Course Project: Parallel & Concurrent Programming

Authors:
Aayush Adlakha (CS22BTECH11001)
Jash Jhatakia (CS22BTECH11028)

Directory Structure and File Explanation:

- `Graphs/` : Contains the test graphs
- `Src_Prjt-CS22BTECH11001-coarse.cpp` : Contains the code for the coarse-grained parallel implementation of the Brandes Algorithm. The outer loop of the Brandes Algorithm is given to each thread.
- `Src_Prjt-CS22BTECH11001-coarseOmp.cpp` : Contains the code for the coarse-grained parallel implementation of the Brandes Algorithm using OpenMP.
- `Src_Prjt-CS22BTECH11001-fineOmp.cpp` : Contains the code for the fine-grained parallel implementation of the Brandes Algorithm using OpenMP. All the neighbours of a BFS node are given to different threads. While performing BFS, the task of exploring the neighbors of a vertex is parallelized. Each thread does the task of exploring the set of neighbors of the vertex.
- `Src_Prjt-CS22BTECH11001-fineThreadPool.cpp` : Contains the code for the fine-grained parallel implementation of the Brandes Algorithm using ThreadPool.
- `Src_Prjt-CS22BTECH11001-mediumChunk.cpp` : Contains the code for the medium-grained parallel implementation of the Brandes Algorithm. Here the exploration of a vertex in one level is parallelized and each thread is given a chunk of vertices to explore.
- `Src_Prjt-CS22BTECH11001-mediumOmp.cpp` : Contains the code for the medium-grained parallel implementation of the Brandes Algorithm using OpenMP.
- `Src_Prjt-CS22BTECH11001-mediumThreadPool.cpp` : Contains the code for the medium-grained parallel implementation of the Brandes Algorithm using ThreadPool.
- `Src_Prjt-CS22BTECH11001-sequential.cpp` : Contains the code for the sequential implementation of the Brandes Algorithm.
- `Src_Prjt-CS22BTECH11001-writeRandomGraphTofile.cpp` : Contains the code to generate a random graph with a given number of vertices and write it to a file.
- `Makefile` : Contains the commands to compile the code.
- `Rpt_Prjt-CS22BTECH11001.pdf` : Contains the report of the project.
- `Turitin_Prjt-CS22BTECH11001.pdf` : Contains the Turnitin report of the project.
  NOTE: The report got 25% percent similarity as per Turnitin, but the similarity is only in the definitions, standard theorems, certain sections of the pseudo codes taken from papers. The observations and analysis has no similarity.
- `output.txt` : Contains the output of the code for the test graphs.

Creating test graphs

To create test graphs with `<number_of_vertices>` nodes, run the following command:

```
g++ Src_Prjt-CS22BTECH11001-writeRandomGraphTofile.cpp -o
writeRandomGraphTofile
```

```
./writeRandomGraphTofile <number_of_vertices>
```

This will create a file `graph-<number_of_vertices>.txt` with the test graph, in the `Graphs/` directory.

Compiling the code

- To compile all the files, run the following command:

```
make all
```

- To compile the `Src_Prjt-CS22BTECH11001-coarse.cpp` file, run the following command:

```
make coarse
```

- To compile the `Src_Prjt-CS22BTECH11001-coarseOmp.cpp` file, run the following command:

```
make coarseOmp
```

- Similarly, use the `make` command to compile any other file.

Running the code

After compiling the code, run the following command to run the code:

```
./<executable> <graph_file> <number_of_threads>
```

Output

The output of the code will be stored in the `output.txt` file.