Published in IET Computers & Digital Techniques
Received on 28th April 2014
Revised on 4th August 2014
Accepted on 28th August 2014
doi: 10.1049/iet-cdt.2014.0074

Special Issue on Energy Efficient Computing with Adaptive and Heterogeneous Architectures

ISSN 1751-8601

Performance/energy trade-off in scientific computing: the case of ARM big.LITTLE and Intel Sandy Bridge

Edson Luiz Padoin[1,2], *Laércio Lima Pilla*[1,3], *Márcio Castro*[1,3], *Francieli Z. Boito*[1,4], *Philippe Olivier Alexandre Navaux*[1], *Jean-François Méhaut*[4]
[1]*Institute of Informatics, Federal University of Rio Grande do Sul (UFRGS), Porto Alegre, RS, Brazil*
[2]*Department of Exact Sciences and Engineering, Regional University of Northwest of Rio Grande do Sul (UNIJUI), Ijui, RS, Brazil*
[3]*Department of Informatics and Statistics, Federal University of Santa Catarina (UFSC), Florianópolis, SC, Brazil*
[4]*Grenoble Informatics Laboratory (LIG), University of Grenoble, Grenoble, France*
E-mail: elpadoin@inf.ufrgs.br

Abstract: Power consumption is one of the main challenges to achieve Exascale performance. Current research trends aim at overcoming power consumption constraints using low-power processors. Although new processors feature sensors that enable precise power measurements, they provide different interfaces to collect data, making it difficult to correlate performance with energy consumption. To overcome this issue, the authors developed a platform-independent tool that collects power and energy data from homogeneous and heterogeneous systems. Using this tool, they provide a detailed comparison between a low-power processor (ARM big.LITTLE) and a high performance processor (Intel Sandy Bridge-EP) using all applications from the NAS parallel benchmarks and a real-world soil irrigation simulator. The results show that the average power demand of Intel Sandy Bridge-EP is within 12.6× to 152.4× higher than ARM big.LITTLE, whereas its average energy consumption is within 1.6× to 7.1× superior. Overall, ARM big.LITTLE presented a better performance/energy trade-off when it takes <9.2× the execution time of Intel Sandy Bridge-EP to solve the same problem.

1 Introduction

Scientific applications that represent natural phenomena, such as molecular dynamics simulations, seismic wave propagation and weather forecasting usually rely on high-performance computing (HPC) platforms to achieve more accurate results and shorter execution times. Research on the HPC field aims at providing faster supercomputers to run even greater simulations.

Until the last decade, the performance of supercomputers was quantified almost exclusively by their computing performance (Flops). For instance, the Top500 list was established to rank supercomputers regarding processing speed. Nonetheless, the development of computing platforms with exponentially scaling performances over the years also led to an exponential growth in power consumption [1, 2]. In this context, saving power is one of the main concerns of current HPC design. Compliant with this idea, more recent initiatives consider speed and power demand to rank supercomputers. One example is the Green500 list, which considers the ratio between Flops and power consumption of the platform (Flops/W) [3, 4].

A main challenge in the development of future HPC platforms lies on increasing their performance while reducing their power consumption. In this sense, energy consumption constraints have to be taken into account [5, 6]. This is key for attaining Exascale systems matching the increased demands of scientific applications. Exascale supercomputers conceived by just scaling current cutting edge technology would demand over a GW of power. To avoid this scenario, specialists alerted on the official DARPA report that the acceptable power budget for Exascale platforms would be 20 MW [7, 8], which requires an energy efficiency of at least 50 GFlops/W. To put things into perspective, the number one supercomputer on the June 2014 Top500 list, Tianhe-2, performs 33.8 PFlops while consuming 17.8 MW, which results at only 1.9 GFlops/W.

A possible approach to increase the computing performance without incurring in the growth of power consumption relies on using low-power processors commonly used on embedded systems. These processors are developed respecting power consumption constraints to improve the battery life of autonomous devices, such as smartphones and tablets. One example of embedded processor architecture is developed by ARM. The first generation of ARM architectures targeted mostly low power consumption, being unsuitable for HPC because of three main reasons: a moderate processing speed, a lack of floating-point units (FPUs) and the absence of hardware support for single instruction, multiple data (SIMD) instructions. However, recent ARM Cortex-A processors feature more powerful cores, FPUs and SIMD instructions while increasing power consumption only slightly when compared with their predecessors. Owing to these characteristics, some research efforts conducted towards Exascale bet on ARM processors. For instance, the Mont-Blanc Project is one of the first to introduce the idea of an ARM-based supercomputer. The project bets on highly heterogeneous multiprocessor systems-on-chip (MPSoCs) combining ARM and graphics processing units (GPUs) to achieve high processing speed at a low power consumption.

To better understand the potential of using low-power processors for HPC systems, we present in this paper a detailed study comparing the use of an ARM big.LITTLE embedded processor and an Intel Sandy Bridge-EP multicore for scientific computing. We benefit from energy sensors available in new processors and boards to measure their power and energy consumption at a fine granularity for several parallel applications, correlating these metrics to the performance attained by these processors. Our main contributions with this research are the following:

(1) We develop an energy monitoring tool (EMonDaemon) to measure the instantaneous power demand and energy consumption on homogeneous and heterogeneous systems;
(2) We study the performance/energy trade-off on ARM big. LITTLE and Intel Sandy Bridge-EP processors, comparing their performance, power demand and energy consumption;
(3) We examine the processors' growth in power demand for different numbers of cores employed; and
(4) We evaluate a wide range of parallel scientific applications on these platforms. More precisely, we use all applications from the NAS parallel benchmark (NPB) suite and a real-world soil irrigation simulator (SIS).

The remaining sections of this paper are organised as follows. Section 2 presents our tool (EMonDaemon) to collect power and energy consumption on homogeneous and heterogeneous systems. Section 3 gives an overview of the processors used in this paper and discusses our measurement methodology and applications. Experimental results are discussed in Section 4. Some relevant related works on the evaluation of energy consumption and performance of low-power and general-purpose processors are presented in Section 5. Finally, our concluding remarks and future work perspectives are presented in Section 6.

