Distributed System

HTTP over MPI

GroupID 3 - ProjectID 4

Tran Kim Quoc Tuan
Phan Manh Tung
Dinh Van Luong

Ngo Thanh Tung
Tuong Duy Linh
Duong Gia Bach

What is MPI?

- **M**essage **P**assing **I**nterface
  + Perform well on distributed memory machines.
  + The standard **parallel programming** interface.
- Different implementations, interfaces (openmpi, mvapich, mpich) in **C/C++**, **Fortran**, **Python** via MPI4Py.

Message Passing Paradigm

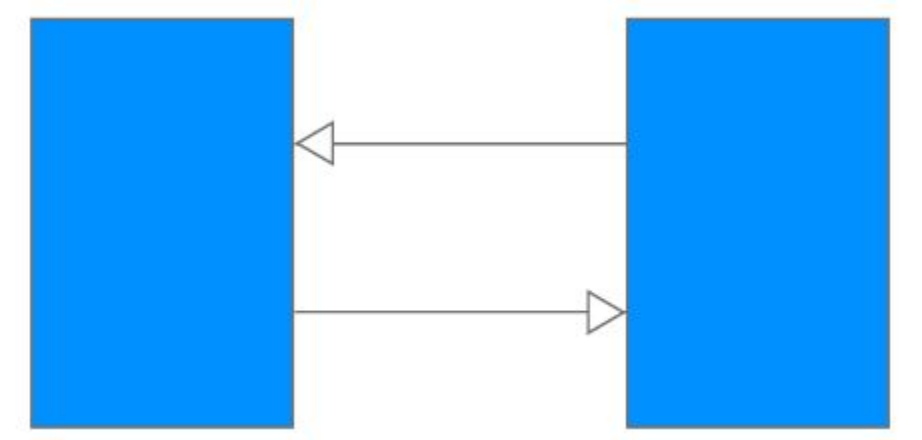
- A parallel MPI program run as **separate processes** with their own address space.
- Data is moved from the address space of one process to that of another process.
- Communication (2 types)
  + **Point-to-point** involve only two tasks (A to B).
  + **Collective messages** involve a set of tasks (among A, B, C, D).

Communicators

- MPI uses **communicators** to identify which processes communicate **only within their set**.
- **MPI_COMM_WORLD** is the beginning of the program, defined as all the processes, required for most MPI calls.
- **Rank**
  - **Unique process ID** within a communicator. (0,1,2,3)
  - Assigned by the system when the process initializes.
  - Used to specify **sources** and **destinations** of messages.

Point-to-point communication

- Data transferred between 2 processes

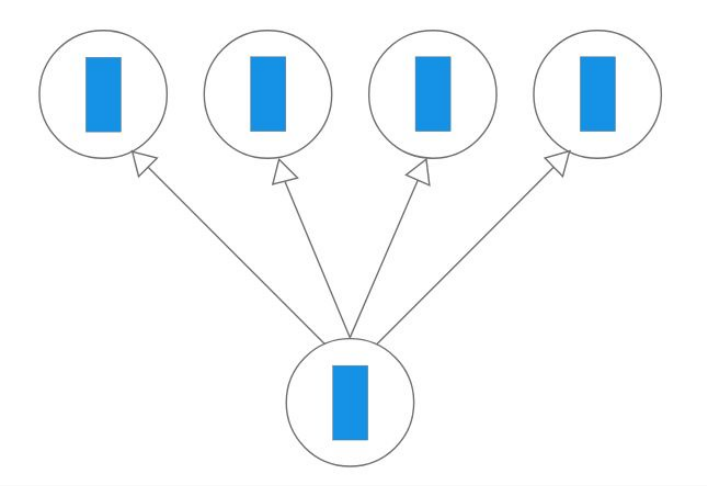


Collective communication

- Broadcast.
- Scatter.
- Gather.
- Reduction.

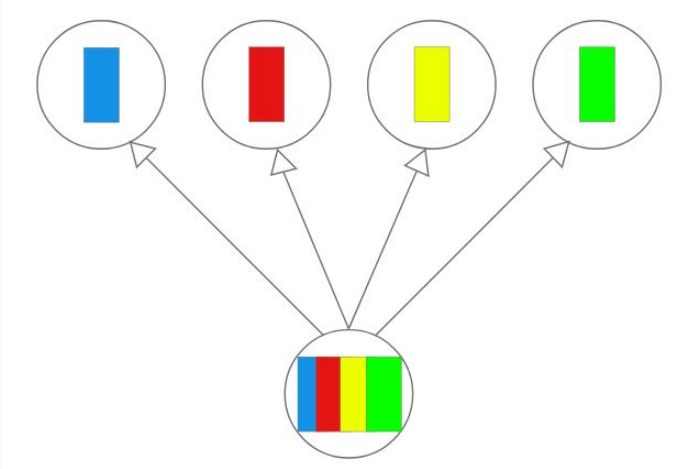
Broadcast communication

- Take one and send it out to all the processes.
- It distributes it to all the processes.



