ULBS
Universitatea "Lucian Blaga" din Sibiu

Interdisciplinary Doctoral School
Doctoral domain: Computers and Information Technology

PhD THESIS

INTEGRATED SYSTEM FOR SCIENTIFIC APPLICATIONS' QUASI-OPTIMAL MAPPING ON HPC PARAMETERISED ARCHITECTURES

PhD Student:
Ing. ION-DAN MIRONESCU

PhD Supervisor:
Prof. univ. dr. ing. LUCIAN N. VINȚAN
Full-member of the Academy of Technical Sciences of Romania

SIBIU 2018

CONTENT

PhD THESIS

Keywords

HPC High Performance Computing; MDE Model Driven Engineering; LBM Lattice Boltzmann Methods; parallel and distributed computing; scheduling algorithms; multicriterial optimisation; Petri nets modeling and simulation.

SUMMARY

The goal of this PhD thesis is to develop a unified methodology for the identification of the optimal mapping of a complex scientific numerical application on a high performance, generic, highly parametrized high performance heterogeneous computing architecture. This methodology is intended for the pre-design phase of the hardware–software co-design process. Optimality means using the available resources as efficiently as possible in order to achieve maximum performance, in the presence of low energy consumption, limited complexity and other constraints, so this is a complex multi-criteria problem [Vin16b].

The objectives of the research presented in this work are as follows:

- The development of a platform that supports the integrated process of generating mapping variants, of evaluating the performances of these variants and selecting the optimal variant. This process will use *Model Driven Engineering* tools to effectively manage mapping variants and artifacts (executable models, code);
- The definition of a software application architecture that can adapt to the particularities of the different organisation levels of the hardware architecture in order to better map the concurrency expressed in software, on the parallelism available in hardware;
- The holistic modeling of the hardware–software architecture at a detail level, which will allow an assessment of the extent to which the application exploits the parallelism of hardware both at the node level and at the network level. The model must also catch the interaction between the two hierarchical levels;
- Verification and evaluation, through complex simulations, of the proposed hardware–software architecture for model and architecture validation;
- Multi-objective optimization, in a unitary analysis framework, of the modelled and implemented hardware–software architecture.

In order to evaluate the performance of the different mapping variants in the pre-design phase, we need a hardware–software model that allows an estimation of this performance in the absence of the application code and with the hardware only in concept phase (generic, parameterized HPC architecture).

The model shall represent the assembly in a uniform way so that:

- The mapping can be described by associating the concurrent components of the Computational Fluid Dynamics (CFD) application with the processing units at the heterogenous node level;
- It can be highlighted to what extent the mapping exploits the parallelism existing in hardware.

To sustain this approach, we will implement the unified methodology in an integrated environment that will be centred on the unified holistic modeling of the hardware–software

ensamble using Petri nets. For the purpose of dynamically exploiting the parallelism, the model will have the degree of detail necessary to catch both the concurrent behaviour of the components of the software application and the parallelism of the processing units on which they are assigned.

Compared to the *cycle accurate* simulators, which are typically limited to the processor, our approach allows a holistic modelling of the network, heterogeneous processing node (CPU, GPU), and processor level.

Another advantage of the Petri nets modeling method is that it allows formal verification of the models expressed in this formalism; this is not possible with the models used in *cycle accurate* simulation (which are *execution driven* simulations, with architectural information for each CPU cycle), usually implemented in code (in medium/ high level languages).

Because models have a graphical representation, a unified modeling/ simulation / verification / optimization framework can be developed. This cannot be built around detailed cycle accurate simulators at the same level. Petri networks, however, cannot simulate software execution with the same accuracy as *cycle accurate* simulators; therefore, the methodology proposed in the thesis is better suited for the pre-design phase. Performance estimations obtained through simulation with Petri networks allows the identification of a class of implementable solutions that need to be further refined at architectural level using *cycle accurate* simulations and classical *Design Space Exploration* methods.

The content of each chapter of the PhD thesis is summarized below.

Chapter 2 presents the state-of-the-art of the field, which highlights:

- The possibilities offered by *Model Driven Engineering* instruments, to integrate the design variants' generation process;
- The particularities of the current HPC arhitectures and the evolution of these architectures, in order to establish the architecture to be modeled, as well as the elements which have to be parameterized;
- The particularities of software frameworks used for the parallelisation of applications;
- The optimization methods that can be used to obtain the optimal mapping;
- The modeling and simulation methods that can be used to evaluate the performances.

