Abstract

Over the last two decades, a lot has changed regarding the way modern scientific applications are designed, written and executed, especially in the field of data-analytics, scientific computing and visualization. Dedicated computing machines are nowadays large, powerful agglomerates of hundreds or thousands of multi-core computing nodes interconnected via network each coupled with multiple accelerators. Those kinds of parallel machines are very complex and their efficient programming is hard, bug-prone and time-consuming. In the field of scientific computing, and of modeling and simulation especially, parallel machines are used to obtain approximate numerical solutions to differential equations for which the classical approach often fails to solve them analytically making a numerical computer-based approach absolutely necessary. An approximate numerical solution of a partial differential equation can be obtained by applying a number of methods, as the finite element or finite difference method which yields approximate values of the unknowns at a discrete number of points over the domain. When large domains are considered, big parallel machines are required in order to process the resulting huge amount of mesh nodes. Parallel programming is notoriously complex, often requiring great programming efforts in order to obtain efficient solvers targeting large computing cluster. This is especially true since heterogeneous hardware and GPGPU has become mainstream. The main thrust of this work is the creation of a programming abstraction and a runtime library for seamless implementation of numerical methods on regular grids targeting different computer architecture: from commodity single-core laptops to large clusters of heterogeneous accelerators. A framework, OpenCAL had been developed, which exposes a domain specific language for the definition of a large class of numerical models and their subsequent deployment on the targeted machines. Architecture programming details are abstracted from the programmer that with little or no intervention at all can obtain a serial, multi-core, single-GPU, multi-GPUs and cluster of GPUs OpenCAL application. Results show that the framework is effective in reducing programmer effort in producing efficient parallel numerical solvers.