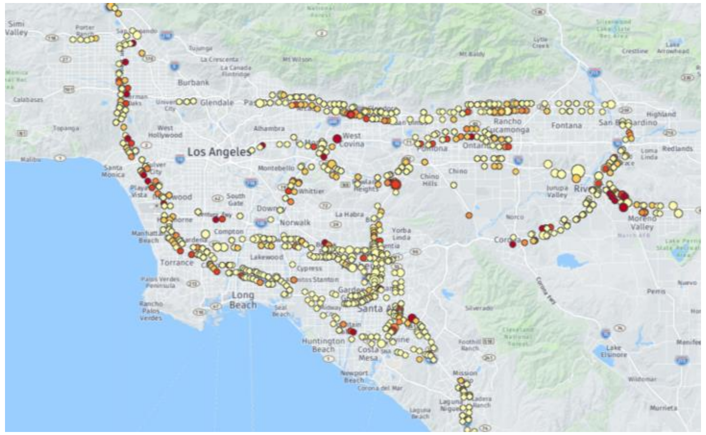
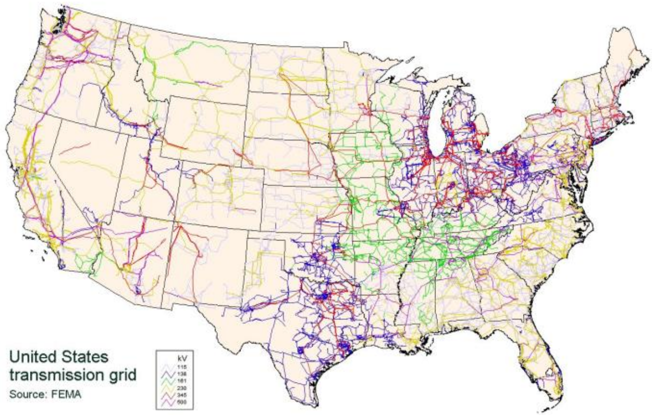
Fograph: Enabling Real-Time Deep Graph Inference with Fog Computing

Liekang Zeng, Peng Huang, Ke Luo, Xiaoxi Zhang, Zhi Zhou, Xu Chen

Sun Yat-sen University

Ubiquitous Graphs in Real-World





Traffic Network

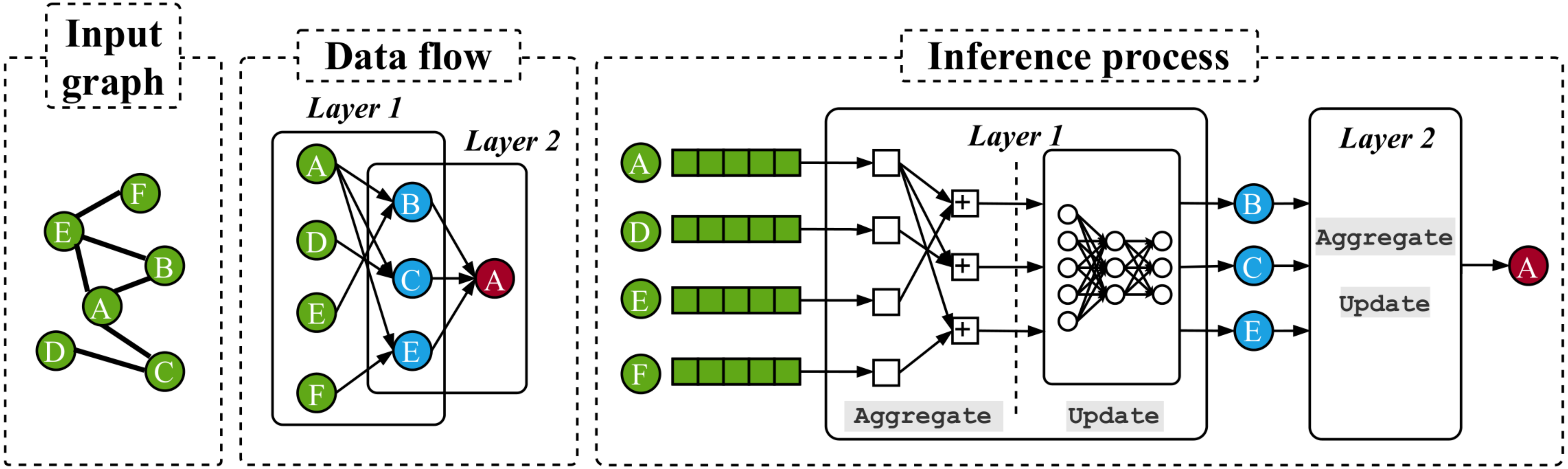
Social Graph

Power Grids

Credit: Google Image

Graph Neural Networks

- **Neural message passing framework**
  - Each vertex *aggregates* features of its neighbors
  - *Update* its feature by combining the aggregation through a neural network operator



