MGG: Accelerating Graph Neural Networks with Fine-grained Intra-kernel Communication-Computation Pipelining on Multi-GPU Platforms

Yuke Wang, Boyuan Feng, Zheng Wang,
*Tong Geng, ^Kevin Barker, ^Ang Li, Yufei Ding.

*: University of Rochester, ^:Pacific Northwest National Laboratory
University of California, Santa Barbara
Graphs are everywhere, and GNN is the key!

Social Networks  Power Grid  Financial Services  Molecular Biology

Credict: Google Image
GNN: Graph Neural Networks

GNN Layer = Graph Ops + DNN Ops

Model view

Operation view
GNNs are Scaling Up!

- **1. Graph Structural Scaling.**
  - Rich structural information (e.g., Community) for various tasks (e.g., link prediction)
  - ogbn-paper100M has 111M nodes and 1.6B edges

- **2. Graph/Node Embedding Scaling.**
  - Fruitful Node/Graph-level Properties for various tasks (e.g., node classification)
  - Reddit graphs has 602 embedding dimension
DL Infrastructures are Scaling to Catch Up!

- Single GPU
- NVIDIA A100
- GPU SuperPod
- NVIDIA DGX-pod

**Building Blocks for Large Clusters**

- High Comp/Mem Capacity
- High Comm. Bandwidth
Powerful Multi-GPU Platforms Cannot Solve Everything!

- Single-GPU solution does not work well for multiple GPUs. (e.g., DGL):
  - Communication and computation in separated phases.
  - Remote neighbor access are fine-grained and irregular.

Weakness:
- High individual neighbor access cost.
- GPU idleness between the computation and communication phases.

Traditional Distributed Graph Solutions Do Not Work Well!

- Schedule Transformation for Dense Communication (e.g., NeuGraph, P3, ROC):
  - Neighbor aggregation is divided into multiple rounds.
  - Neighbor movement are dense, regular and coarse-grained.
  - Neighbor access in each round of aggregation is all local.

- Weakness:
  - Additional algorithmic modification.
  - Redundant data movements.
  - Decreased computation efficiency.

---

Algorithm-Observation: Opportunity for Fine-grained Pipelining

**New opportunities:** we can amortize communication costs by fine-grained overlapping neighbor aggregation from different nodes.
Hardware-Observation: SM Multiplexing for Operation Overlapping
Hardware-Observation: SM Multiplexing for Operation Overlapping


12oz.  16oz.  22oz.  32oz.
MGG Overview

Novel pipeline view/abstraction for hiding communication latency

Sparse and Irregular GNN Workloads

Tackling Input Dynamicity

Tackling Hardware Diversity

Fine-grained and balanced pipelines

Runtime Param. Optimizer

NVIDIA DGX
Contribution-1: Pipeline View and Abstraction.

Key insight: Communication overhead can be offset by fine-grained operation overlapping.

Challenges:
- Communication overweight the local computations/access and dominate the execution.
- Communication exacerbate the workload imbalance.

- The idle cycles of GPUs communication can be fulfilled by other local computing.
- Multi-GPU GNN workload can be abstracted as a fine-grained dynamic software pipeline.
A three-stage dynamic software pipeline.

1. Loading remote neighbors (LR)
2. Loading local neighbors (LL)
3. Aggregation computation (AC)
Contribution-2: Pipeline-aware Workload Management.

Key Insight: Pipeline can be tailored for maximizing efficiency based on diverse GNN inputs properties (e.g., #nodes, #edges and node degree).

Input Dynamicity

Challenge: Input diversity (e.g., graph size/sparsity) would affect the pipeline efficiency (e.g., bubble ratio).

Heterogeneity & Granularity-aware pipeline enhancement.
Facilitate a more balanced workload distribution among pipeline stages.

Heterogeneity & Granularity-aware Pipeline Enhancement.
Contribution-3: GPU-Aware Pipeline Mapping.

Hardware Diversity

Challenges: Hardware diversity (e.g., Comp/Comm Speed) would affect the pipeline execution performance (e.g., SM utilization and occupancy).

Key insight: Dynamically configurable pipeline-workload-to-SM mapping can maximize pipeline execution performance.

Intelligent Runtime Design.

Specialized Memory Design & Optimization.
Evaluation

- **GNN Models.**
  - Graph Convolutional Network (GCN): 2 layers with 16 hidden dimensions.
  - Graph Isomorphism Network (GIN): 5 layers with 64 hidden dimensions.

- **Evaluation Platform.**
  - NVIDIA DGX-A100 with dual AMD Rome 7742 processors (each with 64 cores, 2.25 GHz), 1TB host memory, and 8x(A100-40GB) connected via NVSwitch.

- **Baseline.**
  - Deep Graph Library [ICLR’19].
  - Unified Memory [NVIDIA].
  - ROC [MLSys’20].

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Vertex</th>
<th>#Edge</th>
<th>#Dim</th>
<th>#Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>reddit(RDD) [45]</td>
<td>232,965</td>
<td>114,615,892</td>
<td>602</td>
<td>41</td>
</tr>
<tr>
<td>enwiki-2013(ENWIKI) [23]</td>
<td>4,203,323</td>
<td>202,623,226</td>
<td>300</td>
<td>12</td>
</tr>
<tr>
<td>it-2004 (IT04) [10]</td>
<td>41,291,594</td>
<td>1,150,725,437</td>
<td>256</td>
<td>64</td>
</tr>
<tr>
<td>ogbn-paper100M(PAPER) [12]</td>
<td>111,059,956</td>
<td>1,615,685,872</td>
<td>128</td>
<td>64</td>
</tr>
<tr>
<td>ogbn-products(PROD) [17]</td>
<td>2,449,029</td>
<td>61,859,140</td>
<td>100</td>
<td>47</td>
</tr>
<tr>
<td>ogbn-proteins(PROT) [17]</td>
<td>132,534</td>
<td>39,561,252</td>
<td>8</td>
<td>112</td>
</tr>
<tr>
<td>com-orkut(ORKT) [23]</td>
<td>3,072,441</td>
<td>117,185,083</td>
<td>128</td>
<td>32</td>
</tr>
</tbody>
</table>
Evaluation: Overall Performance

- **Compare with DGL.** (Separated Communication and Computation)
  - Averaged 4.41x speedup in comparison with DGL on GCN and GIN.

- **Compare with ROC.** (Schedule Transformation for Dense Communication)
  - Averaged 10.83x speedup in comparison with ROC (8xA100) on GCN and GIN.
Evaluation: Additional Comparisons

- Compare with MGG-UVM.
  (Pipeline with Coarse-grained Communication)

  ![Graph](image1.png)
  (a) GCN Model.

  ![Graph](image2.png)
  (b) GIN Model.

  Averaged 4.81x speedup in comparison with MGG-UVM on GCN and GIN.

- Neighbor Partitioning (input Dynamicity)

  ![Bar Chart](image3.png)

  Averaged 2.24x with NP

- Workload Interleaving (Hardware Diversity)

  ![Bar Chart](image4.png)

  Averaged 1.88x with WL
Contribution Summary

- Exploiting the joint optimization of the communication and computation.
- Capitalizing pipelining benefits for input dynamicity.
- Enhance pipeline efficiency for diverse hardware.

- A novel and unique multi-stage pipeline view/abstraction
- GNN-tailored pipeline construction
- GPU-aware pipeline mapping
Thank You

Q & A

Contact: yuke_wang@cs.ucsb.edu