Power Consumption Characterization of Synthetic Benchmarks in Multicores

Jonathan Muraña[1(✉)], Sergio Nesmachnow[1], Santiago Iturriaga[1], and Andrei Tchernykh[2]

[1] Universidad de la República, Montevideo, Uruguay
{jmurana,sergion,siturria}@fing.edu.uy
[2] CICESE Research Center, Ensenada, Baja California, Mexico
chernykh@cicese.mx

Abstract. This article presents an empirical evaluation of power consumption of synthetic benchmarks in multicore computing systems. The study aims at providing an insight of the main power consumption characteristics of different applications when executing over current high performance computing servers. Three types of applications are studied executing individually and simultaneously on the same server. Intel and AMD architectures are studied in an experimental setting for evaluating the overall power consumption of each application. The main results indicate the power consumption behavior has a strong dependency with the type of application. An additional performance analysis shows that the best load of the server regarding energy efficiency depends on the type of the applications, with efficiency decreasing in heavily loaded situations. These results allow characterizing applications according to power consumption, efficiency, and resource sharing, and provide useful information for resource management and scheduling policies.

Keywords: Green computing · Energy efficiency · Multicores
Computing efficiency

1 Introduction

Nowadays, data centers are key infrastructures for executing industrial and scientific applications. Data centers have become highly popular for providing storage, computing power, middleware software, and others information technology (IT) utilities, available to researchers with ubiquitous access [3]. However, their energy efficiency has become a major concern in recent years, having a significant impact on monetary cost, environment, and guarantees for service-level agreements (SLA) [4].

The main sources of power consumption in data centers are the computational resources and the cooling system [13]. When focusing on power consumption due to resource utilization, several techniques for hardware and software optimization can be applied to improve energy efficiency. Software characterization techniques [1] are used to determine features that are useful to analyze

the software behavior, including power consumption [2]. Estimation of power consumption is an important issue that can determine the quality and even the feasibility of both software products and big data center infrastructures.

In this line of work, this article focuses on the characterization of power consumption for scientific computing applications over nowadays multicore hardware used in scientific computing platforms. Such characterization is useful for designing energy efficient scheduling strategies for scientific computing platforms. Three synthetic benchmarks are studied over two physical setups from a real High Performance Computing (HPC) platform, registering their power consumption with a power meter device. Furthermore, the experimental analysis studies the power consumption of different applications sharing a computing resource via simultaneous execution. The proposed study is very relevant for nowadays data centers and HPC infrastructures, where many applications are executed at the simultaneously. This impacts on both the energy efficiency and the quality of service (QoS) offered to the users of the platform.

The article is organized as follows. Section 2 reviews related works on energy characterization in multicores. Section 3 describes the proposed methodology for energy characterization, the benchmarks, and the physical setup for experiments. The experimental results are reported and discussed in Sect. 4. The conclusions and the main lines for future work are presented in Sect. 5.

2   Related Work

The analysis of related works allows identifying two types of studies. A first group proposes the empirical evaluation of scientific applications to develop specific models or to adjust existing models for studying power and performance behavior. Other works introduces optimization techniques over existing models and presents optimized results. This section focuses on reviewing those articles about empirical measurements and evaluation of power consumption models.

Iturriaga et al. [7] studied the problem of finding schedules with appropriate trade-off between power consumption and execution time in heterogeneous computers systems, considering uncertainty. Specific versions of well-known heuristic were proposed for scheduling on realistic scenarios, applying the power consumption model in [12] and considering only CPU-bound workloads. A model for uncertainty on power consumption was determined through empirical evaluations using three CPU-bound benchmarks. Regarding scheduling results, online heuristics computed better schedules than offline approaches. Results also confirmed that uncertainly has a significant impact in the accuracy of the scheduling algorithms. The power consumption behavior of CPU-bound benchmarks shown in [7] is consistent with the one in our research. Moreover, we propose a fully empirical power consumption characterization, considering also two additional types of benchmarks: memory bound and disk bound.

Srikantaiah et al. [14] studied workload consolidation strategies for energy optimization in cloud computing systems. An empirical study of the relationship between power consumption, performance, and resource utilization was presented. The experiments were executed in four physical server connected to a

power meter to track the power consumption. The resource utilization was monitorized using the Xperf toolkit. Only two resources were considered in the study: processor and disk. The performance degraded for high levels of disk utilization, and variations in CPU usage did not result in significant performance variations. Energy results were presented in terms of power consumption per transaction (including power consumption in idle state), resources utilization, and performance degradation. On the other hand, results showed that power consumption per transaction, is more sensitive to CPU utilization than disk utilization. The authors also proposed an heuristic method to solve a modified bin packing problem where the servers are bins and the computing resources are bin dimensions. Results were reported for small scenarios where the power consumption of the solutions computed by the heuristic is about 5% more than the optimal solution. The tolerance for performance degradation was 20%.

Du Bois et al. [5] presented a framework for generating workloads with specific features, applied to compare energy efficiency in commercial systems. CPU-bound, memory-bound, and disk-bound benchmarks were executed on a power monitoring setup composed of an oscilloscope connected to the host and a logging machine to persist the data. Two commercial systems were studied: a high-end with AMD processors and a low-end with Intel processors. Benchmarks were executed independently, isolating the power consumption of each resource. Results confirmed that energy efficiency depends on the workload type. Comparatively, the high-end system had better results for the CPU-bound workload, the low-end system was better for disk-bound, and both had similar efficiency for the memory-bound workload. Our work complements this approach by including a study of the power consumption behavior when executing different types of tasks simultaneously on specific architectures for high performance computing.

