GPU as a resource design document

Jun Gong and Thomassong

1. Motivation

GPU are used widely by deep learning related projects now. We use YARN to allocate and isolate GPU cards for docker container, and run programs such as Caffe[1] in it. It is very convenient.

In YARN-4122, we'd like to support GPU as a countable resource just like CPU and memory. AM could request number of GPU cards to run container. Those allocated GPU cards could be only accessed by the corresponding container.

2. Design

GPU Resource Description

We propose describing and scheduling GPU by cards. And add GPU to the resource description.

Scheduling GPU

At RM side, RM knows how many GPU cards each NM has. It could use scheduling method same as CPU/memory to allocate GPU for containers. Then at NM side, NM allocates specific GPU cards to container, and other containers could not use them until they are released.

GPU isolation

As in slurm[2][3], there are two methods to isolate GPU cards for container: 1) environment variable(CUDA_VISIBLE_DEVICES for CUDA, GPU_DEVICE_ORDINAL for OpenCL). 2) cgroups devices[4].

In test, we found there is a problem isolating GPU using cgroups devices for CUDA version less then 7.0[5], however it is OK using environment variable, so we provide both two methods in NM, it could be configured to choose which one.

3. Implementation

GPU Resource Description

Add field gpu to ResourceProto in yarn_protos.proto.

```
message ResourceProto {
   optional int32 memory = 1;
   optional int32 virtual_cores = 2;
+ optional int32 gpu = 3;
}
```

Add field gpu to org.apache.hadoop.yarn.api.records.Resource.

```
+ public abstract int getGPU();
+ public abstract void setGPU(int gpu);
```

Add the following configuration.

```
+ /** Number of GPU cards that can be allocated for containers.*/
+ public static final String NM_GPU_CARDS = NM_PREFIX + "resource.gpu-cards";
+ public static final int DEFAULT_NM_GPU_CARDS = 0;
```

Scheduling GPU

1) RM side change

Consider the GPU resource when scheduling container, and add GPU to DominantResourceCalculator.

2) NM side change

Add two function interface: allocateGPU, releaseGPU. When launching container on NM, NM need call allocateGPU to container if it needs GPU. When cleaning up containers, NM calls releaseGPU to release GPU cards used by container.

We use Round-Robin scheduling to allocate GPU cards.

In order to support NM restart, NM needs save the information that which container use which GPU cards to levelDB. When NM restarts, it could restore these information, and avoid allocating GPUs which have already been allocated.

GPU isolation

1) Docker container

a. Use environment variable

docker run -e CUDA_VISIBLE_DEVICES=... -e GPU_DEVICE_ORDINAL=...

e.g. docker run -e CUDA_VISIBLE_DEVICES=0 -e GPU_DEVICE_ORDINAL=0..

container could only use GPU card 0.

b. Use cgroups
docker run --device=/dev/nvidiaXXX

e.g. Docker run --device=/dev/nvidia0 ...
Container could only use nvidia0.

2) Linux container
Export environment variable in launch script. And write GPU cards' major and minor number to cgroups devices(namely, the file is 'devices.allow').

4. Reference

[1] Caffe: a deep learning framework: http://caffe.berkeleyvision.org/
[2] Generic Resource (GRES) Scheduling: http://slurm.schedmd.com/gres.html
[3] SLURM Resources isolation through cgroups:
http://slurm.schedmd.com/slurm_ug_2011/SLURM_UserGroup2011_cgroups.pdf
[4] cgroups devices: https://www.kernel.org/doc/Documentation/cgroups/devices.txt
[5] Bug with CUDA for cgroups: http://bugs.schedmd.com/show_bug.cgi?id=1421