

High Performance Computing - Programming Paradigms and Scalability

Exercise Sheet 6: MPI

1 Parallel Summation

For the parallel summation of N^2 integers a 2-dimensional torus of size $N \times N$ is to be used. Each node of the torus is initialised with the integer value $\text{val} = \text{rank} + 1$. The parallel summation works as follows:

```
1   sum = val;
2   tmp = val;
3
4   for(i = 1; i < N; i++) {
5       MPI_send(tmp, 1, MPI_INT, left neighbour, ...);
6       MPI_recv(tmp, 1, MPI_INT, right neighbour, ...);
7       sum = sum + tmp;
8   }
9
10  tmp = sum;
11
12  for(i = 1; i < N; i++) {
13      MPI_send(tmp, 1, MPI_INT, lower neighbour, ...);
14      MPI_recv(tmp, 1, MPI_INT, higher neighbour, ...);
15      sum = sum + tmp;
16  }
```

- a) Describe how the program works for a 3×3 torus by showing the content and direction of all the transferred messages for each computational step of the algorithm.

For this purpose \xrightarrow{x} shows that value “ x ” is transferred from left to right and the nodes of the torus (circles) contain the current local value of **sum**.

Example: *before:* $\leftarrow \textcircled{1} \xleftarrow{2}$ *after:* $\leftarrow \textcircled{3} \leftarrow$

- b) Determine the number of required computation steps $T(p)$ depending only on N , neglecting communication. Use this to approximate speedup and parallel efficiency and illustrate the parallel efficiency by plotting it for different values of N . Do you notice anything peculiar, how would you rate this parallel program?

2 Collective Communication

Given is a 2-dimensional torus of size $N \times N$. Nodes are labelled from 1 to N in both dimensions where node $(1, 1)$ resides in the upper left corner and node (N, N) in the lower right corner of this topology. For implementing a broadcast, each node – after receiving – always forwards the received information both to its right and lower neighbour. This procedure – starting from one dedicated root node – successively continues until all nodes have been informed. In order to keep this procedure “symmetric”, the root node should also receive from its other two neighbours.

- a) Sketch the single steps of this algorithm (drawing arrows between sender and receiver nodes) for a 4×4 torus with node $(1, 1)$ as root!
- b) Give an MPI implementation (a communication skeleton is enough) of the above algorithm using correct send/receive statements and also think about dependencies! To keep communication simple, you may refer to a node’s neighbours with left, right, up, and down instead of MPI ranks.