HPC - Skopje

date: Nov. 7, 2011

Some Benchmarks

We propose to read the OpenMP summary as well as the nicely written paper. Have additionally a look at a good OpenMP tutorial besides the general OpenMP homepage.

12. Download the template for the inner product of vectors (example (A)), compile and run it.

    - Try several schedule types and junk sizes in line 13 of the downloaded file *mylib.cpp* and chose the fastest for the remaining tests, see page 8 on the OpenMP summary.
    - Calculate the speedup for using 1-16 (143.50.47.201) and 1–8 (local and 143.50.47.166) threads.
      Use function `omp_set_num_threads(tn)` in your main function or call `export OMP_NUM_THREADS=tn` from the shell in order to run the code on `tn` parallel threads.

13. The same as above for example (B) and for example (D).
    Take care that you use the OMP timing routine `omp_get_wtime()` !!

14. Determine the minimum and the maximum of a vector x together with the appropriate indices. Swap the two vector components with each other.
    Use $x_i := (i \mod 219) + (1.0 + rand())/RAND\_MAX$ for the vector initialization.
    Hints: Try first to determine the maximal value of x by parallel programming (compare the result with the STL-function `max_element`). You might have to use additional omp pragma directives to avoid race conditions and/or undefined values.

15. Implement the OMP-solutions for (A)-(D) not only via the parallel loop (`#pragma omp parallel for`) but also via parallel regions (`#pragma omp parallel`), i.e., you have to determine the chunk of data for thread p by the thread-ID and the overall number of active threads.

16. Use the STL whenever possible in Ex. 15.