An Experimental Study for Parallelizing Basic Kernels From Scientific Computing using Multicore Libraries

Panagiotis D. Michailidis[a], and Konstantinos G. Margaritis[b]

[a]Department of Balkan Studies, University of Western Macedonia, Florina, Greece

[b]Department of Applied Informatics, University of Macedonia, Thessaloniki, Greece

pmichailidis@uowm.gr,kmarg@uom.gr

Key words: Parallel Processing, Linear Algebra, Multicore Programming.

Basic kernels from scientific computing such as matrix computations (i.e. dot product, outer product, matrix transpose, matrix - vector product and matrix product) and solving linear systems (i.e. Gaussian elimination and Jacobi) lie at the core of many computational applications such as computational statistics and combinatorial optimization. Often, these kernels are computation-intensive, so the sequential execution time is quite large. Therefore, it is profitable to use a multi-core platform as a high performance system for the execution. For the parallelization of the kernels from scientific computing on multi-core platforms there are many representative parallel programming libraries. These libraries are Pthreads, OpenMP, Intel Cilk++, Intel TBB, Intel ArBB, SMPSs, SWARM and FastFlow. These libraries based on a small set of extensions to the C programming language and involve a relatively simple compilation phase and potentially much more complex runtime system.

The question that is imposed by programmers is which is the appropriate libary for implementing computational kernels on multi-core so that there is a balance between easy programming effort and the high performance. The aim of this presentation is to show an unified and systematic quantitative and qualitative study of multi-core programming libraries for implementing basic kernels that based on row partitioning method. The quantitative study is based on the assess the performance of the multithread kernels on Dual Opteron 6128 CPU with eight processor cores (16 cores total) and we measured the execution time as a function of the number of cores (i.e. from 1 to 16) and of the problem size (i.e. from 1024×1024 to 5120×5120). On the other hand, the qualitative study is based on the assess the ease of programming effort in the multicore programming tools and we counted the number of lines of code needed to solve the problem. Finally, based on this extensive study we conclude that the Intel ArBB and SWARM parallel programming libraries are the most appropriate because these give good performance and simplicity of programming.