2 Energy monitoring tool: EMonDaemon

Correlating performance with power and energy consumption during run-time can be difficult as current platforms have different interfaces to collect information from their components. Moreover, some of the existing tools provide data to be analysed only after execution. To provide a single solution to measure instantaneous power and energy consumption during the execution of applications on homogeneous and heterogeneous systems, we developed a new tool named EMonDaemon. It works in two consecutive phases when executed along with applications, as described below:

• *Discovery phase*: At the very beginning, the tool obtains mostly static information from the platform, such as processor manufacturer, processor model, available clock frequencies and current clock frequency. This information will guide the decisions on how power and energy data will be collected from the underlying platform.
• *Monitoring phase*: During the execution of the application, EMonDaemon monitors the system to collect power or energy information with a periodicity defined by the user. It maintains some statistical data such as the minimum, maximum and average power, and the energy consumption. In addition to that the tool can also trace instantaneous power and clock frequency of the processors during execution. In this case, data are stored in output files that can be visualised with standard graphing utilities such as Gnuplot.

Our tool collects power and energy through the platform's specific power measurement sensors. Recent Intel and AMD processors provide power and energy information of the whole CPU package (cores and cache memory) from power measurement sensors available in the processor's chip. This information is stored in model-specific registers (MSRs) in the processor, which are mapped to pseudo files by the Linux MSR kernel module. EMonDaemon reads these files using the file input/output application programming interface (API) from the MSR-tools package. Documentation regarding which MSRs a certain processor supports is usually provided by the processor manufacturer. EMonDaemon selects the correct MSRs based on the platform information collected in the discovery phase. The support for power and energy measurements is currently available for Sandy Bridge-EP or newer microarchitectures from Intel and Family 15 h or newer microarchitectures from AMD.

On ARM processors, our tool relies on special on-board power measurement circuits such as current/voltage sensors, since these processors do not feature any on-chip MSRs. This is the case of the Odroid-XU + E ARM big.LITTLE architecture used in this paper, which has four on-board sensors to measure the instantaneous current, voltage and power for the Cortex-A15, the Cortex-A7, the GPU and the dynamic RAM (DRAM) individually. Since these sensors are exposed to the user as MSRs, they can be accessed in the same way as Intel and AMD on-chip MSRs.

3 Experimental methodology

This section describes the methodology used in our performance/energy trade-off study. We first present the execution environment, followed by the measurement methodology and the applications used in our experiments.

3.1 Execution environment

As discussed in the previous section, this work benefits from sensors available in recent architectures in order to measure power and energy consumption at a fine granularity. Two processors were used in this paper: an ARM big.LITTLE and an Intel Sandy Bridge-EP. Their characteristics are presented below and summarised in Table 1.

• *ARM big.LITTLE*: The first platform used in our experiments is an Odroid-XU + E MPSoC. This model features a Samsung Exynos5 5410 Octa based on the recent big.LITTLE architecture. It includes four ARM Cortex-A15 cores at 1.6 GHz and four ARM Cortex-A7 cores at 1.2 GHz organised in two homogeneous clusters. The system contains 2 GB of low-power DDR3 random access memory

Table 1 Detailed configuration of our execution environments

Platform manufacturer	ARM big.LITTLE Samsung		Intel Sandy Bridge-EP Intel
processor model	Cortex-A7	Cortex-A15	E5-4640
clock frequency, GHz	1.2	1.6	2.2
number of cores	4	4	8
memory, GB	2		32
cache L1, kB	64	64	64
cache L2	512 kB	2 MB	256 kB
cache L3	–	–	20 MB
processor technology, nm	28	28	32
instruction set architecture/ extensions	ARMv7l	ARMv7l	AVX
advanced SIMD	NEON	NEON	—
FPU	VFPv3	VFPv3	VFPv3
out-of-order execution	no	yes	yes

and an integrated PowerVR SGX544MP3 GPU. Each core has 32 kB instruction and 32 kB data L1 private caches. The Cortex-A7 cores have 512 kB of L2 cache, whereas the Cortex-A15 cores have 2 MB of L2 cache with error-correcting code.

• *Intel Sandy Bridge-EP*: The second platform features an Intel Xeon E5-4640 Sandy Bridge-EP processor [9]. This processor is a ×86-64 processor with eight cores running at 2.40 GHz. Each core has private 32 kB instruction and 32 kB data L1 caches and a 256 kB L2 cache. The eight cores share a 20 MB L3 cache and 32 GB of DDR3 memory.

The operating systems used in our tests were the ones provided by the manufacturers: GNU/Linux distribution with kernel version 3.4.67 on ARM big.LITTLE and UV2000 GNU/Linux distribution with kernel version 3.0.101–0.29 on Intel Sandy Bridge-EP. The applications were compiled with GCC version 4.8.

3.2 Measurement methodology

We employ different compilation flags in order to compare performance, energy consumption and power demand. Besides the classical optimisation flags '-O', we used: '-march' to enable ARM/Intel architecture specific optimisations; '-mtune' to specify the name of the target ARM processor; '-mfpu' to specify what floating-point hardware (or hardware emulation) is available on the target; and '-mfloat-abi' to specify which floating-point application binary interface to use. We also employ the floating-point operations optimisation flag '-ffast-math'.

Although ARM big.LITTLE contains two heterogeneous processors (Cortex-A15 and Cortex-A7), it is not possible to use both at the same time. The selection between them can be done transparently depending on the application's load, but its simple mechanism results on our HPC applications always running on the Cortex-A15 cores. For this reason, we disabled the cluster switching option and tested the two processors separately.

Overall, we used '-O3 -march = armv7-a -mfpu = neon-vfpv4 -mfloat-abi = hard -ffast-math' on ARM big.LITTLE, including '-mtune = cortex-a7' when using the Cortex-A7 cores and '-mtune = cortex-a15' for the Cortex-A15ones. On Intel Sandy Bridge-EP, we used '-O3 -march = corei7-avx -ffast-math'. We highlight the use of the '-march' flag since it enables the advanced vector extensions (AVXs) support on Intel Sandy Bridge-EP and the new 'armv7-a' instruction set on ARM big.LITTLE.

