ByteGAP: A Non-continuous Distributed Graph Computing System using Persistent Memory

Miaomiao Cheng, Jiujian Chen, Cheng Zhao, Cheng Chen, Yongmin Hu, Xiaoliang Cong, Liang Qin, Hexiang Lin, Rong Hua Li, Guoren Wang, Shuai Zhang and Lei Zhang

Douyin Vision Co., Ltd. & Beijing Institute of Technology
Backgrounds

Risk Control Graphs

Social Networks

E-commercial Graphs

merchants

products

users
Backgrounds

Risk Control Graphs

Social Networks

E-commercial Graphs

Fraud Detective

Centrality Rank

Vertex Classification/Clustering
Motivations

Graph Computation Tasks Run *Periodically* in ByteDance

- **Lending Risk Control:** Repeat In *Hours*
- **Social Recommendation:** Repeat In *Days/Weeks*
- **E-commerce Advertising:** Repeat In *Hours*
- **Fake Fans:** Repeat In *Days*
- **Hotspot Mining:** Repeat In *Minutes*
- **Product Clustering:** Repeat In *Days/Weeks*
- **Centrality Rank**
Motivations

For a task, \( \text{MachineTime} = \#\text{Worker} \times \text{End2End Time} \)
Motivations

Graph Computing #Task-MachineTime Distribution in ByteDance

- **Small-scale tasks**
  - Completed < 1000s
  - > 90% of all tasks
  - < 10% computing resources
### Motivations

Graph Computing #Task-MachineTime Distribution in ByteDance

![Graph](image)

- **Large-scale tasks**
  - Completed > 1000s
  - < 10% of all tasks
  - > 90% computing resources
Motivations

Graph Computing #Task-MachineTime Distribution in ByteDance

Small-scale tasks
- Completed < 1000s
- > 90% of all tasks
- < 10% computing resources

Large-scale tasks
- Completed > 1000s
- < 10% of all tasks
- > 90% computing resources
Problems

Ideal...

Task 1

Cluster 1

Period $N$  

Period $N+1$

Time

Task 2

Cluster 2

Period $N$  

Period $N+1$  

Period $N+2$  

Period $N+3$

Time

In real deal...

Computing resources are scarce.  
A cluster needs to handle multiple tasks.
Problems

In real deal…

Task 1
- Comp (Period 0)
- Waiting...
- Comp (Period 1)

Period 0 → Period 1

Task 2
- Waiting...
- Comp (Period 0)
- Waiting...
- Comp (Period 1)
- Waiting...
- Comp (Period 2)
- Waiting...

Period 0 → Period 1 → Period 2 → Period 3

Cluster 0 → Time

Task2 (P0): Expected to be finished before T0

Task2 (P0): Real finishing time is delayed to T1

Task2 (Px): The effects persist…

T0 → T1
Problems

In real deal…

Rate = \frac{\text{End2End Execution Time}}{\text{Waiting Time} + \text{End2End Execution Time}}

- Waiting too much!
Problems

Ideal...

Task 1

Cluster

Time

If failover ...

Task 1

Failover

Comp (Period 0)

Restart Comp (Period 0)

Waiting...

Comp (Period 1)

Cluster

Time

T0

T1

Task1 (P0): Expected to be finished before T0
Task1 (P0): Real finishing time is delayed to T1
Task1 (Px): The effects persist...
Problems

ByteDance’s Graph Data
- |V|: 10 Billions+
- |E|: 1 Trillions+

Graph Computing Systems
- Distributed
- In-Memory

Naïve Checkpointing
- 10s TB~100s TB
- Ignore Iterativeness

Out-of-core Graph Computing Systems
- How to Checkpoint & Recover?
Problems

Intel® Optane™ Persistent Memory (PMEM)

Capacity

DRAM: 4GB - 128GB

PMEM: 128GB - 512GB

Complex Data Access Types

- Vertex States
- Edge Data
- Message Data
Example: How to run a task in ByteGAP?

- Load Graph
- Initiate
- Repeat: Iteration & Checkpointing
- Dump Results
ByteGAP Overview: Examples

Computing Kernel
CP-Agent
Persistent Memory Manager

Example: How to run a task in ByteGAP?

