doi:10.1093/comjnl/bxs007

Capsule Reviews

FAIROUZ KAMAREDDINE

The Capsule Reviews are intended to provide a short succinct review of each paper in the issue in order to bring it to a wider readership. The Capsule Reviews were compiled by Fairouz Kamareddine. Professor Kamareddine is an Associate Editor of *The Computer Journal* and is based in the Department of Mathematical and Computer Sciences at Heriot-Watt University, Edinburgh, UK.

On the Acceleration of Wavefront Applications using Distributed Many-Core Architectures. S.J. PENNYCOOK, S.D. HAMMOND, G.R. MUDALIGE, S.A. WRIGHT AND S.A. JARVIS

According to the authors, the use of the so-called computational 'accelerators' has gained significant interest throughout the high-performance computing (HPC) community; however, despite the impressive theoretical peak performance of accelerator designs, several hardware/software development challenges must be met before high levels of sustained performance can be achieved by HPC codes on these architectures. The authors add that it is therefore important to develop an understanding of which classes of application, which methods of code porting and which code optimizations/designs are likely to lead to notable performance gains when accelerators are employed. For this reason, the paper investigates the performance of pipelined wavefront applications on devices employing NVIDIA's Compute Unified Device Architecture (CUDA) and uses the so-called LU benchmark as an example. After an introduction to the related work, the LU benchmark is introduced and followed by NVIDIA's CUDA architecture/programming model which will be used in the proposed graphics processing unit (GPU) implementation since the authors claim that it is presently the most mature and stable model available for the development of GPU computing applications. The first stage in the proposed GPU implementation was to convert the entire application to the programming language C and provide a comparison of the performance of the original FORTRAN 77 code with the authors' C port. Then the authors present their optimization techniques applied to both central processing unit (CPU) and GPU, where they discuss memory access/size optimization and GPU-specific optimization. The first set of experiments investigate the performance of a single workstation executing the LU benchmark in both single and double precision and then, in order to illustrate whether these shown benefits transfer to MPI-based clusters of GPUs, the authors compare the performance of their GPU solution running at scale to the performance of two production-grade HPC clusters built on alternative CPU architectures. To do so, the authors use an analytical performance model and to verify their findings, they employ a performance model based on discrete event simulation. It is stated that the high levels of accuracy and correlation between the analytical and simulation-based models provide a significant degree of confidence in their predictive accuracy. Then, these models are used to further assess the behaviour of LU at increased scale and problem size. The authors investigate how the time to solution varies with the number of processors for a fixed problem size per processor (weak scaling) and a fixed total problem size (strong scaling). The results show that the GPU implementation of LU does not scale well beyond a low number of nodes and the authors identify two potential causes of the GPU implementation's limited scalability: domain decomposition and the existing k-blocking policy. The authors analyse these findings.

Performance Characteristics of Hybrid MPI/OpenMP Implementations of NAS Parallel Benchmarks SP and BT on Large-Scale Multi-core Clusters. XINGFU WU AND VALERIE TAYLOR

The authors state that the hybrid MPI/OpenMP programming paradigm is the emerging trend for parallel programming on multi-core clusters and that the NAS Parallel Benchmarks (NPB) are well-known applications with fixed algorithms for evaluating parallel systems and tools. Earlier work developed two hybrid Block Tridiagonal (BT) benchmarks, and compared them with the MPI BT and OpenMP BT benchmarks, and a unified MPI approach was shown to be better for most of the NPB benchmarks. This paper focuses on Scalar Pentadiagonal (SP) and BT benchmarks and implements hybrid MPI/OpenMP implementations of SP and BT benchmarks of MPI NPB 3.3, and compare their performance on three large-scale multi-core clusters. After an introduction to SP and BT, the two large application benchmarks of NPB 3.3, their NPB-MZ 3.3 versions and their differences, the authors give hybrid MPI/OpenMP implementations of the SP and BT and compares them to MPI SP and BT, and to NPB-MZ SP and BT. Three large-scale multi-core clusters are introduced to execute the hybrid SP and BT and to compare their performance with the performance of their MPI counterparts.

Performance Analysis and Optimization of the OP2 Framework on Many-Core Architectures. M.B. GILES, G.R. MUDALIGE, Z. SHARIF, G. MARKALL AND P.H.J. KELLY

The authors argue that exclusively targeting a parallel programming model or a parallel architecture using extensions to traditional sequential languages to write scientific parallel programmes is unsustainable. They argue furthermore that a level of abstraction must be achieved so that computational scientists can increase productivity without having to learn the intricate details of new architectures. OPlus (Oxford Parallel Library for Unstructured Solvers) provides an abstraction framework for performing unstructured mesh-based computations across a distributed-memory cluster of processors. OP2 is the second iteration of OPlus which exploits parallelism on heterogeneous many-core architectures. This paper presents a performance evaluation of the current OP2 library where a performance analysis of the Airfoil unstructured mesh application written using OP2 on a number of multi-core CPU systems is given and the performance issues that distinguish the use of CPU and GPU architectures to execute the Airfoil application are analysed. After an introduction to the related work and the backgrounds for the unstructured mesh applications supported by OP2, the OP2 strategy for building executables for different back-end hardware is presented. This includes covering the data dependency issue encountered when incrementing indirectly referenced arrays and the data layout in memory when there are multiple components for each set element. The first set of experiments is directed at comparing the performance of Airfoil using OpenMP on a single node comprising multi-core, multi-threaded CPUs. The results for both single- and double-precision performance on the CPUs are reported. This is followed by a detailed study of performance analysis and optimization where performance degradation is quantified and two optimizations are implemented to improve the performance of the execution of direct and indirect loops.

