DC PROJECT

PARALLEL BFS

BETWEENESS CENTRALITY BENCHMARKING

DISHIT SHARMA (B22CS082)

GRAPHS USED FOR BENCHMARKING

CONTENT

CODE MODIFICATIONS

Reference 1

1. Header file to read the graphs:
   - In this file, we read the .egr format graphs and use the ECLgraph class to store the graph in the CSR format.
   - nodes: stores the number of nodes
   - edges1: stores the number of edges
   - nindex: stores the indexes between which the corresponding edges of a particular vertex resides. Basically, outdegree of a node with index i is equal to nindex[i+1] - nindex[i]
   - nlist: contains the destination vertex of a particular edge

```
struct ECLgraph
{
    int nodes;
    int edges1;
    int *nindex;
    int *nlist;
    int *eweight;
};

ECLgraph readECLgraph(const char *const fname)
{
    ECLgraph g;
    int cnt;

    FILE *f = fopen(fname, "rb");
    cnt = fread(&g.nodes, sizeof(g.nodes), 1, f);
    cnt = fread(&g.edges1, sizeof(g.edges1), 1, f);

    g.nindex = (int *)malloc((g.nodes + 1) * sizeof(g.nindex[0]));
    g.nlist = (int *)malloc(g.edges1 * sizeof(g.nlist[0]));
    g.eweight = (int *)malloc(g.edges1 * sizeof(g.eweight[0]));

    cnt = fread(g.nindex, sizeof(g.nindex[0]), g.nodes + 1, f);
    cnt = fread(g.nlist, sizeof(g.nlist[0]), g.edges1, f);
    cnt = fread(g.eweight, sizeof(g.eweight[0]), g.edges1, f);
    if (cnt == 0)
    {
        free(g.eweight);
        g.eweight = NULL;
    }
    fclose(f);

    return g;
}
```

CODE MODIFICATIONS

2. Invocation of the header file:
    - Here, we use call the function of the header file and store the graphs in a array.
    - Now, we benchmark the code using all of these graphs one by one.

```
int main(){
    vector<ECLgraph> store;
    store.push_back(readECLgraph("Graphs/sample_1_Gnutella.egr"));
    store.push_back(readECLgraph("Graphs/Email-Enron.egr"));
    store.push_back(readECLgraph("Graphs/citationCiteseer.egr"));
    store.push_back(readECLgraph("Graphs/amazon0601.egr"));
    store.push_back(readECLgraph("Graphs/as-skitter.egr"));
    store.push_back(readECLgraph("Graphs/in-2004.egr"));
    store.push_back(readECLgraph("Graphs/coPapersDBLP.egr"));
    store.push_back(readECLgraph("Graphs/cit-Patents.egr"));
    store.push_back(readECLgraph("Graphs/soc-LiveJournal1.egr"));
    // store.push_back(readECLgraph("Graphs/delaunay_n24.egr"));
    // store.push_back(readECLgraph("Graphs/europe_osm.egr"));
    for(auto &j: store){
        bench(j);
    }
    return 0;
}
```

CODE MODIFICATIONS

Reference 1

3. Utilization of the CSR format:
    - Here, we use graphs stores to make two arrays named adjLst and adjListPtr which basically store the given graph in the CSR format.
    - adjLst contains all the destination vertices of the edges.
    - adjListPtr contains information about the degree of each node.
    - adjListPtr[i] = nindex[i+1] – nindex[i]

```
void bench(ECLgraph& g)
{
    Graph *host_graph = new Graph();
    Graph *device_graph;

    catchCudaError(cudaMalloc((void **)&device_graph, sizeof(Graph)));

    int nodeCount = g.nodes;
    int edgeCount = g.edges1;

    int *adjLst, *adjListPtr;

    // Copy into compressed adjacency List
    adjListPtr = new int[nodeCount];
    adjLst = new int[edgeCount];

    cout << "Number of nodes = " << nodeCount << endl;
    cout << "Number of edges = " << edgeCount << endl;

    for (int i = 0; i < nodeCount - 1; i++) {
        adjListPtr[i] = g.nindex[i+1] - g.nindex[i];
    }

    adjListPtr[nodeCount - 1] = edgeCount - g.nindex[nodeCount - 1];

    for (int i = 0; i < edgeCount; i++) adjLst[i] = g.nlist[i];
```