All results presented in this paper are the average of a minimum of ten runs. The relative error was <5% using a 95% statistical confidence with Student's *t*-distribution.

3.3 Applications

To analyse the performance, power demand and energy consumption of ARM big.LITTLE and Intel Sandy Bridge-EP processors and define their performance/energy trade-off, we chose the widely-used NPBs [10] and a real application from the agroforestry domain for our experiments.

The NPB are a set of benchmarks derived from computational fluid dynamics well recognised for evaluating current and emerging multicore architectures. We used nine different applications, namely, block tri-diagonal (BT), conjugate gradient (CG), embarrassingly parallel (EP), Fourier transform (FT), integer sort (IS), lower upper (LU), multi-grid (MG), scalar penta-diagonal (SP) and unstructured adaptive (UA). For our evaluation, we used Class B workloads.

The case-study application, here called 'SIS', searches for an optimal irrigation system model [11, 12]. The understanding of the behaviour of water absorption is of great interest for agroforestry and agriculture because it can prevent water from being wasted while irrigating the soil. In previous works, this application was parallelised on a cluster using message passing interface (MPI). It was also used as a case-study for investigating the energy efficiency and power supply constraints of heterogeneous CPU + GPU architectures [13].

The application uses a model that represents water infiltrating a cylindrical tube that models the soil. The model considers the irrigation time on the cylinder centre with a fixed continuous flow of water, depicted in Fig. 1. Water infiltrates through the tube on directions *r* and *z*, varying the humidity of each root cell over time. The solution is based on an iterative method with fixed time steps. The computation is divided in two parts: the simulation itself and the fitting of the model with real experiments (at each iteration). The iterations' complexity is related to the matrices' orders, that is, the number of cells on the cylinder.

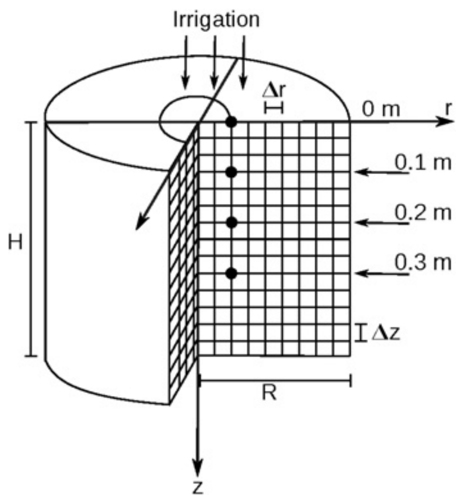

Fig. 1 *Cylindrical tube which models the root while being infiltrated with water and its sampled surface*

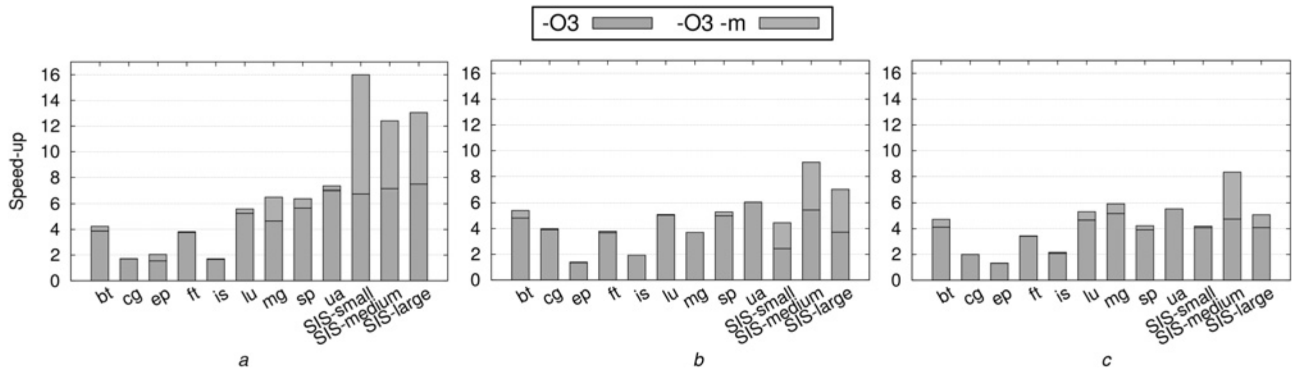

Fig. 2 *Performance improvements over the unoptimised sequential version with compilation flags on different processors*
a Intel Sandy Bridge-EP
b ARM big.LITTLE Cortex-A15
c ARM big.LITTLE Cortex-A7

To run several experiments in feasible time, our tests simulate the water absorption for 18 min with a cylinder with 100 cm of radius and 50 cm of height, with 100 iterations per second. We obtained results for this configuration with three matrices' sizes: 128 × 128 (small), 256 × 256 (medium) and 512 × 512 (large). Although these parameters define simulations of representative phenomena, a specialist on the agroforestry domain would still need larger scale simulations.

To increase this application's performance on multicore systems, we implemented a parallel version using OpenMP, a popular programming model for shared memory systems. The parallel version of the application distributes slices of root cells among the available cores.

We describe our tests and experimental results using NPB and SIS as case studies in the next section.

4 Experimental evaluation of ARM big.LITTLE and Intel Sandy Bridge-EP

This section describes the results obtained from executing the NPBs and the SIS application on the test platforms previously described. Measurements were obtained using the EMonDaemon tool presented in Section 2.

As discussed in Section 3, in order to better compare the platforms, we evaluate the applications using different configurations:

(1) Sequential version without flags ('w/o flags');
(2) Sequential version using optimisation flags ('-O3');
(3) Sequential version with optimisation and architecture specific flags ('-O3 -m'), including all flags discussed in Section 3.2; and
(4) Parallel version using OpenMP ('-fopenmp').

This section is organised as follows: the performance achieved on each platform is discussed in Section 4.1; power demand is analysed in Section 4.2; energy consumption is studied in Section 4.3; and the performance/energy trade-off of the different platforms is the subject of Section 4.4.

