# Exp 2 实验报告

在 `Ring_Allreduce` 函数中，我按照如下流程实现算法：

1. 将 `n` 个数据分成 `comm_sz` 个块，由于在本实验中可以整除，忽略残余部分。
2. 共 `comm_sz-1` 个循环， $i = 0, \cdots, comm\_sz - 2$ ，每个循环，当前线程向下一个线程发送非阻塞信息，信息内容为第 $-i$ 块（模 `comm_sz` 意义下），并将接收到的信息加在自己的sendbuf中。
3. 共 `comm_sz-1` 个循环， $i = 0, \cdots, comm\_sz - 2$ ，每个循环，当前线程向下一个线程发送非阻塞信息，信息内容为第 $-i + 1$ 块（模 `comm_sz` 意义下），并将接收到的信息替换到在自己的sendbuf中。
4. 将sendbuf的内容放入recvbuf中。

`Ring_Allreduce` 函数代码如下：

```
void Ring_Allreduce(void* sendbuf, void* recvbuf, int n, MPI_Comm comm, int com
{
    int block_size = n / comm_sz;
    for (int i = 0; i < comm_sz - 1; i++)
    {
        MPI_Request req[2];
        MPI_Isend((float *)sendbuf + (my_rank + comm_sz - i) % comm_sz * block_
        MPI_Irecv((float *)recvbuf + (my_rank + comm_sz - i - 1) % comm_sz * bl
        MPI_Waitall(2, req, nullptr);
        for (int j = 0; j < block_size; j++)
        {
            ((float *)sendbuf)[(my_rank + comm_sz - i - 1) % comm_sz * block_si
        }
    }
    for (int i = 0; i < comm_sz - 1; i++)
    {
        MPI_Request req[2];
        MPI_Isend((float *)sendbuf + (my_rank + comm_sz - i + 1) % comm_sz * bl
        MPI_Irecv((float *)recvbuf + (my_rank + comm_sz - i) % comm_sz * block_
        MPI_Waitall(2, req, nullptr);
        for (int j = 0; j < block_size; j++)
        {
            ((float *)sendbuf)[(my_rank + comm_sz - i) % comm_sz * block_size +
        }
    }
    for (int i = 0; i < n; i++)
    {
        ((float *)recvbuf)[i] = ((float *)sendbuf)[i];
```

```
    }
}
```

运行 `srun -N 4 -n 4 ./allreduce 10 100000000` 指令测试时间后，得到如下结果：

```
Correct.
MPI_Allreduce:   2582.68 ms.
Naive_Allreduce: 4896.32 ms.
Ring_Allreduce:  3058.9 ms.
```