{p}_{(l+1)} - \hat{\boldsymbol{p}}_{(l+1)}\| \le O(\epsilon) \cdot \|\boldsymbol{p}_{(l)}\| + bc\epsilon(1+O(\epsilon))\|\boldsymbol{p}_{(l)}\|$$

Embedding

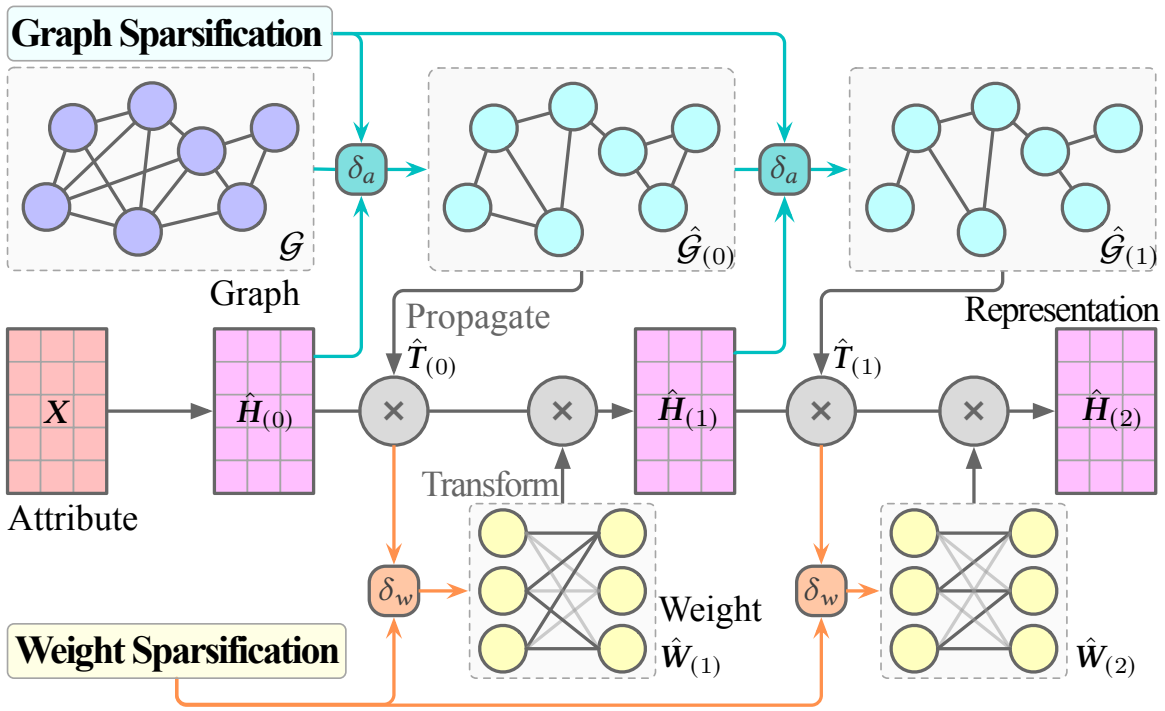
Approx Bound

w/ Weight Sparsification:

$$\|\hat{\boldsymbol{H}}_{(l+1)} - \boldsymbol{H}_{(l+1)}\|_F \le \ell_\sigma bc\epsilon \|\boldsymbol{H}_{(l)}\|_F \|\boldsymbol{W}\|_F + \ell_\sigma q_w \delta_w$$

Representation

Network Distribution



N Liao et al, " Unifews: You Need Fewer Operations for Efficient Graph Neural Networks", *ICML'25*.

Graph Simplification: Takeaways

Layer-wise	Pair-wise	Entry-wise
NIGCN [WWW'23]	ATP [WWW'24]	Unifews [ICML'25]
◦ MP as heat diffusion ◦ Personalized MP hop for each node	◦ Generalized MP norm ◦ Personalized MP hop and norm for each node pair	◦ MP as spectral smoothing ◦ Personalized MM for graph and weight entries

Advances in Designing Scalable GNNs

Conclusion & Future Directions

Outline

Node-wise Similarity

Spectral Embedding

Graph Simplification

25 min
12:05 – 12:30

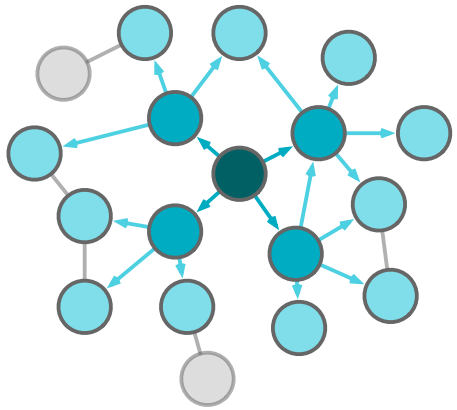
Future Directions
Reynold Cheng

Summary

Large Model	RAG | Graph Transformer
Data Variant	Data Deficiency | Dynamic Data
System Co-design	Device-specific | Distributed Training