Feng et al. [6] evaluated the energy efficiency of a high-end distributed system, with focus on scientific workloads. The authors proposed a power monitoring setup that allows isolating the power consumption of CPU, memory, and disk. The experimental analysis studied single node executions and distributed executions. In the single node experiments, results of executing a memory-bound benchmark showed that the total power consumption is distributed as follow: 35% corresponds to the CPU, 16% corresponds to the physical memory and 7% corresponds to the disk. The rest is consumed by power supply, fans, network, and other components. Idle state represented 66% of the total power consumption. In distributed experiments, benchmarks that are intensive in more than one computing resource were studied. Results showed that energy efficiency increased with the number of nodes used for execution.

Kurowski et al. [9] presented a data center simulator that allows specifying various energy models and management policies. Three types of theoretical energy models are proposed: (i) *static approach*, which consider a unique power value by processing unit; (ii) *dynamic approach*, which consider power levels, representing the usage of the processing unit; and (iii) *application specific approach*, which consider usage of application resources to determine the power consumption. Simulation results were compared with empirical measurements

over real hardware to validate the theoretical energy models in arbitrary scenarios. All models obtained accurate results (error was less than 10% with respect to empirical measurements), and the dynamic approach was the most precise.

Langer et al. [10] studied energy efficiency of low voltage operations in manycore chips. Two scientific applications were considered for benchmarking over a multicore simulator. The performance model considered for a chip was $S = a_k(\sum f_i) + b_k$, where S are the instructions per cycle, f is the frequency of the core i and a_k, b_k are constants that depend on k, the number of cores in the chip. A similar model is used for power consumption. Across 25 different chips, an optimization method based on integer linear programming achieved 26% in energy savings regarding to the power consumption of the faster configuration.

Several works in literature have focused on modeling and characterizing power consumption of scientific applications. However, to the best of our knowledge there is no empirical research focused on the inter-relationship between power consumption and CPU, memory, and disk utilization. Also, there is no experimental analysis of critical levels of resource utilization (close to 100%) and its impact on power consumption and performance. This article contributes in this line of research, proposing empirical analysis for both aforementioned issues.

3   Methodology for Power Consumption Evaluation

This section describes the proposed methodology for power consumption evaluation, the benchmarks and architectures studied, the power evaluation setup, and the experiments designed.

3.1   Overview of the Proposed Methodology

Experiments characterize the power consumption of the most relevant computing resources: CPU, memory, and disk [5–7]. We aim at studying holistic behaviors and analyzing the power consumption of hosts close to 100% of computing resource utilization. The analysis is complemented with performance experiments to study the trade-off between power consumption and performance degradation.

3.2   Benchmarks

The benchmarks used in the analysis are part of the Sysbench toolkit [8]. Sysbench is a cross-platform software written in C that provides CPU, memory, and disk intensive benchmarks for performance evaluation. The components used in the experiments are:

1. *CPU-bound*: an algorithm that calculates $\pi(n)$ (the prime counting function) using a backtracking method. The algorithm contains loops, square root and module operations, as described in Algorithm 1.

Algorithm 1. CPU-bound benchmark code

```
 1: C ← 3
 2: while C < MAX_PRIME do
 3:    T ← sqrt(C)
 4:    L ← 2
 5:    while L < T do
 6:       if C (mod T) = 0 then
 7:          break
 8:       end if
 9:       if L > T = 0 then
10:          N ← N + 1
11:       end if
12:       L ← L + 1
13:    end while
14:    C ← C + 1
15: end while
16: return  N
```
---

2. *Memory-bound*: a program that executes write operations in memory, as described in Algorithm 2, where the *BUF* variable is an array of integers. The cells of the array are overwritten with value of *TMP* until the last position of the array, i.e., the value of the *END* variable.

Algorithm 2. Memory-bound benchmark code

```
1: while BUF < END do
2:    *BUF ← TMP
3:    BUF ← BUF + 1
4: end while
```
---

3. *Disk-bound*: a program that reads/writes content in files. Read or write requests are generated randomly and executed until a given number of requests ($MAX\_REQUEST$) is reached, as described in Algorithm 3.

Algorithm 3. Disk-bound benchmark code

```
 1: while REQS_COUNT < MAX_REQS do
 2:    REQ ← generate_rnd_request()
 3:    if is_read(REQ) then
 4:       read(REQ.FILE)
 5:    end if
 6:    if is_write(REQ) then
 7:       write(REQ.FILE)
 8:    end if
 9:    REQS_COUNT ++
10: end while
```

3.3   Multicore Hosts and Power Monitoring Setup

Experiments were performed on Cluster FING, the HPC platform from Universidad de la República, Uruguay [11]. Two hosts were chosen according to their characteristics and availability: HP Proliant DL385 G7 server (2 AMD Opteron 6172 CPUs, 12 cores each, and 72 GB RAM), and HP Proliant DL380 G9 server (2 Intel Xeon CPU E5-2680v3 CPUs, 12 cores each, and 128 GB RAM).

Figure 1 presents the power monitoring setup. Benchmarks were executed in a host connected to the power source via a Power Distribution Unit (PDU) to register the instant power consumption. In an secondary machine, a polling demon logged data for post-processing. This configuration is similar to the one used in related works [5,7].
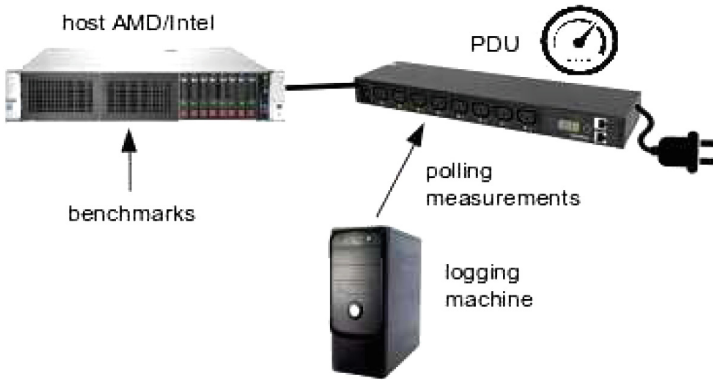

Fig. 1. Power monitoring setup

3.4   Design of Experiments

The power consumption (PC) of each host is computed as the average of 20 independent executions for each benchmark and combination of benchmarks. Idle consumption (IC), i.e., average consumption of the host without load, is registered to compute the effective consumption (EC) as $EC = PC - IC$.

