Construct a sharable GPU farm for Data Scientist

Layne Peng
DellEMC OCTO TRIGr
Agenda

- Introduction
- Problems of consuming GPU
- GPU-as-a Service (project Asaka)
- How to manage GPU in farm?
- How to schedule jobs to GPU?
- How to integrate with cloud platform?
- Next step and contacts
Technology Research Innovation Group (TRIGr)

- Innovation
- Advance Research
- Proof of Concept
- User Feedback
- Agile Roadmap
Layne Peng, Principal Technologist at DellEMC OCTO, experienced in cloud computing and SDDC areas, responsible for cloud computing related initiatives since joint in 2011, 10+ patents and a big data and cloud computing book author.

Email: layne.peng@dell.com
Twitter: @layne_peng
Wechat:
What is GPU & GPGPU?

- **Graphical Processing Unit**
  - Designed for Parallel Processing
    - Latency: Increase Clock Speed
    - Throughput: Increase Concurrent Task Execution

- **General-Purpose GPU**
  - Why: Scientific Computing and Graphical processing => Matrix & Vector
  - Prerequisite: key feature supported in GPU, such as float point computing
  - Easy to use: CUDA => Theano, Tensorflow and etc.
  - **Widely used in machine learning and deep learning area**
Problems of consuming GPU today

- Not widely deployed
- High cost
- Low availability when sharing
- Hard to monitoring in fine-grained

✓ Remote GPU as a Service
✓ Fine-grained resource control
✓ Sharable
✓ Fine-grained resource monitor and tracing

Offering GPU in cloud makes it more complex:

- How to abstract GPU resource?
  - Pass-through solution today
- Performance issues caused by network
- Resource discovery and provisioning
Project Asaka – GPU-as-a Service

Remote GPU Access
- Consume remote GPU resource transparently
- Based on queue model and various network fabric

GPU Sharing
- N:1 model: multiple user consume same GPU accelerator
- Fine-grained control and management

GPU Chaining
- 1:N model: multiple GPU hosts chaining as one to serve a large size to 1 application;
- Dynamical GPU number fit to application

Smart Scheduling
- GPU pooling, discovery and provision
- Fine-grained resource monitor and tracing
- Pluggable intelligent scheduling algorithms:
  - Heterogeneous resource
  - Network tradeoffs
Overview of project Asaka

- Providing transparent remote GPU resource access:
  - CUDA APIs interception
  - Intelligent selection of network fabric: TCP or RDMA

- Abstract the GPU resource to vGPU:
  - Device from client’s view, same with GPU
  - Small management grained

- GPU farm:
  - GPU hosts management
  - Sharable resource pool
  - Chaining features
  - Intelligent scheduling
How to manage GPU in farm?

- **Manage GPU farm:**
  - Consul based
  - Basic cluster features:
    - Add new nodes
    - Remove nodes
    - Detect failure
    - Expel nodes and resurrect nodes

- **Key takeaways:**
  - Abstract the hardware resource in a enough level
  - Learn the experience from MSA

---

**Note:**
- Controller Master: a executable binary based on Consul Server with adding features; It can also deploy to the node with GPU;
- Controller Agent: is an extension of Consul Agent with checking scripts and special configuration.
How to select GPU hosts to chain?

- GPU can be chained and exposed as one large machine to application

- GPU hosts selected for chaining by:
  - Capacity
  - Queue length (crowdedness)
  - Network radius
    - Hops
    - Bandwidth

- Key takeaways:
  - Network is the first citizen of considering which GPU serve to application in remote case and chaining servers

1. Select the nodes according to capacity and network radius

2. Chaining GPU from hosts selected to serve as one big machine has nine vGPU

3. Offer vGPUs to client

Network Radius

Selected hosts

XaaS_Controller

Allocation API

Global Scheduler

vGPU

CUDA APIs

Asaka GPU Library
How to schedule jobs to GPU hosts?

- **Two-level scheduling:**
  - 1\textsuperscript{st} level: find GPU hosts from farm
  - 2\textsuperscript{nd} level: find device for CUDA requests

- **Global smart scheduling:**
  - Global status monitor and trace
  - Network topology and analysis
  - Queue-based priority
  - Pluggable scheduling strategy

- **Key takeaways:**
  - Think resources scheduling in a global view
Leverage ML to solve heterogeneous scheduling?

- Heterogeneous scheduling:
  - Heterogeneous hardware resource
  - Heterogeneous network fabric
  - NP-hard problem
    - Greedy Algorithm
    - Or ...?

- Pluggable scheduler design:
  - Abstract of scheduler API
  - Machine learning jobs has workload pattern
  - Metadata matters for scheduling

- Use machine learning to solve the scheduling problem of machine learning

- Status: on-going
How to integrate with cloud platform?

- Offering allocation strategies as a service:
  - High-level abstraction of allocation/scheduling functions
  - Non-invasion design, easy to integrate with existed scheduler using in Cloud Native Platform

- Access GPU transparently from applications deployed in Cloud Native Platform

- Ease of integrating to Cloud Native Platform:
  - Mesos/Marathon
  - Kubernetes
  - Cloud Foundry (demo in DellEMC World 2017)
Example: Cloud Foundry Integration

- As a friendly machine learning interface at project Asaka
  - Receive jobs from users
  - Manage cluster
  - Start the containers for jobs

- Cloud Foundry offering
  - Machine learning platform doesn't need to consider how environment configure and hardware accelerations
  - “Run my app with GPU! I don’t care how!”

- Reach out:
  - Victor Fong from DellEMC Dojo
    › victor.fong@dell.com
    › @victorfong

---

1. Deploy machine learning task/application with CLI
2. Provision GPU service for application according to XaaS_Controller allocation strategies, and bind to applications
3. Users consume the task or application.
Next step and contacts

- Intelligent scheduling
  - Schedule according to application metadata
  - Model and calculate network distance
  - Introduce machine learning to solve heterogeneous resource scheduling

- Keep improve Asaka performance
  - Intelligent data movement and loading
  - Enterprise features such as snapshot and etc.

- XaaS
  - More hardware resource as a service, such as FPGAaaS
  - Global-wide maximized performance

- Reach out the team
  - Jack Harwood: jack.harwood@dell.com (distinguished engineer in DellEMC)
  - Frank Zhao: junping.zhao@dell.com (consultant engineer, leads data path)
  - Layne Peng: layne.peng@dell.com
DELL EMC