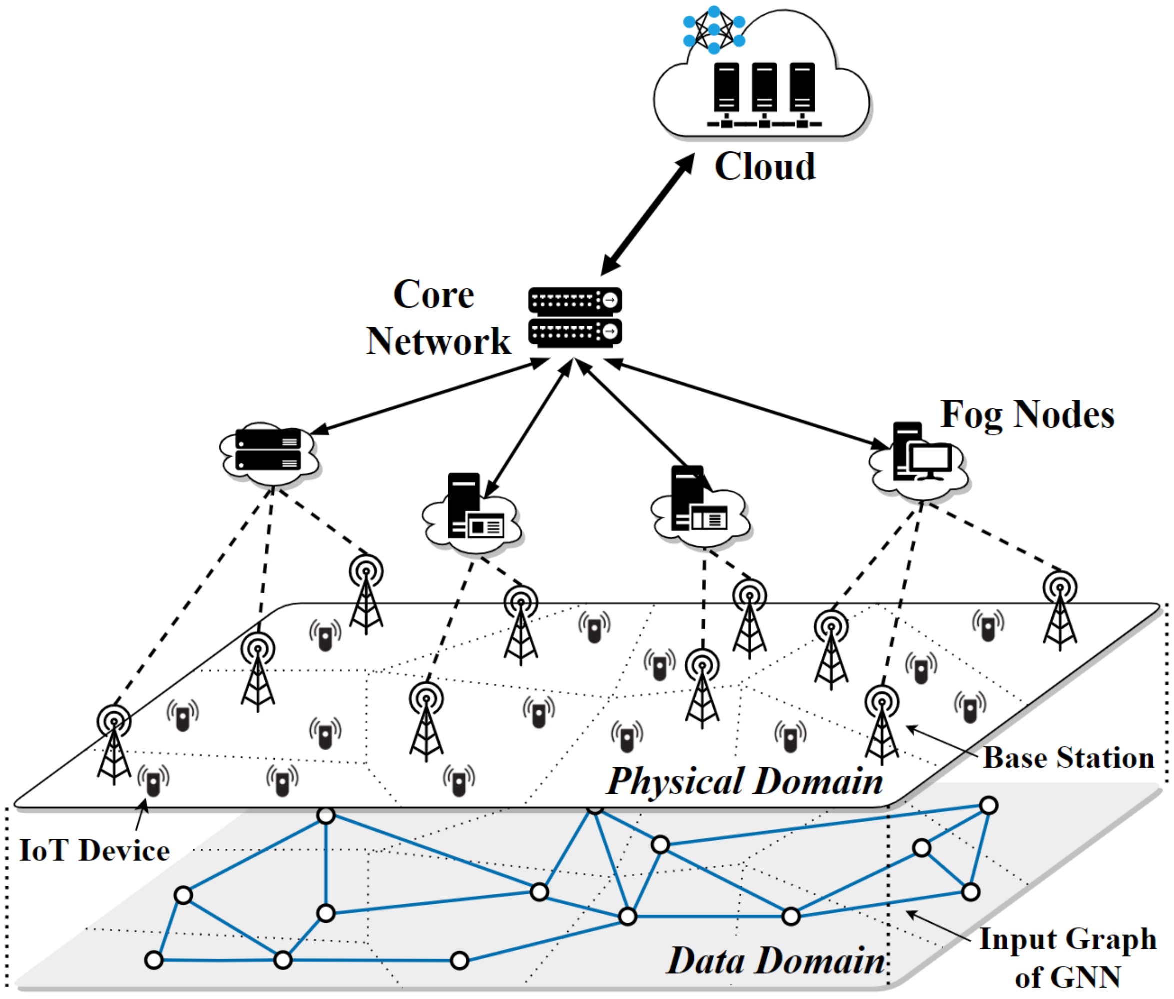
Analytics with Graph Neural Networks

- **Why use GNNs?**
  - High classification **accuracy**
  - Superior **generality** for diverse graphs
  - Advanced **expressiveness** to interpret topology

- **What applications?**
  - **Graph prediction**: traffic flow forecasting
  - **Link prediction**: locations-based social recommendation
  - **Node classification**: power grid failure detection

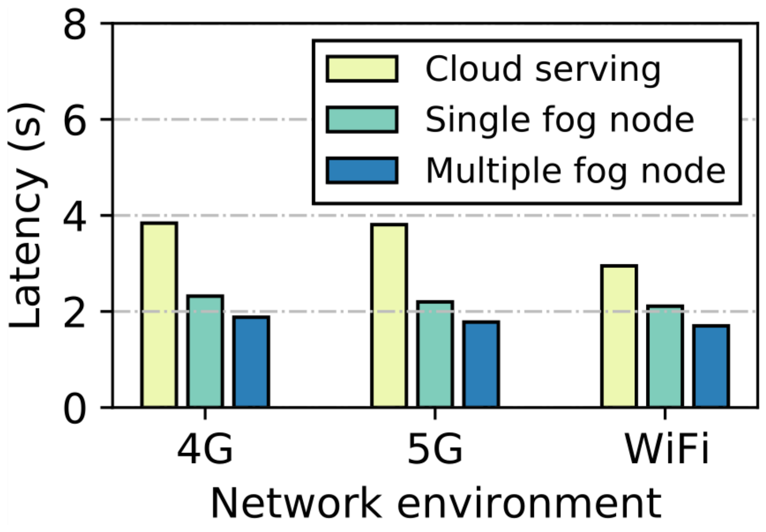
Status Quo of GNN Serving

- **Cloud-based GNN serving**
  - Data generation from geo-distributed end devices
  - Data collection through fog nodes and wide-area network
  - GNN processing at a centralized cloud server

