Practical Work 3: MPI File Transfer

Phạm Tưởng Ngạn
Distributed Systems

3rd December

1 System Overview

This practical work implements a file transfer system using **mpi4py** (MPI for Python). The system allows the transfer of files between two processes running within the same MPI communicator.

2 Service Design

The service assigns roles based on the MPI process Rank:

- **Rank 0 (Sender):** Reads the file from disk and sends data.
- **Rank 1 (Receiver):** Receives data and writes the file to disk.

Communication is performed using point-to-point blocking `send` and `recv` operations.

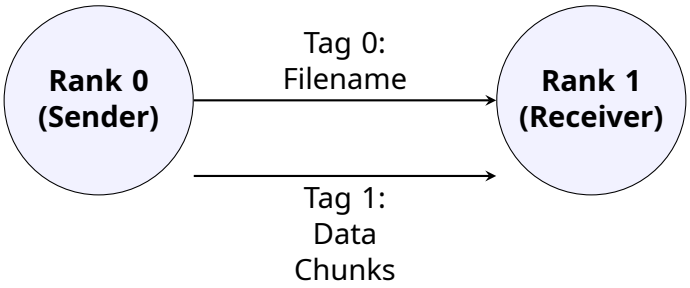

Figure 1: MPI Service Design

3 System Organization

The system runs as a single script where execution paths diverge based on the MPI Rank. The `COMM_WORLD` communicator handles synchronization between the sender and receiver.

4 Implementation

The implementation uses specific Tags to manage the data flow:

- **Tag 0:** Transmits the filename.
- **Tag 1:** Transmits file content chunks.

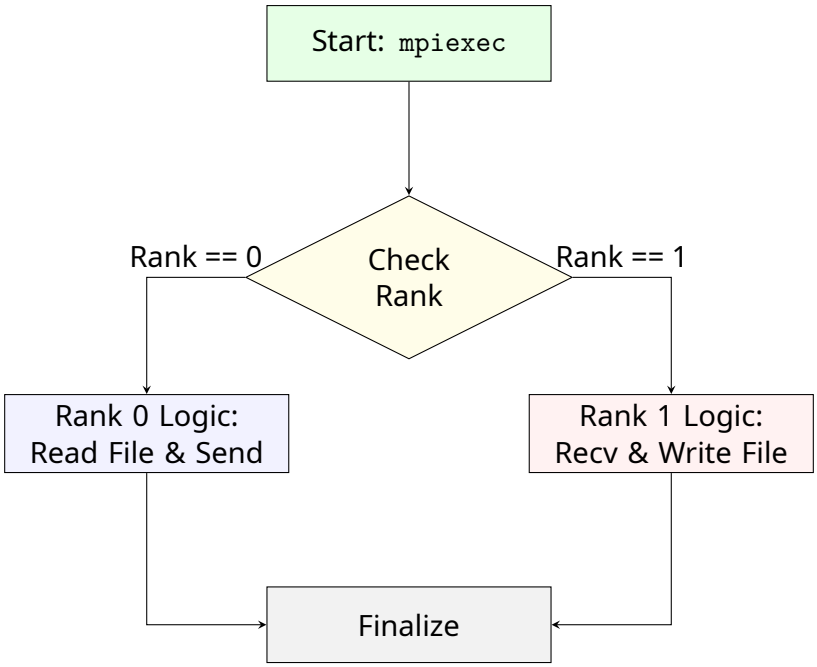

Figure 2: System Flow

A `None` object is sent at the end of the transfer to signal EOF (End of File).

```python
from mpi4py import MPI

def run():
    comm = MPI.COMM_WORLD
    rank = comm.Get_rank()

    if rank == 0:
        # Sender
        filename = "test.txt"
        comm.send(filename, dest=1, tag=0)

        with open(filename, 'rb') as f:
            while True:
                chunk = f.read(4096)
                if not chunk: break
                comm.send(chunk, dest=1, tag=1)

        comm.send(None, dest=1, tag=1) # EOF Signal

    elif rank == 1:
        # Receiver
        filename = comm.recv(source=0, tag=0)

        with open("recv_" + filename, 'wb') as f:
            while True:
                chunk = comm.recv(source=0, tag=1)
                if chunk is None: break
                f.write(chunk)
```

Listing 1: MPI Transfer Logic