In a first stage, benchmarks are evaluated independently from each other, analyzing only one resource. Utilization level (UL) is defined as the percentage of processors being used regarding the total number of processors in the host. Eight ULs were considered for single benchmark execution: 12%, 25%, 37.5%, 50%, 62.5%, 75%, 87.5% and 100%. Figure 2 shows an example of 37.5% utilization level: the host has 24 processors of which 9 execute instances of the CPU-bound benchmark. The remaining processors are in *idle* state.

In a second stage, the simultaneous execution of benchmarks is evaluated, analyzing several combinations of resource utilization at the same time. Two and three benchmarks are executed together in different ULs. In this case, UL is a vector where each entry represents the percentage of processors being used by
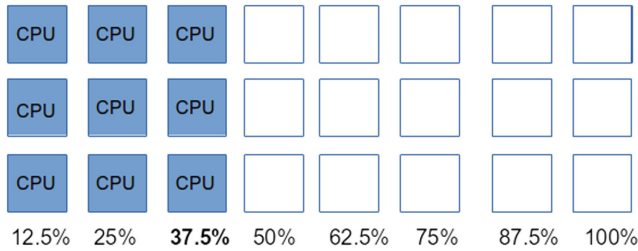
Fig. 2. CPU-bound benchmark, UL of 37.5%

each type of benchmark considered in the study (CPU-bound, memory-bound, and disk-bound, in that order). The utilization levels chosen were $(25\%, 25\%)$, $(25\%, 50\%)$, $(25\%, 75\%)$, $(50\%, 25\%)$, $(50\%, 50\%)$, $(75\%, 25\%)$ in pair executions and $(25\%, 25\%, 25\%)$, $(25\%, 25\%, 50\%)$, $(25\%, 50\%, 25\%)$, $(50\%, 25\%, 25\%)$ in triple executions. Figure 3 shows CPU-bound and memory-bound executing together with UL of $(25\%, 50\%)$ and Fig. 4 shows CPU-bound, memory-bound and disk-bound executing combined with UL of $(25\%, 25\%, 50\%)$.
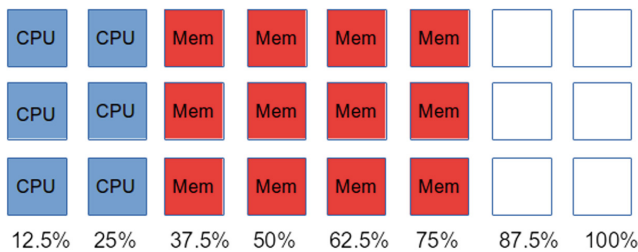
Fig. 3. CPU-bound and memory-bound benchmarks, UL of $(25\%, 50\%)$.
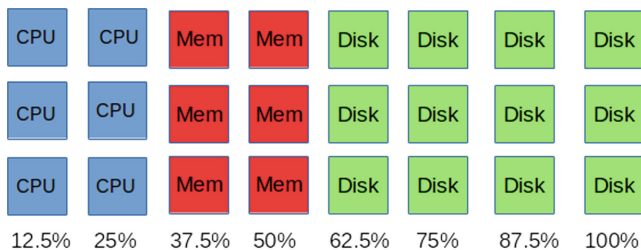
Fig. 4. CPU-, memory-, and disk-bound benchmarks in UL of $(25\%, 25\%, 50\%)$.

Each instance of the memory-bound benchmark is configured to use $(100/N)$ percent of the available memory, where N is the number of processors of the host (100% of the memory is used in full utilization mode). Each instance of the

disk benchmark is configured to use 4 GB of disk size in AMD experiments and 2 GB in Intel experiments. These sizes were chosen taking into account the disk size available in each host. Instances were executed and monitored for 60 s.

Finally, the impact on performance is analyzed. The *makespan* when executing multiple applications at the same time is compared with the makespan of single executions. The reported makespan values are the average computed over 20 independent executions.

4   Experimental Results

This section reports the experimental results of the power consumption and the performance evaluation. The average idle consumption was 183.4 W for the AMD host and 57.0 W for the Intel host.

4.1   Single Benchmark Executions

Figure 5 reports PC and EC values for the CPU-bound benchmark and a graphic comparison of EC values in both hosts. Results show an average difference of 56 W between the EC of the Intel host and the AMD host for all ULs. The almost linear behavior indicates that power consumption is proportional to the UL.