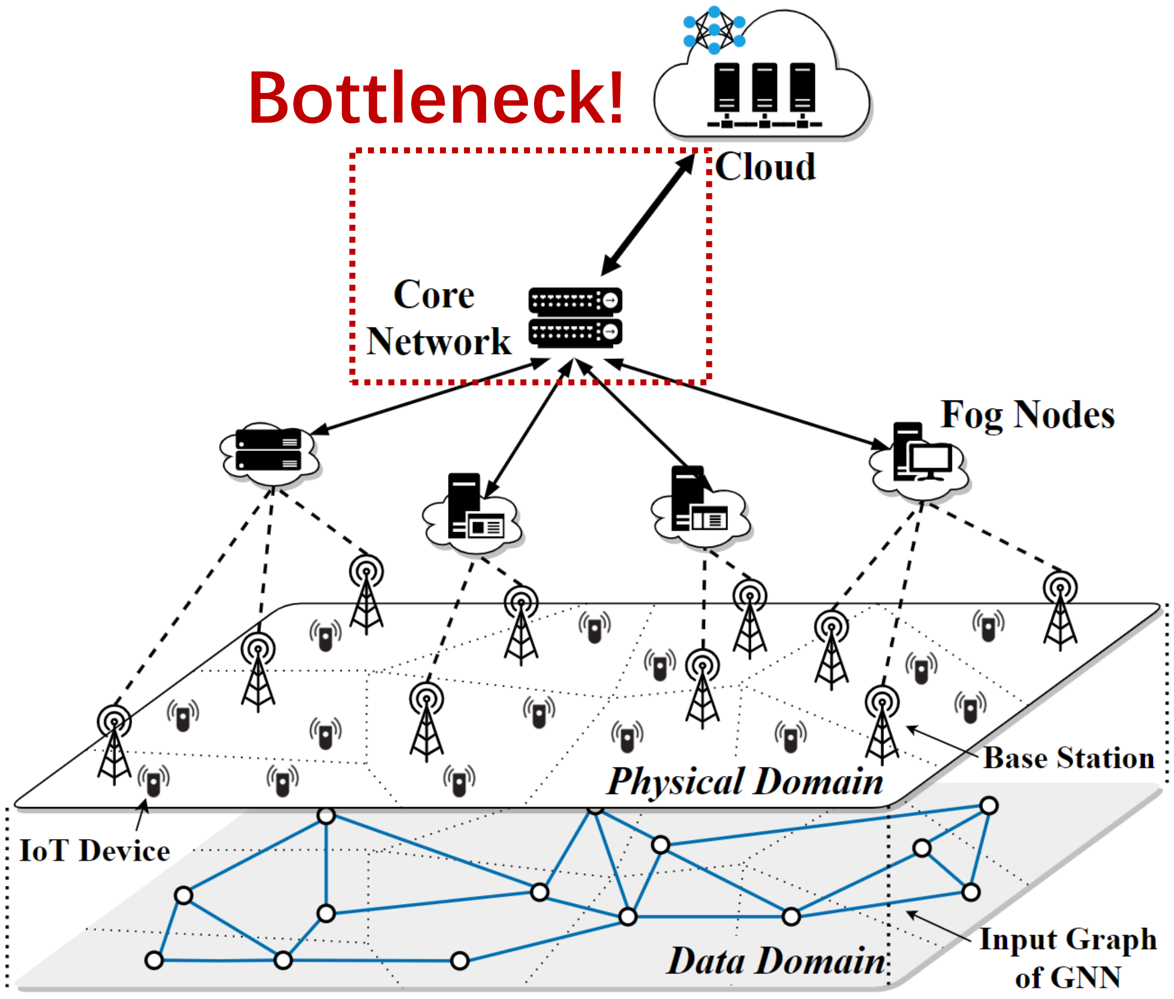


Status Quo of GNN Serving

- **Cloud-based GNN serving**

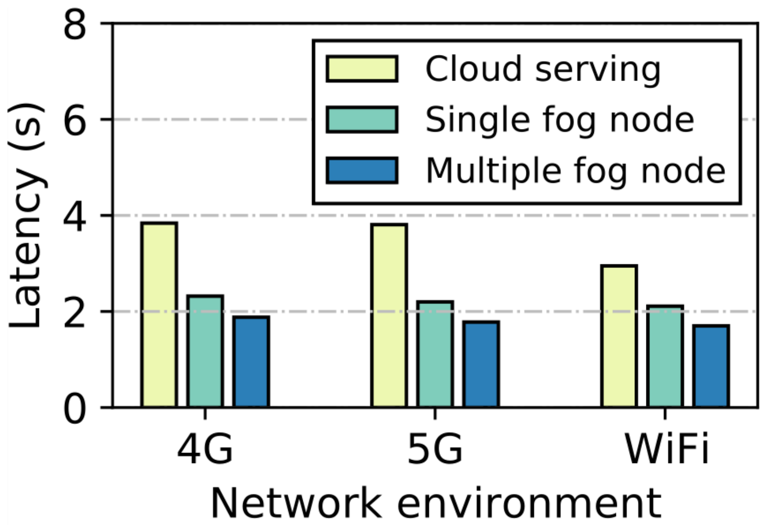


Avoiding remote Internet can reduce at most **53.2%** latency



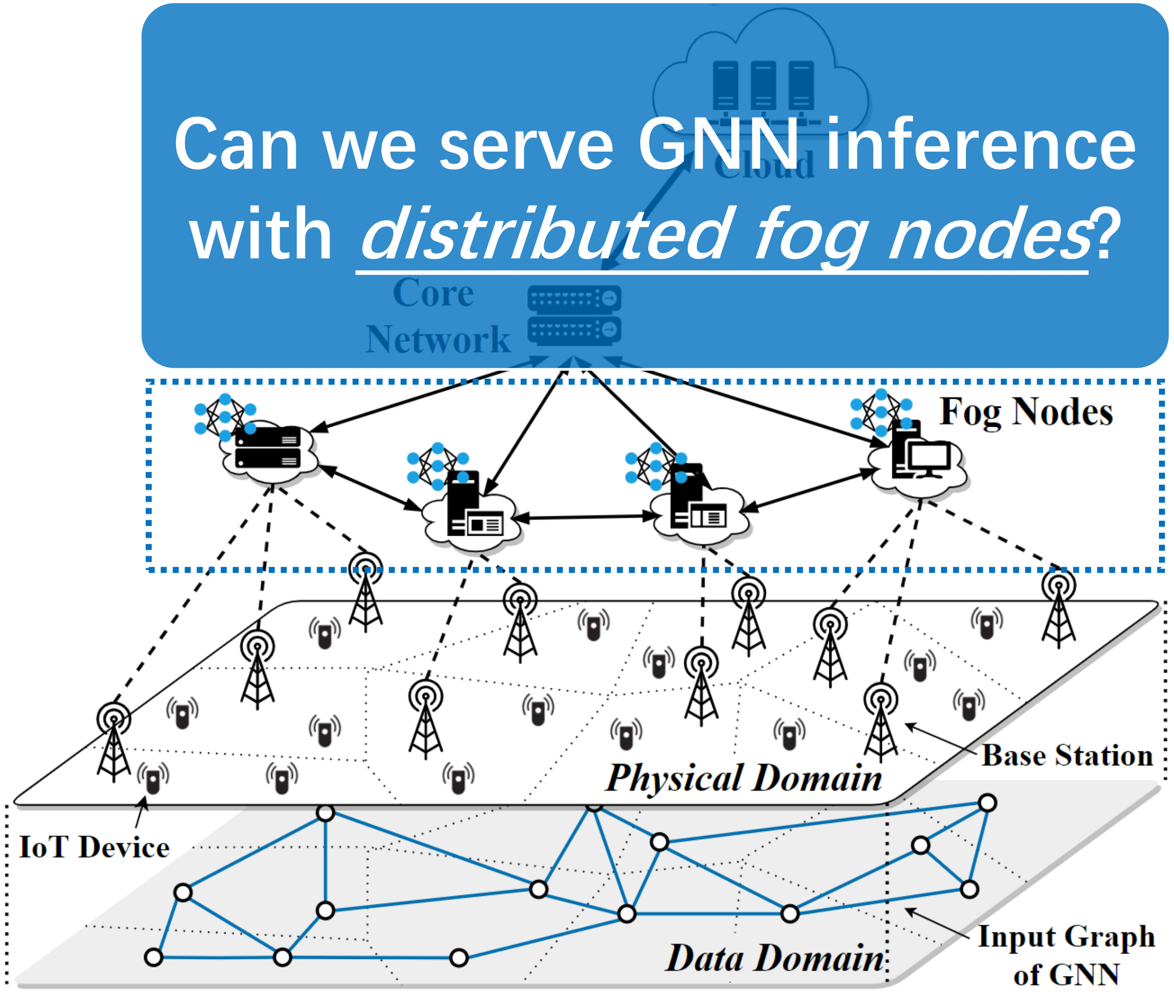
Status Quo of GNN Serving

- **Cloud-based GNN serving**



Avoiding remote Internet can reduce at most **53.2%** latency

Can we serve GNN inference with ***distributed fog nodes***?



Fograph

- **The first fog-enabled distributed GNN inference system**
  - Efficient distributed execution with **resource-aware inference execution planning**
  - Communication-effective data collection via **GNN-specific compression**
  - Better performance: outperform existing cloud serving by up to **5.39x speedup**

