exercises corresponding to chapter 6

The exercises below are ordered chronologically, using a double numbering. The first element in this double numbering refers to the lecture. A good understanding of the lecture notes (what was written on the blackboard in class) should suffice to find the answers.
The programs we have seen in class are available on the class web site.

18.1) Change pipe_ring.c (of lecture 15) using MPI_Sendrecv() commands.

18.2) The algorithm to implement a tree barrier we have seen in class uses an ordering of the labels of the processors which emphasizes that in every step the number of processors which have the messages doubles, in case of a fan out broadcast; or, in case of a gather, the labeling emphasizes that the number of processors which yet have to send halves in every step. Implement the algorithm corresponding to the picture in 6.5.

18.3) Compare the linear barrier with the tree barrier. To make the cost of one send/recv heavier, you could perform multiple instances of the barrier, e.g.: run it a million times; or alternatively, instead of the null message, send a large matrix of doubles through. Report timings obtained with MPI_Wtime().

18.4) Implement the butterfly barrier algorithm we have discussed in class. How does it compare with the MPI_Barrier() command?

19.1) In class we stated that the prefix sum algorithm would be an efficient implementation for the MPI_Reduce() command with the sum operand. Implement this idea and illustrate it modifying the compute_pi program of lecture 13.

19.2) The power method is an iterative method to approximate the eigenvector of a matrix A corresponding to the dominant eigenvalue (eigenvalue with largest complex modulus), using the formula $x^{(k+1)} = Ax^{(k)}$, for $k = 0, 1, \ldots$. One iteration uses only two statements: (1) $x := A * x$; (2) $x := x/||x||$, for some norm $||.||$. Give the pseudo code for a parallel version of this method. Analyze the computation and communication cost, the computation/communication ratio, and the speedup.

19.3) Implement the power method, described in exercise 19.2, to see whether your analysis matches the timings of the program.

20.1) In the lecture we used fictitious numbers for the startup time and the time to send data through the network. Find realistic approximations for these times for our cluster computer (Megabit Ethernet) and UIC's supercomputer (Gigabit Ethernet). By how much does the optimal number of processors for the analysis we did in class increase with a reduced startup time? You may use the Maple worksheet posted at the course web site for your computations.

20.2) Write your own implementation of a parallel Jacobi method. Do some runs on random dense matrices and report whether the timings correspond to the analysis we have seen in class.

Homework collection is every Friday, at 1PM. Homework (unlike projects) must be completed individually. Feel free to contact me with questions about these assignments.