MPI: Practical session 2

1 MPI Version

First, check the version of MPI provided by all implementations of MPI that exist on your computer. Use the example on slide 65/66, adapted for your language of choice.

2 Quadrature

Type in the program from the end of the first lecture, compile it and run it on as many processors as you can.

The number 27720 has been carefully chosen to be divisible by any integer up to 15. However, this is pointless. Modify the program to print out `start` and `end` on every processor and convince yourself of this.

Increase the number of steps by a factor of 1,000 (to make it take a reasonable amount of time) and rerun the program:

```
time mpirun -np 16 ./int1
```

2.1 Speed

Compare the speed of the program as run on 1 core with that run on 2, 4, 6, 8, 10, 12, and 16 (or more) cores. Do you attain a factor of 16 speed-up by running on 16 cores? If not, why not?

2.2 Accuracy

Is the final result `Grand total` or `integral` identical regardless of the number of processors the code is run on?

If you find yourself getting different results, consider why this might be happening. Is this a problem? You may need to print more decimal places to see a difference. Recall that double precision corresponds to approximately 16 significant decimal figures.

2.3 Modifications

- Try changing the code so that the result is made available on process `N-1` where `N` is the number of processors on which the code is run.
- Change the code so that the result is available on all processors simultaneously.
- What happens if you replace `MPI_SUM` by `MPI_MAX`, `MPI_MIN`, `MPI_PROD`? Can you think of algorithms where each of these might be required?
- What happens if you replace `MPI_DOUBLE` by `MPI_FLOAT`, `MPI_INTEGER`, or `MPI_LONG`? What does this tell you about type-checking in MPI?