AMD		
UL	*PC*	*EC*
12.5%	194.2±0.6	10.8
25.0%	204.6±0.7	21.2
37.5%	214.6±0.5	31.2
50.0%	224.4±0.9	41.0
62.5%	234.6±1.6	51.2
75.0%	245.2±1.3	61.8
87.5%	253.8±1.2	70.4
100.0%	262.8±2.0	79.4

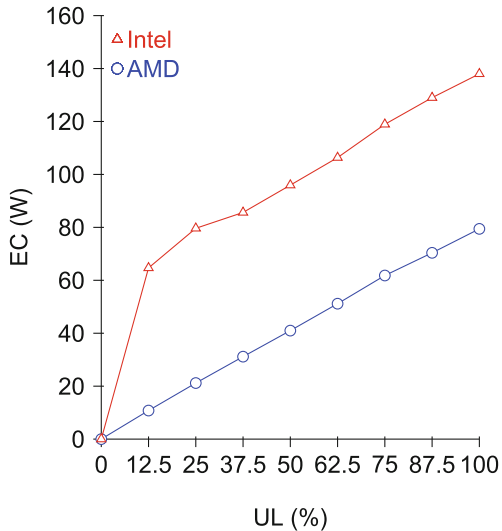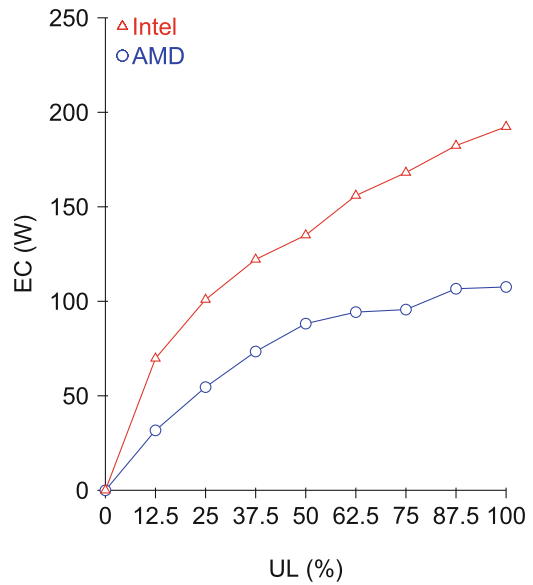| Intel		
UL	*PC*	*EC*
12.5%	121.7±1.7	64.7
25.0%	136.7±2.0	79.6
37.5%	142.7±2.1	85.6
50.0%	153.0±2.8	96.0
62.5%	163.4±2.7	106.4
75.0%	176.0±1.8	118.9
87.5%	186.0±2.2	129.0
100.0%	195.0±4.1	138.0

Fig. 5. CPU-bound PC and EC results

Figure 6 reports the PC and EC values for the memory-bound benchmark and a graphic comparison of the EC values in both hosts. Results show a significant increment in EC with regard to CPU-bound executions for all ULs (104% for the AMD host and 36% for the Intel host, on average).

AMD		
UL	PC	EC
12.5%	215.1±2.0	31.7
25.0%	238.0±1.1	54.6
37.5%	256.8±2.2	73.4
50.0%	271.6±2.9	88.2
62.5%	277.7±6.3	94.3
75.0%	279.0±5.9	95.6
87.5%	290.0±5.6	106.6
100.0%	290.9±13.0	107.6

Intel		
UL	PC	EC
12.5%	126.8±5.9	69.8
25.0%	158.0±4.3	100.9
37.5%	179.2±4.4	122.2
50.0%	192.0±6.7	135.0
62.5%	213.0±3.9	156.0
75.0%	225.1±5.4	168.1
87.5%	239.4±3.9	182.4
100.0%	249.4±8.9	192.4

Fig. 6. Memory-bound PC and EC results

AMD		
UL	PC	EC
12.5%	188.1±0.4	4.7
25.0%	189.3±0.6	5.9
37.5%	189.5±0.6	6.1
50.0%	189.3±0.3	5.9
62.5%	189.8±0.3	6.4
75.0%	190.1±1.5	6.7
87.5%	190.7±1.5	7.3
100.0%	190.4±0.6	7.0

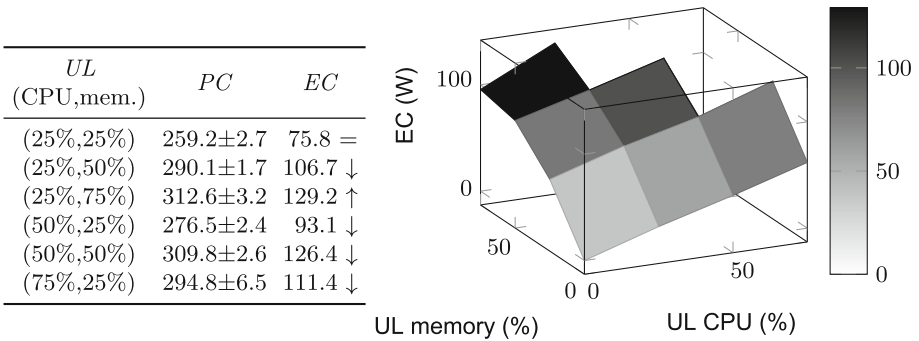| Intel		
UL	PC	EC
12.5%	67.5±0.9	10.5
25.0%	68.0±0.6	11.0
37.5%	67.7±0.6	10.7
50.0%	68.9±0.7	11.8
62.5%	70.5±0.8	13.5
75.0%	70.7±0.8	13.7
87.5%	71.7±0.6	14.7
100.0%	72.4±0.9	15.4
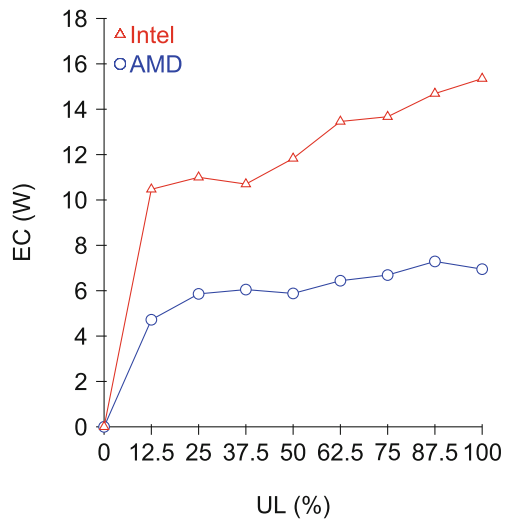
Fig. 7. Disk-bound PC and EC results

A logarithmic behavior is observed for both PC and EC, which does not occur in CPU-bound case, mainly due to the bottleneck in the access to the main memory that reduces the CPU usage. No significant increase is detected on high/critical ULs, possibly by effective resource contention by the operating system for solving conflicts over access to shared resources.

