Simulation Modelling Practice and Theory 72 (2017) 168–201

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Manycore simulation for peta-scale system design: Motivation, tools, challenges and prospects

Javad Zarrin[a,*], Rui L. Aguiar[a,b], João Paulo Barraca[a,b]

[a] ATNoG Research Group, Instituto de Telecomunicações (IT), Aveiro, P-3810-193 Aveiro, Portugal
[b] Departamento de Electrónica, Telecomunicações e Informática Universidade de Aveiro, Campus Universitário de Santiago, 3810-193 Aveiro, Portugal

ARTICLE INFO

Article history:
Received 14 July 2016
Revised 18 November 2016
Accepted 14 December 2016

Keywords:
Architectural simulations
Manycore simulators
Full-system simulation
Manycore systems
Computer architecture

ABSTRACT

The architecture design of peta-scale computing systems is complex and presents lots of difficulties to designs, as current tools lack support for relevant features of future scenarios. Novel systems must be designed with great care and tools, such as manycore architecture simulators, must be adapted accordingly. However, current simulation tools are very slow, often specific-purpose-oriented, suffer from various issues and are rarely able to simulate thousands of cores. The emergence of peta-scale systems and the upcoming manycore era brings nevertheless new challenges to computing systems and architectures, adding further difficulties and requirements on the development of the corresponding simulators. Furthermore, the design of architecture simulators for manycore systems involve methods and techniques from various interdisciplinary research areas, which in turn brings more challenges in different aspects. As system complexity grows, the growth of the simulation capacity is being outpaced (reaching the so called simulation wall). In this paper, we present the challenges for simulating future large scale manycore environments, and we investigate the adequacy of current modeling and simulation tools, methodologies and techniques. The aim of this work is to highlight how current approaches can best deal with the identified problems, smoothing the challenges of research in future peta-scale systems.

1. Introduction

Current trends in computation technology have focused in improving performance by increasing the number of cores per die (parallelism) rather than by increasing the rate of clock frequency of each core, due to the exhaustion of the Moore's law. Many companies and academic communities pushed this trend, designing multicore and manycore systems with capacity of tens to hundreds of cores per single die. These manycore processors are more like data-centers-on-a-chip than previous single processors, as a complex communication network connects the different cores. It is predictable in a near future to consider systems with a very large interconnect network in manycore machines with dimensions from thousands to millions of cores.

A consequence is that both system design and programming concepts must increasingly focus in heterogeneous parallelism. Future parallel and distributed applications, compilers, operating systems and tools must be able to scale well with the hardware nature of manycore and distributed execution. However, increasing the number of cores on a die increases the

* Corresponding author.
E-mail addresses: javad@av.it.pt (J. Zarrin), ruilaa@ua.pt (R.L. Aguiar), jpbarraca@ua.pt (J.P. Barraca).

http://dx.doi.org/10.1016/j.simpat.2016.12.014
1569-190X/

complexity of hardware designs, and has considerable impacts which result in enlargement of the potential design-space. Moreover, it brings serious challenges particularly for memory hierarchies and on-die interconnect bandwidth, both within the die and off die.

Systems must then be designed with great care and tools, such as manycore architecture simulators, must be adapted to address these disruptive challenges. Manycore simulators (i.e. manycore architecture simulators) can assist researchers from different areas. These areas include software (such as programming models, operating systems, compilers, etc.), hardware and computing architectures to model and assess future systems. The upcoming manycore era brings new challenges to computer architects that must be paralleled by the development of adequate architecture simulators. Ideally, to simulate a fully parallel system we can expect that an efficient architecture simulator should (at-least) be able to be parallelizable, and use the benefit of concurrency, enabling faster evaluation of future systems. Moreover, architecture simulators should provide a highly scalable, fast and accurate model to describe, emulate/simulate and measure the hardware details, memory hierarchy and interconnection networks. Furthermore, they must meet the stringent requirements along the lines of productivity, multi-modeling, synchronization, modularity, and event sampling capabilities. However, these properties are not true for the majority of the currently available architecture simulators. As complexity grows, the performance of a single simulated CPU core slows down, and the usage of these sequential simulators (i.e. architecture simulators) will be mainly limited by the performance of simulating a single CPU.

