ARTICLE IN PRESS

Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at SciVerse ScienceDirect

Future Generation Computer Systems

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

Exascale design space exploration and co-design

S.S. Dosanjh [a], R.F. Barrett [b,*], D.W. Doerfler [b], S.D. Hammond [b], K.S. Hemmert [b], M.A. Heroux [b], P.T. Lin [b], K.T. Pedretti [b], A.F. Rodrigues [b], T.G. Trucano [b], J.P. Luitjens [c]

[a] Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, CA, 94720, USA
[b] Center for Computing Research, Sandia National Laboratories, Albuquerque, NM, 87185, USA
[c] NVIDIA Corporation, 2701 San Tomas Expressway, Santa Clara, CA, 95050, USA

HIGHLIGHTS

- A codesign-based methodology is described for exploring the exascale design space.
- Codesign requires a multi-faceted approach.
- Architecture testbeds are being used to study performance issues of key algorithms.
- Network bandwidth degradation studies help define requirements for future systems.
- The Structural Simulation Toolkit is described, with some example use cases.

ARTICLE INFO

Article history:
Received 2 June 2012
Received in revised form
21 February 2013
Accepted 13 April 2013
Available online xxxx

Keywords:
High performance computing
Scientific computing
Co-design
Exascale preparation

ABSTRACT

The co-design of architectures and algorithms has been postulated as a strategy for achieving Exascale computing in this decade. Exascale design space exploration is prohibitively expensive, at least partially due to the size and complexity of scientific applications of interest. Application codes can contain millions of lines and involve many libraries. Mini-applications, which attempt to capture some key performance issues, can potentially reduce the order of the exploration by a factor of a thousand. However, we need to carefully understand how representative mini-applications are of the full application code. This paper describes a methodology for this comparison and applies it to a particularly challenging mini-application. A multi-faceted methodology for design space exploration is also described that includes measurements on advanced architecture testbeds, experiments that use supercomputers and system software to emulate future hardware, and hardware/software co-simulation tools to predict the behavior of applications on hardware that does not yet exist.

Published by Elsevier B.V.

1. Introduction

The United States Department of Energy's mission needs in energy, national security and science are predicted to require a thousand-fold increase in supercomputing performance during the next decade [1]. However, the transition to Exascale systems that are operable within affordable power budgets will not be possible based solely on existing computer industry roadmaps [2]. The conclusion is therefore that we not only need to support an acceleration of industry roadmaps to deliver power efficient architectures but that we also need to augment this with modifications, or in some cases rewriting of applications to utilize new approaches to hardware design and significantly increased scale [3]. The benefits of doing so are profound as they impact the entire computing industry, addressing cross-cutting issues such as energy efficiency, concurrency and programmability for users of single workstations, data centers and large supercomputers. For the users of Exascale machines there will be additional challenges including the scalability and reliability that are brought about by the extreme size of such systems.

Given the complexity of constructing such large systems and the problems associated with modifying applications to run on them, a dialog needs to be established between computer companies and application developers where feedback is able to rapidly assess and optimize designs as they are created. In this process we envisage assessment based on balancing performance benefit versus cost in terms of software complexity, portability, silicon area, *etc*. In order for trust to exist in this dialog a number of approaches might be considered including execution on prototype or early design hardware, the construction of application models including simulators or analytic performance models and an attention to creating solutions that work across a broad range of applications and do not benefit a single problem.

* Corresponding author.
E-mail address: rfbarre@sandia.gov (R.F. Barrett).

0167-739X/$ – see front matter. Published by Elsevier B.V.
http://dx.doi.org/10.1016/j.future.2013.04.018

The scale of modern scientific applications is however a potential limiter for rapid prototype assessment. Applications are typically millions of lines of source and are written to use complex algorithms and data structures. Whilst our eventual goal is to run applications on such large systems the effort required to port them is likely to be prohibitive if multiple platforms must be assessed in short time frames. It is in this context that the notion of a *mini-application* has been developed—a mini-app is a condensed implementation of one or multiple key performance issues that affect parent codes, written to be amenable to refactoring or change but representative enough to be useful in the scientific problem domain.

Our proposed methodology for Exascale design space exploration, which we discuss in this paper, includes measurement on prototype hardware, experimentation in the form of refactoring and re-implementation using a variety of programming models and algorithms and prediction using architectural simulators. To this end, we are investigating several architectural testbeds which are representative of industry trends including Intel's Many Integrated Core (MIC) processors, GPUs from NVIDIA, Fusion APUs from AMD and nodes from Convey and Tilera. Such studies are providing useful feedback to computer architects, application developers and algorithm researchers. Our experimentation is also wider than just hardware, including evaluation of execution models such as ParalleX and low-level activities such as direct measurement of energy use in contexts such as the variation of network injection bandwidth.

