

GPU Cluster Usage Tutorial

--How to make caffe and enjoy tensorflow on Torque

Yunfeng Wang

wangyf11@mail.ustc.edu.cn

PBS and Torque

- PBS: **P**ortable **B**atch **S**ystem, computer software that performs job scheduling
- versions of PBS
 - i. OpenPBS: original open source version
 - ii. Torque: A fork of OpenPBS
 - iii. PBS Pro: the commercial version of PBS
- Torque: **T**erascale **O**pen-source **R**esource and **Q**UEue Manager, a distributed resource manager providing control over batch jobs and distributed compute nodes
- All commands of Torque are compatible with PBS

Basic info about cluster

- TorqueServer: 192.168.17.240, 8 cores
- GPU cluster: 192.168.6.[101-107], aliased as Gpu101-Gpu107, 32 cores each node
- Shared data disks and home dir

```
[yunfeng@torqueServer ~]$ lsb_release -a # Check distro info
LSB Version:          :core-4.0-amd64:core-4.0-noarch:graphics-4.0-am
Distributor ID: CentOS
Description:          CentOS release 6.3 (Final)
Release:              6.3
Codename:             Final
[yunfeng@torqueServer ~]$ cat /proc/cpuinfo #Get num of CPU
```

Workflow of run job in Torque

- First login to torqueserver, then debug on Gpu107
- Since everything is OK, write the script and send your task to queue
- Wait
- Check your results and errors

Login to cluster

Register an account

You need a ssh client, which has been included in XShell, cgywin and Linux.

```
$MY-PC ssh yunfeng@192.168.17.240
Last login: Fri Nov 11 10:38:16 2016 from 192.168.102.198
[yunfeng@torqueServer ~]$ df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sdb1	985G	393G	543G	42%	/home
/dev/sdc1	1.8T	865G	868G	50%	/data
/dev/sdd1	8.1T	7.3T	415G	95%	/data0
192.168.6.103:/data3	1.4T	679G	718G	49%	/data3
192.168.6.104:/data4	1.4T	1.1T	314G	78%	/data4
192.168.6.105:/data5	1.4T	552G	845G	40%	/data5
192.168.6.106:/data6	1.4T	1.3T	102G	93%	/data6
192.168.6.107:/data7	1.4T	1.4T	13G	100%	/data7
192.168.6.101:/data1	1.3T	770G	561G	58%	/data1
192.168.6.102:/data2	1.3T	447G	883G	34%	/data2

tips: SSH without password

```
$ ssh-keygen #Simply type enter all the way down  
$ ssh-copy-id -i yunfeng@192.168.17.240 # Enter password  
$ ssh yunfeng@192.168.17.240 #No password needed since now!
```

tips: Use aliases to speed up workflow

```
$ echo "alias ssh240='ssh yunfeng@192.168.17.240'" >> ~/.bashrc  
$ source ~/.bashrc  
$ ssh240 # Same as 'ssh yunfeng@192.168.17.240'
```


tips: Some utilities to improve your shell experience

1. [oh-my-zsh](#)
2. tmux
3. ipython

Login to Gpu107 to debug your Caffe

```
[yunfeng@torqueServer ~]$ ssh Gpu107  
Last login: Fri Nov 11 11:02:43 2016 from torqueserver  
[yunfeng@Gpu107 ~]$ nvidia-smi
```

0	Tesla K80	Off	0000:04:00.0	Off	
N/A	26C	P8	26W / 149W	22MiB / 11519MiB	0
1	Tesla K80	Off	0000:05:00.0	Off	
N/A	23C	P8	29W / 149W	22MiB / 11519MiB	0
2	Tesla K80	Off	0000:84:00.0	Off	
N/A	48C	P0	98W / 149W	9106MiB / 11519MiB	99
3	Tesla K80	Off	0000:85:00.0	Off	
N/A	24C	P8	30W / 149W	22MiB / 11519MiB	0
4	Tesla K80	Off	0000:8A:00.0	Off	
N/A	34C	P0	60W / 149W	2181MiB / 11519MiB	0
5	Tesla K80	Off	0000:8B:00.0	Off	
N/A	47C	P0	153W / 149W	8349MiB / 11519MiB	98
6	Tesla K80	Off	0000:8E:00.0	Off	
N/A	40C	P0	96W / 149W	4404MiB / 11519MiB	91
7	Tesla K80	Off	0000:8F:00.0	Off	
N/A	58C	P0	147W / 149W	5361MiB / 11519MiB	95

1. Copy compiled caffe to your dir