Figure 7 reports PC and EC values for the disk-bound benchmark and a comparison of EC values in both hosts. The maximum EC variation through ULs is 4 W in Intel and 2 W in AMD. These low power variation indicate that disk usage has low impact in power consumption in comparison with CPU and memory, mainly due to waits generated by bottlenecks in disk access.

4.2   Combined Benchmark Executions

CPU and memory. Figure 8 reports PC and EC when executing CPU- and memory-bound benchmarks together and a comparison of EC values on AMD. Figure 9 reports the same analysis on Intel. Symbol ↑ indicates the EC of the combined benchmarks is higher than the sum of the ECs of each benchmark executed independently, i.e., the combined execution is less efficient than the independent execution. Symbol ↓ indicates the opposite, that is, the combined execution is more efficient. Symbol = indicates that the values are equal (less than 1 W of difference). Results show that for AMD, combined executions reduces EC compared to independent executions. On the contrary, EC of combined executions is higher than independent executions for most cases on Intel.

UL (CPU,mem.)	PC	EC
(25%,25%)	259.2±2.7	75.8 =
(25%,50%)	290.1±1.7	106.7 ↓
(25%,75%)	312.6±3.2	129.2 ↑
(50%,25%)	276.5±2.4	93.1 ↓
(50%,50%)	309.8±2.6	126.4 ↓
(75%,25%)	294.8±6.5	111.4 ↓

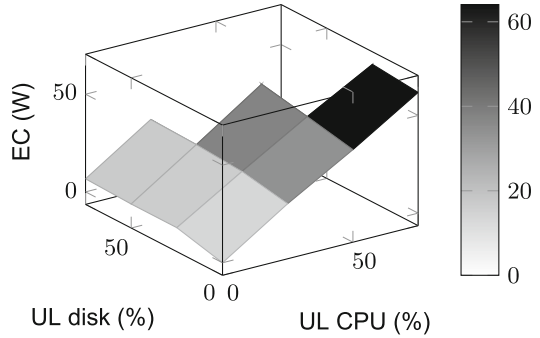**Fig. 8.** Combined CPU- and memory-bound PC and EC on AMD

CPU and disk. Figure 10 (AMD) and Fig. 11 (Intel) report PC and EC when executing CPU- and disk-bound benchmarks combined, and the EC graphics on each host. Results show that the combined execution of CPU and disk benchmarks improves energy efficiency for most ULs in both hosts, with regard to EC.

UL (CPU,mem.)	PC	EC
(25%,25%)	263.9±1.9	206.8 ↑
(25%,50%)	296.4±2.9	239.3 ↓
(25%,75%)	325.4±2.1	268.4 ↑
(50%,25%)	278.5±2.0	221.4 ↑
(50%,50%)	320.0±4.0	262.9 ↑
(75%,25%)	302.2±2.6	245.2 ↑
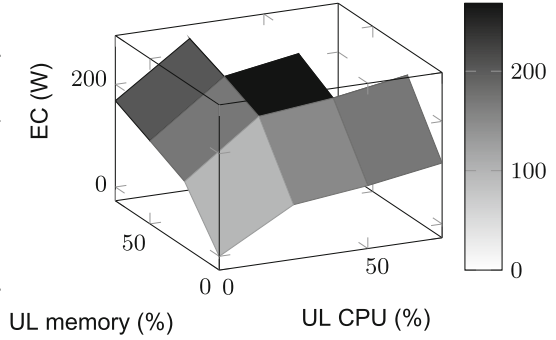
Fig. 9. Combined CPU- and memory-bound PC and EC on Intel

UL (CPU,disk)	PC	EC
(25%,25%)	209.6±1.1	26.2 =
(25%,50%)	211.2±1.3	27.8 =
(25%,75%)	212.0±1.2	28.6 =
(50%,25%)	229.2±2.3	45.8 ↓
(50%,50%)	233.7±4.9	50.3 ↑
(75%,25%)	247.5±3.4	64.1 ↓

Fig. 10. Combined CPU- and disk-bound PC and EC on AMD

UL (CPU,disk)	PC	EC
(25%,25%)	136.5±2.3	79.5 ↓
(25%,50%)	139.1±1.9	82.1 ↓
(25%,75%)	137.3±3.2	80.3 ↓
(50%,25%)	154.4±2.6	97.3 ↓
(50%,50%)	156.6±2.7	99.6 ↓
(75%,25%)	178.7±2.8	121.0 ↓
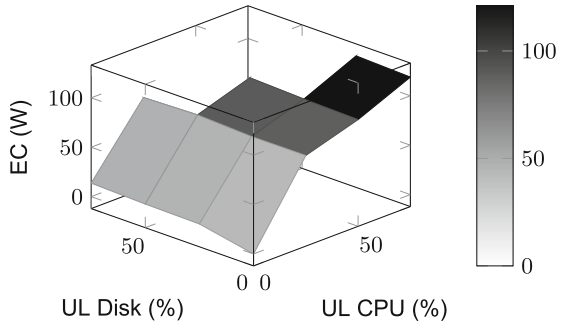
Fig. 11. Combined CPU- and disk-bound PC and EC on Intel

Memory and disk. Figure 12 (AMD) and Fig. 13 (Intel) report PC and EC for memory- and disk-bound benchmarks combined, and the EC graphics on each host. Results show that the combined execution presents higher values of EC than their independent execution, except for low ULs of the AMD host.