4.1 Performance evaluation

4.1.1 Analysis of compilation flags: The performance gains resulting from the cumulative use of the different optimisation flags are illustrated in Fig. 2 as stacked speedups over the original version without optimisation. The gains for BT, EP, FT, IS and LU were similar on the three processors. In general, the use of '-O3' had the largest impact and its use conducted to an average speedup of 4.71 on Intel Sandy Bridge-EP. The speedups achieved for SIS with the large problem size were 7.47, 3.72 and 4.09 on Intel Sandy Bridge-EP, Cortex-A15 and Cortex-A7, respectively. In other words, the impact of '-O3' was almost two times larger on Intel Sandy Bridge-EP than on ARM big.LITTLE.

The architecture specific optimisation flags ('-m') enable the AVXs support on Intel Sandy Bridge-EP and the new 'armv7-a' instruction set on ARM big.LITTLE. These flags affect the performance of the different applications by reducing their total execution time when compared with using only '-O3' by 19% on all platforms on average. The larger impact in performance seen for SIS comes from the use of SIMD instructions by the compiler.

Optimisation flags have a larger impact on the performance of SIS and benchmarks such as MG, SP and UA on the most complex processor Intel Sandy Bridge-EP than Cortex-A15 and Cortex-A7. For instance, Fig. 3 compares the execution times of the benchmarks with different flags enabled on the different platforms. The performance difference between Intel Sandy Bridge-EP and Cortex-A15 (Cortex-A7) for SIS-small goes from 1.6 (6.6) when using no compilation flags in Fig. 3*a*, to 4.5 (10.9) with '-O3' in Fig. 3*b* and up to 5.8 (25.1) with '-O3 and –m' in Fig. 3*c*. The obtained results pointed that the use of compilation flags improve the performance of the applications by 9.44× , 6.28× and 4.89× on average for Intel Sandy Bridge-EP, Cortex-A15 and Cortex-A7, respectively.

4.1.2 Analysis of the parallel version: The benefits of executing the application in parallel using OpenMP are illustrated in Fig. 4 and compared in Fig. 3*d*. It is important to highlight that Intel Sandy Bridge-EP has double the number of cores than Cortex-A15and Cortex-A7, which enables bigger performance gains with the introduction of parallelism. This is the reason why the scale in Fig. 3*d* is two times larger than the other illustrations in Fig. 3.

When compared with the optimised sequential version using '-O3 and –m', speedups vary from 2.00 to 7.63 on Intel Sandy Bridge-EP, depicted in Fig. 4*a* and from 1.33 (1.74) to 3.83 (3.87) on Cortex-A15 (Cortex-A7). The greatest performance gains over the optimised were

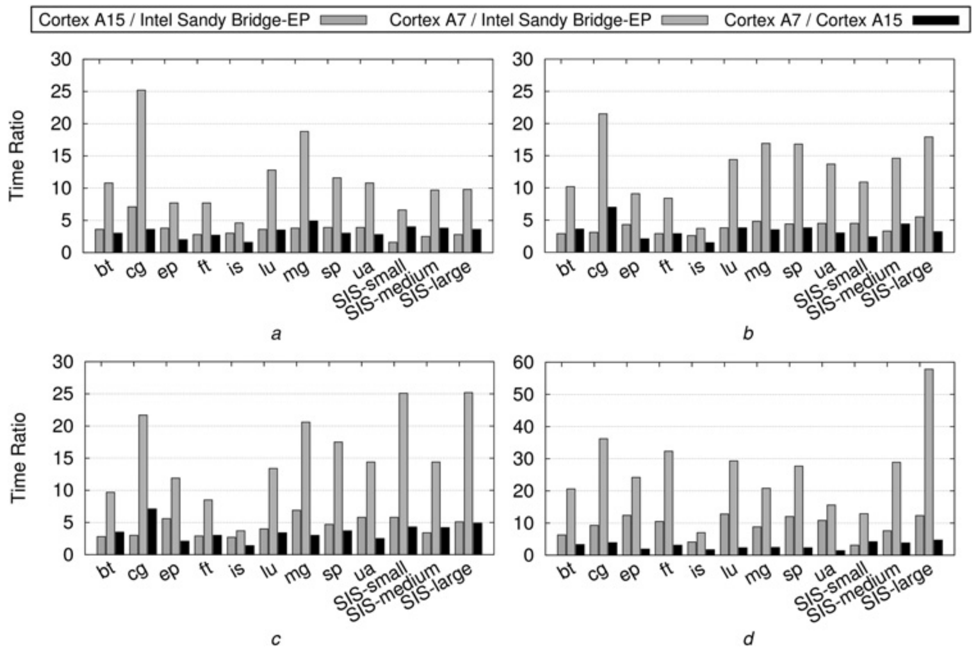

Fig. 3 *Performance comparison between ARM big.LITTLE (Cortex-A15 and Cortex-A7) and Intel Sandy Bridge-EP*

a w/o flags
b -O3
c -O3 –m
d -O3 -m -openmp

achieved with SIS-medium on Intel Sandy Bridge-EP, which benefits from the use of the compilation flags. Meanwhile, the largest speedups over the optimised version between the NPB were achieved for EP, which is the benchmark that most benefits from parallelisation.

The small workload of SIS-small limits the performance gains on Intel Sandy Bridge-EP to a speedup of 2.00. Nevertheless, it did not represent the same kind of limitation on ARM big.LITTLE, where speedups of 3.83 and 3.87 were achieved for Cortex-A15 and Cortex-A7, respectively.

A performance degradation with the increase in problem size for SIS can be noted on the Intel Sandy Bridge-EP processor, as seen in Fig. 4*a* and on the ARM big.LITTLE processor, in Figs. 4*b* and *c*. Several factors are responsible for this. SIS has a complexity of O(n^4), where the computation of humidity in each cell depends of the neighbouring cells. This data dependency is a strong constraint of the parallel solution [13]. Another factor is that several matrices are concurrently accessed by the threads. This results in multiple copies of data on caches and a large number of invalidations. These limitations together with an intense memory usage were noted on different scales on the different tested architectures. We also see a memory bandwidth limitation on ARM. Although the bandwidth of the DDR3 on Intel Sandy Bridge-EP is 32 GB/s, the bandwidth of the LP-DDR3 on our ARM big.LITTLE architecture is only 12.8 GB/s. This memory bandwidth bottleneck became evident on both Cortex-A15and Cortex-A7clusters. Although on Intel Sandy Bridge-EP, the speedup was decreased in 10%, from 7.1 (medium size) to 6.5 (large size), on the Cortex-A15it was decreased in 18%, from 3.2 (medium size) to 2.7 (large size) and in 28%, from 3.5 (medium size) to 2.6 (large size), on the Cortex-A7.

