Lab 1 Parallel Programming with MPI

Point to Point Communications

1.Muc tiêu:

- SV tìm hiểu và sử dụng các hàm cơ bản trong thư viện MPI

- Viết một chương trình MPI đơn giản minh họa việc truyền thông điệp qua lại giữa các process.

- Một số hàm SV cần tìm hiểu :

  - o MPI_Init(), MPI_Comm_rank(), MPI_Comm_size(), MPI_Finalize().

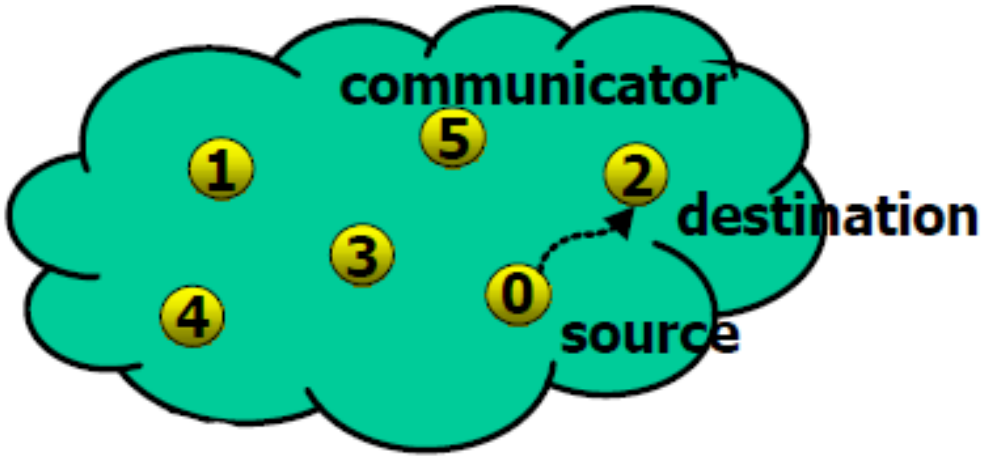
  - o MPI_Send(), MPI_Ssend(), MPI_Bsend(), MPI_Rsend(), MPI_Issend(), …

  - o MPI_Recv(), MPI_Irecv().

  - o MPI_Wtime(), MPI_Get_count(), MPI_Wait(), MPI_Test().

2. Nội dung:

2.1 Giới thiệu



- Sự giao tiếp giữa 2 process

- Process nguồn gửi thông điệp đến process đích

- Process đích nhận thông điệp

- Hoạt động giao tiếp diễn ra trong một “communicator”

- Process đích được nhận biết thông qua định danh (rank) trong “communicator”.

2.2 Chương trình minh họa

2.2.1 Chương trình Hello world:

```
#include <mpi.h>

#include <stdio.h>

#include <stdlib.h>

int main(int argc, char **argv){

      int i,rank,size;

      MPI_Init(&argc,&argv);

      MPI_Comm_rank(MPI_COMM_WORLD,&rank);

      MPI_Comm_size(MPI_COMM_WORLD,&size);

      printf("Hello world, I have rank %d out of %d processes
\n",rank,size);

      MPI_Finalize();

      return 0;

}
```

☆ Để biên dịch và thực thi chương trình, SV thực hiện các lệnh sau:

$ mpicc hello.c –o hello

$ mpirun –np 10 –hostfile fileName ./hello

(trong đó fileName là tên của file host đã tạo)

☺ SV nhận xét thứ tự các dòng hello được xuất trên màn hình.

2.2.2 Chương trình minh họa giao tiếp point to point: pPoint.c

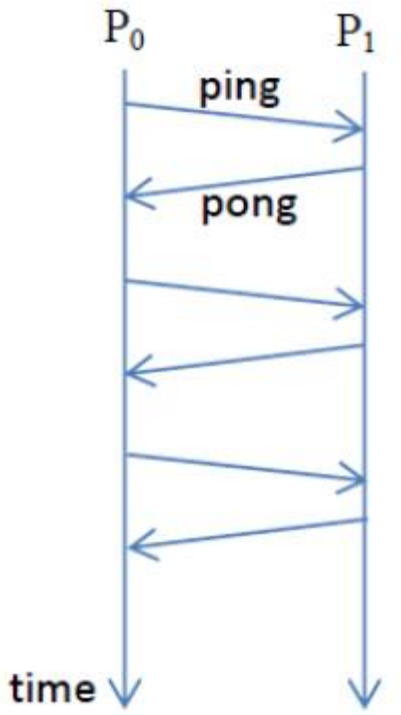
```
#include <mpi.h>

#include <stdio.h>

#include <stdlib.h>
```

```
int main(int argc, char **argv){

    int  rank,size;

    double a,b,s;

    MPI_Status status;

    MPI_Init(&argc,&argv);

    MPI_Comm_rank(MPI_COMM_WORLD,&rank);

    MPI_Comm_size(MPI_COMM_WORLD,&size);

    fprintf(stdout,"\n Process %d of %d processes starts \n", rank, size);

    if(rank == 1) {
        b = 12.2;

        MPI_Recv(&a,1,MPI_DOUBLE,MPI_ANY_SOURCE,MPI_ANY_TAG,MPI_COMM_WORLD,
        &status);

        s = b - a ;

        fprintf(stdout," The result is : %f \n", s);

    }

    else {

        a = rank;
        MPI_Send(&a, 1, MPI_DOUBLE, 1, 11, MPI_COMM_WORLD);

    }

    MPI_Finalize();

    return 0;

}
```

