SF2568: Homework 2

Michael Hanke

January 22, 2021

1. The broadcast operation is a one-to-all collective communication operation where one of the processes sends the same message to all other processes.

   (a) Assume our simple communication model for point-to-point communication. A straightforward implementation would require $P-1$ communication steps. Design an algorithm for the broadcast operation using only poit-to-point communications which requires only $O(\log P)$ communication steps. (2)

   (b) Do a (time-)performance analysis for your algorithm. (2)

   (c) How can the scatter operation be implemented using $O(\log P)$ communication steps? (2)

2. Consider a matrix A distributed on a $P \times P$ process mesh. An algorithm has been given in the lecture for evaluating the matrix-vector product $y = Ax$. While x is column distributed, y is row distributed. In order to carry out a further multiplication Ay, the vector y must be transposed (the row-distributed y must be redistributed to become column-distributed).

   (a) Design an algorithm for this transposition. (1)

   (b) Make a performance analysis of your transposition algorithm. (1)

   (c) An extra credit will be given for a good (!) solution in the case that A is distributed in a $P \times Q$ process mesh with $P \neq Q$.

3. We consider the problem of solving the 1-dimensional differential equation

$$u'' + r(x)u = f(x), \quad 0 < x < 1$$
$$u(0) = 0, \quad u(1) = 0$$

   Assume that $r(x) \leq 0$ for all $x \in (0,1)$. The equation can be discretized as follows: For a give $N > 0$ let $h = 1/(N+1)$ and $x_n = nh$, $0 \leq n \leq N+1$. Then the discrete system reads

$$\frac{1}{h^2}(u_{n-1} - 2u_n + u_{n+1}) + r(x_n)u_n = f(x_n), \quad n = 1, \ldots, N$$

where $u_n \approx u(x_n)$ and $u_0 = u_{N+1} = 0$. One possibility for solving this linear system of equation is Jacobi iteration. Let $u_n^{[0]}$ be a given guess of the solution. The sequence $u^{[k]}$ given by

$$u_n^{[k+1]} = (u_{n-1}^{[k]} + u_{n+1}^{[k]} - h^2 f(x_n))/(2.0 - h^2 r(x_n))$$

converges (slowly!) towards the exact solution.

Your task is to write a program which implements this algorithm using MPI. Test it out on a parallel computer. As a result, hand in a matlab plot of the solution and another one of the error for a nontrivial problem of your choice, that is, $r(x)$ should be a nonconstant function. Additionally, hand in the source code of your program. (4)

Use $N = 1000$. Since the speed of convergence is very slow, around 1000000 steps may be necessary.

Hints:

- Your starting point may be the skeleton `poisson1D.skel.c` to be found in canvas. A Fortran version is also available (`poisson1D.skel.f`).
- Implement red/black communication for the transfer of the overlap between adjacent processors.
- In order to avoid the gathering of the complete array at the master processor, let each processor print out its own part of the discrete solution.
- Do not forget to switch on optimization during compilation!
- In order to be able to plot the error of your numerical approximation you must know the exact solution. This can be done as follows: Fix an "exact" solution u of your choice. Then, for your selected r, insert u into the equation and compute the corresponding right-hand side f by $f := u'' - ru$.