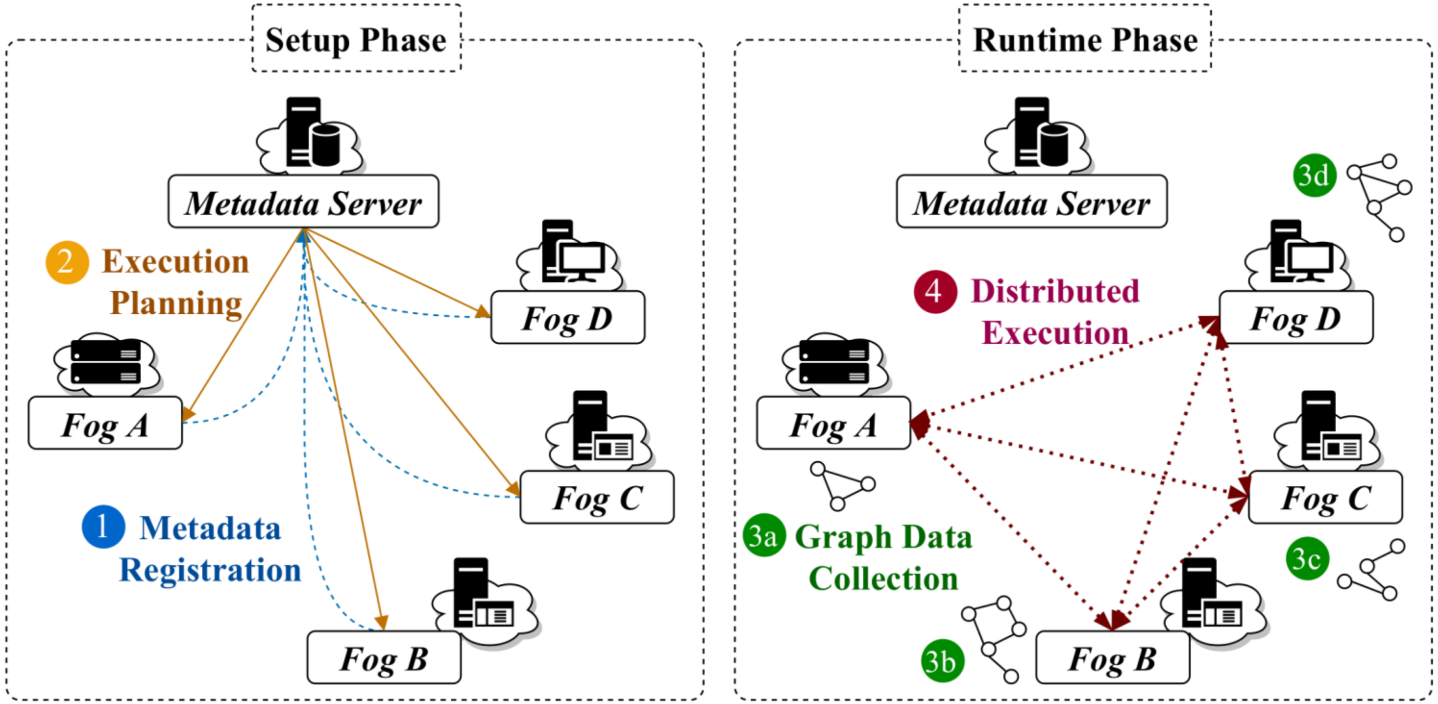
Fograph Overview

- **Workflow**

Metadata Registration → Execution Planning → Graph Data Collection → Distributed Execution

Offline

Online



Metadata Registration

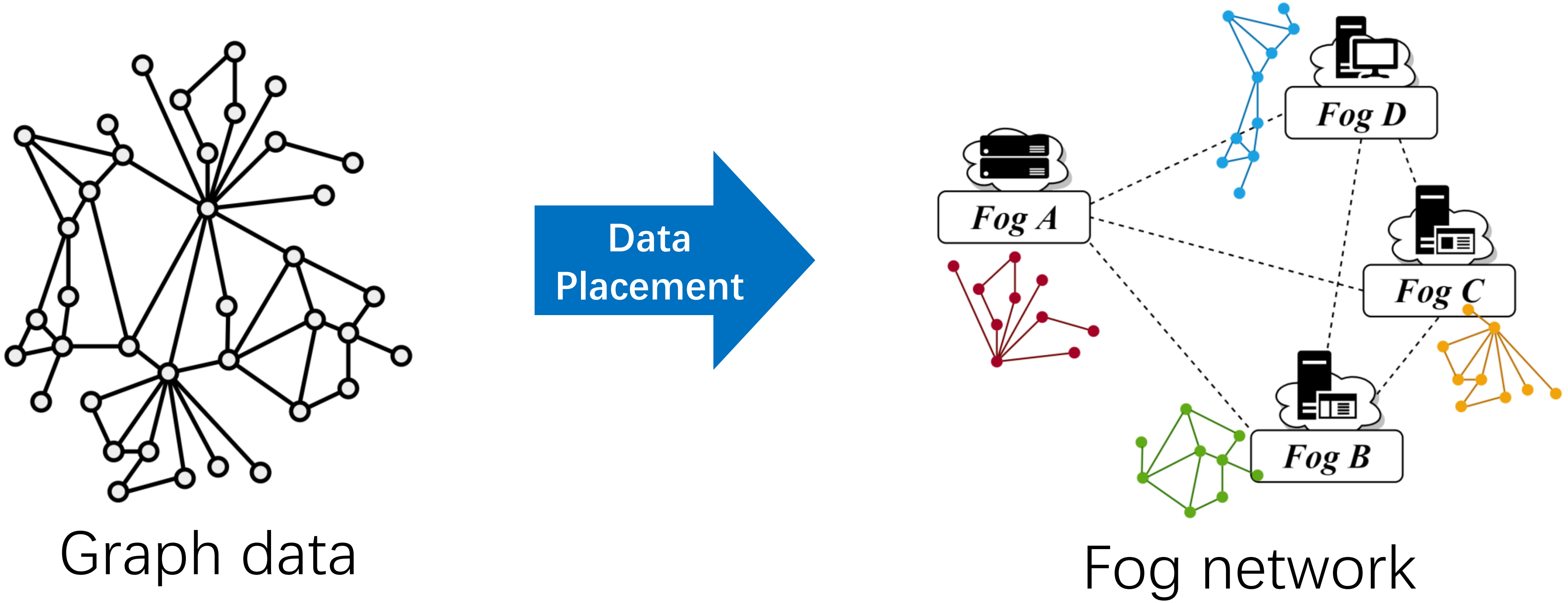
- **Goal**
  - Provision fundamental model configurations
  - Characterize the heterogeneity of fog nodes

- **Metadata**
  - **Device-independent**: Parameters determined in a trained given GNN model
    - Adjacency matrix, size of feature vectors, etc.
  - **Device-dependent**: Computing capability profiles specific to each fog node
    - A regression-based latency estimation model that accepts a graph and predicts its execution latency

Inference Exec. Planning

Metadata Registration → **Execution Planning** → Graph Data Collection → Distributed Execution

- **Goal:** optimize end-to-end latency for **data collection** and **distributed execution**
  - Decide a *graph data placement* to direct the data flow from end devices to fog nodes



Inference Exec. Planning

Metadata Registration → **Execution Planning** → Graph Data Collection → Distributed Execution

- **Goal:** optimize end-to-end latency for **data collection** and **distributed execution**

The corresponding problem is NP-hard!

- **Insight 1: Efficient distributed execution** desires load balance and minimized cross-server data exchange

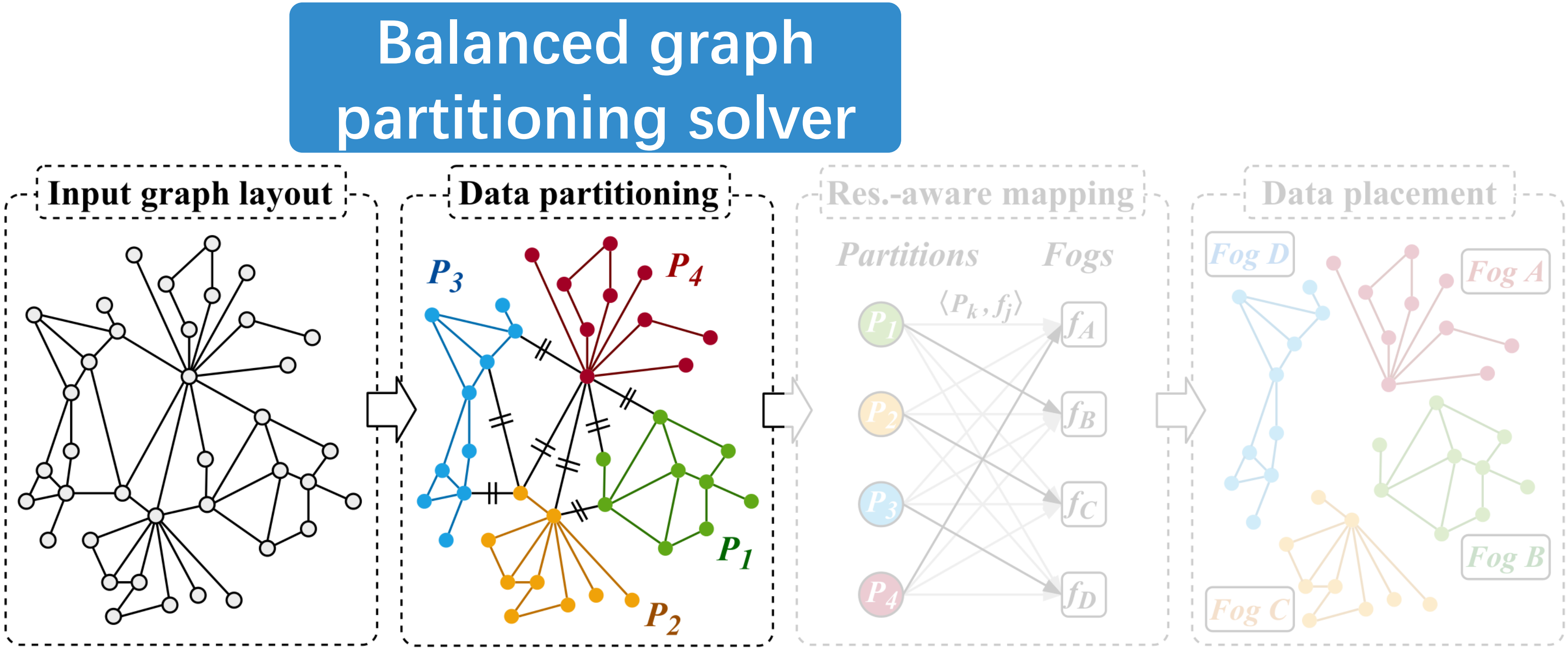
  ➡ **Locality-preserved graph partitioning**