3. Bài tập:

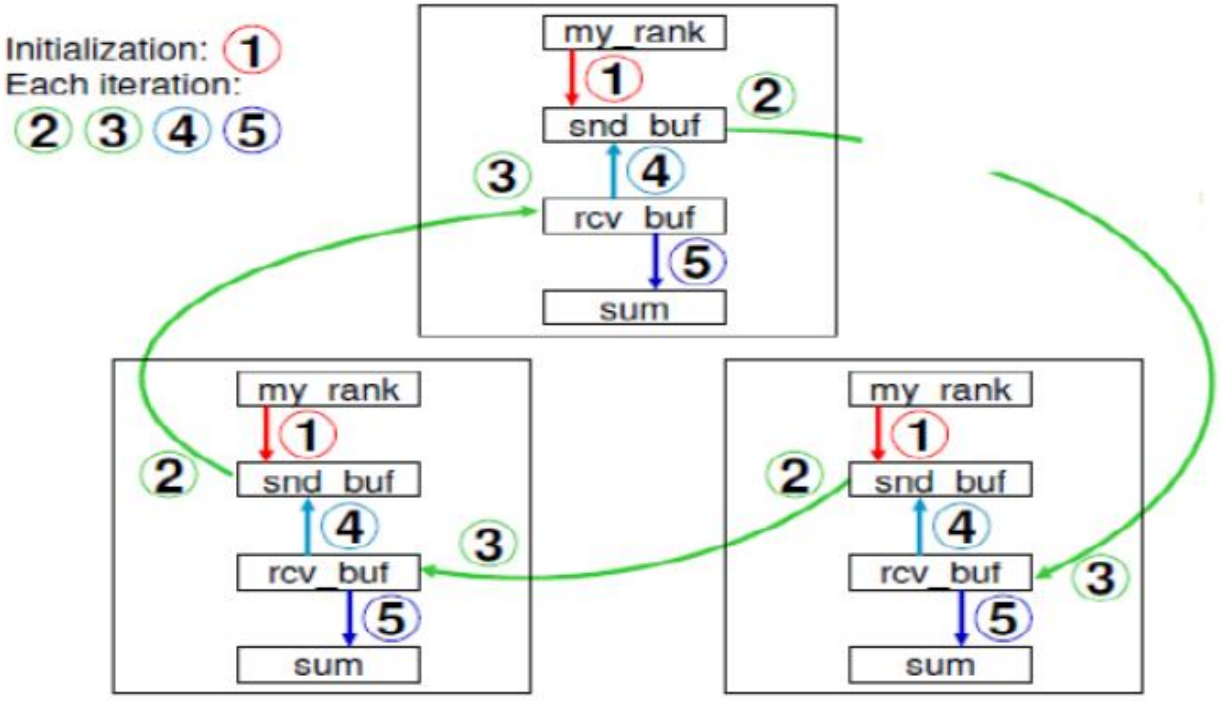
SV sử dụng MPI_Send, MPI_Recv,… của thư viện MPI viết các chương trình sau:

3.1 Viết chương trình minh họa giao tiếp của hai process như hình sau:



- Process 0 gửi thông điệp đến Process 1 (ping)
- Sau khi nhận thông điệp này, Process 1 gửi thông điệp về cho Process 0 (pong)
- Lặp lại quá trình ping-pong với độ lớn tùy ý, vd: 50, 80, 90 …
- Dùng hàm đo thời gian của MPI (MPI_Wtime()) để đo thời gian truyền một thông điệp.

3.2 Viết chương trình gửi dữ liệu giữa một tập các process theo dạng xoay vòng (ring), mỗi process đều cập nhật giá trị dữ liệu cho riêng nó. Kết quả là mỗi process đều có cùng giá trị dữ liệu sau một chu kỳ truyền nhận thông điệp. Hình sau minh họa cho quá trình thu thập thông tin của 3 process theo vòng tròn:



SV hãy hiện thực chương trình trên theo hình minh họa với số lượng process tùy ý (có thể lớn hơn 3), kết quả của chương trình là giá trị sum được in ra từ mỗi process.

- my_rank : giá trị process muốn gửi đi
- snd_buf : buffer dùng để gửi dữ liệu
- rcv_buf : buffer dùng để nhận dữ liệu

Tham khảo cho hàm MPI_Send và MPI_Recv trong C:

int MPI_Send(void *buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm)

buf : địa chỉ của dữ liệu được gởi.

count : số phần tử cần gửi đi.

datatype : Kiểu dữ liệu được gửi đi.

dest : Định danh (rank) của tiến trình nhận.

tag : Giá trị dùng để nhận biết các thông điệpkhác nhau. Do người dùng chọn.

comm : Không gian giao tiếp (thường được chỉ định là tất cả các tiến trình được tạo ra).

int MPI_Recv(void* buf, int count, MPI_Datatype datatype, int src, int tag, MPI_Comm comm, MPI_Status *status)

buf : địa chỉ vùng nhận dữ liệu .

count : số phần tử nhận.

datatype : Kiểu dữ liệu nhận

src : Định danh (rank) của tiến trình gửi.

tag : Giá trị dùng để nhận biết các thông điệpkhác nhau. Do người dùng chọn.

comm : Không gian giao tiếp (thường được chỉ định là tất cả các tiến trình được tạo ra).

status : Biến lưu thông tin về thông điệp nhận.

Tham khảo các hàm MPI:

https://computing.llnl.gov/tutorials/mpi/