HARP: an Efficient and Elastic GPU-sharing System

Heterogeneous Accelerator Resource Pooling

Pengfei Fan, Lingling Jin
Alibaba Cloud
October, 2019
Background

• Who are we
• AI compute demand
• GPU usage problems in cloud
Blooming AI Applications

- Facial recognition
- Chat robots
- Search engines
- Personalized shopping
- Credit scoring
- NLP & Translation
- ...

Copyright © 2019 by Alibaba Group
AI Compute Demand

https://openai.com/blog/ai-and-compute/
GPU Usage Problems in Cloud

- Low utilization
- Fragments in scheduling
GPU Utilization Sample Data

[Graph showing GPU utilization over a period of days, with lines representing different metrics such as 'Active Card Num (smActivity > 0)', 'smActivity sum of active cards / 100', and 'Total Card Num'.]
GPU Utilization Sample Data

Optimizing space among applications <-- HARP

Optimizing space inside applications
Fragments in Scheduling

Task 1: 2CPU + 1GPU
Task 2: 1CPU + 2GPU
Task 3: 2CPU + 1GPU
Task 4: 3CPU + 1GPU
Task 5: 3CPU + 1GPU

Server #0
CPU
CPU
GPU

Server #1
CPU
CPU
GPU

Copyright © 2019 by Alibaba Group
HARP System

- Framework
- Features
- Performance tuning
Dynamic GPU Allocating
Dynamic GPU Allocating
Dynamic GPU Allocating
Elastic Worker Pool Size - Grow

1. Request GPU
2. Add more workers
3. Send request to kubelet
4. Start more workers
Elastic Worker Pool Size - Grow

6. Allocate worker and return

5. Register to scheduler

7. Communicate with worker

Worker Pool

- Worker0 Container
- Worker1 Container
- Worker2 Container
- Worker3 Container

Client

Scheduler

Kubernetes Master

Kubelet
**Elastic Worker Pool Size - Shrink**

1. **Mark workers to be retired when idle worker number is larger than threshold**

2. **Stop specified containers**

3. **Send request to kubelet**

4. **Stop workers**
Elastic Worker Pool Size - Shrink

6. Update worker status in DB

5. Worker retired notification or timeout

Scheduler

Kubernetes Master

DB

Worker Pool

Worker0 Container

Worker1 Container

Kubelet

Kubelet

Copyright © 2019 by Alibaba Group
## Comparing with vGPU

<table>
<thead>
<tr>
<th>Feature</th>
<th>HARP</th>
<th>vGPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Support graphics</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Security</td>
<td>Low</td>
<td>High</td>
</tr>
<tr>
<td>Support remote GPU</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Environment requirement</td>
<td>VM/Docker/Bare-metal</td>
<td>VM</td>
</tr>
<tr>
<td>Performance overhead vs direct running</td>
<td>Low in local pass-through mode</td>
<td>Higher vs. local pass-through mode</td>
</tr>
</tbody>
</table>
### Comparing with TensorFlow Serving

<table>
<thead>
<tr>
<th>Feature</th>
<th>HARP</th>
<th>TensorFlow Serving</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flexibility</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>High-level app agnostic</td>
<td></td>
<td>Only deployed model for Infer</td>
</tr>
<tr>
<td>Model &amp; data security</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Hard to retrieve high-level info</td>
<td></td>
<td>Easy to get model &amp; input data</td>
</tr>
<tr>
<td>Performance overhead</td>
<td>Depend on cases</td>
<td>Low</td>
</tr>
<tr>
<td>vs direct running</td>
<td></td>
<td>Only I/O transfer overhead</td>
</tr>
</tbody>
</table>

Copyright © 2019 by Alibaba Group
Performance Tuning

01 Scheduling Strategy
- network-topology aware
- client-worker Locality
- local pass-through
- remote forwarding

02 Single GPU Sharing
- context-switch vs utilization
- resource limiting

03 Forwarding Layer Optimizing
- async execution
- caching
- coalescing
Network BW/Latency Really Matters

single GPU resnet50 training performance

latency-Bandwidth (ms-Gbps)
Improve Locality By Topology-Aware Scheduling

distance calculation by IP and location encoding
Single GPU Sharing by Multiple Clients
Forwarding Layer Optimizing

1. request
2. send request via network
3. request
4. return
5. send result via network
6. return

Note:
**FE**: Frontend, forwarding layer in client side
**BE**: Backend, serving layer in worker side

Async Execution
Coalescing
Caching
Async Execution Benefits

2x Improvement
AI Models Training Performance

![Bar Chart](image)

- **Models**: googlenet-bs128, resnet50-bs96, resnet152-bs64, densenet121-bs64, synNet-bs16
- **Batch Sizes**: direct, harp
- **Performance Metric**: imgs/sec normalized by direct
- **Performance Comparison**:
  - **direct**
  - **harp**

**Key:**
- **0.08ms, 10Gbps**

*Copyright © 2019 by Alibaba Group*
Use Case Sharing

• R&D cluster
Research & Development Cluster

Before using HARP

- Custom environment with Jupyter/Web IDE etc.
- Docker
- Kubernetes
- GPU Servers

**GPU Server with 2 GPU + 96 CPU Cores**

- **User0**
  - CPU 0-15
  - GPU0

- **User1**
  - CPU 16-31
  - GPU1
  - CPU 32-95

- **No** GPUs shared by users
- **Low** hardware utilization. **Has** CPU resource waste.
- Concurrent user number is determined by **GPU** number

---

Copyright © 2019 by Alibaba Group
Research & Development Cluster

After using HARP

- **GPU shared** by users in time-division multiplexing
- **Improved** hardware utilization. **No** CPU resource waste.
- Concurrent user number is determined by CPU number. **2.5x of original**

**HARP Client**
- Custom environment with Jupyter/Web IDE etc.

**Docker**

**Kubernetes**

**GPU Servers**

**Worker**
- CPU 80-96
- GPU0
- GPU1
- Worker idle
  - mapping to GPUs on other hosts
  - GPU Server with 2 GPU + 96 CPU Cores

**User0**
- CPU 0-15
- vGPU0

**User1**
- CPU 16-31
- vGPU1

**User2**
- CPU 32-47
- vGPU2

**User3**
- CPU 48-63
- vGPU3

**User4**
- CPU 64-79
- vGPU4
Future Work

• Work in progress
Work In Progress

• Performance optimizing - Faster
• More testing - More Stable
• Supporting AI Chips – More chips
Thank You
Rate today’s session

Cyberconflict: A new era of war, sabotage, and fear

We’re living in a new era of constant sabotage, misinformation, and fear, in which everyone is a target, and you’re often the collateral damage in a growing conflict among states. From crippling infrastructure to sowing discord and doubt, cyber is now the weapon of choice for democracies, dictators, and terrorists.

David Sanger explains how the rise of cyberweapons has transformed geopolitics like nothing since the invention of the atomic bomb. Moving from the White House Situation Room to the dens of Chinese, Russian, North Korean, and Iranian hackers to the boardrooms of Silicon Valley, David reveals a world coming face-to-face with the perils of technological revolution—a conflict that the United States helped start when it began using cyberweapons against Iranian nuclear plants and North Korean missile launches. But now we find ourselves in a conflict we’re uncertain how to control, as our adversaries exploit vulnerabilities in our hyperconnected nation and we struggle to figure out how to deter these complex, short-of-war attacks.

David Sanger
The New York Times

David E. Sanger is the national security correspondent for the New York Times as well as a national security and political contributor for CNN and a frequent guest on CBS This Morning, Face the Nation, and many PBS shows.