Example: How to recover a failed/interrupted task in ByteGAP?

- Read Last Checkpoint ID
- Read Local Checkpoints
- Continue…

① Read LCI & Broadcast
② Read Local Checkpoints
③ Continue Processing…

Disk / HDFS / ...

Compute() Combine() Aggregate()

Application

Graph Loader Result Dumper

Batch Processing

Thread Pool

Sender

Communication

Receiver

Vertex Table

Checkpoint Manager

Computing Kernel

Persistent Memory Manager

Edge Table

Message Table

Checkpoints

Result Dumper

Graph Loader

CP-Agent
Data Layout

Machine 1
- thread 1
  - 1
  - 2
  - 3
  - 4
  - 5
- thread 2
  - 6
  - 7

Machine 2
- thread 1
  - 8
  - 9
  - 10
  - 11
  - 12
- thread 2
  - 13
  - 14
  - 15
  - 16
  - 17

One Edge Page
- 10
- 12
- 13
- 14
- 15

One Msg Page
- vertex 1
  - msg 5
- vertex 2
  - msg 2
- vertex 3
  - msg 3
- vertex 4
  - msg 1

Vertex Table
- ID
- Value
- Active
- Offset
- ...!
- 1
  - 1.0
  - true
  - 0
- 2
  - 1.0
  - true
  - 2
- 3
  - 1.0
  - true
  - 5
- 4
  - 1.0
  - false
  - 6
- 5
  - 1.0
  - true
  - 8
- 6
  - 1.0
  - true
  - 5

Edge Table
- ...!
- ...!
- ...!
- ...!

Msg Table
- vertex 1
  - msg 5
- vertex 2
  - msg 2
- vertex 3
  - msg 3
- vertex 4
  - msg 1

MEM
- 10
- 12
- 13
- 14
- 15

NVM
- 17
- 18
Data Layout

Build in Load Graph

One Edge Page

Vertex Table

Edge Table

MEM

NVM

One Msg Page

<table>
<thead>
<tr>
<th>vertex</th>
<th>msg</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>12</td>
</tr>
</tbody>
</table>

Message Table

<table>
<thead>
<tr>
<th>msg</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>12</td>
</tr>
</tbody>
</table>
Data Layout

Machine 1

- Thread 1
- Thread 2

Machine 2

- Thread 1
- Thread 2

One Edge Page

- Vertex Table
- Edge Table
- Message Table

Build by Computing Algorithms
Lightweight Distributed Checkpointing

Load Graph:
- Superstep 0: Init
- Superstep 1: Iter 0
- Superstep 2: Iter 1
- Superstep 3: Iter 2
- ...: Iter 3
- Superstep I: Iter
- Dump Results

Leader-Follower Checkpointing:
- Worker 0
- Worker 1
- ...: Worker w
- Local CP
- Last Checkpoint ID (LCI)
- PMEM

Converged / Stopped
Lightweight Distributed Checkpointing

Load Graph

Superstep 0 | Superstep 1 | Superstep 2 | Superstep I
---|---|---|---
Init | Iter 0 | Iter 1 | Iter 2 | Iter 3 | ... | Iter | Iter | Dump Results

Converged / Stopped

CP 0

CP 1

CP 2

CP I

Vertex Table

ID
1
2
3
4
5
6
...
Offset
0
2
5
6
8
5
...

Partition Metadata

#Partition
Part Range
Data Paths

ID
0
2
5
6
8
5
...
Lightweight Distributed Checkpointing
Dual-Mode Persistent Memory Management

- **PMEM device**
  - New chunk
  - Global free list

- **Object Cache**
  - T1
  - T2
  - T3

- **Deallocate**

- **PMEM device**
  - New chunk
  - Large chunk

- **Object Cache**
  - T1
  - T2
  - T3

- **Allocate**

- **Large size allocation**
Dual-Mode Persistent Memory Management

PMEM device

New chunk

Global free list

Object Cache

Object Cache

Object Cache

Deallocate

Deallocate

Deallocate

PMEM device

New chunk

Large chunk

T1

T2

T3

Allocate

Allocate

Large size allocation

T1

T2

T3

Object Cache

Object Cache

Object Cache

Deallocate

Deallocate

Deallocate
Dual-Mode Persistent Memory Management

PMEM device

New chunk

Global free list

Object Cache

Deallocate

Object Cache

Object Cache

Deallocate

Object Cache

Deallocate

PMEM device

New chunk

Large chunk

Allocate

Object Cache

Allocate

Deallocate

Large size allocation

T1

T2

T3

T1

T2

T3
Our Solutions

ByteGAP: Non-continuous distributed graph computing system based on PMEM
ByteGAP: Non-continuous distributed graph computing system based on PMEM

