Multi-node distributed training

IBM DDL

Background

IBM's DDL library is included in the PowerAI module (the most recent version is 'wmlce'). It supports TensorFlow, Pytorch, and Caffe frameworks. It requires very little code modification, and is well-documented at the IBM Knowledge Center. In the context of the HAL cluster, this is one of the easiest ways to run a training job across multiple nodes and allows users to train using more than the 4 GPUs hosted on a single node. However, it is not portable to machines without the WML CE (PowerAI) software distribution.

Distributed (batch) job submission with SLURM

The only setup that is required is to load the WML CE (PowerAI) module.

```
module load wmlce
```

If you also want a copy of the example scripts provided by IBM, they can be copied to your home directory with

```
ddl-tensorflow-install-samples [example_dir]
```

The following job submission script can be run with any valid DDL program using 'sbatch'. For instance, this submission script runs IBM's MNIST example script across all 4 GPUs on two nodes.

```
#!/bin/bash

#SBATCH --job-name="ddl_mnist"
#SBATCH --output="ddl_mnist.%j.%N.out"
#SBATCH --error="ddl_mnist.%j.%N.err"
#SBATCH --partition=gpu
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=4
#SBATCH --cpus-per-task=36
#SBATCH --gres=gpu:v100:4
#SBATCH --export=ALL
#SBATCH -t 1:00:00

NODE_LIST=$( scontrol show hostname $SLURM_JOB_NODELIST | sed -z 's/\n/,/g' )
NODE_LIST=${NODE_LIST%?}
echo $NODE_LIST

ddlrun --no_ddloptions -cores 18 -H $NODE_LIST python ~/[example_dir]/ddl-tensorflow/examples/mnist/mnist-init.
py --ddl_options="-mode b:4x2" --mpiarg "--mca btl_openib_allow_ib 1 --mca orte_base_help_aggregate 1"
```

Horovod

Background

Horovod is a framework that allows users of popular machine learning frameworks such as TensorFlow, Keras, and PyTorch to easily adapt their applications to run across multiple GPUs or nodes. It requires a short environment setup and (like DDL) minor code modifications, but is portable and popular. Newer versions of PowerAI may not require the environment setup, and this page will be updated accordingly once testing is finished on our system.

Environment setup tutorial

The first step is to clone the PowerAI conda environment. This copies the packages in the environment to the user's home directory and as such allows the user to install or uninstall packages.

```
conda create --name [your_env_name] --clone powerai_env
```

The next step is to remove Spectrum MPI from the cloned environment, which will cause it to default to OpenMPI. First, we will remove the script that sets Spectrum MPI environment variables.

```
rm /home/[username]/.conda/envs/[your_env_name]/etc/conda/activate.d/spectrum-mpi.sh
```

Next, load your cloned environment.

```
conda activate [your_env_name]
```

Now, we'll uninstall both Spectrum MPI and Horovod, as we'll need to re-install with OpenMPI as the default MPI implementation.

```
conda uninstall spectrum-mpi
pip uninstall horovod -y
```

Finally, reinstall Horovod.

```
HOROVOD_GPU_ALLREDUCE=NCCL pip install --no-cache-dir horovod
```

Distributed (batch) job submission with SLURM

Once your environment is properly configured, you can submit batch jobs that utilize the GPUs from multiple nodes to perform a single distributed training run. The following submission script can be run with any valid Horovod program. For instance, the following runs the Horovod example script at https://github.com/horovod/horovod/blob/master/examples/tensorflow_mnist.py.

```
#!/bin/bash

#SBATCH --job-name="hvd_tutorial"
#SBATCH --output="hvd_tutorial.%j.%N.out"
#SBATCH --error="hvd_tutorial.%j.%N.err"
#SBATCH --partition=gpu
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=4
#SBATCH --cpus-per-task=36
#SBATCH --gres=gpu:v100:4
#SBATCH --export=ALL
#SBATCH -t 1:00:00

NODE_LIST=$( scontrol show hostname $SLURM_JOB_NODELIST | sed -z 's/\n/\:4,/g' )
NODE_LIST=${NODE_LIST%?}
echo $NODE_LIST
mpirun -np $SLURM_NTASKS -H $NODE_LIST -bind-to none -map-by slot -x NCCL_DEBUG=INFO -x NCCL_SOCKET_IFNAME=^lo -
x LD_LIBRARY_PATH -x PATH -mca pml ob1 -mca btl ^openib -mca btl_openib_verbose 1 -mca btl_tcp_if_incle
192.168.0.0/16 -mca oob_tcp_if_include 192.168.0.0/16 python tensorflow_mnist.py
```

PyTorch Distributed Mixed-precision Training

Github repo for all source code and details: https://github.com/richardkxu/distributed-pytorch

Jupyter notebook tutorial for the key points: https://github.com/richardkxu/distributed-pytorch/blob/master/ddp_apex_tutorial.ipynb

ImageNet benchmark results for performance analysis and visualization: HAL Benchmarks 2020