TIME COMPARISON FOR DIFFERENT GRAPHS

The time comparison is done for the parallel work efficient method with coarse grained parallelism

```
PS C:\Users\dishi\Desktop\Benchmarking> ./_parallel_work_coarse
Number of nodes = 6301
Number of edges = 20777
Maximum Betweenness Centrality = 579.08
Time Taken (Parallel) = 3 ms

Number of nodes = 36692
Number of edges = 367662
Maximum Betweenness Centrality = 2219794.25
Time Taken (Parallel) = 962 ms

Number of nodes = 268495
Number of edges = 2313294
Maximum Betweenness Centrality = 110606.77
Time Taken (Parallel) = 4949 ms

Number of nodes = 403394
Number of edges = 4886816
Maximum Betweenness Centrality = 64581512.00
Time Taken (Parallel) = 21643 ms
```

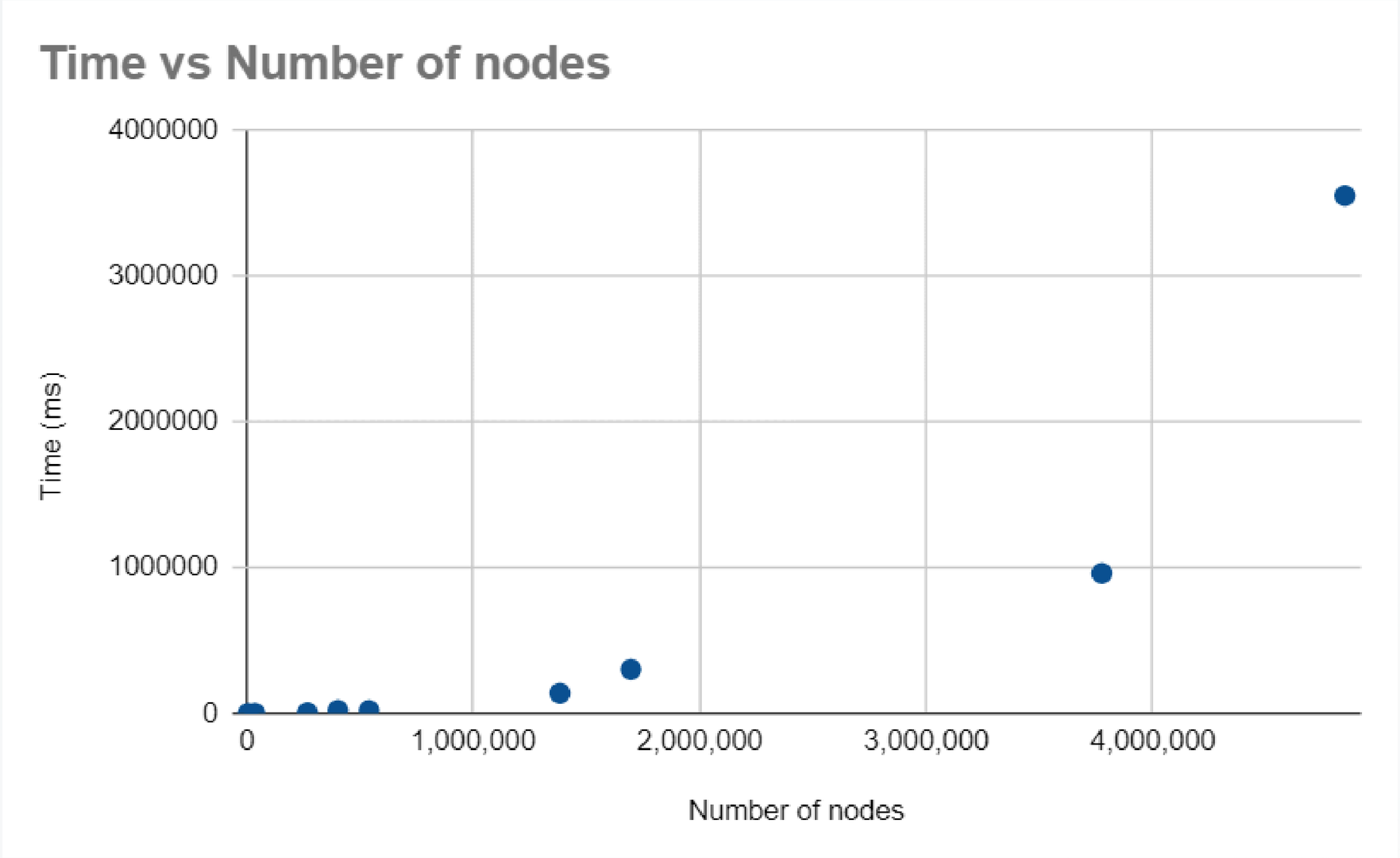
```
Number of nodes = 1696415
Number of edges = 22190596
Maximum Betweenness Centrality = 417029728.00
Time Taken (Parallel) = 302152 ms

Number of nodes = 1382908
Number of edges = 27182946
Maximum Betweenness Centrality = 83918.40
Time Taken (Parallel) = 138279 ms

Number of nodes = 540486
Number of edges = 30491458
Maximum Betweenness Centrality = 2893632.00
Time Taken (Parallel) = 20384 ms

Number of nodes = 3774768
Number of edges = 33037894
Maximum Betweenness Centrality = 6319996.00
Time Taken (Parallel) = 961595 ms

Number of nodes = 4847571
Number of edges = 85702474
Maximum Betweenness Centrality = 8498392576.00
Time Taken (Parallel) = 3554475 ms
```

