MPI and OpenMP Pathfinding Program

This program is a parallel implementation of Dijkstra's algorithm and Yen's K-Shortest Paths algorithm, using MPI for distributed computation and OpenMP for thread-level parallelism. The code is designed to find multiple shortest paths in a graph.

Prerequisites

An MPI implementation (e.g., OpenMPI or MPICH)
An OpenMP-compatible compiler (e.g., GCC or Clang)
For questions related to the OS, visit: https://www.kali.org/docs/

Compilation Instructions

To compile the program with both MPI and OpenMP support, ensure you have the necessary packages installed, then use the following command:

```
mpicxx -fopenmp -o parallel_file parallel_file.cpp
```

This command compiles the C++ code with support for OpenMP and MPI. The -fopenmp flag is required for OpenMP. Ensure that mpicxx is in your system's PATH or provide the full path to the MPI compiler.

Execution Instructions

To execute the program, you need to run it with mpirun or mpiexec, specifying the number of processes. Use the following command to execute with a desired number of MPI processes:

```
mpirun -np <number_of_processes> ./parallel_file
```

Replace <number_of_processes> with the desired number of MPI processes. You can control the number of OpenMP threads with the OMP_NUM_THREADS environment variable. To set it to 4 threads, for example, run:

```
export OMP_NUM_THREADS=4
mpirun -np <number_of_processes> ./parallel_file
```

For serial version of the code

Run the following command in the terminal:

g++ -o serialfile serialfile.cpp

To execute the serial file, run the following command in the terminal:

./serialfile

Graph Input File

The program expects a graph input file named ‘Email-EuAll.txt’, ‘Email-Enron.txt’ or ‘classic-who.csv’ in the same directory as the executable. Ensure this file is in the correct format:

Output

The program outputs:
The shortest paths found by each process
Execution time for each MPI process

Troubleshooting

If you encounter issues, consider the following:

Ensure MPI and OpenMP are correctly installed and configured.
Verify that mpicxx and related binaries are in your system's PATH.
Check environment variables like OMP_NUM_THREADS for thread-level control.
If using a cluster, ensure it's properly configured for MPI.

Additional Notes

If you need to run with a different number of OpenMP threads on each MPI process, you can set OMP_NUM_THREADS individually for each process.
Consider tuning the number of MPI processes and OpenMP threads for optimal performance based on your system's hardware resources.