When comparing the performance per core for the different processors, which can be easily computed by dividing by two the time ratios involving the Intel Sandy Bridge-EP processor presented in Fig. 3*d*, Intel Sandy Bridge-EP cores outperform Cortex-A15 and Cortex-A7 core by 4.58 and 13.28 times on

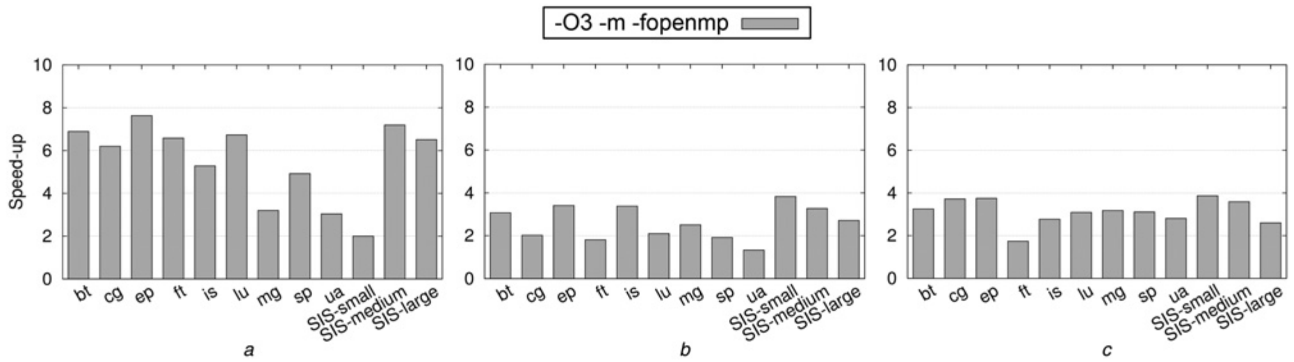

Fig. 4 *Performance improvements over the optimised sequential version with parallel execution on different processors*

a Intel Sandy Bridge-EP
b ARM big.LITTLE Cortex-A15
c ARM big.LITTLE Cortex-A7

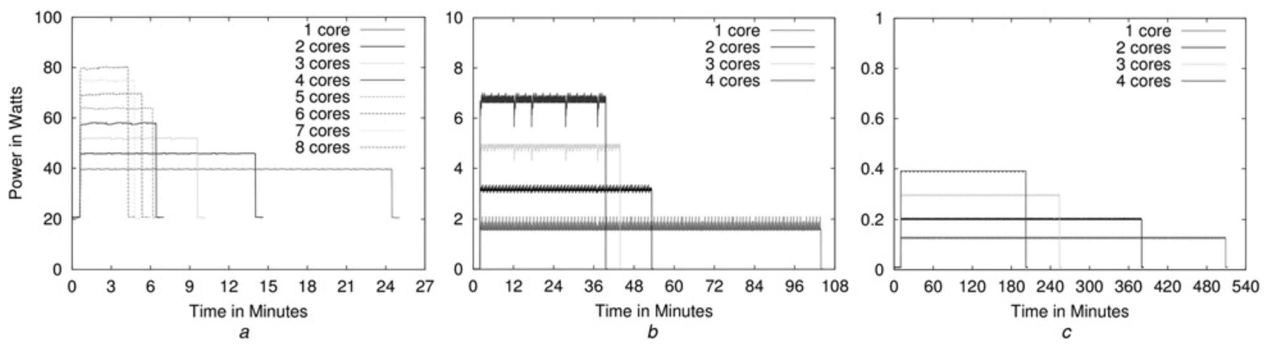

Fig. 5 *Instantaneous power against execution time to different number of cores in each processor*

a Intel Sandy Bridge-EP
b ARM big.LITTLE Cortex-A15
c ARM big.LITTLE Cortex-A7

average, respectively. In other words, one Intel Sandy Bridge-EP core is able to outperform a whole Cortex-A15 processor or three Cortex-A7 processors for these applications. Nevertheless, this comes at a price of an increased power demand, as discussed in the following section.

4.2 Power demand evaluation

This section presents an analysis of the power demand behaviour of the processors using different numbers of cores. We focus on SIS-large for this analysis, as it has shown the largest power demand among the tested benchmarks, as illustrated in Fig. 6*c*. This power demand comes from an intense use of SIMD instructions.

Fig. 5 illustrates the power demand measured for ARM big. LITTLE (Cortex-A15 and Cortex-A7) and Intel Sandy Bridge-EP when executing SIS-large using one or more cores. The horizontal axis represents time, whereas the vertical axis represents power demand. Each line represents the power consumption behaviour measured for a different number of used cores. Intel Sandy Bridge-EP shows the smallest proportional power demand increase (2×), as the average consumption goes from 39.59 W on one core only to 79.90 W for eight cores. This small difference happens because the static power consumption of Intel Sandy Bridge-EP is already very high when compared with the other processors. Additionally, each core employed by Intel Sandy Bridge-EP increases its power consumption by almost the whole power demand of Cortex-A15.

The power demand changes on ARM big.LITTLE are much smaller in number, but bigger in proportion to the ones on Intel Sandy Bridge-EP. The power consumed by one core running SIS-large on Cortex-A15is 1.61 W on average, whereas 6.57 W were measured when using all four cores, which represents an increase of 4.07×. Meanwhile, the power demanded by Cortex-A7varies from 0.13 to –0.39 W, which represents an increase of 3.05×.