- **Insight 2: Efficient placement** requires jointly considering fog nodes' computing capabilities and available bandwidth

  ➡ **Resource-aware partition-fog mapping**

Inference Exec. Planning

- **Inference Execution Planning Algorithm**
  - **Key 1:** Locality-preserved graph partitioning

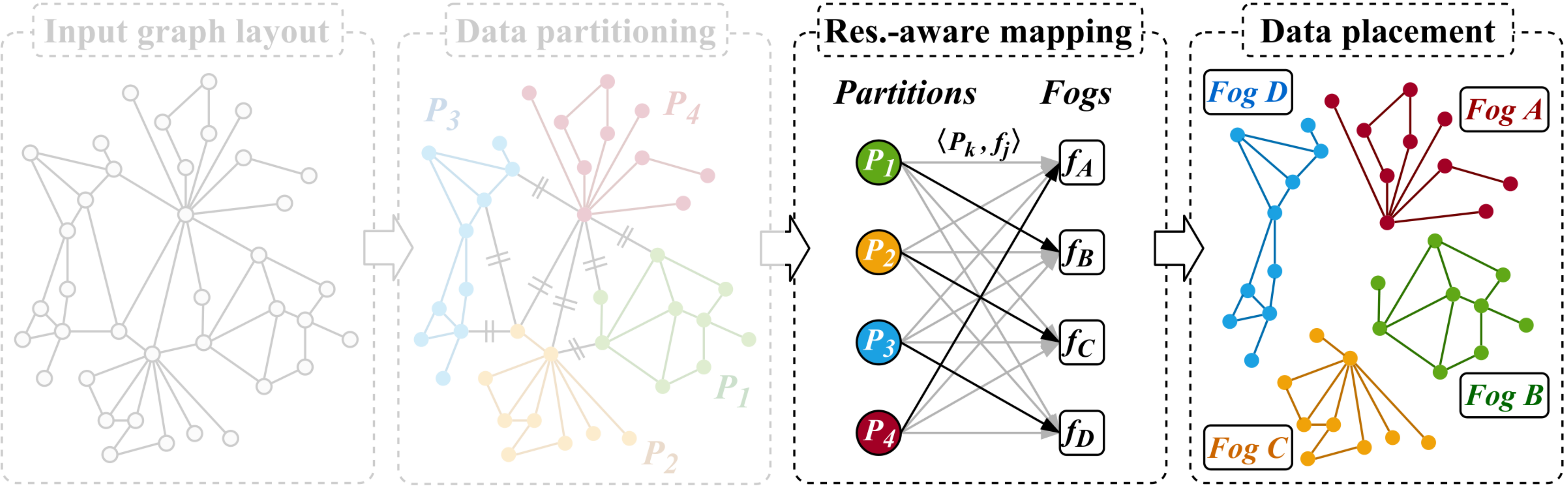


Inference Exec. Planning

Metadata Registration → **Execution Planning** → Graph Data Collection → Distributed Execution

- **Inference Execution Planning Algorithm**
  - **Key 1:** Locality-preserved graph partitioning
  - **Key 2:** Resource-aware partition-fog mapping

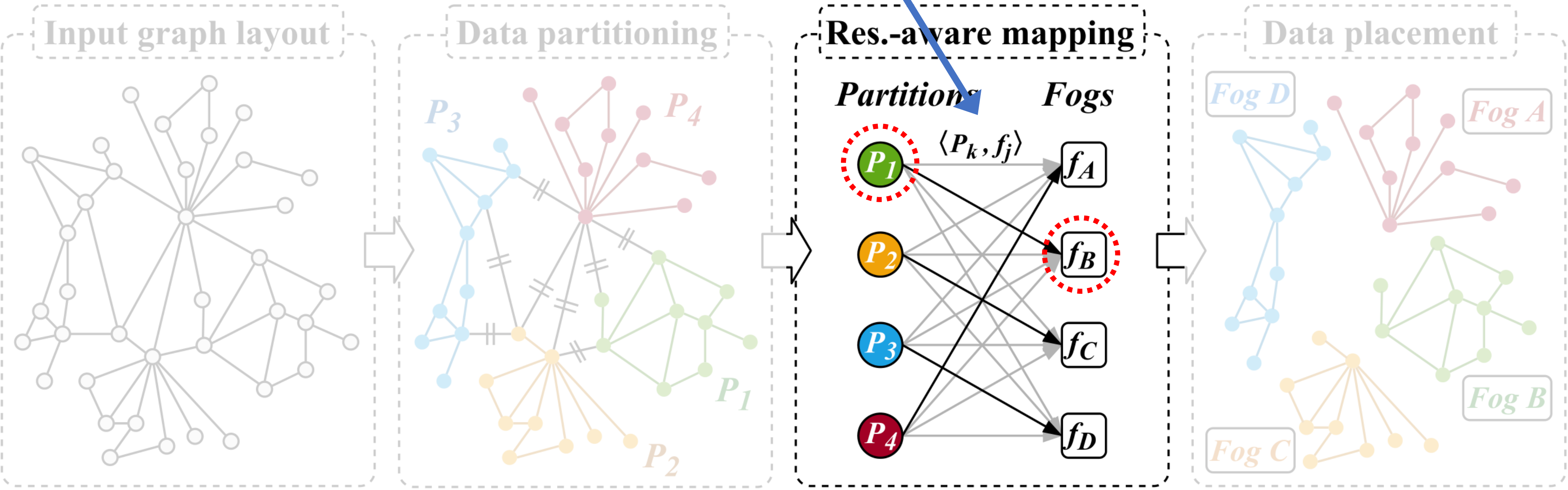
Greedy min-weight mapping



Inference Exec. Planning

- **Inference Execution Planning Algorithm**
  - **Key 1:** Locality-preserved graph partitioning
  - **Key 2:** Resource-aware partition-fog mapping