Summary: Challenges & Solutions

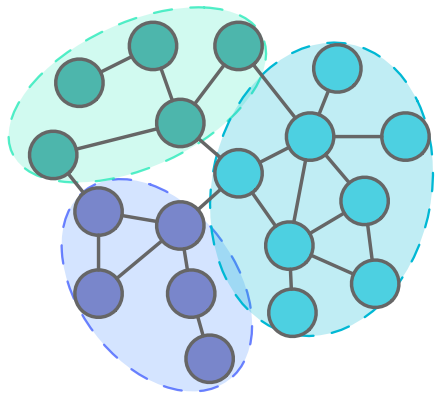
Neighbor Explosion



Node-wise Similarity

- PPR: faster message passing
- SimRank: global similarity
- Hub Labeling: core hub nodes

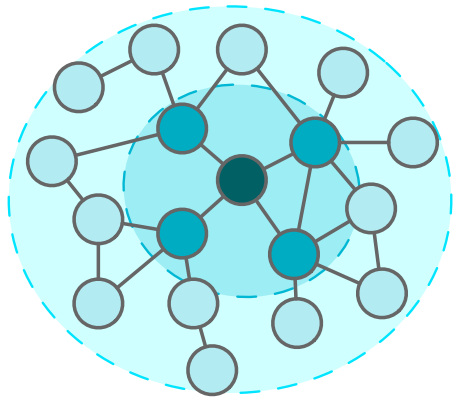
Limited Memory



Subgraph Extraction

- Partition: reduce transmission
- Generation: tailored message passing
- Slicing: disk lookup