TIME COMPARISON FOR DIFFERENT GRAPHS

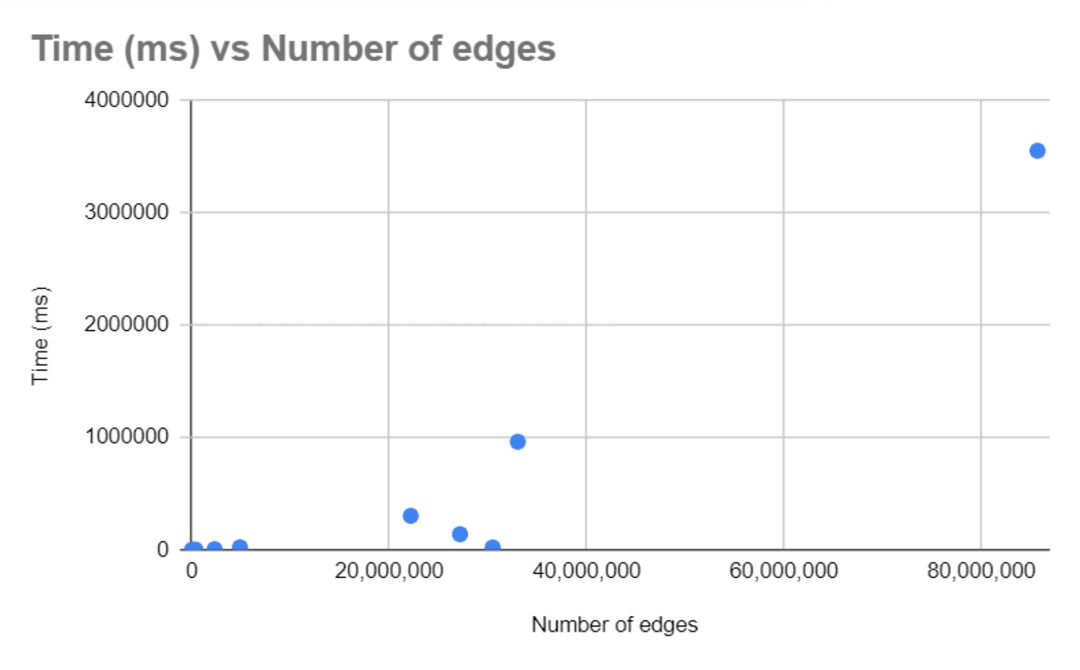
The time comparison is done for the parallel work efficient method with coarse grained parallelism