Note that, in this paper, we use the term “simulator” to refer to “architecture simulator” or “manycore architecture simulator” which differs from its more general meaning. We use the term “manycore simulation tools” in a slightly more general way to cover a wider range of contexts (methods and techniques). In other words, a simulator (architecture simulator) is a type of manycore simulation tool. Furthermore, the term “emulator” (or architecture emulator), as used in this paper, also differs from its general meaning in other research fields. In this paper, an emulator means a simulator which lacks support for performance measurements. In fact, we associate an emulator with functional correctness only. This means that the notion of time for an emulator is imprecise and often just a representation of the wall-clock time of the host. We use the term “emulation” to refer both to “functional simulation” and to describe the act of an emulator.

The rest of this paper is organized as follows. In Section 2, we discuss why manycore architectural simulation is needed, particularly for research on peta-scale systems. Section 3 presents in detail our taxonomy to addresses current modeling and simulation tools, as well as the methodologies that could be exploited and enhanced in order to design next generation efficient simulators. In Section 4, we provide an overview of recently proposed simulation tools for architectural analysis, which are able to simulate the entire execution cycle of application for the target systems. Other simulator types, such as those that mainly focus on physical modeling aspects (e.g. power, energy and thermal) or interconnect simulators (e.g. Network on Chip simulators), are out of the scope of paper for conciseness. In Section 5, we extract and elaborate a set of major problems and challenging issues created by manycore simulation. Finally, in Section 6, we present our conclusions, followed by a discussion of future directions for research. This includes possible approaches and solutions which can be used to solve the problems and deal with the challenges identified.

2. Simulation and peta-scale systems

Peta-scale systems are defined as systems which are able to provide peta-FLOPS, millions of billions of FLoating OPerations per Second, computational power [1,2]. They can be described as the increasingly massive and dynamic networks of interconnected diverse processors and components (i.e. elements). Such as system, as a whole, exhibits a set of properties and behaviors among the elements, which are not distinguishable from the properties of the individual processors and components. These systems are only on its infancy currently, but in future peta-scale manycore systems, we can expect to have computing nodes with more than 10,000 cores per node. We can also expect to have much more diversity (heterogeneity) of cores, interconnections and architecture designs compared to today systems.

Architectural simulation is a common method for studying and analyzing different architectures, designs and algorithms for various target systems through imitating the operation of real-world processes, processors and systems over time. Architectural simulation acts as a low cost alternative to experimentation on real systems by representation of key characteristics, behaviors and functions of the real systems. The objective of simulation is to provide capability to researchers and designers to flexibly and efficiently explore a design space. This can include analyzing the performance of current systems (e.g. architecture assessment), acquiring and predicting processor/system behaviors and evaluating novel designs. Simulation enables today’s designers to analyze and predict different aspects (such as performance, reliability and efficiency) of future’s machines. This means that simulation is a particularly useful tool when the desired target systems, such as peta-scale systems, do not currently exist in reality. However, current simulation tools are very slow, often specific-purpose-oriented, suffer from various issues and are rarely able to simulate more than 2000 cores (we discuss current simulation tools further in Section 4). Furthermore, introducing the concept of peta-scale system presents more requirements that must be fulfilled by current simulation tools. We discuss these requirements and their corresponding issues further in Section 5.

Fig. 1 demonstrates a generic structure for (software-based) simulation. As it is shown in the figure, a simulator is an application software which runs on single host or multiple networked host machines (distributed simulation). The target system is the system which needs to be simulated by simulator. Depending on user requirements the target system might be a partial system or a full system including target (simulated) OS. The term “target Instruction Set Architecture (or target ISA)”, as used in this paper, refers to the ISA of the processor architecture simulated. Similarly, the term “host ISA” refers to