Multi-scale



Spectral Embedding

- Combination: local & global channels
- Selection: adaptive filtering

Graph + Large Models

Message-passing

- Mature & diverse options
- Tailored for graph data
- Scalable solutions

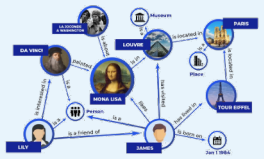
Transformer-based

- Powerful & attentive
- Stackable to deep layers
- Integrate with existing LMs

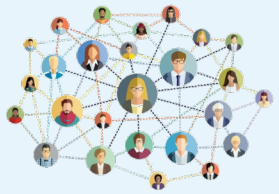
Model

Scalable Graph Data Management

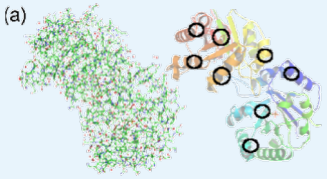
Data



Knowledge Graph



Social Network



Protein & Molecules

Task

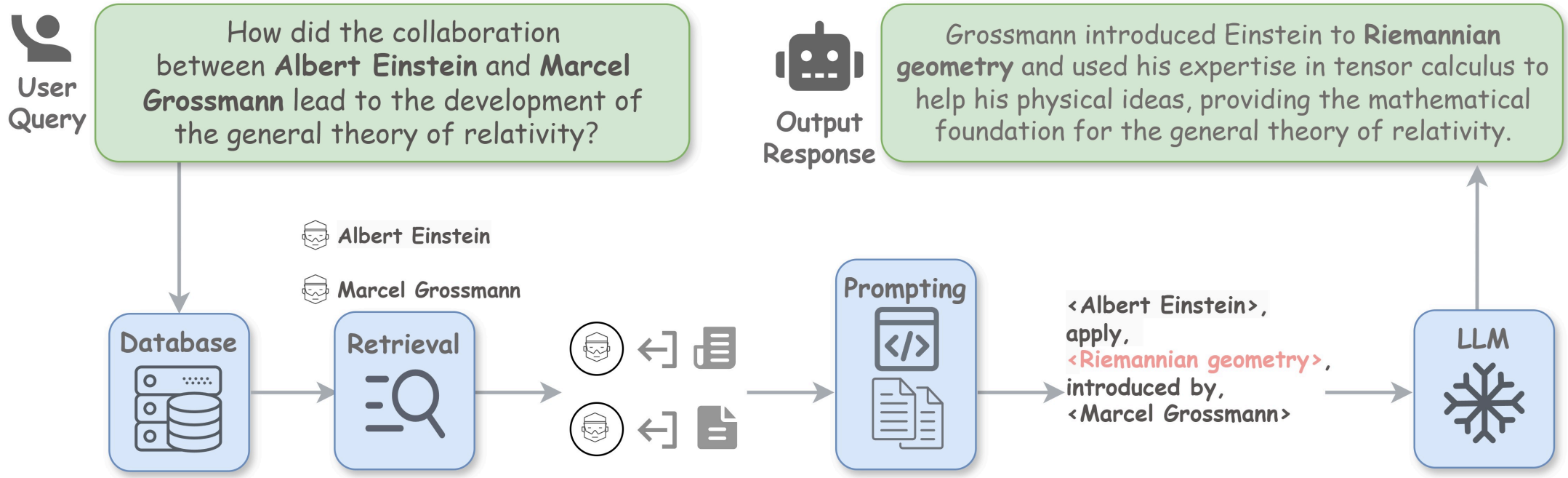
Analysis

Retrieval

Recommendation

LM: Graph for Retrieval-Augmented Generation

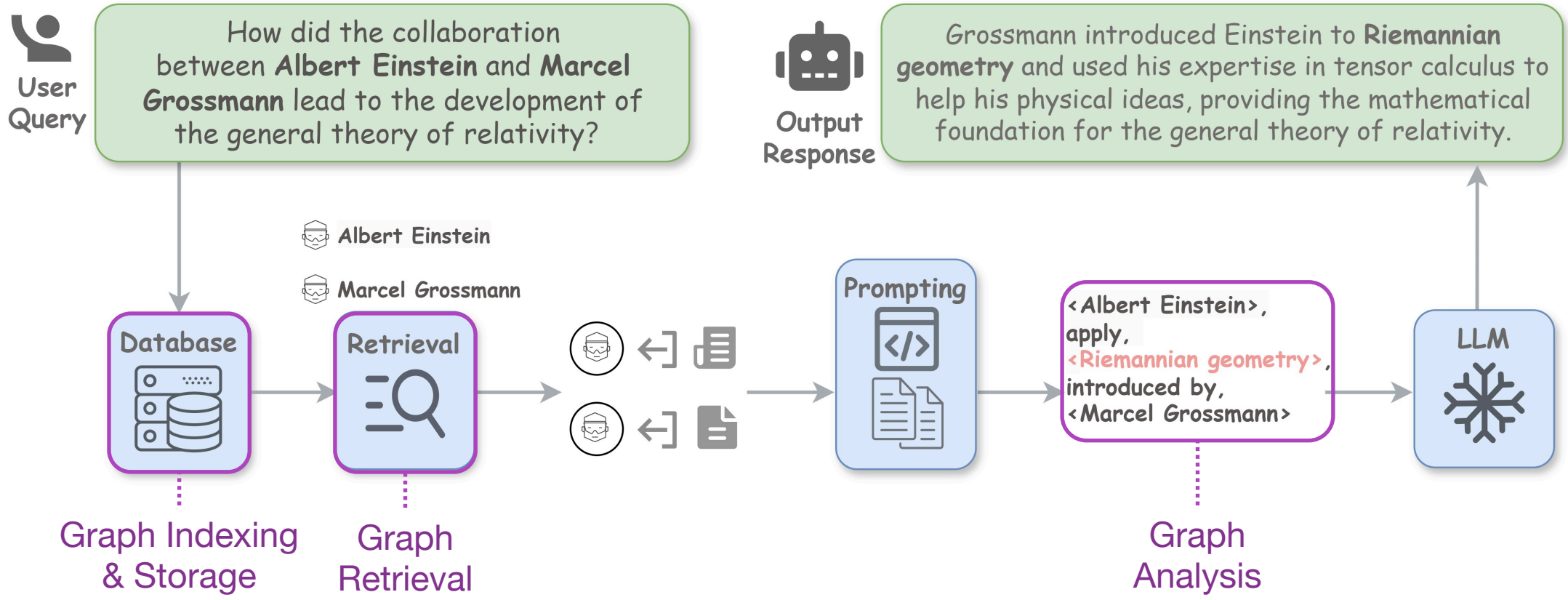
Standard RAG



Z Zhu et al, "Graph-based Approaches and Functionalities in Retrieval-Augmented Generation: A Comprehensive Survey", arXiv: 2504.10499.

LM: Graph for Retrieval-Augmented Generation

Opportunities for Graph + RAG



Z Zhu et al, "Graph-based Approaches and Functionalities in Retrieval-Augmented Generation: A Comprehensive Survey", arXiv: 2504.10499.