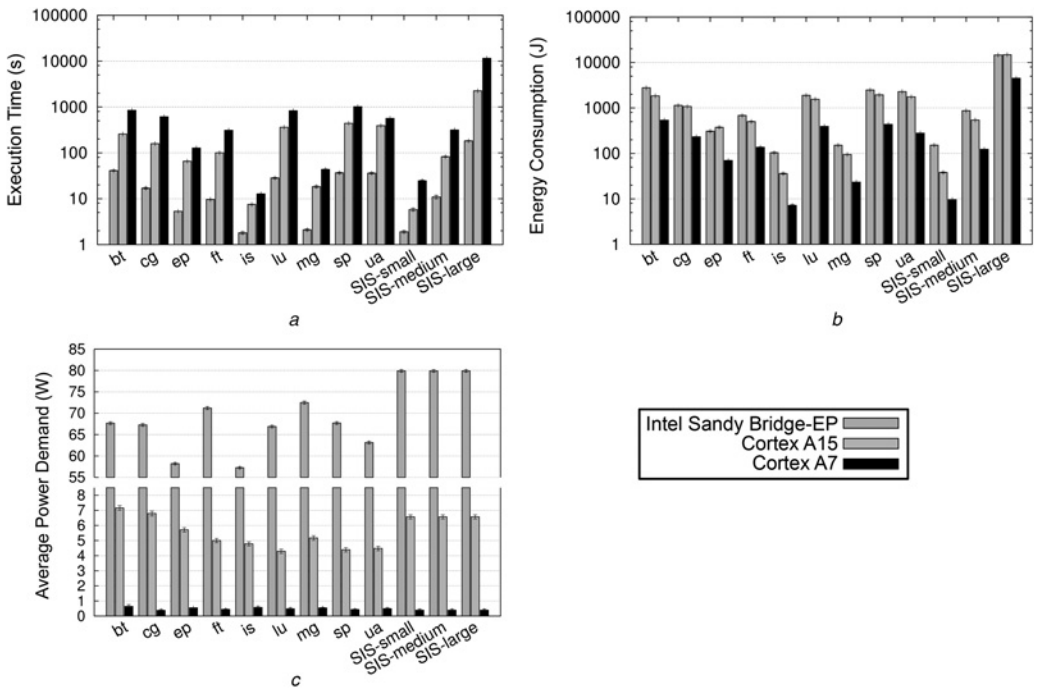

Fig. 6 *Evaluated metrics on ARM big.LITTLE and Intel Sandy Bridge-EP*

a Execution time
b Energy consumption
c Average power demand

Nevertheless, the power consumption of Cortex-A7 running SIS-large on all its cores is still 100× smaller than one Intel Sandy Bridge-EP core only. More precisely, when considering the use of all cores in a processor, Intel Sandy Bridge-EP, Cortex-A15 and Cortex-A7 present power demands per core of 9.99, 1.64 and 0.097 W, respectively. In other words, Intel Sandy Bridge-EP demands 6.07× (102.2×) more power per core than Cortex-A15 (Cortex-A7). These differences are very important because they define the base of the trade-off between the studied platforms.

4.3 Energy consumption evaluation

This section presents the processors' energy consumption analysis. The energy consumption represents the cost of maintaining the processors turned-on during the execution of an application. Fig. 6*b* reports the total energy consumption measured for the different applications on Intel Sandy Bridge-EP and ARM big.LITTLE.

Although Intel Sandy Bridge-EP is the most power demanding processor tested, as illustrated in Fig. 6*c*, Cortex-A15 was the one that consumed the most energy for EP. This happens because Intel Sandy Bridge-EP is more adapted to handling compute-intensive applications that work with double precision floating-point data. Cortex-A15 still consumes 39% less energy than Intel Sandy Bridge-EP on average for the evaluated applications, but this is mostly influenced by the results achieved for IS and SIS-small. When considering the other applications only, this difference is reduced to 22%.

In general, Cortex-A7 consumes almost one order of magnitude less than Intel Sandy Bridge-EP (more specifically, 7.10×). This is most clear in the case of the IS benchmark, where Cortex-A7consumes 14.46× and 5.01× less than Intel Sandy Bridge-EP and Cortex-A15, respectively. As Cortex-A7cores are well-optimised for integer workloads, the processor is able to achieve a fair performance while still keeping its low power consumption.

4.4 Performance/energy trade-off evaluation

An analysis of the performance/energy trade-off between the evaluated architectures requires understanding the time, power and energy results obtained for the different applications considered. The results indicate that Cortex-A15 and Cortex-A7 perform 9.16× and 26.55× slower than Intel Sandy Bridge-EP on average as illustrated in Fig. 6*a*. This performance difference is changed to 9.67× and 23.74× when considering the applications from NPB only. These differences for Cortex-A15 vary from 4.1× and 6.3× for benchmarks IS and BT, to 12.4× and 12.8× for EP and LU. For Cortex-A7, the performance difference varies from 7× for IS to 32.3× and 36.2× for benchmarks FT and CG. Meanwhile, the performance differences for SIS on Cortex-A15 (Cortex-A7) vary between 3.1× (12.9×) for the small size and 12.3× (63.1×) for the large size. Finally, the performance differences between Cortex-A15 and Cortex-A7 vary between 1.4× and 1.7× for EP and IS, and 3.9× and 5.1× for CG and SIS-large.

Although Cortex-A7 presented the worst performance among the evaluated processors, it also showed the best energy efficiency with an energy consumption 7.10× and 4.37× smaller than Intel Sandy Bridge-EP and Cortex-A15 on average. The largest energy consumption difference between Cortex-A7 and Intel Sandy Bridge-EP was registered for SIS-small (15.79×). For this benchmark, Cortex-A7 takes 12.94× longer to execute, but it demands $\frac{1}{204.4}$ of the power. Nevertheless, this kind of behaviour is only sustained for small workloads or applications working with integers, such as IS.

