Report for HW9

Cluster username: bds23-ghcstu08
Student name: Ng Tze Kean
Student type: Virutal Exchange student

Preparation

Note that the code has to be first copied from hw9_GridGraph. We will run the code using a gridgraph of partition comprising of different sizes to test the program effectiveness. The partition size tested will be in the range of [2,4,8,16] for pagerank delta and kcores. Please copy the the dataset using the below snippet in the cluster first. Note that we make the graph unweighted for kcores computation.

```
cp /data/hw9_data/livejournal .
./bin/preprocess -i /data/hw9_data/livejournal  -o ./LiveJournal_Grid -v
4847571 -p 4 -t 0
make all
```

Pagerank delta

Implementation

PageRank-Delta: Only update vertices whose PageRank value has changed by more than some Δ-fraction

We pass the gridgraph and initialize the vertex for computation first. The code then calculates the degree of each vertex by streaming the edges of the graph and incrementing the degree of the source vertex for each edge. After the degrees are calculated, it initializes the PageRank of each vertex to be the reciprocal of its degree and the sum to zero. In each iteration, it streams the edges of the graph again. For each edge, if the PageRank of the source vertex is greater than the threshold, it adds the PageRank to the sum for the target vertex. After all edges have been processed, it updates the PageRank of each vertex based on the sum of the PageRanks of its incoming neighbors, and resets the sum to zero for the next iteration.

The following command can be used to invoke the compiled program. `./bin/pagerank_delta [path] [number of iterations] [threshold] [memory budget]`

Output

```
# ------ 2 partition
./bin/pagerank_delta ./LiveJournal_Grid 100 10  8
degree calculation used 0.22 seconds
100 iterations of pagerank took 9.00 seconds
# ------ 4 partition
./bin/pagerank_delta ./LiveJournal_Grid 100 10  8
degree calculation used 0.23 seconds
100 iterations of pagerank took 7.97 seconds
# ------ 8 partition
```

```
./bin/pagerank_delta ./LiveJournal_Grid 100 10  8
degree calculation used 0.21 seconds
100 iterations of pagerank took 7.63 seconds
# ------ 16 partition
./bin/pagerank_delta ./LiveJournal_Grid 100 10  8
degree calculation used 0.20 seconds
100 iterations of pagerank took 7.20 seconds
```

Kcores

A k-core of a graph G is a maximal connected subgraph of G in which all vertices have degree at least k. The graph is made undirected first, and then subsequently run with the following command. The code loops to check if the degree of the vertex is less than k and is not yet removed, if such a vertex exist, then it will be marked for removal. The process is repeated till the graph is left with k cores.

The following command can be used to invoke the compiled program. `./bin/kcores [path] [core] [memory budget]`

Output

```
# ------ 2 partition
./bin/kcores ./LiveJournal_Grid 10 8
degree calculation used 0.23 seconds
kcores took 0.81 seconds, removed 3258030 vertices in 2 iterations
# ------ 4 partition
./bin/kcores ./LiveJournal_Grid 10 8
degree calculation used 0.21 seconds
kcores took 0.73 seconds, removed 3258030 vertices in 2 iterations
# ------ 8 partition
./bin/kcores ./LiveJournal_Grid 10 8
degree calculation used 0.20 seconds
kcores took 0.80 seconds, removed 3258030 vertices in 2 iterations
# ------ 16 partition
./bin/kcores ./LiveJournal_Grid 10 8
degree calculation used 0.22 seconds
kcores took 0.72 seconds, removed 3258030 vertices in 2 iteration
```