MPI documentation

Requirement:

Write an MPI-based program that adds two large numbers (numbers with more than 10 digits). The representation of a number will be done through an array of digits (unsigned integers) in which the least significant digit is in the first position. The two large numbers are read from files. Each of these files contains a number at the beginning representing the number of digits and then the digits of the corresponding number. Three versions will be implemented:

1. Sequential
2. Using MPI_Send and MPI_Recv
3. Using MPI_Scatterv and MPI_Gatherv

Testing results:

Matrix type	Version	Number of processes	Execution time (ms)
Number1 = 1234567891234567 89 = Number2	0	1	0.6249
	1	4	1.8597
		8	3.1324
		16	5.0469
	2	4	1.2391
		8	2.9193
		16	5.0784
N1 = 1000 N2 = 1000 (random digits)	0	1	6.5007
	1	4	6.5041
		8	6.9273
		16	7.9512
	2	4	6.7138
		8	8.8415
		16	10.2221
N1 = 100 N2 = 100000 (random digits)	0	1	275.2385
	1	4	243.7756
		8	250.4942

		16	251.8639
	2	4	243.906
		8	245.3292
		16	251.1624

Anaysis:

- In all cases, the execution time increased as the number of processes increased. The major difference is observed in the case of the two numbers with only 18 digits. The reason could be that creating processes is more costly than executing the calculations themselves.
- For the first two files, the sequential version is the fastest, but in the case of the third file, which contains numbers with many more digits, the sequential version is the slowest.
- Versions 1 (MPI_Send + MPI_Recv) and 2 (MPI_Scatterv + MPI_Gatherv) are very similar in terms of execution time.