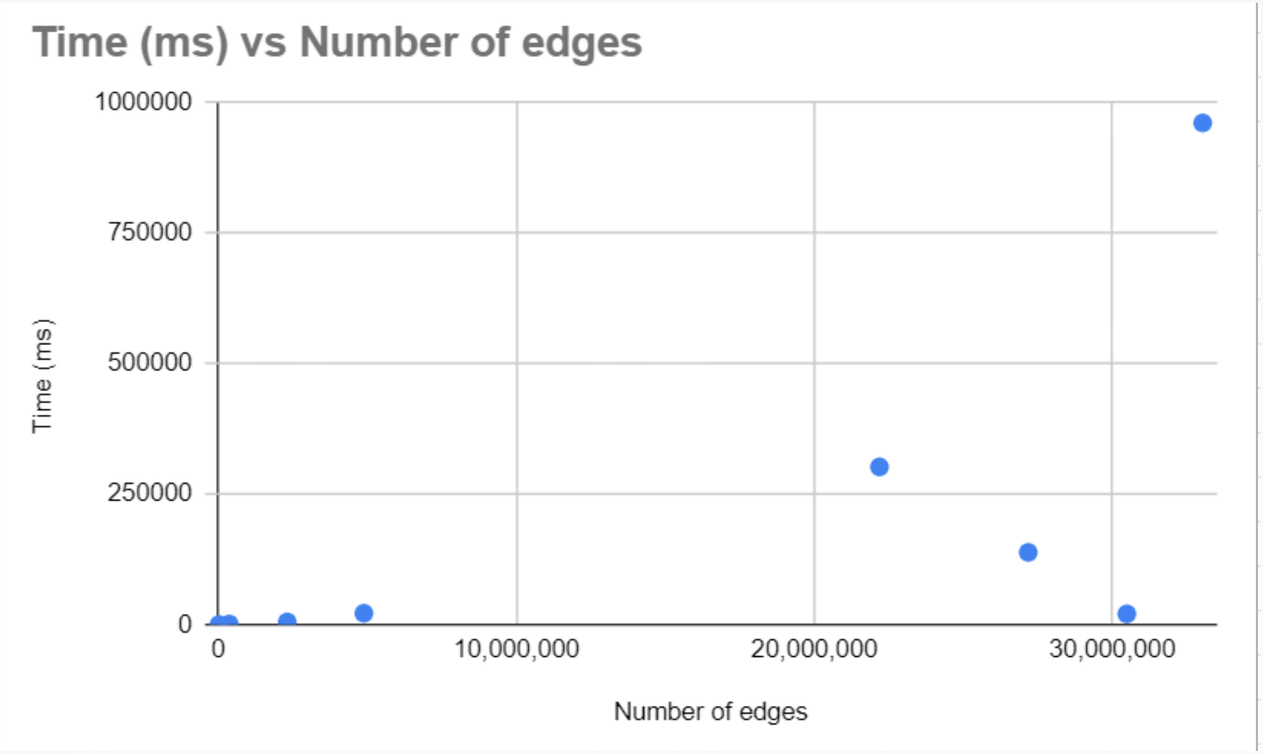
Name of graph	Type of graph	Number of nodes	Number of edges	Time (ms)	Time (min, sec)	Maximum BC
sample_1_Gnutella	P2P network	6,301	20,777	3	0.003 s	579
Email-Enron	Email network	36,692	367,662	962	1 s	2,219,794
citationCiteseer	Publication citations	268,495	2,313,294	4,949	5 s	110,607
amazon0601	Product co-purchases	403,394	4,886,816	21,643	22 s	64,581,512
as-skitter	Internet topology	1,696,415	22,190,596	302,152	5m 2s	417,029,728
in-2004	Web links	1,382,908	27,182,946	138,279	2m 18s	83,918
coPapersDBLP	Publication citations	540,486	30,491,458	20,384	20 s	2,893,632
cit-Patents	Patent citations	3,774,768	33,037,894	961,595	16m 1s	6,319,996
soc-LiveJournal1	Journal community	4,847,571	85,702,474	3,554,475	59m 14s	84,989,392,576

VISUALIZATION USING GRAPHS



VISUALIZATION USING GRAPHS





REFERENCES

- Graphs: https://snap.stanford.edu/data/index.html
- Graphs: https://userweb.cs.txstate.edu/~burtscher/research/ECLgraph/index.html
- Code: https://developer.nvidia.com/blog/accelerating-graph-betweenness-centrality-cuda/
- Code: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.728.2926&rep=rep1&type=pdf
- Brandes Algorithm: https://matteo.rionda.to/centrtutorial/BonchiDeFranciscisMoralesRiondato-CentralityBigGraphsTutorial-Slides.pdf
- Further reading: https://github.com/barisbatuhan/centrality_project?tab=readme-ov-file

THANK YOU