SST: A Scalable Parallel Framework for Architecture-Level Performance, Power, Area and Thermal Simulation. MINGYU HSIEH, ARUN RODRIGUES, KEVIN THOMPSON, WILLIAM SONG AND ROLF RIESEN

The authors claim that traditional architectural simulators for HPC narrowly focus on the performance and power dissipation of part of the system making it difficult to manage the energy of the whole system. This paper introduces the technology interface in the structural simulation toolkit (SST), the core of integrated energy, power and temperature simulation. SST provides a parallel framework for simulating large-scale HPC systems to understand both performance and energy consumption. After a brief introduction to related work, STT is presented together with its component-based discrete event model of computation that carries out the simulation, its technology and introspection interfaces. The technology interface is the core of power and thermal simulation, whereas the introspection interface is a unified way to report and record simulation data for analysis and display. The four technology models (HotSpot, McPAT, IntSim and ORION) that are currently supported by STT are described and a validation of the STT is studied on three levels: the component level, the component–technology interface level and the component–component level. Various interconnect options of many-core processors are evaluated to analyse the performance of the proposed framework especially vis-a-vis to leakage feedback and within-chip temperature variation.

Benchmarking Energy Efficiency, Power Costs and Carbon Emissions on Heterogeneous Systems. SIMON MCINTOSH-SMITH, TERRY WILSON, AMAURYS AVILA IBARRA, JONATHAN CRISP AND RICHARD B. SESSIONS

According to the authors, heterogeneous, massively parallel processors are likely to become ubiquitous and hence hybrid multi-core CPUs-many-core GPUs processors will soon become mainstream. This means for the authors that the HPC community will need to embrace this next major architectural paradigm in order to maximize the performance benefits of future systems. Hence, the goal of the paper is to present a benchmarking methodology for measuring a number of performance metrics for heterogeneous systems and to investigate software energy efficiency and resulting carbon emissions in an HPC context. After a review of related work, the authors introduce their molecular docking engine BUDE (which has been under development since 2001) and a many-core parallelization of BUDE using the emerging industry standard, OpenCL, as the assumed parallel programming language. OpenCL is used to describe the target many-core hardware and the software design and to facilitate running the same code on multiple target platforms. Then the authors introduce their benchmarking methodology where the performance and energy efficiency of a variety of test systems is examined and a specific BUDE test problem involving the calculation of millions of pose energies is defined. The proposed benchmarking methodology shows that it is possible to compare energy efficiency between different systems by carefully selecting the test equipment and methodology.

Memory Trace Compression and Replay for SPMD Systems Using Extended PRSDs. SANDEEP BUDANUR, FRANK MUELLER AND TODD GAMBLIN

Experience with current petascale systems has shown that production codes tend to face scalability problems each time the core count is increased by a factor of 10. The authors argue that the most common causes of scalability problems are communication, I/O and memory inefficiencies and propose to focus on memory inefficiencies (particularly in multi-threaded shared-memory models with large multi-cores that require efficient use of the memory hierarchy across threads). This requires scalable compression of communication and I/O traces; however, in the current literature, these traces do not reflect memory access patterns across threads. To overcome these difficulties, this paper

develops a memory trace generator, a generic trace compression template library and a signature tree library in C++, and builds a memory trace compression tool, ScalaMemTrace, that generates near-constant size memory traces that preserve the temporal order of accesses irrespective of problem/concurrency size. The authors introduce their memory trace compression scheme EPRSDs which are extended versions of PRSDs which preserve the order of memory references and generalize memory access patterns across threads and processes along with loop dependencies. Intra-thread compression, inter-thread compression and inter-process compression are introduced, and this is followed by the components of ScalaMemTrace and the trace compression implementation. The memory trace generation consists of a number of steps that include binary instrumentation and intermediate data structures and techniques (such as stackwalking) to ensure scalability. The C++ EPRSD template library is introduced and experiments are carried out with ScalaMemTrace to assess the scalability of the EPRSD compression scheme for different concurrency levels and problem sizes. The correctness of the compression scheme is discussed and so is the run-time performance of the various stages.

Leveraging Service Discovery in MANETs with Mobile Directories. SERGIO GONZALEZ-VALENZUELA SON T. VUONG AND VICTOR C.M. LEUNG

There is a large interest in Service Discovery Protocols (SDPs) for MANET use and a MANET-wide flooding of service discovery packet traffic aimed at preserving bandwidth and battery power. However, the task overlap between service discovery and packet routing protocols leads to the inefficient use of resources. Furthermore, commercial software application vendors and distributors tend to favour proprietary SDPs. This paper advances the Service Directory Placement Protocol (SDPP), which is a multi-directory deployment scheme that enables the efficient replication of service advertisements closer to their potential consumers. The SDPP aims to leverage the efficiency of any existing, directory-based SDP to advertise service-related information, and is agnostic to any underlying routing protocol. After an overview of existing work, a brief overview of SLPv2 is given and will be used to better understand the role of SDPP in a service discovery subsystem. Thereafter, the authors explain how SDPP enables directory information exchange with SLPv2 and promotes the duplication of one or more fixed directory contents as well as the processing of queries. Then the performance SDPP is improved by mathematically defining the directory replication task as a semi-Markov Decision Process (SMDP) and a Reinforcement Learning technique is used to solve (namely through a Q-learning algorithm) the SMDP. The SDPP was implemented and tested as a computer simulation and the performance of the proposed system is examined for networks of 50, 100 and 200 mobile hosts and a predetermined number of stationary service providers. Finally, the evaluation results are reported and a discussion is given of the practicality of the SDPP in a real MANET.