File description hierarchical_structure.c

The file in question aims to describe the behavior of the code within the hierarchical_structure.c file. To implement communication within a hierarchical structure, we exploited the use of the MPI (Message Passing Interface) library, which is usually used for parallel computing. The code in question aims to test the hierarchy within the structure (responsibilities and various tasks for processes) and also to create ad hoc communication for the structure.

The structure designed includes:

- **A central server:** the process that covers this role is the process with the global rank equal to 0. It has the task of initializing the message (which for the purposes of the Knn algorithm is the point to be classified, in fact the communication has been tested with data types that will then be officially used for the knn algorithm). Once the message has been initialized (an array of four floating point elements) the central server will take care of sending it to the intermediate servers.

- **Three intermediate servers:** processes with global rank 1,2,3 have been identified as intermediate servers. They will receive the message with the point to be classified by the central server, and will forward it to the various groups of processes called local devices.

- **N local devices:** Local devices will be all remaining processes. Their task will be to execute the knn algorithm in multithreading on their CPU and to send the group classification to their competent intermediate server. The local devices will therefore be divided into three groups, each of which will be managed by a different intermediate server.

Intermediate servers and servers will also be located within a group with a value of 0, so as to allow communication between these processes. The groups are numbered from 1 to 3, the association of the intermediate servers with the various groups is made based on the number of the intermediate server.

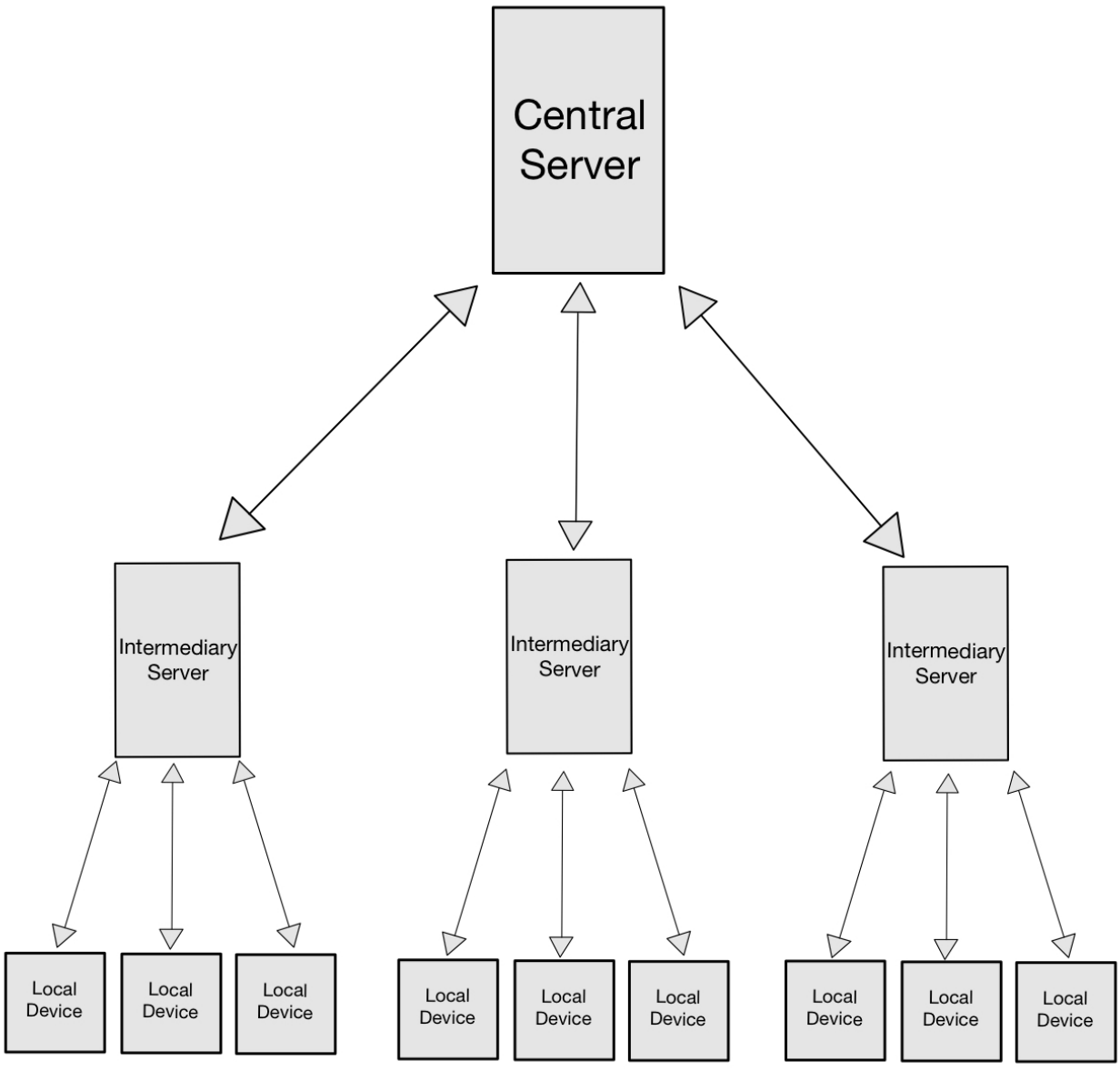
The test carried out (as can be seen from the output.txt file) followed the following communication flow:

1. central server broadcasts the message to the group of intermediate servers.
2. each intermediate server receives and forwards the broadcast message to its own group for which he is responsible.

3. Local devices receive the message, perform processing and send the results to their intermediate server (MPI_Gather()).

4. Intermediate servers receive the results from local devices in their group, calculate the average for the group and forward it to the central server.

5. The central server receives the averages of each group and calculates a further average which it assigns as the definitive label.



The groups were created using the MPI_Comm_Split() function, thus dividing the global MPI_COMM_WORLD communicator into the different group communicators, thus managing to have a clear division of processes and linear intra-communication