The energy efficiency of Cortex-A15 was highly variable, depending on the application and the input problem size. For instance, Cortex-A15 consumed more energy than Intel Sandy Bridge-EP to compute both EP and SIS-large, but consumed 2.89× and 3.9× less to execute IS and SIS-small. Even though Cortex-A15 presents, in general, a better performance/energy trade-off than Intel Sandy Bridge-EP (with power demands and energy consumption measured for the processors 12.62× and 1.63 smaller on average, respectively), its use may not be justifiable for high-performance scientific computing yet, since its 9.16× slower execution time on average could incur in a larger energy consumption of the whole system.

5 Related work

Energy consumption is a central issue on the development of the next generation of supercomputers. Research efforts have been focusing on evaluating the power consumption of general-purpose and low-power processors. However, there is still a lack of studies with the ARM Cortex-A15 architecture.

Blake *et al.* [14] compared multicore processors, such as ARM Cortex-A9, Intel Atom, XMOS XS1-G4, Intel Core i7 and Sun Niagara T2, according to aspects such as cache and microarchitecture. Dongarra and Luszczek [15] analysed the energy efficiency of ARM, Intel, AMD and NVIDIA processors. Their results point to a better energy efficiency of ARM processors. However, their energy consumption measurements considered the whole system and not the processor only. Valero [16] presented similar results with the Cortex-A9 architecture and estimated that ARM Cortex-A15 may achieve 8 GFlops/W. Stanley-Marbell and Cabezas [17] showed that ARM-based platforms present low power dissipation than PowerPC and Intel Atom when running light-weight workloads. Nonetheless, they also showed that Intel Atom was two times more energy efficient than the others for heavy-weight workloads.

The energy efficiency of ARM processors has been well researched outside the scope of HPC systems. Ou *et al.* [18] compared ARM and Intel Nehalem-based clusters for web services, and conclude that ARM provides 1.3 times better energy efficiency on average and is able to outperform the Intel one in some scenarios. McKenney *et al.* [19] achieved energy efficiency gains of 10% on Cortex-A15 systems for mobile workloads by tuning pre-existing Linux parameters. The use of different clock frequencies on Cortex-A9 and A8 processors for web pages was studied by Zhu and Reddi [20]. To measure energy consumption, they built an external power-sensing circuit. Using this circuit and web pages with different characteristics, they were able to build statistical inference models that estimate web page load time and energy consumption. In the context of web servers, Aroca and Gonçalves [21] analysed the feasibility of building servers based on low-power processors. They compared the power usage, CPU load, temperature and other characteristics of ×86 and ARM-based computers running as web and database servers, and concluded that

ARM systems are from 3 to 4 times more power efficient for running hypertext transfer protocol and SQL services.

The use of ARM and Intel processors for scientific applications is an important research topic. In a previous work, we evaluated the feasibility of using Cortex-A9 processors as building blocks for HPC systems [22]. Nonetheless, when using the metrics 'time-to-solution', power and 'energy-to-solution', these processors were outperformed by Intel Xeon ones. Additionally, energy measurements were done at a system granularity, as current sensors for the processor only were not available at the time. Meanwhile, Goddeke *et al.* [23] also conducted a comparison between ARM and classic ×86 architecture considering three HPC kernels. They evaluate the scalability of ARM-based system using 96 dual core processors, and concluded that clusters of ARMs are potentially more efficient than ×86 clusters.

The features present on current ARM big.LITTLE systems are the subject of studies outside scientific computing. For instance, Muthukaruppan *et al.* [24] introduced a hierarchical power management framework for asymmetric multicores. It coordinates multiple controllers in a synergistic manner to achieve optimal power-performance efficiency while respecting the thermal design power budget.

Differently from the other researches discussed here, our focus lies on evaluating the latest ARM big.LITTLE system, which features two heterogeneous quad-core processors (Cortex-A15 and Cortex-A7) and its use for parallel scientific applications. To do so, we proposed a platform-independent EMonDaemon that collects power and energy data from the heterogeneous systems. We thus used this tool to carry out our study on all applications from the NPB suite and on a real-world SIS.

6 Conclusions

Processors are responsible for a large percentage of total energy consumption in parallel platforms, so managing the power demands of processors is crucial to save energy in HPC. New processor architectures feature sensors that allow measurements of instantaneous power and/or accumulated energy consumption of the processors separately. This allows us to have a greater control over power demand and energy consumption at a granularity level that was not possible before. In this context, we developed a platform-independent tool named EMonDaemon that collects performance, power demand and energy data from homogeneous and heterogeneous systems. Using this tool, we provided a 'performance/energy trade-off' and detailed comparison between ARM big.LITTLE and Intel Sandy Bridge-EP processors using the NPBs and an application from the agroforestry domain that optimises the use of water during irrigation.

We compared performance, power demand and energy consumption using a sequential and parallel versions of the applications. The obtained results pointed that the use of compilation flags improve the performance of the applications by 9.44×, 6.28× and 4.89× on average for Intel Sandy Bridge-EP, Cortex-A15 and Cortex-A7, respectively, showing that processors with more complex organisations are able to benefit more from this kind of optimisation.

When using all cores available in the different processors, results show that Intel Sandy Bridge-EP has a performance per core 4.58× and 13.28× superior to Cortex-A15 and Cortex-A7. The biggest performance differences were seen for highly vectorised or memory-intensive applications, whereas the smallest one was reported for a benchmark that works with integer data.

Although the average power demand of Intel Sandy Bridge-EP was 12.62× and 152.40× bigger than the ones seen on Cortex-A15and Cortex-A7, its average energy consumption was only 1.6× and 7.10× superior. In this context, although Cortex-A15 presents a better performance/energy trade-off than Intel Sandy Bridge-EP, it may still not be usable for high-performance scientific computing because of a possible increase in the total energy consumption of the whole platform when the processor takes a much longer time (9.16×) to compute a solution.

Our future works will focus on investigating energy consumption and power demand using a cluster with a larger number of processors and also extend the 'performance/energy trade-off' evaluation to AMD processors. Moreover, we intend to use the EMonDaemon as a means to decide at run-time whether the executed application should be migrated or not from one processor to another in order to optimise the attained performance/energy efficiency on ARM big.LITTLE architectures. However, because of the limitation of current ARM big. LITTLE architectures, it would only be possible to migrate the entire application from one processor to another. If future generations of ARM big.LITTLE allow parallel applications to be executed on both processors simultaneously, it will also be possible to map specific threads to the Cortex-A15 and Cortex-A7 cores based on the information collected by EMonDaemon at run-time.