LM: Graph for Retrieval-Augmented Generation

Example: PageRank in RAG

Standard RAG

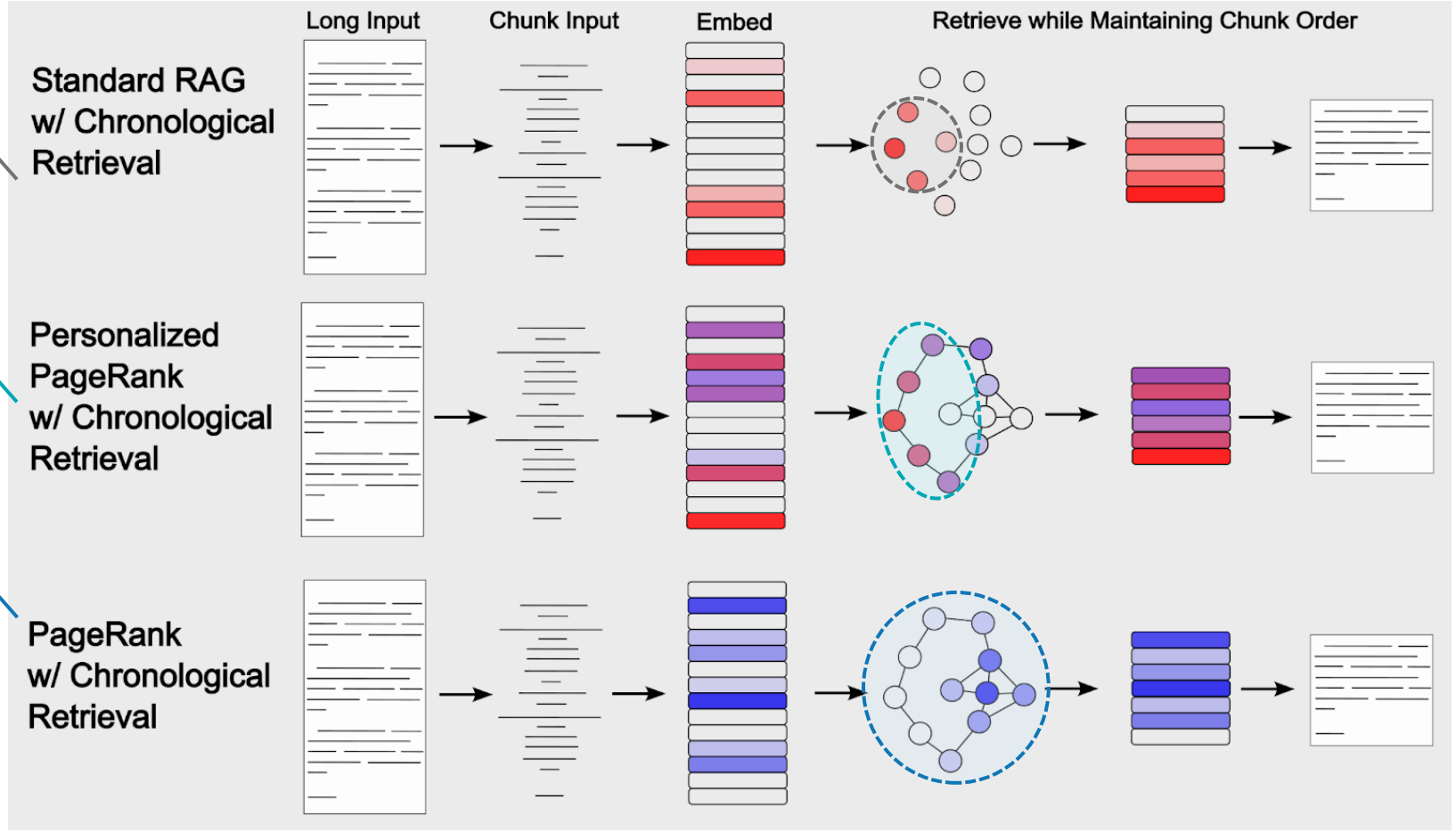
Direct Concepts

PPR

Local Relation

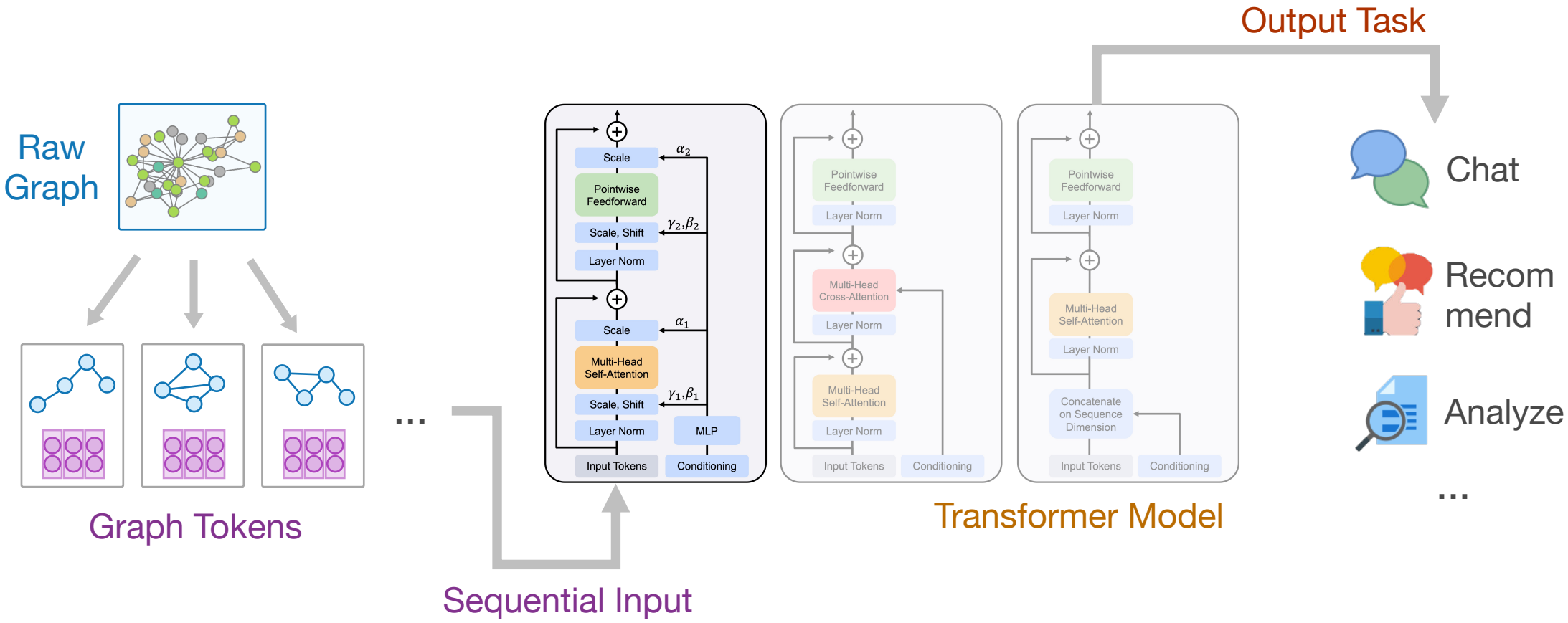
PR

Global Relation



N Alonso et al, "The Mixture-of-PageRanks Retriever