UNIVERSITY OF AMSTERDAM
Institute of Informatics
System and Network Engineering Lab

Dr. Ana Varbanescu
Dr. Clemens Grelck
Giulio Stramondo, MSc
Feb 6, 2018

Programming Multi-core and Many-core Systems

— Assignment Series 1 —

Assignment 1.1: Sequential C base version of heat dissipation simulation

Your initial task is to write a sequential C program that simulates heat dissipation on the surface of a cylinder according to the specification given in the lab project description. Pay attention to both maintainability as well as extensibility of your code and aim at "fast" code. For example, avoid function calls in (inner) loop bodies and consider the cost of different memory storage layouts for the matrices involved in the simulation. Also experiment with C compiler options and possibly different C compilers to study their effect on the runtime performance of your code.

Assignment 1.2: Experimentation with sequential heat dissipaton code

Run experiments with your sequential base implementation of the heat dissipation simulation with the following parameter sets:

- square surfaces with M=N=100, 1000, 2000;
- rectangular surfaces with N=100 and M=50, 200, 1000, 2000.
- rectangular surfaces with M=100 and N=50, 200, 1000, 2000.

Follow the guidelines for experimentation set out in the project description. Observe and compare, in particular, the FLOP/s achieved. Does the orientation of the cylinder surface in the non-sqare experiments affect the performance achieved? Explain your observations.

Assignment 1.3: Compiler optimisations

Explore the impact of compilers and compiler optimisations on the effective runtime performance of your heat dissipation simulation code. Try out both the GNU C compiler `gcc` and the Intel C compiler `icc` and compare the performance of the codes they generate.

Experiment with different optimisation levels and try to improve performance by adapting code generation to the DAS-4 hardware.

Assignment 1.4: Vectorisation with compiler intrinsics

Vectorise your heat dissipation simulation code using compiler intrinsics for the DAS-4.

Compare the performance of your hand-vectorised code with your previous version. Do the compilers automatically vectorise your code? Check compiler reporting and/or the generated assembly code. Can you beat the compiler in terms of vectorisation?

Submission due date: Feb 15, 2018