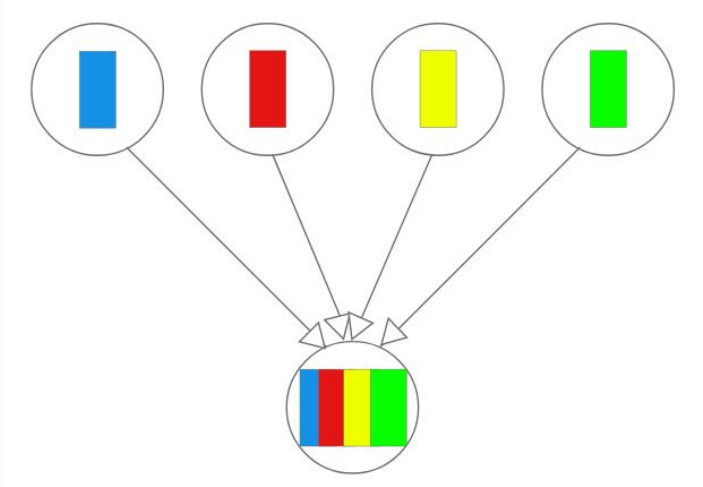
Scatter communication

- Take one and decompose it into multiple.
- Send one of those out to each process.



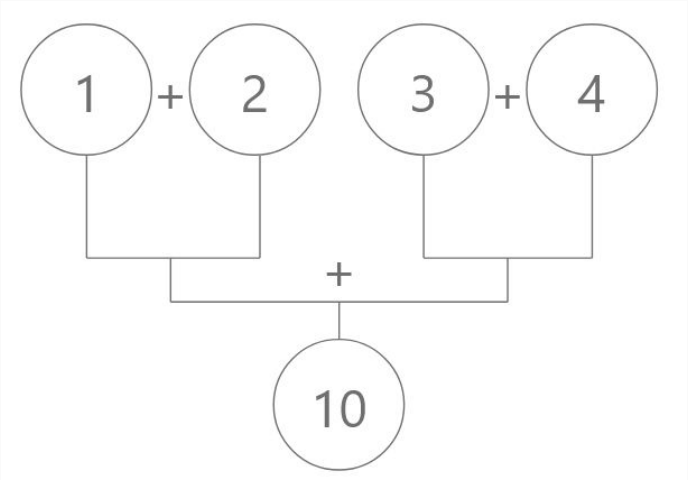
Gather communication

- **Reverse** of the scatter communication,
- Take one from each process and gather them into a single one.



Reduction communication

- Do operation in **parallel**.
- Addition (+), subtraction (-), multiply (*), divide (/), max, min,...



Method

- **MPI4Py** is very similar to the MPI standard C++ interface.
- Communication with Python objects.
- Lost in performance, gain in **shorter** development time.
- Platform: **Google Colab**.

Application with HTTP

- **HTTP** (**H**yper**T**ext **T**ransfer **P**rotocol)
- Client/Server
- **Parallel programming** with MPI

Transferring messages

- Point-to-point communication
- comm.send()
- comm.recv()

Transferring messages

```
[ ]  %%file share.py

     from mpi4py import MPI
     import time

     comm = MPI.COMM_WORLD

     rank = comm.rank
     size = comm.size
     name = MPI.Get_processor_name()
     message = "Hello"

     if rank == 0:
       data = message
       comm.send(data, dest=1)
       print("from node", rank, "sent data")
       time.sleep(5)

     elif comm.rank == 1:
       data = comm.recv(source=0)
       print("Node", rank, "received mess", data)
```

```
Overwriting share.py
```

```
[ ]  ! mpiexec -np 2 --allow-run-as-root python share.py
```

```
from node 0 sent data
Node 1 received mess Hello
```

Parallel programming (HTTP request)

- Collective communication
- comm.scatter()
- comm.gather()

Parallel programming (HTTP request)

```
[ ] %%file gather.py

    import requests
    from mpi4py import MPI

    comm = MPI.COMM_WORLD

    rank = comm.rank
    size = comm.size
    name = MPI.Get_processor_name()

    sendbuf = []
    root = 0
    if comm.rank==0:
      urls = [
        "https://httpbin.org/ip",
        "https://httpbin.org/xml",
        "https://httpbin.org/json",
        "https://httpbin.org/image",
        ]
      sendbuf = urls

    v = comm.scatter(sendbuf, root)
    print("I got the url")
    v = requests.get(v).content

    recvbuf = comm.gather(v, root)
    if comm.rank == 0:
      for i in recvbuf:
        print(i)
```

```
Overwriting gather.py
```