7 Acknowledgments

This work was partially supported by CNPq, FAPERGS, FINEP, CAPES (under grants 3471-13-6, 5854-11-3 and 5847-11-7) and by the HPC-GA research project, which has received funding from the European Community's Seventh Framework Programme (FP7-PEOPLE) under grant agreement number 295217. This work was developed in the context of LICIA, an associated international laboratory between UFRGS and University of Grenoble.

8 References

1 Dong, Y., Chen, J., Tang, T.: 'Power measurements and analyses of massive object storage system'. Int. Conf. on Computer and Information Technology (CIT), West Yorkshire, UK, 2010, pp. 1317–1322
2 Laros, J., Pedretti, K., Kelly, S., *et al.*: 'Topics on measuring real power usage on HPC platforms'. Int. Conf. on Cluster Computing (ICCC), New Orleans, USA, 2009, pp. 1–8
3 Feng, W., Lin, H.: 'The green500 list: Year two'. Parallel & Distributed Processing, Workshops (IPDPSW), Atlanta, Georgia, USA, 2010, pp. 1–8
4 Subramaniam, B., Feng, W.-c.: 'Understanding power measurement implications in the green500 list'. Green Computing and Communications (GREENCOM), Washington, DC, USA, 2010, pp. 245–251, [Online]. Available at http://www.dx.doi.org/10.1109/GreenCom-CPSCom.2010.140
5 Younge, A., von Laszewski, G., Wang, L., Lopez-Alarcon, S., Carithers, W.: 'Efficient resource management for cloud computing environments'. Int. Green Computing Conf. (IGCC), Chicago, IL, USA, 2010, pp. 357–364
6 Barker, K., Davis, K., Hoisie, A., *et al.*: 'Using performance modeling to design large-scale systems', *IEEE Comput.*, 2009, **42**, (11), pp. 42–49
7 Kogge, P., Bergman, K., Borkar, S., *et al.*: 'Exascale computing study: technology challenges in achieving exascale systems'. DARPA IPTO, 2008, pp. 1–297

8 Beckman, P., Dally, B., Shainer, G., Dunning, T., Ahalt, S.C., Bernhardt, M.: 'On the road to exascale', *Sci. Comput. World*, 2011, **1**, (116), pp. 26–28
9 Rotem, E., Naveh, A., Rajwan, D., Ananthakrishnan, A., Weissmann, E.: 'Power-management architecture of the Intel microarchitecture code-named sandy bridge', *IEEE Micro*, 2012, **32**, (2), pp. 20–27
10 Bailey, D.H., Barszcz, E., Barton, J.T., *et al.*: 'The NAS parallel benchmarks', *Int. J. High Perform. Comput. Appl.*, 1991, **5**, (3), pp. 63–73
11 Miyazaki, T.: 'Water flow in soils' (Florida: CRC Press, 2006)
12 Doussan, C., Jouniaux, L., Thony, J.: 'Variations of self-potential and unsaturated water flow with time in sandy loam and clay loam soils', *J. Hydrol.*, 2002, **267**, (3), pp. 173–185
13 Padoin, E.L., Pilla, L.L., Boito, F.Z., Kassick, R.V., Velho, P., Navaux, P.O.A.: 'Evaluating application performance and energy consumption on hybrid CPU + GPU architecture', *Cluster Comput.*, 2013, **16**, (3), pp. 511–525
14 Blake, G., Dreslinski, R., Mudge, T.: 'A survey of multicore processors', *Signal Process. Mag.*, 2009, **26**, (6), pp. 26–37
15 Dongarra, J., Luszczek, P.: 'Anatomy of a globally recursive embedded linpack benchmark'. High Performance Extreme Computing Conf. (HPEC), Waltham, USA, 2012, pp. 1–6
16 Valero, M.: 'Towards exaflop supercomputers'. High Performance Computing Academic Research Network (HPC-net), Rio Patras, Greece, 2011, pp. 1–117
17 Stanley-Marbell, P., Cabezas, V.: 'Performance, power, and thermal analysis of low-power processors for scale-out systems'. Parallel & Distributed Processing, Workshops (IPDPSW). Anchorage, Alaska, USA, 2011, pp. 863–870
18 Ou, Z., Pang, B., Deng, Y., Nurminen, J., Yla-Jaaski, A., Hui, P.: 'Energy-and cost-efficiency analysis of ARM-based clusters'. Cluster, Cloud and Grid Computing (CCGrid), Ottawa, Canada, 2012, pp. 115–123
19 McKenney, P.E., Eggeman, D., Randhawa, R.: 'Improving energy efficiency on asymmetric multiprocessing systems'. Technical Report, 2013
20 Zhu, Y., Reddi, V.J.: 'High-performance and energy-efficient mobile web browsing on big/little systems'. High Performance Computer Architecture (HPCA), Shenzhen, China, 2013, pp. 13–24
21 Aroca, R., Gonçalves, L.G.: 'Towards green data-centers: a comparison of ×86 and ARM architectures power efficiency', *J. Parallel Distrib. Comput. (JPDC)*, 2012, **72**, (12), pp. 1770–1780
22 Padoin, E.L., de Oliveira, D.A.G., Velho, P., Navaux, P.O.A.: 'Time-to-solution and energy-to-solution: a comparison between ARM and xeon'. Workshop on Applications for Multi-Core Architectures (WAMCA), New York, USA, 2012, pp. 48–53
23 Göddeke, D., Komatitsch, D., Geveler, M., *et al.*: 'Energy efficiency vs. performance of the numerical solution of PDEs', *J. Comput. Phys.*, 2012, **237**, pp. 132–150
24 Muthukaruppan, T.S., Pricopi, M., Venkataramani, V., Mitra, T., Vishin, S.: 'Hierarchical power management for asymmetric multi-core in dark silicon era'. Design Automation Conf. (DAC). San Francisco, USA, 2013, pp. 1–9