for Long-Context Pre-Processing", 2024. https://www.zyphra.com/post/

LM: Scalable Graph Transformer

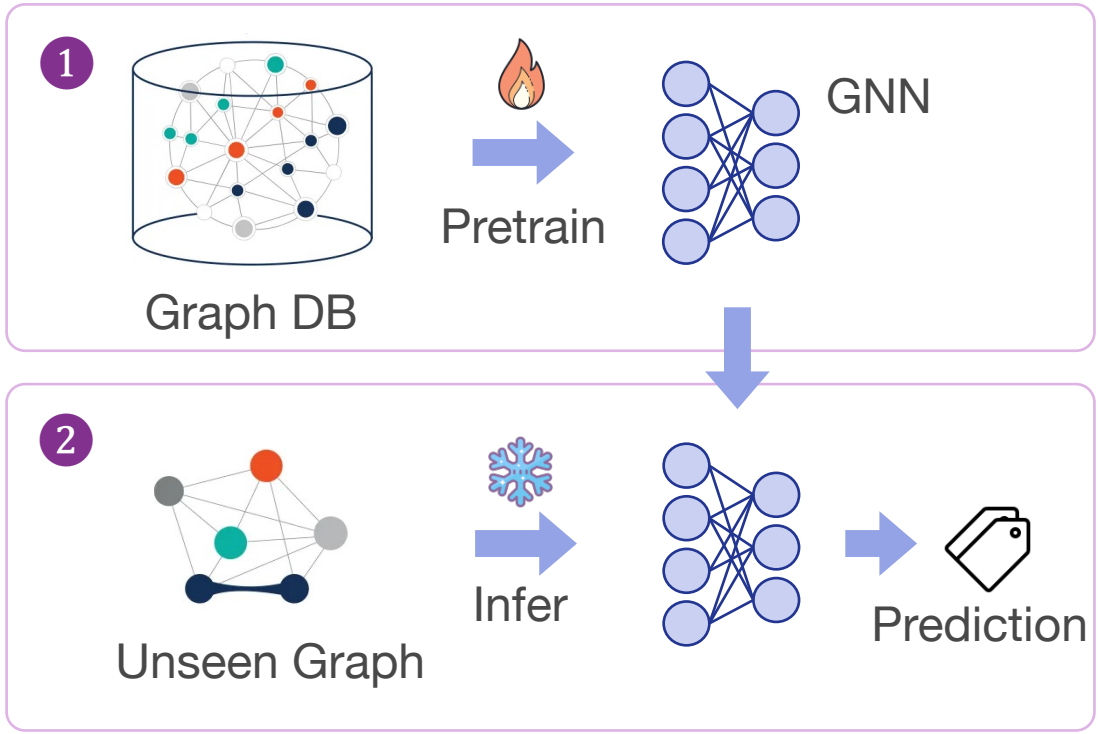


Efficient sequential graph representation?

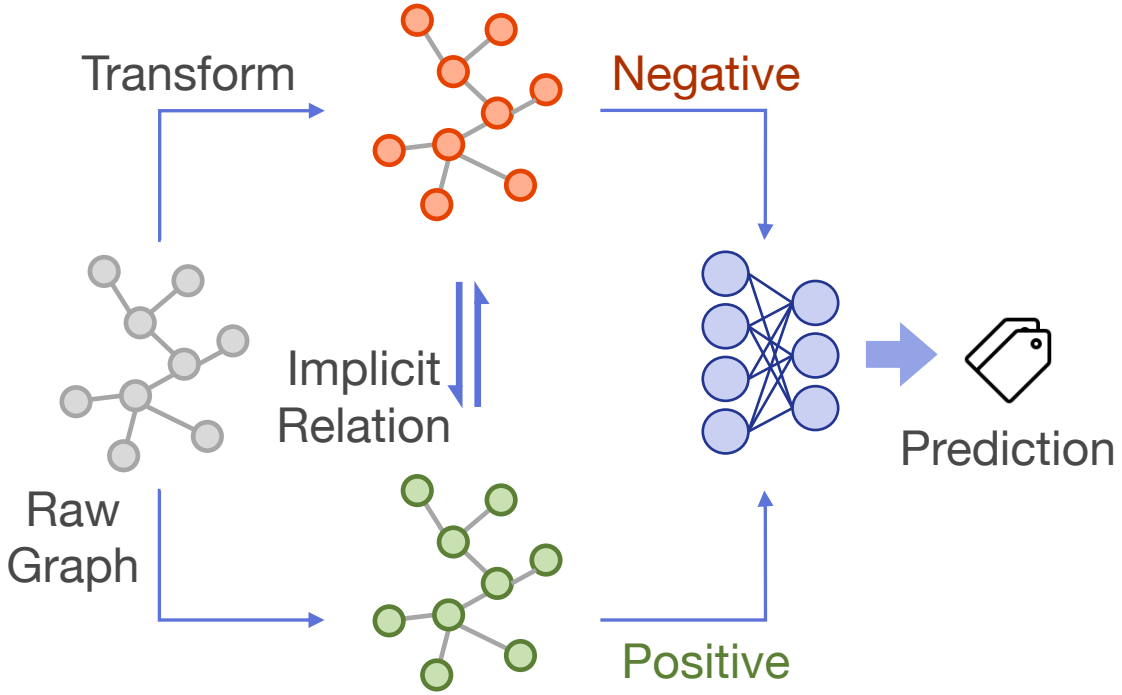
Efficient graph Transformer architecture?