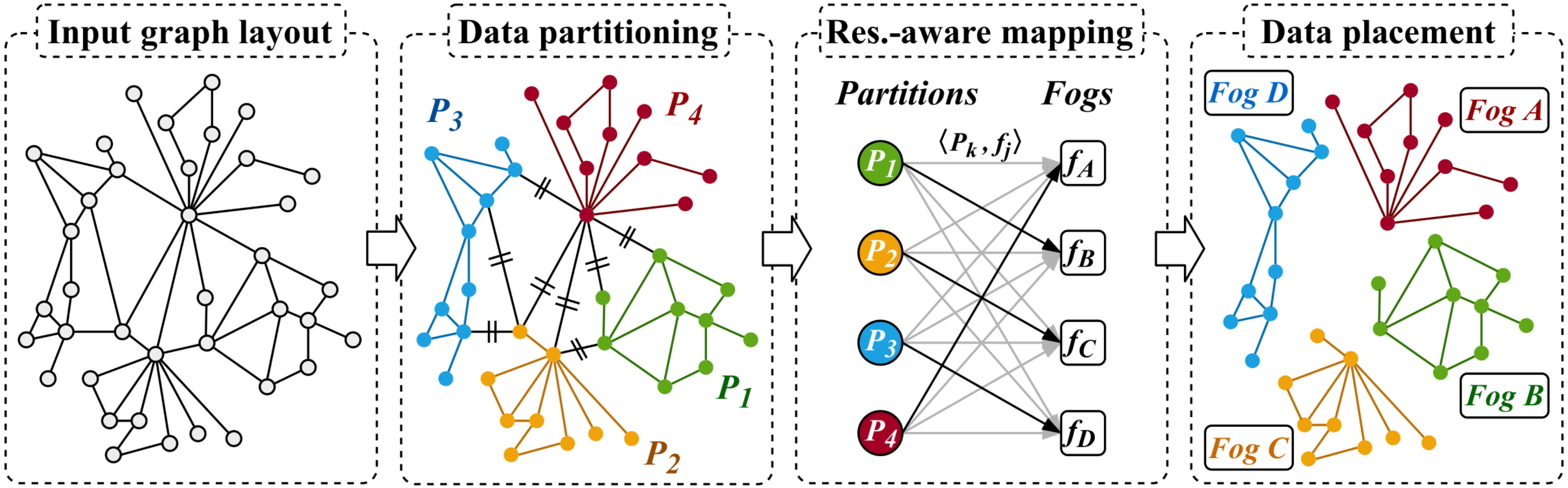
$\langle P_k, f_j \rangle$ = Data Collection Latency + Computation Latency



Inference Exec. Planning

- **Inference Execution Planning Algorithm**
  - **Key 1:** Locality-preserved graph partitioning
  - **Key 2:** Resource-aware partition-fog mapping

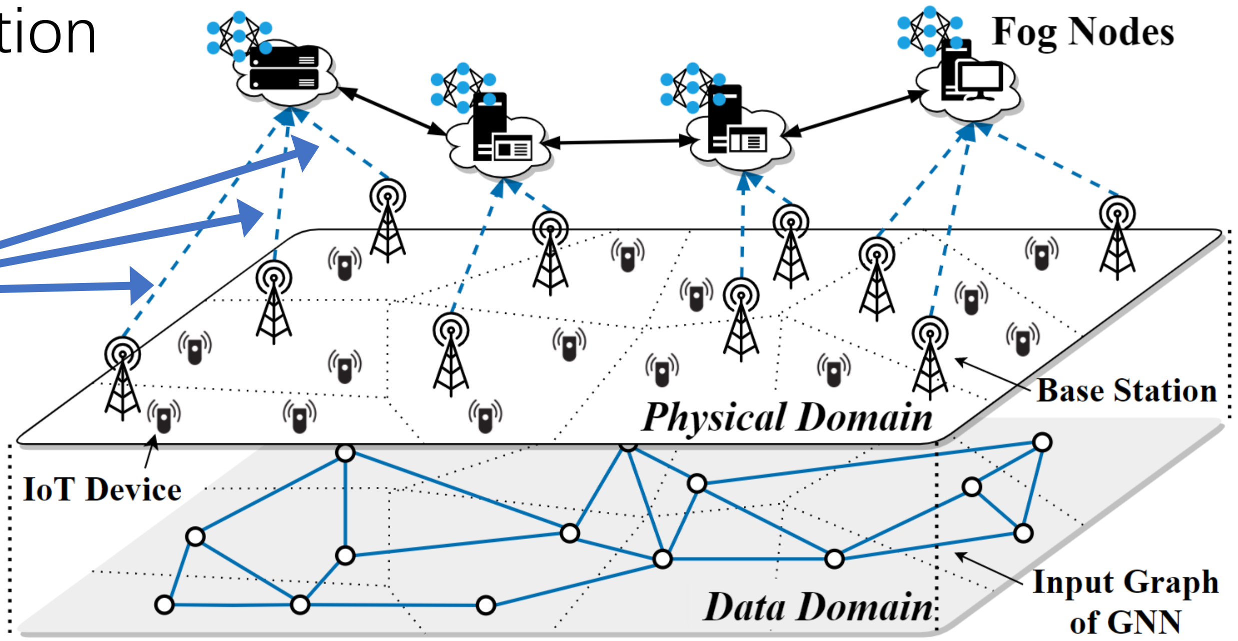
Output: a resource-aware data placement



Data Collection

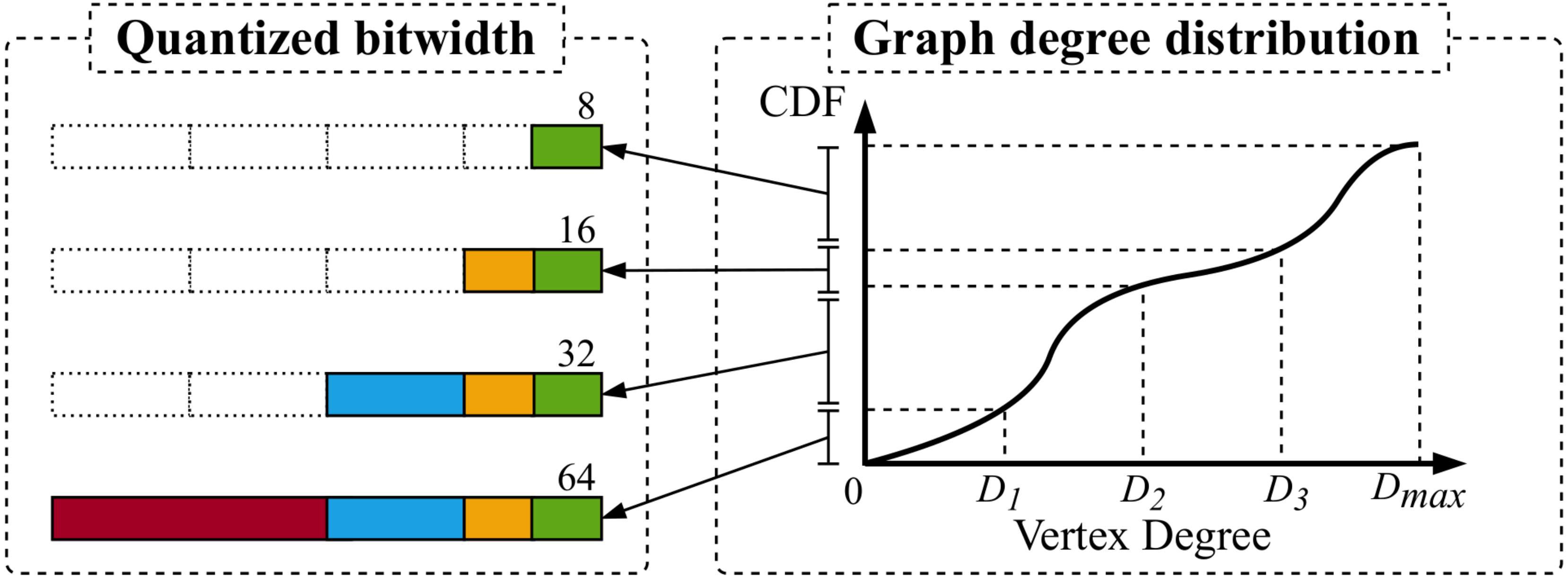
- **Goal:** Communication-effective data transmission
  - **Quantization:** Degree-aware quantization
  - **Compression:** Sparsity elimination

Quantization & Compression



Data Collection

- **Quantization:** Degree-aware quantization
  - GNNs are resilient to low-precision representation [Tailor, et al.]
  - A vertex with a higher degree is more robust to low bit widths [Feng, et al.]



