J Supercomput (2017) 73:3327–3332
DOI 10.1007/s11227-017-2107-6

Embedded multi-core computing and applications

Che-Lun Hung[1] · Frédéric Magoulès[2] · Meikang Qiu[3] · Robert C. Hsu[4] · Chun-Yuan Lin[5]

In the past decades, computing power has benefited from multi-core hardware architectures and state-of-the-art parallel programming models. For the multi-core architectures, the trend is to integrate many powerful computing cores into a single computing component. Meanwhile, many processors can be integrated into a single device as GPU. To leverage the power of these hardwares, many novel programming frameworks and models, such as Hadoop, Spark, OpenMP, OpenCL and CUDA, have been released to facilitate software development by launching the cores or threads simultaneously. In addition to both of hardware and software architectures, another important issue is a suitable parallel algorithm which can efficiently perform on these architectures. Nowadays, many successful applications through the parallel algorithms

✉ Che-Lun Hung
clhung@pu.edu.tw, clhung@gm.pu.edu.tw

Frédéric Magoulès
frederic.magoules@centralesupelec.fr

Meikang Qiu
mqiu@pace.edu

Robert C. Hsu
chh@chu.edu.tw

Chun-Yuan Lin
cyulin@mail.cgu.edu.tw

1 Providence University, Taichung, Taiwan

2 CentraleSupelec, Châtenay-Malabry, France

3 Pace University, New York, USA

4 Chung Hua University, Hsinchu, Taiwan

5 Chang Gung University, Taoyuan, Taiwan

have unveiled lot of impressive results. This special issue contains thirteen papers, and each paper covers a particular application and state-of-the-art technology.

As today's scientific and technical documents are published at a great scale, a tool that can rapidly transform image information into text is invaluable for further processing. Lin et al. integrate the GPU technology to enhance the automatic recognition tool—EqnEye—for mathematical equations. The whole process is carried out in seven steps, namely image capture, image preprocessing, yaw correction, symbol segmentation, symbol recognition, structure analysis and expression exhibition. The parallel acceleration by GPU happens in computing the threshold for binarization. Since CUDA 6, data can be shared between CPU and GPU, and they implement the Ostu method much simpler than before. After that, a yaw correction is performed in order to increase the recognition accuracy. They use an equal symbol, which appears in any equation as indication for the rotation. In symbol segmentation, glyphs are recombined to form correct symbols by considering the geometric feature and position. Then, an open-source OCR engine Tesseract is used to identify the segmented symbols. Finally, an ME parse tree is generated by bounding boxes' relative position and expression is obtained. Results show that for high-resolution images, this method achieves 1.6× speedup in average, and the yaw correction effectively increases the recognition accuracy. In addition, experiments are conducted in three conditions, i.e., normal, skew and shadowy; their method achieves 90, 80 and 50% correctness for each condition, respectively. Failure reason is also studied. They mainly lie in image preprocessing and yaw correction.

Hu et al. propose an energy-efficient design of a microkernel-based on-chip OS for an NOC-based manycore system. As number of cores on a chip grew fast, network on chip (NOC) has been proposed as a promising model to solve the connection problem of the cores. The operating system (OS) is partitioned into the microkernel and the other OS modules; they are distributed on the network to provide services to the user programs. There are two levels of networks in this kind of OS: the on-chip network and the software network, and software modules can be mapped to the whole on-chip network. The key issue of this designing is how to fully benefit from the on-chip network and the cores. The authors approach the problem by modeling program as some cooperating threads. The microkernel and local OS threads provide the appropriate services as much as possible but not all possible services; rarely used ones are routed to remote nodes. However, there are two principles in mapping. The first is that the threads from the same programs are mapped together. The second is that the most frequently used OS threads are mapped as local threads or as close to the cores as possible. A modified communication energy mapping (CE) based on the proposed OS is implemented within Noxim and then compared to other two mapping methods in experiments, first-fit mapping (FF) and nearest neighbor mapping (NN). Results of some criteria are reported in the paper, such as messages traffic, latency and energy efficiency. In all of these tests, the CE mapping exhibits better performance than others.

Paulino and Delgado firstly propose to shift memory hierarchy-related concerns to the run-time system, which takes on the responsibility of distributing the computation's data across the target memory hierarchy in data parallel computations. So the main idea is to explore the domain decomposition at node level to enhance the locality properties

of data parallel computations. To that end of the analysis, a cache-conscious strategy that takes into account the characteristics of some (or all) levels of the target memory hierarchy can then be devised. The author implements the proposed algorithms in the context of the Elina framework for Java parallel computing. The results of comparison between the cache-conscious approach and the classical, cache-neglectful strategy show significant performance gains with the proposed method.