Graph Variants: Data Deficiency

Pretrain



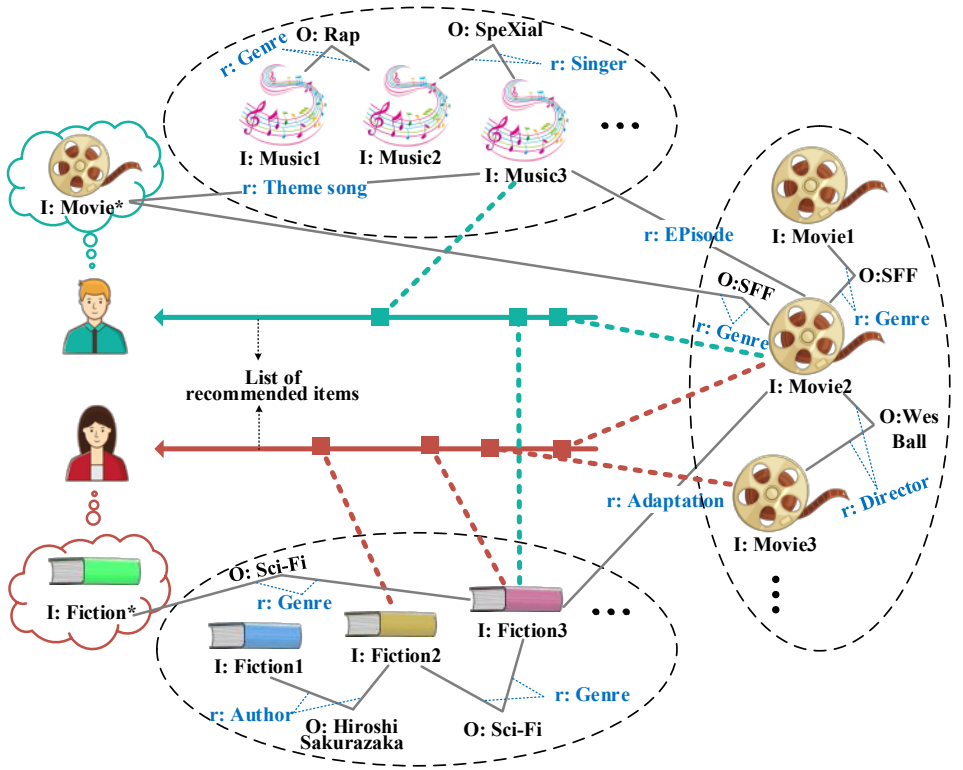
Contrastive



Graph Variants: Data Deficiency

Example: Knowledge Sharing in Recommendation

- Challenge: Data Deficiency
  - Overlap Sparsity: few overlap data between domains
  - Data Sparsity: limited data available in some domains
- Solution: Cross-domain learning by knowledge alignment

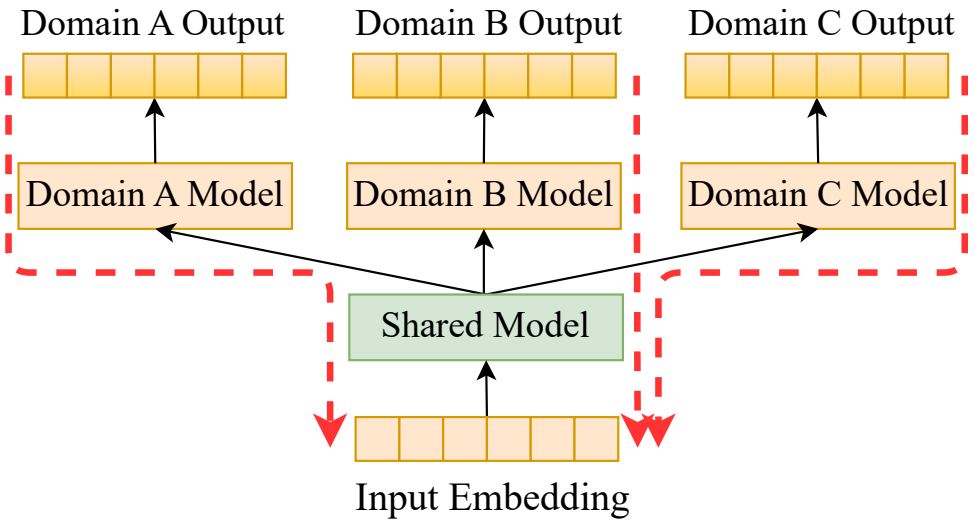


J Liu et al. “Cross-domain knowledge graph chiasmal embedding for multi-domain item-item recommendation”. *TKDE’22*
W Ning et al. “Multi-domain Recommendation with Embedding Disentangling and Domain Alignment”. *CIKM’23*.