[Tailor, et al.] Degree-Quant: quantization-aware training for graph neural networks, ICLR 2020.
[Feng, et al.] SGQuant: Squeezing the last bit on graph neural networks with specialized quantization, ICTAI 2020.

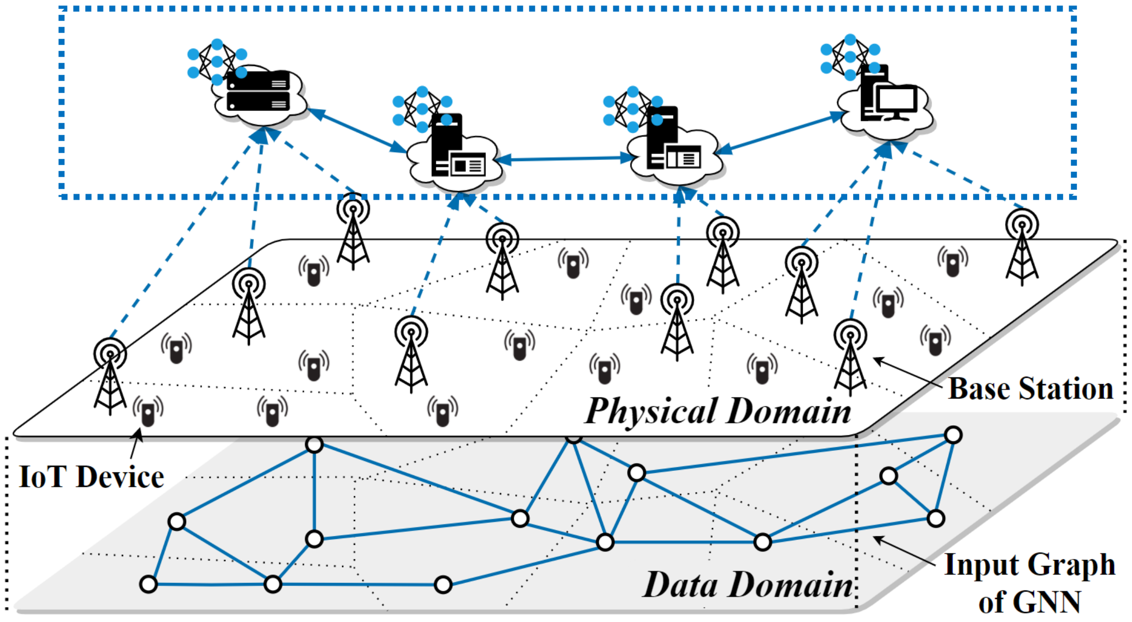
Data Collection

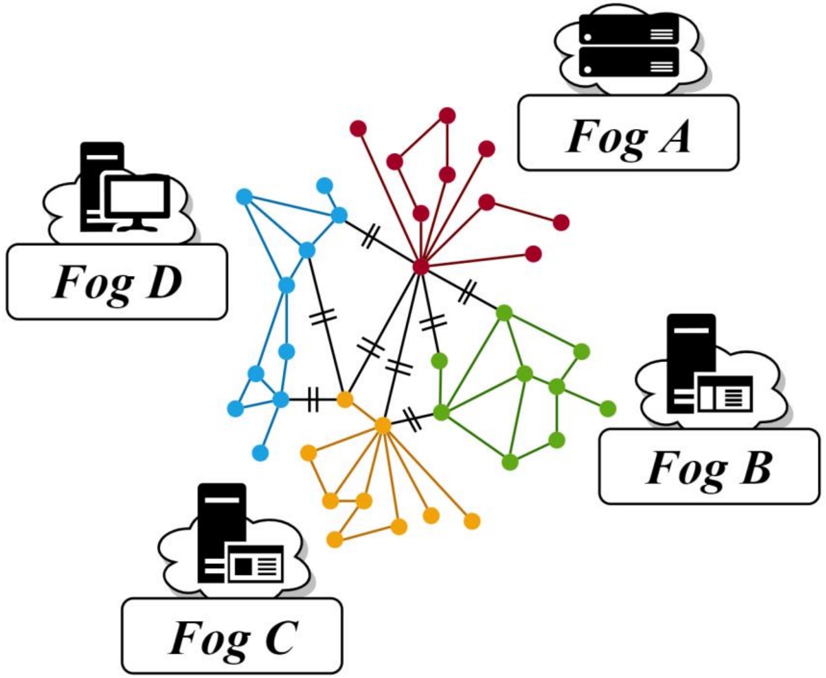
- **Quantization:** Degree-aware quantization
  - GNNs are resilient to low-precision representation [Tailor, et al.]
  - A vertex with a higher degree is more robust to low bit widths [Feng, et al.]
- **Compression:** Sparsity elimination
  - A major fraction of feature vectors are sparse
  - The sparsity is further magnified by precision reduction after quantization

Distributed Execution

Metadata Registration → Execution Planning → Graph Data Collection → **Distributed Execution**

- **Computation**
  - **Bulk Synchronous Parallel** model for iterative layer processing
- **Communication**
  - **Neighbor data exchange** through message passing across fog nodes





Evaluation

- **Models:** GCN, GAT, GraphSAGE
- **Baselines:** cloud serving, vanilla fog serving
- **Datasets**

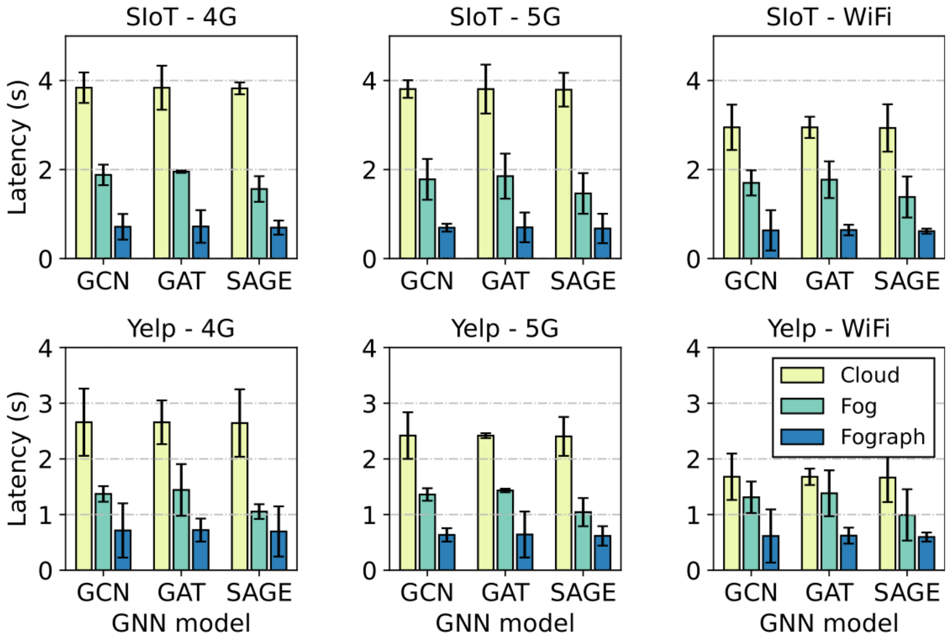
Dataset	Vertex	Edge	Feature	Label	Duration
SIoT	16216	146117	52	2	1
Yelp	10000	15683	100	2	1
PeMS	307	340	3	N/A	12

