Introduction to Parallel Computing
Exercise 1 (Deadline: December 4)

Wolfgang Schreiner
RISC-Linz

November 6, 2000

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 apply the automatic parallelization features of the C compiler.

1. Compile the sequential program with `-O3`. Measure the execution time of function `path` for matrix sizes 768 *and* 1024 using the function `clock_getttime()` (see the man page) and take these times as the base times of all your comparisons with the parallel program.

2. Compile the program with `-O3` and `-apo` without changes; analyze the output generated by the compiler (`-apo keep` or `-apo mplist`) and explain them.

3. Modify the sequential program such that compilation with `-O3` and `-apo` gives a reasonably good parallel program *without using parallelization pragmas.* Measure the execution time of the program for both input sizes and 1, 2, 4, 8, 16 processors.

4. Add parallelization pragmas to the sequential program such that compilation with `-O3` and `-apo` gives a reasonably godd parallel program *with only small modifications to the program.* Measure the execution time of the program for both input sizes and 1, 2, 4, 8, 16 processors.

5. The deliverable of this exercise is a written report (separate cover page with title and author, pages stapled on the left) that documents your results including

   - the source codes of the parallel programs,
   - the compiler reports and an analysis/explanation of them,
   - the execution times (processor/time tables and diagrams),
   - the parallel speedups (processor/speedup tables and diagrams),
   - the parallel efficiencies (processor/efficiency tables and diagrams),
   - an explanation of the behavior/performance of the program and any other findings, problems, comments, etc.

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

Speedup: $S_n = \frac{T_s}{T_n}$

Efficiency: $E_n = \frac{S_n}{n}$

(T_s ...time of sequential program, T_n ...time of parallel program with n processors).