Graph Variants: Data Deficiency

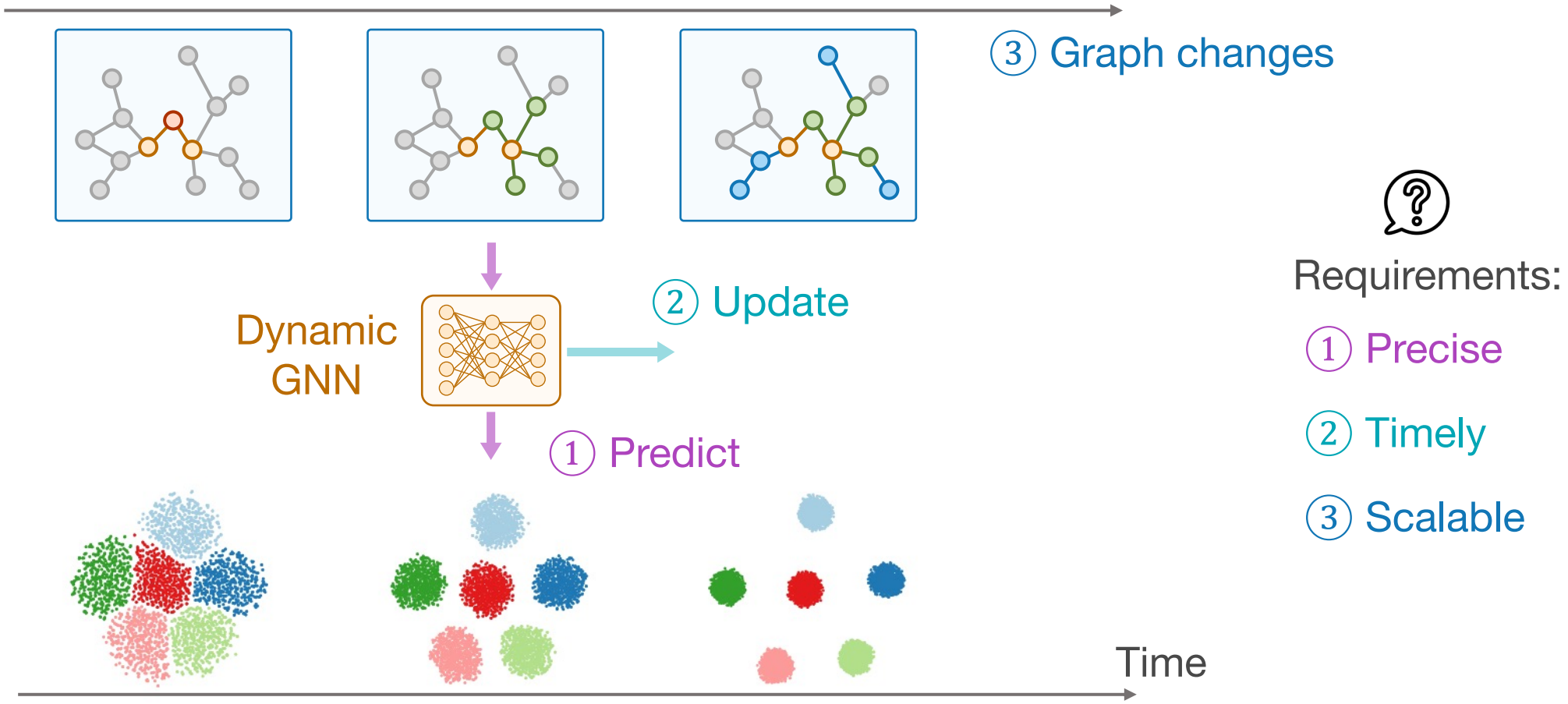
Example: Knowledge Sharing in Recommendation

- Challenge: Model Scalability
  - Inter-domain: effectively learn domain-specific knowledge
  - Intra-domain: efficiently share knowledge without learning from scratch
- Solution: Model disentangling
  - Domain Model
  - Shared Model



W Ning et al. "Multi-domain Recommendation with Embedding Disentangling and Domain Alignment". *CIKM'23*.