Chapter 2 ends with a synthesis (Section 2.4), which contains the decisions taken and the opportunities identified for original contributions to the development of the domain.

Chapters 3, 4, 5 and 6 represent the original contribution of the author of this PhD thesis, to the domain. Thus, chapter 3 presents the models and transformations implemented within the thesis using the *Model Driven Engineering* tools. The models generated at this level are descriptive models, expressed in the *System Modelling Language* (SysML). The descriptive models allow effective management of the various investigated variants and the application of the changes imposed by the optimization process. Based on these, the executable models for the mapping variants are generated.

Chapter 4 presents the original architecture proposed for the software application, which implements a numerical method from the Lattice Boltzmann Methods (LBM) family. The architecture is resulting from the combination of the execution environments of two software platforms for parallel application development: CHARM++ and StarPU. Each of the two platforms maps better the concurency on one of the two organization levels of High Performance Computing (HPC) hardware. Thus, the parallelism model used by the CHARM++ language exploits better the node network, and the StarPU environment allows more efficient exploitation of each heterogeneous node. Combining the two platforms results in an application architecture, which allows the efficient use of both levels. In the proposed architecture, the application consists of OpenCL tasks and CHARM++ objects (chares). The OpenCL tasks implement the calculation part of the numerical algorithm. *Chares* implement the communication part of the same algorithm through a negotiation protocol.

Chapter 5 presents the original infrastructure developed in the PhD thesis for the dynamic mapping of the application on the hardware architecture. The proposed solution is an algorithm for cvasi-optimal mapping which combines a dynamically distributed scheduler (which balances the computational load between network nodes) with a lists-based dynamic scheduler (distributes the computational load at the node level). The network scheduler is based on a negotiation protocol between chares that implements a diffusive load-balancing algorithm (*work-stealing*). The scheduler from the node level uses a *Heterogeneous Earlier Finish Time* (HEFT) algorithm (to choose the processing unit on which to distribute the task) and a local load transfer algorithm (to redistribute the task, when required).

Chapter 6 describes the executable models developed by the author for hardware and for all the software levels involved in the application architecture: the two execution environments, the dynamic mapping infrastructure, the application components (chares and tasks).

By using timed Colored Petri Nets, the full hardware–software model of the application mapping on the generic parameterisable HPC system was developed and tested.

Using the hardware model of a network of homogeneous nodes, several variants of static mapping of load distribution on the nodes were simulated. The simulations were compared to those of the BigSim simulator, used to estimate the performances of CHARM++ applications on HPC systems. The results showed good qualitative concordance between the two simulations.

The behavior of the dynamic mapping algorithms, implemented in the StarPU environment, with different load types, was simulated using a heterogeneous node. The results were compared with the result of running the real application on the modeled architecture. The simulations allowed a comparative study of the different mapping algorithms. They also confirmed that the proposed model can be used to evaluate such an algorithm in the pre-design stage.

Through the integrated environment, the model has been transformed into a representation using extended timed Petri Nets. The resulting model was used for the testing through simulation of the algorithm for cvasi-optimal mapping presented in Chapter 5. Using the times obtained by simulation with the Multi2sim simulator for the task execution, it was estimated a reduction of the total execution time:

- with 15% if both, the scheduler at the network level and the scheduler at the node level, are used;
- with 8.6%, if only the scheduler at the node level is used

compared with the case when only a static distribution is used at both levels.

Based on data from literature ([Calore2017] [Martinez2009] [van Werkhoven2014]), for a cluster with 16 heterogeneous nodes running the numerical computing application, we have developed:

- An original mathematical model, which estimates the time necessary and energy consumption for calculating a given computational load for each node;
- An original heuristic algorithm that approximates the Pareto set of the problem of minimizing execution time, consumed energy, and load to the calculation of a given computational load, distributed on the cluster.

The model was used to test through simulation the cvasi-optimal mapping algorithm, described in Chapter 5 and adapted for multicriterial optimization. The ability of the alghoritm to dynamically redistribute the load in normal (current iterations) or special (initiation, increase of load due to grid refining, increase of load at collapse of a knot) situations has been tested. It has been demonstrated that the mapping algorithm has the ability to manage all these situations and generates a mapping near to the approximation of the Pareto set.

Chapter 7 presents the general conclusions of the thesis: the degree of solving the proposed objectives, the original contributions and the research perspectives.