- **Testbed**

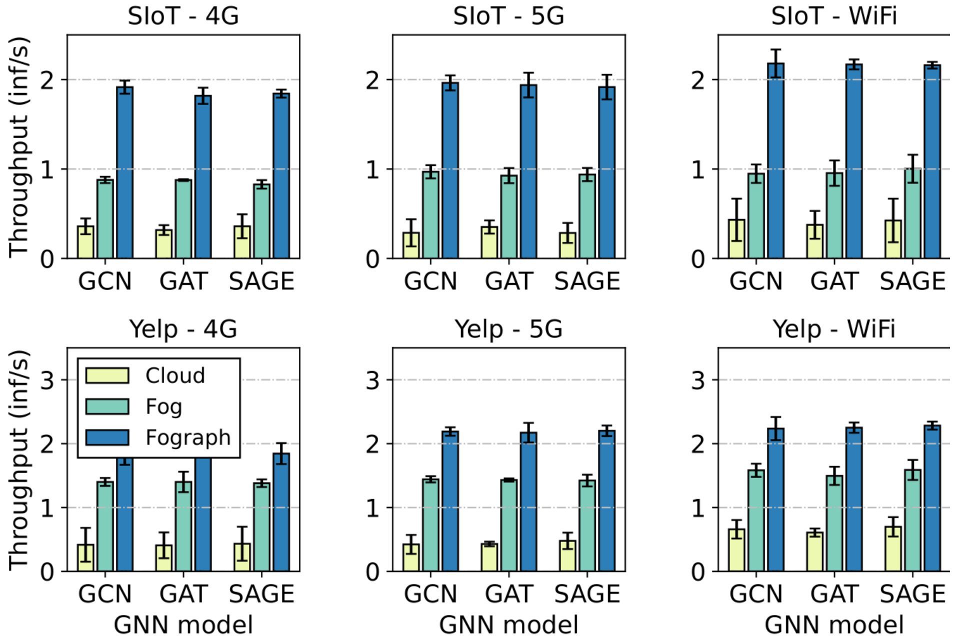
Type	Processor	Memory	Capability
Cloud	8vCPUs & Tesla V100 GPU	32GB	Highly Powerful
Fog A	3.40GHz 8-Core Intel i7-6700	4GB	Weak
Fog B	3.40GHz 8-Core Intel i7-6700	8GB	Moderate
Fog C	3.70GHz 16-Core Xeon W-2145	32GB	Powerful

Performance Comparison

- Six fog nodes: 1xA, 4xB, 1xC
- **Latency reduction** up to 82.18% and 63.70% for SIoT and Yelp
- **Throughput improvement** up to 6.84x and 2.31x for SIoT and Yelp



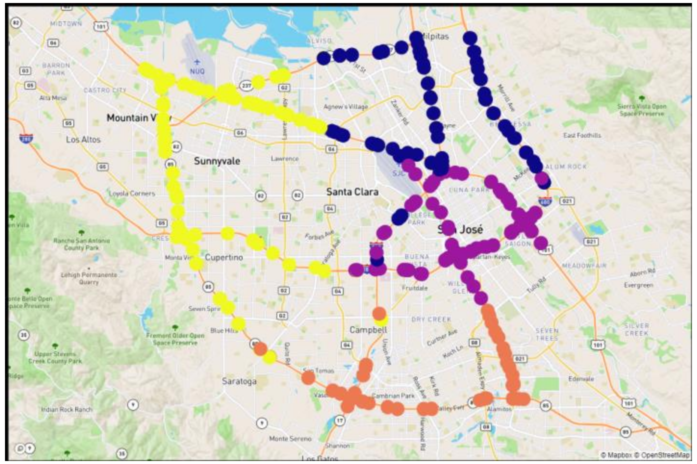
Latency results



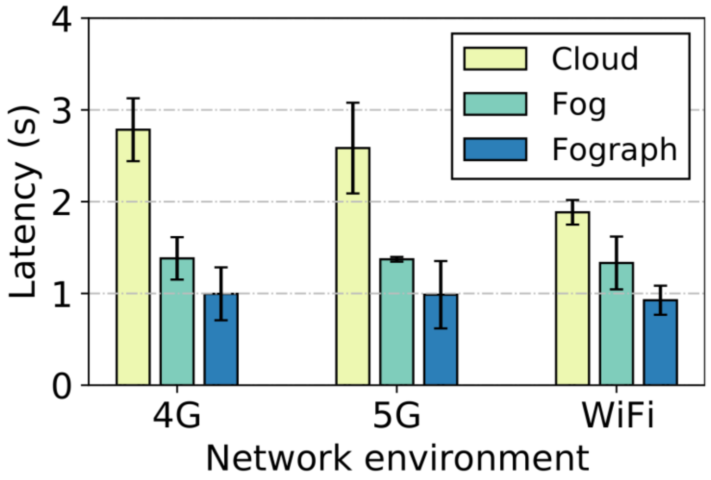
Throughput results

Case Study

- Traffic flow forecasting with ASTGCN model and PeMS dataset
- Four fog nodes: 1xA, 2xB, 1xC
- **Heterogeneity-aware** data placement
- **Inference speedup** up to 2.79x and 1.43x over cloud and fog



Sensor distribution



Latency result

Accuracy Results

- **Minimal accuracy drops** by <0.1% for SIoT and Yelp
- **Tiny error expansion** of ~0.1 for traffic flow forecasting

Inference accuracy on SIoT and Yelp

Method	**SIoT (%)**			**Yelp (%)**		
	GCN	**GAT**	**SAGE**	**GCN**	**GAT**	**SAGE**
Cloud	89.98	86.08	95.50	92.19	86.30	91.73
Fog	89.98	86.08	95.50	92.19	86.30	91.73
Fograph	89.97	86.08	95.48	92.12	86.20	91.70

Traffic flow forecasting errors

Method	**15min**			**30min**		
	MAE	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
Cloud	17.71	29.92	11.84	18.66	30.97	12.27
Fog	17.71	29.92	11.84	18.66	30.97	12.27
Fograph	17.75	30.05	11.93	18.73	31.12	12.38
Uni. 8-bit	18.79	30.26	12.97	19.74	32.01	13.38

Fograph

- **The first fog-enabled distributed GNN inference system**
  - **Efficient distributed execution** with **resource-aware inference planning**
  - **Communication-effective data collection** via **GNN-specific compression**
  - **Better performance**: outperform existing cloud serving by **5.39x speedup**

Thanks!

- **Liekang Zeng**, Peng Huang, Ke Luo, Xiaoxi Zhang, Zhi Zhou, Xu Chen

zenglk3@mail2.sysu.edu.cn, Sun Yat-sen University