UL (mem.,disk)	PC	EC
(25%,25%)	243.2±1.4	59.8 =
(25%,50%)	241.2±5.2	57.8 =
(25%,75%)	245.4±2.1	62.0 =
(50%,25%)	267.5±9.0	84.1 ↓
(50%,50%)	269.0±12.6	85.6 ↓
(75%,25%)	286.4±4.1	102.0 ↑

Fig. 12. Combined memory- and disk-bound PC and EC on AMD

UL (mem.,disk)	PC	EC
(25%,25%)	174.4±8.3	117.4 ↑
(25%,50%)	176.7±6.2	119.7 ↑
(25%,75%)	172.6±7.5	115.6 ↑
(50%,25%)	206.7±5.7	149.7 ↑
(50%,50%)	208.8±4.0	151.8 ↑
(75%,25%)	228.8±4.8	171.7 ↓
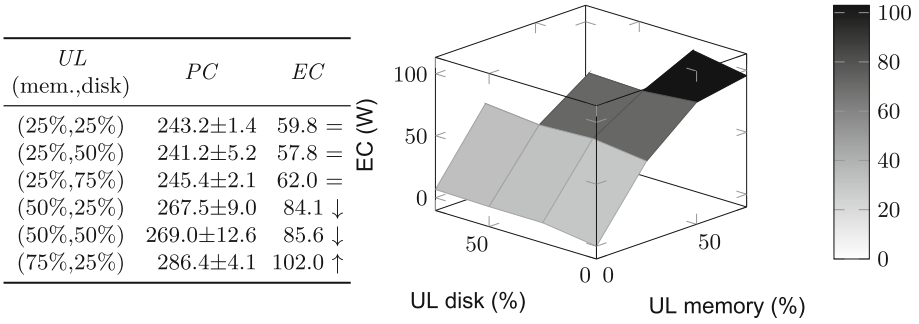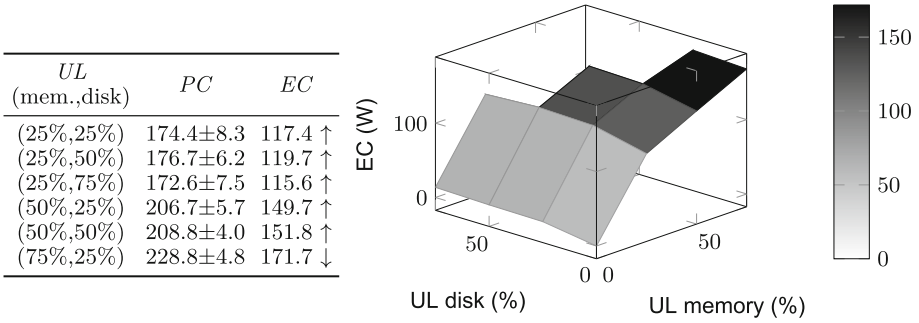
Fig. 13. Combined memory- and disk-bound PC and EC on Intel

CPU, memory, and disk. Table 1 reports PC and EC of CPU-, memory- and disk-bound benchmarks executed together. Results show that the combined execution on AMD has higher EC compared to their independent execution, mainly at high ULs. However, on Intel, combined executions reduce EC compared to independent executions for all ULs.

Table 1. Combined CPU-, memory- and disk-bound PC and EC

UL	AMD		Intel	
	PC	EC	PC	EC
(25%, 25%, 25%)	265.9±4.3	82.5 =	176.9±3.8	119.9 ↓
(25%, 25%, 50%)	266.6±4.8	83.2 ↓	178.2±3.3	121.6 ↓
(25%, 50%, 25%)	303.0±3.1	119.6 ↑	221.3±5.4	164.3 ↓
(50%, 25%, 25%)	287.9±1.8	104.5 ↑	194.5±1.8	137.5 ↓

4.3   Performance Evaluation

This subsection analyzes the performance evaluation experiments.

Figure 14 reports the makespan of the CPU-bound benchmark and a graphic comparison for both hosts. Results show that increasing the UL does not impact significantly the completion time, due to the absence of resource competition. However, both hosts present a slight degradation for UL 100%, possibly due to conflicts with operating system processes.
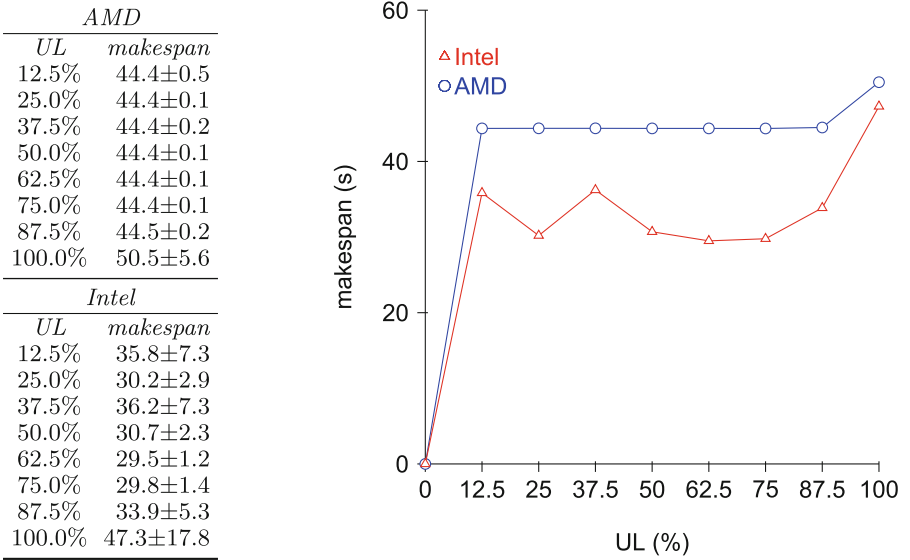
AMD	
UL	makespan
12.5%	44.4±0.5
25.0%	44.4±0.1
37.5%	44.4±0.2
50.0%	44.4±0.1
62.5%	44.4±0.1
75.0%	44.4±0.1
87.5%	44.5±0.2
100.0%	50.5±5.6