Graph Variants: Dynamic Data



Requirements:

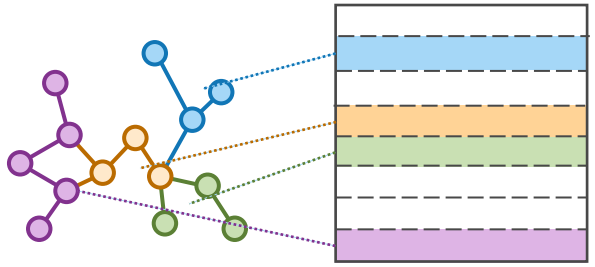
① Precise

② Timely

③ Scalable

Systems Co-design: Device-Specific

Graph Storage

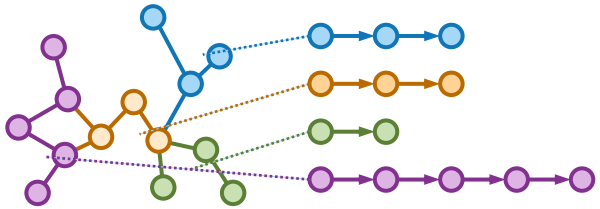


On-disk

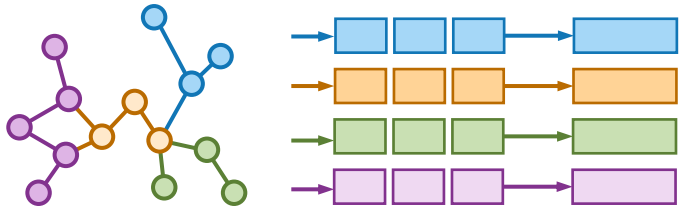
- Large storage
- Transmission overhead
- Update overhead

In-memory

- Random access
- Device loading
- Cache & buffer



Graph Computation

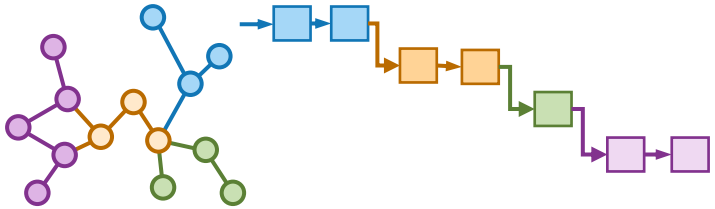


GPU-oriented

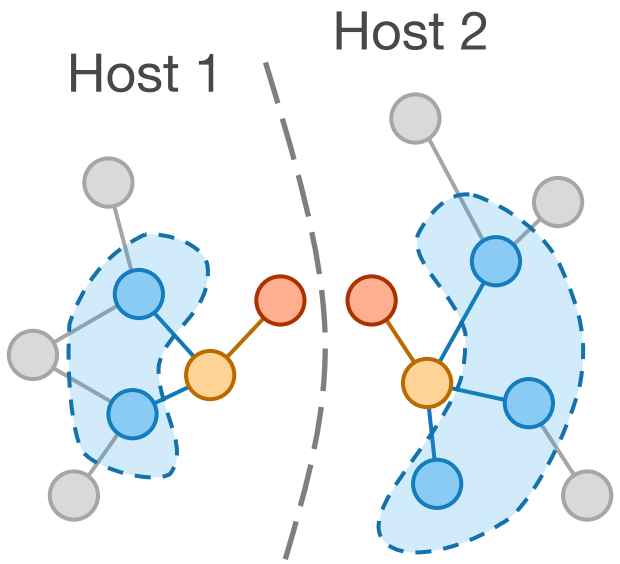
- Limited memory
- Batch processing
- Predefined operators

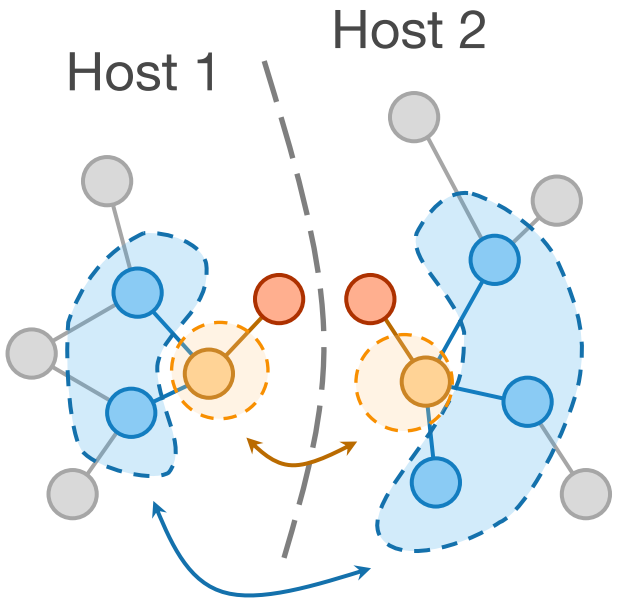
CPU-oriented

- Sequential/parallel
- Dedicated algorithms
- Sparse data structures



Systems Co-design: Distributed Training





Neighbor Explosion

- Neighbors affected by partitioning
- More communication for accessing neighbors

Multi-scale

- Local subgraph structure affected by partitioning
- Communication for global info

THANK YOU

Slides

香 港 大 學
THE UNIVERSITY OF HONG KONG

2025 International Conference on Management of Data
Berlin, Germany – June 22, 2025