Exercise 1: Computing PI with point-to-point communication*

12/06/2019

An approximation to the value π can be obtained from the following expression

$$\frac{\pi}{4} = \int_0^\infty \frac{dx}{1+x^2} \approx \frac{1}{N} \sum_{i=1}^{N} \frac{1}{1+\left(\frac{i-1/2}{N}\right)^2}$$

where the answer becomes more accurate with increasing N. Iterations over i are independent so the calculation can be parallelised.

For the following exercises you should set $N = 840$. This number is divisible by 2, 3, 4, 5, 6, 7 and 8 which is convenient when you parallelise the calculation!

1. Modify your Hello World program so that each process independently computes the value of π and prints it to the screen. Check that the values are correct (each process should print the same value).

2. Now arrange for different processes to do the computation for different ranges of i. For example, on two processes: rank 0 would do $i = 1, 2, ..., \frac{N}{2}$; rank 1 would do $i = \frac{N}{2} + 1, \frac{N}{2} + 2, ..., N$. Print the partial sums to the screen and check the values are correct by adding them up by hand.

*Designed by David Henty (epcc)

3. Now we want to accumulate these partial sums by sending them to the master (rank 0) to add up:

   - all processes (except the master) send their partial sum to the master
   - the master receives the values from all the other processes, adding them to its own partial sum

   You should use the MPI routines `MPI_Ssend` and `MPI_Recv`.

4. Use the function `MPI_Wtime` (see below) to record the time it takes to perform the calculation. For a given value of N , does the time decrease as you increase the number of processes? Note that to ensure that the calculation takes a sensible amount of time (e.g. more than a second) you will probably have to perform the calculation of π several thousands of times.

5. Ensure your program works correctly if N is not an exact multiple of the number of processes P.

The function `double MPI_Wtime(void)` returns the time in seconds from an arbitrary time in the past. Example:

```
1  double start_t = MPI_Wtime();
2  //.
3  //some MPI code
4  //.
5  double stop_t = MPI_Wtime();
6  if(rank == 0){
7    printf("\nTime␣taken␣was␣␣%f␣seconds\n",
8           stop_t-start_t);
9  }
```