CompressGraph

1. Description

This is the open-source implementation for CompressGraph:

CompressGraph: Efficient Parallel Graph Analytics with Rule-Based Compression", Zheng Chen, Feng Zhang, Jiawei Guan, Jidong Zhai, Xipeng Shen, Huanchen Zhang, Wentong Shu, Xiaoyong Du. SIGMOD/PODS '23: Proceedings of the 2023 International Conference on Management of Data. https://dl.acm.org/doi/10.1145/3588684

2. Getting Started

2.1 System Dependency

- CMake
- OpenMP and C++17
- CUDA
- Optional(CPU): Ligra
- Optional(GPU): Gunrock

2.2 Compilation

```
 git clone https://github.com/ZhengChenCS/CompressGraph.git --recursive
cd CompressGraph
mkdir -p build
cd build
cmake .. -DLIGRA=ON -DGUNROCK=ON
make -j
```

If you just want to run application on CPU, you can set `-DGUNROCK=OFF` .

2.3 Graph Input Format

The initial input graph format should be in the adjacency graph format. For example, the SNAP format(edgelist) and the adjacency graph format for a sample graph are shown below.

SNAP format:

```
 src dst
0 1
0 2
2 0
2 1
```

Adjacency Graph format:

```
 AdjacencyGraph
3 <The number of vertices>
4 <The number of edges>
0 <o0>
2 <o1>
2 <o2>
1 <e0>
2 <e1>
0 <e2>
1 <e3>
```

3. CompressGraph Compression

3.1 Data Preparation

The Compression Mudule accepts binary CSR(Compressed Sparse Row) graph data as input, which contains a `vlist` and a `elist` array. We provide two programs for converting graph files from edgelist and adjacency graph format to CSR format. User can invoke them as follows:

- Edgelist to CSR

```
edgelist2csr < <edgelist.txt>
```

- Adjacency graph to CSR

```
adj2csr < <adjgraph.txt>
```

The two programs will generate two output files in CSR format: `csr_vlist.bin` and `csr_elist.bin` in the current directory.

3.2 Graph Compression

The `dataset` folder provides an example.

To compress a input graph, run:

```
compress <csr_vlist.bin> <csr_elist.bin>
```

To filter the rule by the threshold(16 by default), run:

```
filter <csr_vlist.bin> <csr_elist.bin> <info.bin> 16
```

The `filter` program will filter out rules that meet `(freq - 1) * (len - 1) - 1 <= threshold;` , where `freq` is the frequency of rules, `len` is the length of rules.

We also provide a filtering program `filter_decmp` that can filter out rules that do not meet the frequency and length requirements separately.

```
filter_decomp <csr_vlist.bin> <csr_elist.bin> <info.bin> <freq_threshold> <len_threshold>
```

The `filter_decomp` will filter out rules that meet `freq < freq_threshold || len < len_threshold` .

4. ComprassGraph Analytics

We have implemented the CompressGraph analytic engine based on Ligra on CPU and Gunrock on GPU.

4.1 Data Prepareation

Before running the graph analytic program, we need to convert the CSR format to the format required by Ligra. Additionaly, we generate some auxiliary structures, including `order.bin` and `degree.bin` (used in `pagerank` and `hits`).

```
 $convert2ligra $csr_vlist $csr_elist > $output
$save_degree $csr_vlist
$gene_rule_order $csr_vlist $csr_elist $info
```

We provide a script `data_prepare.sh` and `data_prepare_gpu.sh` to execute the data prepareation process in `script` directory.

4.2 Run Applications

Users can execute graph applications using the following approach:

```
 ./bfs_cpu -r 1 $file
./cc_cpu $file
./sssp_cpu -r 1 $file
./pagerank_cpu -maxiters 10 -i $info -d $degree -o $order $file
./topo_cpu -i $info $file
./hits_cpu -maxiters 10 -i $info -o $order $file
```

```
 ./bfs_gpu $file 0
./cc_gpu $file
./sssp_gpu $file $info 0
./pagerank_gpu $file
./hits_gpu $file
./topo_gpu $file $info
```

We provide scripts in `script/cpu` and `script/gpu` directory to execute these programs.

4.3 Run applications with script

```
 cd script
bash data_prepare.sh
cd cpu
bash bfs.sh
bash sssp.sh
bash cc.sh
bash pagerank.sh
bash topo.sh
bash hits.sh
cd gpu
bash bfs.sh
bash sssp.sh
bash cc.sh
bash pagerank.sh
bash topo.sh
bash hits.sh
```

5. Citation

If you use our code, please cite our paper:

```
 @article{chen2023compressgraph,
  title={CompressGraph: Efficient Parallel Graph Analytics with Rule-Based Compression},
  author={Chen, Zheng and Zhang, Feng and Guan, JiaWei and Zhai, Jidong and Shen, Xipeng and Zhang, Huanchen and Shu, Wentong and Du
  journal={Proceedings of the ACM on Management of Data},
  volume={1},
  number={1},
  pages={1--31},
  year={2023},
  publisher={ACM New York, NY, USA}
}
```