Abstract

The move from single-core processor systems to multi-core and manyprocessor systems comes with the requirement of implementing computations in a way that can utilize these multiple computational units efficiently. This task of writing efficient parallel algorithms will not be possible without improving programming languages and compilers to provide the supporting mechanisms. Computer aided mathematical modelling and simulation is one of the most computationally intensive areas of computer science. Even simplified models of physical systems can impose a considerable computational load on the processors at hand. Being able to take advantage of the potential computational power provided by multi-core systems is vital in this area of application. This thesis tries to address how to take advantage of the potential computational power provided by these modern processors in order to improve the performance of simulations, especially for models in the Modelica modelling language compiled and simulated using the OpenModelica compiler and runtime environment.

Two approaches of utilizing the computational power provided by modern multi-core architectures for simulation of Mathematical models are presented in this thesis: Automatic and Explicit parallelization respectively. The Automatic approach presents the process of extracting and utilizing potential parallelism from equation systems in an automatic way without any need for extra effort from the modellers/programmers. This thesis explains new and improved methods together with improvements made to the OpenModelica compiler and a new accompanying task systems library for efficient representation, clustering, scheduling, profiling, and executing complex equation/task systems with heavy dependencies. The Explicit parallelization approach allows utilizing parallelism with the help of the modeller or programmer. New programming constructs have been introduced to the Modelica language in order to enable modellers to express parallelized algorithms to take advantage of the computational capabilities provided by modern multicore CPUs and GPUs. The OpenModelica compiler has been improved accordingly to recognize and utilize the information from these new algorithmic constructs and to generate parallel code for enhanced computational performance, portable to a range of parallel architectures through the OpenCL standard.