The ability to predict the performance, and more importantly the performance limitations, of hardware which does not currently exist or is significantly different from contemporary systems is a key facet of design exploration. Since many proposed Exascale point designs are currently proprietary or encumbered with intellectual property, many of our early evaluations are being conducted using the notion of an Abstract Machine Model (or AMM) which defines the key architectural building blocks but no specific detail. We then are able to augment AMMs with details provided by performance models, architectural simulators and information obtained from our mini-applications running on test-bed platforms to inform us of performance trade offs and available design decisions.

In this work we provide a detailed overview of our Exascale methodology including descriptions of currently running projects to produce relevant mini-applications, accurate and relevant architectural simulation tools and our prototype test bed program which is being used to drive programming model assessments and improvements in our simulation and modeling capabilities. In addition we describe a validation methodology which is being developed to demonstrate the applicability of mini-applications to their parent codes enabling HPC vendors and researchers to have a high degree of confidence in results obtained from studies using mini-apps.

2. Miniapplications

Full-scale computational science and engineering (CSE) applications are often large and complex, depend upon numerous third-party libraries and require substantial systems programming expertise in order to compile and execute. Because of this, we are compelled to use application performance proxies for early-phase design studies meant to target a particular suite of applications. Numerous types of proxies are useful for design studies, depending on the specific context. Fig. 1 summarizes some of the key proxies used by the systems performance community.

Application performance is determined by a combination of many choices: hardware platform, runtime environment, languages and compilers used, algorithm choice and implementation, and more. In this complicated environment, we find that the use of mini-applications is an excellent approach for rapidly exploring the parameter space. Furthermore, use of mini-applications enriches the interaction between application, library and computer system developers by providing explicit functioning software and concrete performance results that lead to detailed, focused discussions of design trade-offs, algorithm choices and runtime performance issues.

Unlike a benchmark, the result of which is a metric to be ranked, the output of a miniapp is a richer set of information, which must be interpreted within some, often subjective, context. We distinguish this from a *compact-application* whose purpose is to replicate a complex domain-specific behavior being used in a parent application. Miniapps are designed specifically to capture some key performance issue in the full application but to present it in a simplified setting which is amenable to rapid modification and testing. Note that this is also distinct from a *skeleton application*, which is typically designed to focus on inter-process communication often producing a "fake" computation. Miniapps instead create a meaningful context in which to explore the key performance issue. Within many of the ASC programs, miniapps are developed and owned by application code teams; are limited to $O(1\,\mathrm{K})$ source lines of code (SLOC) and are intended to be modified with the only constraint being the continued relevance to parent applications.

2.1. Mantevo

The Mantevo project [4] provides a set of proxies, or "miniapps", which enable rapid exploration of key performance issues that impact a board set of scientific applications of interest to the ASC and broader HPC community. Mantevo miniapps are tools with uses throughout the co-design space [5]. They are intended to be fluid, and a mechanism to explore issues relating to hardware performance, programmability, porting, etc. As part of the ongoing work in developing miniapps under Mantevo, a comprehensive initial validation exercise [6] has recently been conducted to ensure the first full release of codes is able to provide strong behavioral correlation to parent physics and engineering application currently in use.

The Mantevo Project started in 2006 as an effort to develop tools and environments for studying computational science and engineering application performance. Very early on in the project, miniapps emerged as important tools. They provide the right balance of complexity – capturing the nuances of performance coupling between distinct computational phases – and ease of use and refactoring to be accessible and meaningful co-design tools.

The initial miniapp HPCCG was designed to study the performance characteristics of preconditioned iterative methods used in Trilinos [7]. System developers were looking for a small representative code to answer questions about the direction of some coding implementations targeting emerging and expected future architectures, including multi-core, many-core, and GPU-accelerated high performance computers. HPCCG was frequently used for compiler studies, processor comparison and more. Furthermore, the depth of conversation between algorithms and systems developers grew with a small, concrete target as the means for exploration. Based on this experience we started identifying and deploying miniapps across other application areas of interest.

Each Mantevo miniapp is designed to focus attention on one or a few key performance characteristics of an application or class of applications, enabling agile exploration of a variety of issues that impact performance, ranging from low-level hardware capabilities to the application.

The current set of miniapps in the Mantevo project are listed in Table 1. All of the miniapps in Table 1 are available via the GNU

Please cite this article in press as: S.S. Dosanjh, et al., Exascale design space exploration and co-design, Future Generation Computer Systems (2013), http://dx.doi.org/10.1016/j.future.2013.04.018