HPC Assignment 1

Yuwei Jiang, N11031694

1 MPI ring communication

1.1 Sending the sum around the ring

The program *int_ring.c* does the task of adding ranks to the sum around the ring.
Input: N is the number of loops around the ring. Assume there are P processes, then there will be $2NP$ communications in total (NP send and NP receive).
Output:

- Good/Bad: check if the result after N loops is consistent with the theoretical value: $N \cdot \frac{P(P-1)}{2}$
- Latency: Since the time of communication is much larger compared with the time of addition, thus the time of each communication can be estimated by the total time divided by the number of communications.

Some typtical results are:

System	# of processes	# of loops	Total time(s)	Latency(s)
Crunchy1	20	10^5	1.719129	4.297822×10^{-7}
	20	10^6	17.75312	4.438280×10^{-7}
Crunchy3	20	100	7.605181	1.901295×10^{-3}
	20	1000	67.92891	1.698223×10^{-3}

It can be seen that Crunchy1 has much smaller latency than Crunchy3, i.e. the former takes less time for the same amount of computation than the latter.

1.2 Sending a large array round the ring

In the program *vec_ring.c*, we communicate exactly an int array of size $2 \times 10^6/sizeof(int)$. The total time of N loops around the ring and the bandwidth (data communicated per second, estimated by $\frac{N*P*2\times10^6}{total\ time}$).

System	# of processes	# of loops	Total time(s)	Bandwidth(GB/s)
Crunchy1	20	1000	25.572582	1.564175
	20	10000	262.153428	1.525824
Crunchy3	20	10	8.896732	0.0449603
	20	100	79.981089	0.0500118

It can be seen that the approximate bandwidth of Crunchy1 is about 30 times larger than Crunchy3. Thus, Crunchy1 is a better machine than Crunchy3 in both latency and bandwidth.

2 Parallel Jacobi smoother

The program *jacobi_mpi.c* realizes the parallel Jacobi smoother as described under MPI. It inputs the discretization size N and max number of iterations M, and outputs the time elapsed and the L2 norm of residue

once reaching the accuracy required or after iterated M times.
It's easy to confirm that the residue after a fixed number of iterations for fixed large N and initial guess is independent of p. For example, for $N = 10^6$ and zero initial guess, after 10 iterations, the L2 norm of residue is all 999.998906.
GS is much more difficult to be parallelized because the operations at mesh point need to be done in order.