Intel	
UL	makespan
12.5%	35.8±7.3
25.0%	30.2±2.9
37.5%	36.2±7.3
50.0%	30.7±2.3
62.5%	29.5±1.2
75.0%	29.8±1.4
87.5%	33.9±5.3
100.0%	47.3±17.8

Fig. 14. CPU-bound makespan results

Figure 15 reports the makespan of the memory-bound benchmark and a graphic comparison for both hosts. Performance degrades in AMD; there is a gap of 400 s between the lowest and the highest UL. For Intel the difference is only 48 s. The difference in gaps is possibly due to specific disk features of each host, such as transfer speed.

Figure 16 reports the makespan of the disk-bound benchmark and a graphic comparison for both hosts. The disk-bound case presents a notorious degradation in performance when increasing UL when compared with other benchmarks.

4.4   Energy Efficiency Analysis

This subsection analyzes the energy efficiency from the collected measurements. The energy efficiency metric $\frac{PC \times makespan}{number\ of\ instances \times 3600}$ is defined for comparing results. The lower the metric value, the higher energy efficiency of the host.

UL	makespan
AMD	
12.5%	90.3±6.6
25.0%	136.8±21.4
37.5%	179.8±26.8
50.0%	220.7±33.8
62.5%	258.9±39.3
75.0%	326.7±42.6
87.5%	408.2±44.6
100.0%	490.6±58.9
Intel	
12.5%	34.7±6.4
25.0%	26.5±1.4
37.5%	41.0±4.4
50.0%	41.5±1.9
62.5%	67.5±7.2
75.0%	62.1±3.1
87.5%	74.4±5.6
100.0%	82.5±4.1
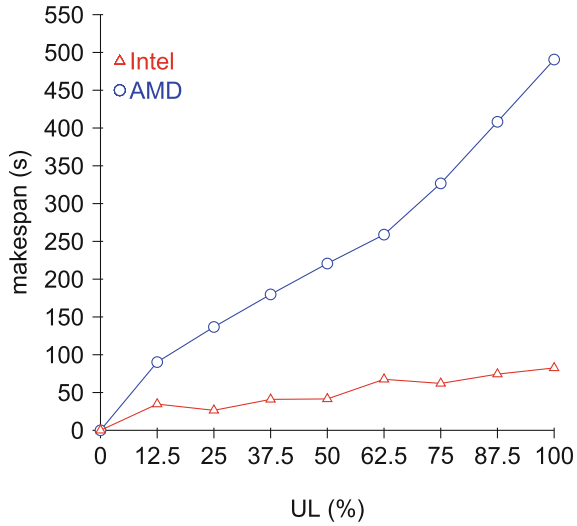
Fig. 15. Memory-bound *makespan* results

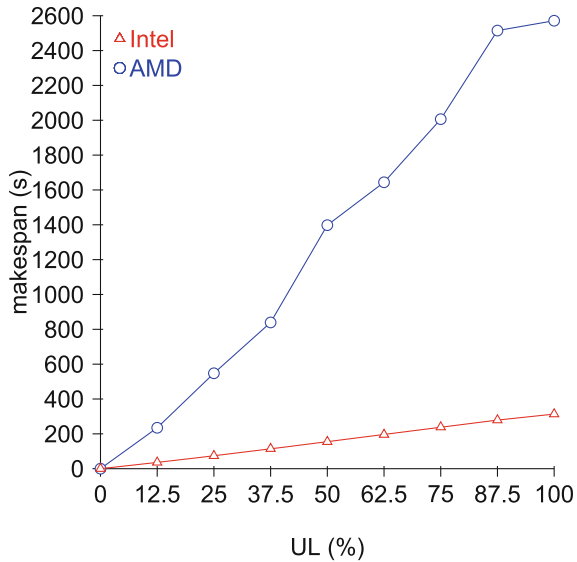| *AMD*	
UL	makespan
12.5%	234.8±8.5
25.0%	547.7±30.4
37.5%	839.4±39.5
50.0%	1397.8±144.5
62.5%	1644.0±171.1
75.0%	2006.3±169.6
87.5%	2514.8±306.1
100.0%	2571.3±217.2
Intel	
UL	makespan
12.5%	36.1±1.7
25.0%	73.9±1.5
37.5%	114.7±3.6
50.0%	154.6±6.0
62.5%	195.7±7.9
75.0%	238.8±10.5
87.5%	279.0±19.5
100.0%	313.5±42.9

Fig. 16. Disk-bound *makespan* results

Table 2 shows the average energy efficiency of all tests for all ULs and all hosts. Best values for each host are presented in bold. The study shows that for both hosts the CPU-bound benchmark is more efficient at high ULs, memory-bound benchmark is more efficient at medium ULs and disk-bound benchmark is more efficient at low ULs. Intel host is more efficient than AMD for all ULs and all types of benchmarks. Finally, the high-critic UL (100%) is less efficient than the high-medium UL (87.5%), except for disk-bound executions.