Lin et al. propose two strategies, the inter- and intra-task parallelization (abbreviated as ETP and RTP, respectively) for compressing 3D sparse arrays, especially on Intel Xeon and Xeon Phi processors. CRS and CCS Schemes are applied; parallelization is achieved by employing the technique of OpenMP. In inter-task parallelization scheme, multiple sparse arrays are processed concurrently, while in intra-task parallelization scheme, only one sparse array is compressed, but the single array is decomposed into many local ones that can be processed simultaneously. Memory limitation is also taken into consideration. By the results of experiments, both the ETP strategy and the RTP strategy accelerate the compression process for different practical applications and achieve considerable speedup ratios (up to 18.2× and 4.5×, respectively) according to the sizes of sparse arrays and available resources based on Intel Xeon E5-2670 v2 and Intel Xeon Phi, but generally the ETP scheme outperforms the RTP scheme; however, in some cases the RTP may be more suitable.

Cheik Ahamed and Magoules propose in this work an original implementation of Jacobi iterative method with fast linear algebra operations, such as dot product, scaling, and matrix-vector multiplication, for large sparse linear systems on graphic processing units. The CSR storage format of a sparse matrix is used to gain efficiency both in terms of arithmetic operations and in terms of memory usage. Experiments are carried out on large linear systems generated from finite element discretization of three differential equations—3D Laplace equation, 3D gravitational potential equation and 3D Heat equation. Results show that the proposed implementation of the parallel algorithm is robust and efficient as a speedup of up to 23× is achieved in double-precision arithmetics. Furthermore, the speedup increases with the number of iterations.

Song et al. focus on combining the contemporary computing technologies—general-purpose computing on the graphics processing unit (GPGPU)—with Incremental K-SVD to achieve the fast sparse representation method toward large image datasets. The GPU-aided IK-SVD employs the following optimization techniques to maximize the amplification of GPU computing power. First, a batch-OMP algorithm based on a recursive Cholesky decomposition algorithm which alleviate data dependency is used to target the sparse coding, which is a hotspot of the computation. Second, an adaptive sparse matrix format—blocked row–column (BRC)—is chosen for matrix computations. And third, a hybrid data-task parallel scheme on one NVIDIA's GPU based on Kepler architecture is proposed to dramatically increase GPU utilization and significantly reduce CPU idle times. The experimental results show that without losing any quality of dictionary learning as IK-SVD, the GPU-aided IK-SVD achieves an impressive speedup of 20 times overall. In the phase of sparse coding, the batch-OMP algorithm can achieve up to 30 times speedups than the sparse coding part of IK-SVD, the optimized sparse matrix operations improve the whole procedure of IK-SVD up to 15 times and the proposed hybrid parallel scheme can further accelerate the procedure of sparsely representing one large image dataset up to 24 times.

Ferreira Junior et al. aim at increasing the automatic diagnosis accuracy of lung cancer based on similar pulmonary nodule retrieval by integrating 3D attributes of margin sharpness and texture features. They use three normalized feature vectors to characterize the nodules, in which the first vector consists of 36 texture attributes (TA) extracted from a 3D texture analysis, the second vector has 12 margin sharpness attributes (MSA) extracted from the 3D margin sharpness analysis and the third one is a concatenation of the previous two, thus composing 48 integrated attributes (IA). Experimental results show that the integrated method performs the best in diagnosis accuracy against the methods separately using TA and MSA. In addition, the authors also employ the GPU to accelerate the run time of nodule comparison process and obtain a speedup of 23.7 times.

Just an Asynchronous Communication Tool (JACK) is the first MPI-based C++ library for both classical and asynchronous iteration. Magoules and Gbikpi-Benissan present in this article different computation models to illustrate the advantage of asynchronous iterative methods, which leads to three communication schemes. As the management of resources is an annoying problem while programming the asynchronous iterative algorithms, the main features of JACK include a unique high-level interface for the communication mode transition and use a buffer manager and a continuous message reception manager to ease the conception of code architecture. They test the new library with both a classical parallel scheme and a sub-structuring approach of the Jacobi iterative method. The global convergence detection algorithm is implemented with the method proposed in 2005 by Bahi, which provides a robust and fast termination detector. Experimental result shows that the performance of the asynchronous iterative algorithm based on the JACK increases both the efficiency and the scalability. In the case of very large matrices, it may even beat synchronous algorithm right from relatively small number of processors.

