Introduction to Parallel Computing
Exercise 2 (Deadline: January 29)

Wolfgang Schreiner
RISC-Linz

January 5, 2001

Write on the Origin a parallel C program solving the "All Pairs Shortest Path" problem. Use the sequential program on the course Web page as the starting point of your work and use the MPI library to develop the parallel program in the message passing model.

1. Take as the comparison basis the execution times of the sequential program for matrix sizes 768 and 1024 measured in the previous exercise.

2. Write an MPI program that implements a row/column-oriented solution organizing all processors in a ring (as for the matrix multiplication algorithm discussed in class). After each "communication round", the result has to be *gathered* and *scattered* for the next round.

   For developing the MPI program

   - call `cc -O3 -lmpi` *`prog`*`.c -o` *`prog`* to create program *prog*
   - call `mpirun -np` *`n prog`* to run *prog* with n processes,
   - use `MPI_Wtime` to measure wall clock times,
   - use `xmpi` to visualize the dynamic behavior of the program for (performance) debugging.

   Manual pages are available for `mpi`, `mpirun`, and for all the MPI library calls. Various environment variables can be used to customize the execution (see the handouts respectively man pages).

3. Measure the execution times for both input sizes and 1, 2, 4, 8, 16 processors.

   For your measurements with n processors, use `top` to find times when there are n idle processors. If you cannot get n idle processors on 3 different days, use the best timings you got and mention this fact in your report.

4. Determine the asymptotic *isoefficiency function* for the algorithm using $T(s) = s+1$ as the time required for sending a message of s words (assume that a scatter to n processors takes the same time as sending n partial messages). By which factor should the dimension of the matrix grow to maintain constant efficiency if we increase the processor number from 1 to 2, 4, 8, 16? Test this claim by increasing your matrix size (from 768) correspondingly and measuring the execution times.

5. Write a report documenting your results including

   - the source code of the parallel program,
   - the execution times (processor/time tables and diagrams),
   - the parallel speedups (processor/speedup tables and diagrams),
   - the parallel efficiencies (processor/efficiency tables and diagrams),
   - the isoefficiency analysis and the corresponding time measurements,
   - a screenshot of (part of) a trace of a sample run of your program (an appropriate selection of the xmpi "Timeline" window),
   - an explanation of the behavior/performance of the program and any other findings, problems, comments, etc.