Table 2. Efficiency ($PC \times makespan / 3600 / number\ of\ instances$)

UL	AMD			Intel		
	CPU	mem.	Disk	CPU	mem.	Disk
12.5%	0.801	1.804	**4.106**	0.404	0.407	**0.226**
25.0%	0.422	1.511	4.817	0.191	0.193	0.233
37.5%	0.295	1.429	4.926	0.159	0.226	0.240
50.0%	0.231	1.391	6.147	0.109	**0.185**	0.246
62.5%	0.193	**1.335**	5.801	0.089	0.266	0.255
75.0%	0.168	1.410	5.907	**0.081**	0.216	0.260
87.5%	**0.150**	1.570	6.367	0.083	0.235	0.265
100.0%	0.154	1.656	5.686	0.107	0.238	0.262

5 Conclusions and Future Work

This article presented an empirical analysis of the power consumption of synthetic benchmarks in high-end multicore systems. The main contribution is an exhaustive study of the inter-relationship among the power consumption of the main computing resources at different ULs, over AMD and Intel architectures.

The experimental methodology consisted on executing synthetic benchmarks over high-end hosts connected to a PDU, considering different ULs and combinations, aimed at characterizing the power consumption of each computing resource (CPU, memory, and disk). The operations performed by the benchmarks include mathematical functions and read/write of main memory and disk.

The study was complemented with performance experiments. A total number of 144 experiments were performed, 96 evaluating the power consumption and 48 evaluating the performance. For each experiment, 20 independent executions were performed. Results showed that in single executions, CPU utilization has a linear relation with power consumption. Memory utilization has significant impact on power consumption when compared to CPU usage, up to 157% more EC for AMD and 46% more EC for Intel. On the other hand, disk usage presented low EC variation for all ULs.

Combined executions are able to reduce EC with regard to independent executions manly for CPU and disk combined execution. Efficiency analysis showed

that different benchmarks performed more efficiently at different ULs: CPU at high ULs, memory at medium ULs and disk at low ULs. Critic UL (100%) showed worse efficiency than high-medium UL (87.5%), except for disk.

The main lines for future work are related to extend the power and performance characterization of different benchmarks (including GPU-bound, network-bound, and no-synthetic benchmarks) and other high-end hosts. We are also working on using the characterization to build energy models for evaluating energy-aware scheduling strategies on HPC infrastructures and datacenters.

References

1. Anghel, A., Vasilescu, L., Mariani, G., Jongerius, R., Dittmann, G.: An instrumentation approach for hardware-agnostic software characterization. Int. J. Parallel Prog. **44**(5), 924–948 (2016)
2. Brandolese, C., Corbetta, S., Fornaciari, W.: Software energy estimation based on statistical characterization of intermediate compilation code. In: International Symposium on Low Power Electronics and Design, pp. 333–338 (2011)
3. Buyya, R., Vecchiola, C., Selvi, S.: Mastering Cloud Computing: Foundations and Applications Programming. Morgan Kaufmann, San Francisco (2013)
4. Dayarathna, M., Wen, Y., Fan, R.: Data center energy consumption modeling: a survey. IEEE Commun. Surv. Tutorials **18**(1), 732–794 (2016)
5. Du Bois, K., Schaeps, T., Polfliet, S., Ryckbosch, F., Eeckhout, L.: Sweep: evaluating computer system energy efficiency using synthetic workloads. In: 6th International Conference on High Performance and Embedded Architectures and Compilers, pp. 159–166 (2011)
6. Feng, X., Ge, R., Cameron, K.: Power and energy profiling of scientific applications on distributed systems. In: 19th IEEE International Parallel and Distributed Processing Symposium, pp. 34–44 (2005)
7. Iturriaga, S., García, S., Nesmachnow, S.: An empirical study of the robustness of energy-aware schedulers for high performance computing systems under uncertainty. In: Hernández, G., Barrios Hernández, C.J., Díaz, G., García Garino, C., Nesmachnow, S., Pérez-Acle, T., Storti, M., Vázquez, M. (eds.) CARLA 2014. CCIS, vol. 485, pp. 143–157. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45483-1_11
8. Kopytov, A.: Sysbench repository https://github.com/akopytov/sysbench. Accessed 1 May 2017
9. Kurowski, K., Oleksiak, A., Piątek, W., Piontek, T., Przybyszewski, A., Węglarz, J.: Dcworms-a tool for simulation of energy efficiency in distributed computing infrastructures. Simul. Model. Pract. Theory **39**, 135–151 (2013)
10. Langer, A., Totoni, E., Palekar, U.S., Kalé, L.V.: Energy-efficient computing for HPC workloads on heterogeneous manycore chips. In: Proceedings of the Sixth International Workshop on Programming Models and Applications for Multicores and Manycores, pp. 11–19 (2015)
11. Nesmachnow, S.: Computación científica de alto desempeño en la Facultad de Ingeniería. Universidad de la República. Revista de la Asociación de Ingenieros del Uruguay, **61**(1), pp. 12–15 (2010). Text in Spanish
12. Nesmachnow, S., Dorronsoro, B., Pecero, J., Bouvry, P.: Energy-aware scheduling on multicore heterogeneous grid computing systems. J. Grid Comput. **11**(4), 653–680 (2013)

13. Nesmachnow, S., Perfumo, C., Goiri, I.: Holistic multiobjective planning of data-centers powered by renewable energy. Cluster Comput. **18**(4), 1379–1397 (2015)
14. Srikantaiah, S., Kansal, A., Zhao, F.: Energy aware consolidation for cloud computing. In: Conference on Power Aware Computing and Systems, vol. 10, pp. 1–5 (2008)