Code availability:

Pap407 submission code is now available as a pull request to main DGL code repository. The code can be accessed from url: https://github.com/dmlc/dgl/pull/2914

Location: https://github.com/dmlc/dgl/pull/2914 (DGL Pull Request Id:2914, commit: 8b27954)

DGL installation details can be found at https://docs.dgl.ai/install/index.html#install-from-source

Benchmark datasets:

All the benchmark datasets are automatically downloaded when the application is executed.

Dependencies:

PyTorch v1.7.1 – Please refer to https://pytorch.org/ for installation

OneCCL - https://github.com/ddkalamk/torch-ccl/tree/working_1.7 (commit: 633a77e)

LIBXSMM library added as a submodule to DGL, please download the submodule using "git submodule update –init --recursive" after cloning the repository.

Installation steps:

1. Copy dgl/setup_env.sh && dgl/env.sh to a desired location XYZ (After this you may discard dgl folder, as the scripts below will setup dgl separately)
2. cd XYZ
3. Set compiler, gcc 8.3.0
4. Run “XYZ/setup_env.sh”
   a. It creates a XYZ/sub407 sub-folder
   b. It installs all the dependencies (anaconda, OneCCL, Pytorch, and other dependencies) as well as DGL in a new conda environment called “sub407”.
   c. The DGL code is now present in sub407 sub-folder
   d. The conda environment can be enabled as “source sub407/miniconda3/bin/activate sub407”
5. Run “source XYZ/env.sh”
   a. It activates the conda environment “sub407” and sets up all the environment variables
6. The DGL (DistGNN) installation is ready to run the single socket as well as distributed experiments, with DGL code present in XYZ/sub407/dgl (follow “How to run” described below).
7. If you wish to rerun setup_env.sh then remove sub407 folder and rerun the scripts.

How to run (Instructions are also present in <path_to_dgl>/dgl/examples/pytorch/graphsage/experimental/README.md):

1. Single Socket experiments

```
cd <path_to_dgl>/dgl/examples/pytorch/graphsage

numactl -N 0 -m 0 python train_full.py --n-epochs 200 --dataset reddit

numactl -N 0 -m 0 python train_full_ogbn-products.py --n-epochs 300 --dataset ogbn-products

numactl -N 0 -m 0 python train_full_proteins.py --n-epochs 200 --dataset proteins
```

```
cd <path_to_dgl>/dgl/examples/pytorch/rgcn-hetero

numactl -m 0 -N 0 python entity_classify.py -d am --l2norm 5e-4 --n-bases 40 --testing --gpu -1 --n-epochs 20
```

2. Distributed-memory experiments

```
cd <path_to_dgl>/dgl/examples/pytorch/graphsage/experimental
```

2.1 Graph partitioning

```
python ../../../../python/dgl/distgnn/partition/main_Libra.py cora

python ../../../../python/dgl/distgnn/partition/main_Libra.py reddit

python ../../../../python/dgl/distgnn/partition/main_Libra.py ogbn-products

python ../../../../python/dgl/distgnn/partition/main_Libra.py proteins

python ../../../../python/dgl/distgnn/partition/main_Libra.py ogbn-papers100M
```

Note:

- Output partitions are created in the current directory.
- By default it creates 2, 4, & 8 partitions of the input graph. The number of partitions can be changed in dgl/python/dgl/distgnn/partition/main_Libra.py:213.
- As of now the Libra partitioning code is single threaded, so for large dataset, it takes time (in hrs) to produce the partitions.

2.2 Distributed-memory runs

Note: By default the partitions are read from current directory.

cd-0:

```
sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym.py --dataset reddit --n-epochs 200 --nr 1  --lr 0.03

sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym_ogbn-products.py --dataset ogbn-products --n-epochs 300 --nr 1 --lr 0.03

sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym_proteins.py --dataset proteins --n-epochs 200 --nr 1  --lr 0.03

sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym_ogbn-papers.py --dataset ogbn-papers100M --n-epochs 200 --nr 1 --lr 0.08
```

cd-5:

```
sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym.py --dataset reddit --n-epochs 200 --nr 5  --lr 0.03

sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym_ogbn-products.py --dataset ogbn-products --n-epochs 300 --nr 5 --lr 0.03

sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym_proteins.py --dataset proteins --n-epochs 200 --nr 5  --lr 0.08

sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym_ogbn-papers.py --dataset ogbn-papers100M --n-epochs 200 --nr 5 --lr 0.08
```

0c:

```
sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym.py --dataset reddit --n-epochs 200 --nr -1  --lr 0.03

sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym_ogbn-products.py --dataset ogbn-products --n-epochs 300 --nr -1 --lr 0.03

sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym_proteins.py --dataset proteins --n-epochs 200 --nr -1  --lr 0.08

sh run_dist.sh -n <num_nodes> -ppn <ppn>  python train_dist_sym_ogbn-papers.py --dataset ogbn-papers100M --n-epochs 200 --nr -1 --lr 0.08
```

Software Details:

Location: https://github.com/dmlc/dgl/pull/2914 (DGL Pull Request Id:2914, commit: 8b27954)

Artifact name: DistGNN

Citation of artifact (if known):

Relevant hardware details: Intel Xeon 8280/9242 CPU (64 nodes cluster with dual socket 9242 CPU)

Operating systems and versions: CentOS 7.6/8.0

Compilers and versions: GCC 8.3.0

Applications and versions: DGLv0.6.0; PyTorch 1.7.1

Libraries and versions: OneCCL (https://github.com/ddkalamk/torch-ccl/tree/working_1.7 (commit: 633a77e)); LIBXSMM (git commit:55c6a9f)

Key algorithms: SPMM, Libra graph partitioning

Input datasets and versions: Reddit, OGBv1.1.1, AM, Proteins

Python version: 3.7.10