Hi!

CS 744: MARIUS

Shivaram Venkataraman
Fall 2021
- Midterm grades today! → Pick up papers in my office hours by Monday.
- Course Project: Check in by Nov 30th
- Don't panic!
- On the class on Tuesday
  - Skip the discussion
  - Solution walkthrough
EXAMPLE: LINK PREDICTION

Task: Predict potential connections in a social network

Find K-nearest neighbors

- [0.25, 0.45, 0.30]
- [0.15, 0.85, 0.92]
- ...

Captures the structure of this graph

List of recommendations

Embedding of recommendations

One embedding per vertex
BACKGROUND: GRAPH EMBEDDING MODELS

Score function  
for a particular edge
Capture structure of the graph given source, destination embedding

Loss function
Maximize score for edges in graph
Minimize for others (negative edges)

\[ \mathcal{L} = \sum_{e \in G} \sum_{e' \in S_e} \max(f(e) - f(e') + \lambda, 0) \]
**TRAINING ALGORITHM**

- **SGD/AdaGrad optimizer**
- **Sample positive, negative edges**
- **Access source, dest embeddings for each edge in batch**

```
for i in range(num_batches):
    B = getBatchEdges(i)
    E = getEmbeddingParams(B)
    G = computeGrad(E, B)
    updateEmbeddingParams(G)
```

Rennet inputs take a batch of edges

1 epoch = all positive edges

return the embeddings at the end.
CHALLENGE: LARGE GRAPHS

Large graphs $\rightarrow$ Large model sizes

Example

3 Billion vertices, $d = 400$
Model size = $3 \text{ billion} \times 400 \times 4 = 4.8 \text{ TB}$!

Need to scale beyond GPU memory, CPU memory!
CHALLENGE: DATA MOVEMENT

(a) Sample edges, embeddings from CPU memory (DGL-KE)

(b) Partition embeddings so that one partition fits on GPU memory. Load sequentially (Pytorch-BigGraph)

Data movement overheads $\rightarrow$ low GPU util
MARIUS

I/O efficient system for learning graph embeddings

Marius Design
- Pipelined training
- Partition ordering
PIPELINED TRAINING

Splits the algorithm into 5 stages and pipelines stages to utilize CPU & GPU & PCIe at same time.

```
<table>
<thead>
<tr>
<th>Edges</th>
<th>Load</th>
<th>Transfer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node Embedding Parameters</td>
<td>Update</td>
<td>Transfer</td>
</tr>
</tbody>
</table>
```

CPU Memory

GPU Memory

Queue size controls staleness!

Edges, embs, params

Relation Embedding Parameters

If $b_2$ has shared embeddings with $b_1$, then $b_2$'s embeddings are stale.

Sparse access patterns not too many conflicts

Stale data!

If $b_2$ has shared embeddings with $b_1$, then $b_2$'s embeddings are stale.
OUT OF MEMORY TRAINING

Key idea: Maintain a cache of partitions in CPU memory

Questions
- Order of partition traversal?
- How to perform eviction?

Adjacency matrix of my graph

Caching in CPU memory

Node embedding parameters

Partitions in Buffer

Partitions on disk
ELIMINATION ORDERING

1. Initialize cache with c partitions
2. Swap in partition that leads to highest number of unseen pairs
3. Achieved by fixing c-1 partitions and swap remaining in any order

number swaps close to the lower bound
SUMMARY

Graph Embeddings: Learn embeddings from graph data for ML

Marius: Efficient single-machine training
  Pipelining to use CPU, GPU
  Partition buffer, BETA ordering $\rightarrow$ minimize number of swaps
DISCUSSION

https://forms.gle/LtoT8nEmw3oLvXuo9
If you were going to repeat the COST analysis for knowledge graph embedding training, what would you expect to find and why?

- Distributed systems need 4-8 GPUs to match a **Single GPU**
- Marius 1 GPU
- Comm. cost if we go to many GPUs

KG training: compute -> more compute for PageRank

- I/O the bigger bottleneck?!
- I/O data dependent
How does the partitioning scheme used in this paper differ from partitioning schemes used in PowerGraph and why?

Workload was focused on

- vertex and its neighbors → PageRank job

- edge drives the computation

- source, dest for this edge
Next class: New module!
Project check-ins by Nov 30th