AntMan: Dynamic Scaling on GPU Clusters for Deep Learning

Wencong Xiao, Shiru Ren, Yong Li, Yang Zhang, Pengyang Hou, Zhi Li, Yihui Feng, Wei Lin, Yangqing Jia

Alibaba Group
10/22/2020
Deep Learning in productions

- Computer Vision
- Natural Language Processing
- Speech Understanding
- Recommendation
- Advertisement
- ...

Large company runs DL in shared GPU clusters!
Observations: Low utilization

5000+ GPU cluster statistic

- Low utilization in GPU SM usage
- Low utilization in GPU memory
Observations: Low utilization

More GPUs, more resource wastes

Dynamic resource demand
Observations: Low utilization

Challenges of GPU resource sharing:

- Performance isolation for resource-guarantee jobs
- Prevent failure from GPU memory contention

More GPUs, more resource wastes

Dynamic resource demand
Opportunities

- Small model size
  - Most GPU memory schedulable

- Short mini-batch
  - Fast resource coordination

- Similar mini-batch
  - Metrics to quantify interference

10K sampled production tasks

(a) Model size distribution. (b) Mini-batch time distribution.
AntMan: Dynamic scaling for DL jobs

- Co-executing jobs on shared GPUs

- Resource-guarantee jobs
  - Ensure performance same as dedicated execution

- Opportunistic jobs
  - Best effort utilize spare resources
  - Maximize cluster utilization
Outline

• Introduction
• AntMan: dynamic scaling mechanism
• AntMan: architecture
• Evaluation
• Conclusion
Dynamic scaling memory

• Adjust memory to an appropriate fit
Dynamic scaling memory

- Adjust memory to an appropriate fit
- Cache memory burst to prevent failure, raise upper-bound
- Ensure resource-guarantee job performance
Dynamic scaling memory

- Adjust memory to an appropriate fit
- Cache memory burst to prevent failure, raise upper-bound
- Ensure resource-guarantee job performance
Dynamic scaling memory

- Adjust memory to an appropriate fit
- Cache memory burst to prevent failure, raise upper-bound
- Ensure resource-guarantee job performance
- Best-effort utilize the spare memory
- Opportunistic jobs train with universal GPU and CPU memory
Dynamic scaling computation

Exclusive mode

Job packing

AntMan
AntMan architecture

- Global Scheduler
  - Scheduler
  - Cluster Stats

- Local Coordinator
  - Coordinator
  - Local Stats

- Device Info
  - GpuUtil
  - GpuMem
  - MiniBatch
  - PeakMem
  - MinMem
  - HostMem

- DL Job Info

- Data statistic flow → Control flow

- TF Job
- GPU0
- GPU1

- PyTorch Job
- GPU0

Job Stats
Device Stats
Outline

• Introduction
• AntMan: dynamic scaling mechanism
• AntMan: architecture
• Evaluation
• Conclusion
Micro-benchmark: Memory grow-shrink

- Efficient memory shrinkage and growth
  - Resnet50
    - Shrink: 17ms
    - Growth: 115ms
  - Only 0.4% overhead at one minute interval

(a) A shrink-growth profiling on (b) Overhead of GPU memory scaling for typical models.
Micro-benchmark: Adaptive computation

Setup
• ESPnet(resource-guarantee)
• ResNet50(opportunistic)

Results
• Naïve packing
  • 5.23x slowdown for ESPnet
• Adaptive scaling
  • Same performance as in a dedicated GPU

(a) Packing mode.  (b) Adaptive computation adjustment mode.
Trace experiment

Setup

- 64 V100 GPUs
- 9 SOTA workloads in two tenants

Achievement

- JCT: 2.05x(YARN-CS), 1.84x(Gandiva)
- MakeSpan: 1.76x(YARN-CS), 1.67x(Gandiva)
- Ensure SLAs for resource-guarantee jobs
Large-scale experiment

Setup

- 5000+ GPU
- Production cluster

Achievement

- Up to 17.1% extra GPUs for jobs
- 42% improvement in GPU memory utilization
- 34% improvement in GPU SM utilization
- Avg. queuing delay reduces by 2.05x

<table>
<thead>
<tr>
<th></th>
<th>Avg.</th>
<th>90% tile</th>
<th>95% tile</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dec. 2019</td>
<td>1132</td>
<td>1978</td>
<td>5960</td>
</tr>
<tr>
<td>Apr. 2020</td>
<td>550</td>
<td>124</td>
<td>489</td>
</tr>
</tbody>
</table>

Table 4: One-week queuing delay statistic in seconds.

<table>
<thead>
<tr>
<th>Interference</th>
<th>0%</th>
<th>0~1%</th>
<th>1~2%</th>
<th>2~3%</th>
<th>3~4%</th>
</tr>
</thead>
<tbody>
<tr>
<td># of jobs</td>
<td>9895</td>
<td>26</td>
<td>30</td>
<td>20</td>
<td>29</td>
</tr>
</tbody>
</table>

Table 5: Interference analysis on mini-batch time for 10K production jobs
Conclusion

AntMan: Dynamic Scaling on GPU Clusters for Deep Learning

• Deployed DL infrastructure at Alibaba
• Introduces dynamic scaling primitives
• Maximize utilization using opportunistic jobs while avoiding job interference
• 42% ↑in GPU memory utilization, 34% ↑in GPU SM utilization

[Code] https://github.com/alibaba/GPU-scheduler-for-deep-learning
[Production] PAI-DLC: a cloud-native deep learning training platform
Thanks

Q&A