```
[yunfeng@Gpu107 ~]$ mkdir /data2/yunfeng  
[yunfeng@Gpu107 ~]$ cd /data2/yunfeng  
[yunfeng@Gpu107 ~]$ cp -r /opt/caffe-master .  
[yunfeng@Gpu107 ~]$ mv caffe-master caffe
```

2. Write shell script to run your job

```
# /home/yunfeng/run_mnist.sh Example script of running mnist  
cd /data3/yunfeng/caffe  
./data/mnist/get_mnist.sh  
./examples/mnist/create_mnist.sh  
./examples/mnist/train_lenet.sh
```

3. Debug and check everything is OK

```
[yunfeng@Gpu107 ~]$ cd ~  
[yunfeng@Gpu107 ~]$ chmod +x run_mnist.sh  
[yunfeng@Gpu107 ~]$ ./run_mnist.sh
```

Cluster types

GPU used detail:

	0	1		2	3	4	5	6	7
GPU101:	[x]	[x]		[x]	[x]				
GPU102:	[x]	[x]		[x]	[x]				
GPU103:	[x]	[x]		[x]	[x]	[x]	[x]	[x]	[x]
GPU104:	[x]	[x]		[x]	[x]	[x]	[x]	[x]	[x]
GPU105:	[x]	[x]		[x]	[x]	[x]	[x]	[x]	[x]
GPU106:	[x]	[x]		[x]	[x]	[x]	[x]	[x]	[x]
GPU107:	[x]	[x]		[x]	[x]	[x]	[x]	[x]	[x]

Type S:{Gpu101,Gpu102};D:{Gpu103,Gpu104};M:{Gpu105,Gpu106,Gpu107}

Submit your job to queue

1. go to head node, write pbs script

```
# /home/yunfeng/run_mnist.pbs configuration file of your job
#PBS -N run_mnist #Name of your job
#PBS -o /home/yunfeng/run_mnist.out #the file stdout will write
#PBS -e /home/yunfeng/run_mnist.err #the file stderr will write
#PBS -l nodes=1:gpus=1:S #cluster type to use
#PBS -r y #run the job immediately or not

## Put out debug info,don't modify this part
cd $PBS_O_WORKDIR
echo Time is `date`
echo Directory is $PWD
echo This job runs on following nodes:
cat $PBS_NODEFILE

## The name of script to run main job
./run_mnist.sh
```

2. Submit job and check status

```
[yunfeng@torqueServer ~]$ qsub run_mnist.pbs # Send job to queue  
[yunfeng@torqueServer ~]$ chk_gpu # Check the status of your job
```

[yunfeng@torqueServer ~]\$ vi run_mnist.pbs

[yunfeng@torqueServer ~]\$ chk_gpu

Jobid	User	JobName	Req_parm	Start_time	S	Run_time	Alloc_GPUS
4126	liuyj	conv1_0.004_96	1:gpus=2:D	20161028 22:43:28	R	283:21:39	Gpu103-gpu/3/2
4179	liuyj	myjob1	1:gpus=2:D	20161101 22:36:09	R	236:00:29	Gpu103-gpu/5/4
4240	liuyj	conv1_hard	1:gpus=2:M	20161105 11:23:04	R	151:25:14	Gpu107-gpu/7/6
4299	zhangyh	T_Step_B	1:gpus=2:M	20161107 22:03:25	R	80:11:25	Gpu107-gpu/3/2
4314	huangjl	lstm_train	1:gpus=2:M	20161108 10:31:20	R	80:16:31	Gpu105-gpu/1/0
4316	yann	IFCNN_Q22	1:gpus=1:S	20161108 10:38:51	R	76:51:11	Gpu102-gpu/1
4324	fw092	caffe_train_dog_rand	1:gpus=2:D	20161108 20:34:40	R	70:13:49	Gpu103-gpu/1/0
4340	yann	IFCNN_end2end_Q32	1:gpus=2:D	20161109 10:33:47	R	31:41:36	Gpu104-gpu/5/4
4345	liuyj	conv1_hard	1:gpus=1:S	20161109 17:21:15	R	49:27:14	Gpu102-gpu/2
4346	liuyj	conv1_hard	1:gpus=1:S	20161109 17:23:27	R	49:25:01	Gpu102-gpu/0
4348	fanhz	net2_13	1:gpus=2:D	20161109 23:15:39	R	28:44:13	Gpu104-gpu/1/0
4351	huangjl	lstm_train01	1:gpus=2:M	20161110 10:18:12	R	32:30:06	Gpu107-gpu/5/4
4357	zenglh	zenglhjob2	1:gpus=1:S	20161110 18:05:59	R	24:42:26	Gpu101-gpu/0
4358	zenglh	zenglhjob3	1:gpus=1:S	20161110 18:19:23	R	24:29:03	Gpu101-gpu/1
4361	fengxy	test_lr	1:gpus=6:M	20161110 21:25:25	R	21:22:26	Gpu105-gpu/7/6/5/4/3/2
4362	yangjw	lrelu_alexnet_01	1:gpus=2:D	20161110 21:27:43	R	21:20:34	Gpu104-gpu/3/2
4363	fanhz	net2_14	1:gpus=2:D	20161110 21:43:07	R	14:19:37	Gpu104-gpu/7/6
4367	fanhz	net2_15	1:gpus=2:D	20161110 21:54:01	R	05:20:56	Gpu103-gpu/7/6
4397	liyao	Barc-29	1:gpus=1:S	20161111 15:05:08	R	03:43:17	Gpu101-gpu/3
4398	liyao	Barc-33	1:gpus=1:S	20161111 15:05:12	R	03:43:16	Gpu102-gpu/3
4399	housh	jointvegfru_vggcp	1:gpus=2:M	20161111 15:08:10	R	03:40:05	Gpu106-gpu/1/0
4401	yangjw	elu_alexnet	1:gpus=4:M	20161111 15:11:29	R	03:36:46	Gpu106-gpu/5/4/3/2
4406	housh	jointvegfru_vggcpv2	1:gpus=2:M	20161111 15:24:03	R	03:24:12	Gpu106-gpu/7/6
4408	housh	jointaircrafts_vggcp	1:gpus=2:M	20161111 15:29:34	R	03:18:44	Gpu107-gpu/1/0
4411	sunshy	train_idx	1:gpus=1:S	20161111 15:48:50	R	02:59:36	Gpu101-gpu/2
4412	yann	IFCNN_ph22_Q32	1:gpus=1:S	20161111 16:30:11	Q		
4413	yann	IFCNN_ph22_Q37	1:gpus=1:S	20161111 16:30:20	Q		
4415	liux	train_ucf	1:gpus=1:S	20161111 17:51:36	Q		

GPU used detail:

```

    0 1 | 2 3 4 5 6 7
GPU101: [x][x] | [x][x]
GPU102: [x][x] | [x][x]
GPU103: [x][x] | [x][x][x][x][x][x]
GPU104: [x][x] | [x][x][x][x][x][x]
GPU105: [x][x] | [x][x][x][x][x][x]
GPU106: [x][x] | [x][x][x][x][x][x]
GPU107: [x][x] | [x][x][x][x][x][x]

```

Type S:{Gpu101,Gpu102}; D:{Gpu103,Gpu104}; M:{Gpu105,Gpu106,Gpu107}

Total 28 jobs.

Other useful pbs commands

```
[yunfeng@torqueServer ~]$ qstat # Show status of pbs jobs  
[yunfeng@torqueServer ~]$ qhold job_id # Hold pbs jobs  
[yunfeng@torqueServer ~]$ qrls job_id #Release hold of jobs  
[yunfeng@torqueServer ~]$ qel job_id # Delete job  
[yunfeng@torqueServer ~]$ pbsnodes # Show status of nodes
```

How to run Tensorflow on Cluster?

1. Use system tensorflow (version:0.10.0)

```
[yunfeng@Gpu107 ~]$ ipython
Python 2.7.5 (default, Nov 20 2015, 02:00:19)
Type "copyright", "credits" or "license" for more information

In [1]: import tensorflow as tf

In [2]: tf.__version__
Out[2]: '0.10.0'
```

2. Or [install latest version](#) of tensorflow for yourself

```
$ export TF_BINARY_URL=https://storage.googleapis.com/tensorflow
$ pip install --user --upgrade $TF_BINARY_URL
```

3. Write your command in shell script and then submit it, just like the example above

How to prepare data for my experiment?

1. Use `scp` to copy your data to cluster

```
$ scp -r my_data/ yunfeng@192.168.17.240:/data3/yunfeng/
```

2. Since data disks are shared, you can then use your data in code

```
# example.py  
data_dir = '/data3/yunfeng/my_data'  
result = my_func(data_dir)
```

How to check logs of my job

Normally, we can only see log when job finished. However, we can

1. Use `>>` or `tee` to

```
# run_mnist.sh
...
./examples/mnist/train_lenet.sh >> my_mnist_log
```

```
# run_mnist.sh
...
#Please check the manual of tee to fully use it.
./examples/mnist/train_lenet.sh | tee my_mnist_log
```

2. Use tensorflow's tensorboard to check the result

```
$ tensorboard --logdir=path/to/log-directory
```

Docs location

<https://github.com/vra/docs-and-slides>

Q&A