In order to guarantee that applications are efficiently carried out without wasting the cloud resources, recognition of different types of applications is crucial, as it can be used as the basis of reasonably selecting strategies to process different types of applications. Peng et al. devised in this paper a comprehensive method that can generally and efficiently predict and classify the types of cloud applications. According to the resource consumption state, the author classifies applications in cloud into four types: CPU intensive, I/O intensive, memory intensive and network intensive. Nine out of more than 30 parameters that can be monitored are selected to distinguish among these types of cloud applications by a thorough analysis of them. Extensive experiments on different types of applications (500 times experiments for each) have proved the correctness and effectiveness of the proposed model.

Chen et al.'s paper proposes a shared write buffer (SWB) among the speculative multi-threading (SpMT) cores, so that it is possible to confine the speculative read and write in the SWB. Thus, first there will not be coherence issue with those speculative data since the SWB captures all the speculative written data, and hence, the coherence protocol does not need redesign; second by buffering the written data and share them among the cores, the SWB helps to reduce cache coherence and consequently improve the overall cache performance by noticeably reducing the average access delay. Experiments show that the SWB can capture a big portion of inter-core data sharing, reduce cache coherence, and drastically improve data access performance of SpMT threads;

for most cases, the average access delay is reduced by at least 7%, and for some cases this could reach more than 75%.

Cai et al. propose an SLA-aware (Service Level Agreements) energy-efficient scheduling scheme which allocates appropriate amount of resources to MapReduce applications with YARN architecture since YARN is one of the key features in the second-generation Hadoop, which provides resource management and scheduling for large-scale MapReduce environments. The scheduling scheme enables automatic resource inference and allocation based on the job profiling information, especially the data locality is taken into consideration and thus leads to the reduction in MapReduce network traffic. Furthermore, the slack time between the actual execution time of completed tasks and expected completion time of the application is utilized to improve the energy efficiency of the system. In fact, an online userspace governor-based dynamic voltage and frequency scaling (DVFS) scheme is designed to dynamically change the CPU frequency so that tasks can be run at low CPU frequency during slack times without violation of the application's SLA; actually, this scheme may also avoid accepting jobs that will lead to deadline misses and improve the cluster utilization. Experimental evaluation shows that the proposed scheme achieves better SLA conformance with low resource cost and energy consumption than the existing MapReduce scheduling policies.

Real-time video contents with high quality consume the major traffic of the Internet today. Content providers tend to offload the task of content hosting to content delivery networks (CDNs) providers. Chen et al., in this paper, propose a flexible CDN architecture in attempts to fulfill requests for high-quality videos across the Internet by integrating software-defined networking (SDN) technology. In the design, both server workloads and user-side requirements are considered, and the requests finally end up with efficient and balanced workload distribution. The authors revised load balancing algorithms upon SDN and developed simulations to verify their approaches. The results through simulations show that the proposed algorithms have the potential to meet bandwidth requirement of real-time video streaming and fully utilize links in CDNs and thus would achieve much higher user satisfaction in bandwidth-idle environments.

Lu et al. design and implement a co-simulation system, named as the virtual media network (ViMediaNet) aiming at the system simulations and performance evaluations for inter-active media-based TPS services, through which the TPS designer can efficiently establish and readily deploy the telepresence service (TPS) system over a heterogeneous network virtualized by the emulator for comprehensive scheme testing, such as investigation of transmission delay and packet jitter. The users may even observe control results in real time. The network simulator takes EMANE as the backbone and introduces different add-ons over EMANE to build critical function blocks. The main blocks include a CSMA/CD plugin for wired network simulations in EMANE, a OpenCV-based video processing engine for video data coding and packing, and a time-triggered synchronization scheduler for multi-task scheduling in simulations. The authors take a mobile device-based vision navigation system (VNS) as an example to establish a testing scenario to validate the effectiveness of ViMediaNet in emulating the real-setting TPS services. A series of experiments are conducted to evaluate three critical kinds of performance in ViMediaNet, including performance

of VNS messages, performance of a Multi-user and Multi-robot VNS case, and system costs of ViMediaNet. The experimental results show that ViMediaNet is an efficient tool for TPS designers to test their designs efficiently.

All of the papers in this journal special issue represent an important idea that is the need of high-performance computing, easy programming models and cheap devices to deliver the significant results. Similarity, these papers cover a broad spectrum of scientific areas, and they presented that more and more scientific applications will be considered to apply multi-core systems.