Our Solutions
Our Contributions

**ByteGAP**: Non-continuous distributed graph computing system based on PMEM

Contribution 1: Lightweight iterative distributed checkpointing

Contribution 2: Efficient dual-mode PMEM management
Evaluations

| Dataset       | |V|               | |E|               |
|---------------|----------------|----------------|
| Twitter       | 41,652,230     | 1,468,364,884  |
| Friendster    | 65,608,366     | 1,806,067,135  |
| UK-2007       | 105,896,555    | 3,738,733,648  |
| UK-union      | 133,633,040    | 5,475,109,924  |

Testbed:
- 10 Machines
- Two Intel Xeon Platinum 8260 CPUs (48 cores)
- 128GB of DRAM
- 320GB NVMe SSD INTEL SSDPE2KX020T8
- 512GB Optane DC PMEM

Baselines:
- Spark 3.0 GraphX[1]

Algorithms:
- PageRank (PR)
- Connected Components (CC)

Evaluations: Checkpointing

Run-time Breakdown

Recovery Time

- $T_{iter}$
- $T_{cp0}$
- $T_{cpi}$

<table>
<thead>
<tr>
<th>#Machine</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time/s</td>
<td>19.59</td>
<td>9.88</td>
<td>5.45</td>
<td>7.16</td>
<td>7.26·10^{-2}</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#Machine</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time/s</td>
<td>16.45</td>
<td>8.88</td>
<td>6.05</td>
<td>4.85</td>
<td>4.19</td>
</tr>
</tbody>
</table>

- $T_{rec}$-PMEM
- $T_{rec}$-SSD
Evaluations: Checkpointing

Run-time Breakdown

<table>
<thead>
<tr>
<th>#Machine</th>
<th>Time/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.22</td>
</tr>
<tr>
<td>4</td>
<td>9.88</td>
</tr>
<tr>
<td>6</td>
<td>12.27</td>
</tr>
<tr>
<td>8</td>
<td>5.45</td>
</tr>
<tr>
<td>10</td>
<td>7.16</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#Machine</th>
<th>$T_{iter}$</th>
<th>$T_{cp0}$</th>
<th>$T_{cpi}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$5.65 \cdot 10^{-2}$</td>
<td>$1.45 \cdot 10^{-2}$</td>
<td>$2.71$</td>
</tr>
<tr>
<td>4</td>
<td>$4.32$</td>
<td>$3.2$</td>
<td>$4.65 \cdot 10^{-2}$</td>
</tr>
<tr>
<td>6</td>
<td>$3.2$</td>
<td>$5.65 \cdot 10^{-2}$</td>
<td>$7.16$</td>
</tr>
<tr>
<td>8</td>
<td>$4.23$</td>
<td>$4.98$</td>
<td>$12.27$</td>
</tr>
<tr>
<td>10</td>
<td>$7.26 \cdot 10^{-2}$</td>
<td>$4.98$</td>
<td>$19.59$</td>
</tr>
</tbody>
</table>

Recovery Time

<table>
<thead>
<tr>
<th>#Machine</th>
<th>$T_{rec}$-PMEM</th>
<th>$T_{rec}$-SSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>16.45</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>9.88</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>8.88</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>6.05</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>4.85</td>
<td>4.19</td>
</tr>
</tbody>
</table>

Run-time Breakdown

Recovery Time
Evaluations: Checkpointing

Compare with GraphX

10 worker Speedup 6.3x
Evaluations: Checkpointing

Datasets: Twitter(TW), UKunion(UK)

Compare with GraphX

10 worker Speedup 6.3x
Thanks
Q & A