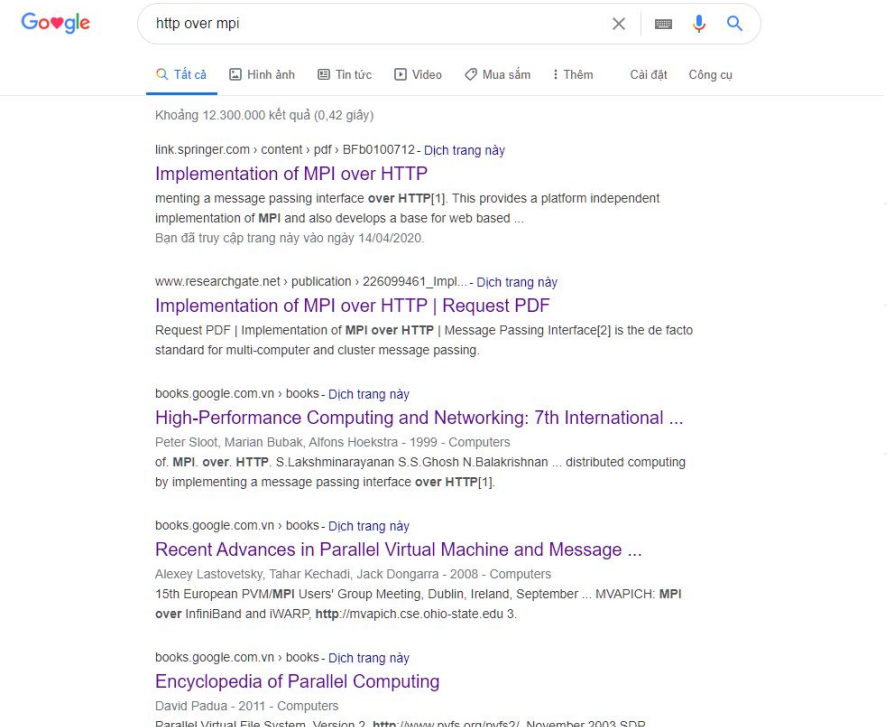
```
[ ] ! mpiexec -np 4 --allow-run-as-root python gather.py
```

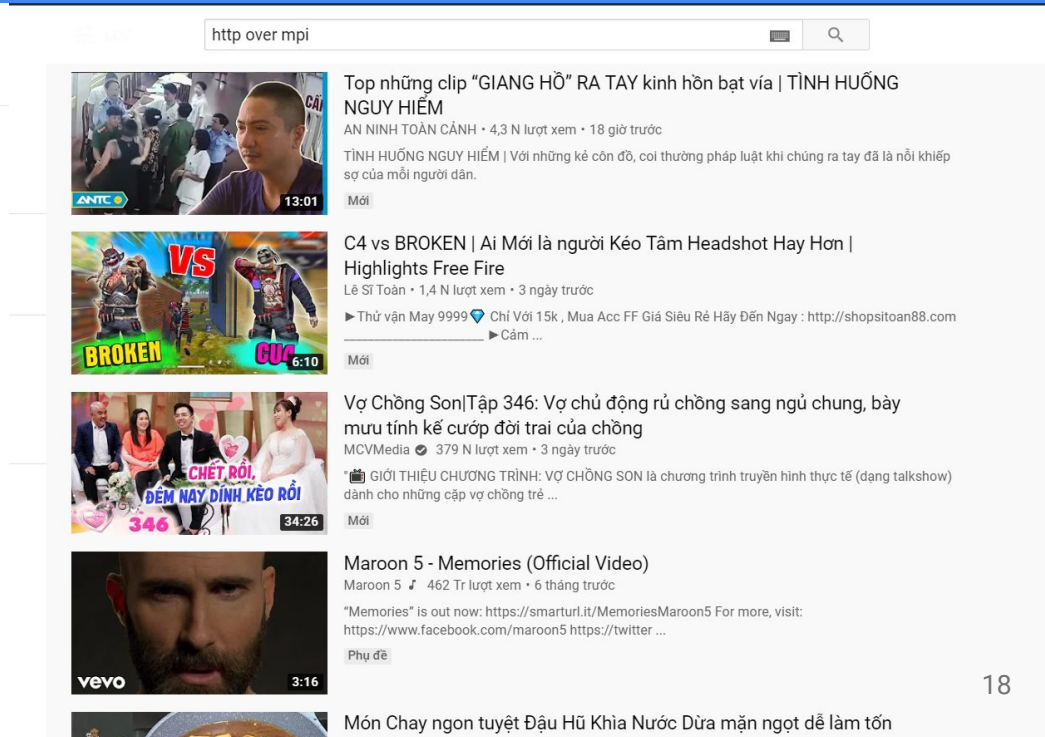
```
I got the url
I got the url
I got the url
I got the url
b'{\n  "origin": "35.247.24.83"\n}\n'
b'<?xml version=\'1.0\' encoding=\'us-ascii\'?>\n\n<
b'{\n  "slideshow": {\n    "author": "Yours Truly",
b'{"message": "Client did not request a supported me
```

Evaluation

- Quick system response
- Efficiency

State of the art





Demo