A Generic Methodology to Derive Empirical Power Consumption Prediction Models for Multi-core Processors

Robert Basmadjian
University of Passau
94032, Passau – Germany
Email:basmadji@fim-uni.passau.de

Sebastian Rainer
ZF Friedrichshafen AG
94030, Passau – Germany
Email:sebastian.rainer@zf.com

Hermann De Meer
University of Passau
94032, Passau – Germany
Email:demeer@fim-uni.passau.de

***Abstract*—With the emergence of multi-core processors and their P-states, deriving power consumption models expressed in terms of easily extractable parameters becomes a necessity. In this paper, we introduce a generic methodology to devise power consumption estimation models for multi-core processors. The proposed approach takes into account three metrics: The number of active cores (i.e. executing instructions), frequency and utilization rate of the processor. Consequently, the derived models are expressed in terms of the above-mentioned parameters which can be extracted by any monitoring system. In order to assess the correctness as well as the accuracy of the proposed methodology, an experimental analysis is performed on Intel quad- and hexa-core processors. The results confirm the exactitude of the proposed methodology.**

I. Introduction

During the last decade, Information Technology (IT) services (e.g. social networks) have become omnipresent in our daily activities. The provision of such services required the development of an ubiquitous infrastructure with huge datacenters and an ultimately global network as providers of all those ICT needs. Consequently, an enormous growth of the energy use of datacenters has taken place due to the hosting of powerful ICT resources to guarantee the Quality of Service (QoS) of the delivered IT services. Among those resources, servers are the major contributors (i.e. 40% – 50%) to the energy consumption of datacenters [1]. Furthermore, it was shown in [2] and [3] that processors contribute between 23% – 40% to the total server's power drain.

Given the importance of the topic, several prediction models were proposed in the literature in order to estimate the dynamic (not idle) power consumption of processors. These can be classified into two classes: hardware- and software-level. Examples of hardware-level models are [4] (CPU cycles), [5] (CMOS circuits) and [6], [7] (register transfer level) where all these models provide a high level of accuracy. However, monitoring the activities of a processor at low (transistor) level is a tedious task since a processor possesses millions (billions) of transistors and monitoring each transistor is not trivial. To overcome this complexity, software-level models were proposed. The power consumption of the underlying processor is predicted based on the power consumed by each executed instruction [8] or function [9]. One key inconvenience is that software-level models depend upon *tracing tool*s that parse an application to extract all of its constituent instructions or functions. In case these tools are unable to fetch the complete information regarding instructions, software-level models suffer from inaccuracy in power prediction.

From the above-mentioned proposed models of the literature, we conclude that the input parameters of the model as well as the ease with which those can be extracted play a major role in the modeling methodology and concept. To this end, in this paper, we introduce a generic modeling methodology to derive power consumption prediction models for multi-core processors. In short, the proposed methodology consists of the following generic steps:

1) The power consumption of the processor is expressed by varying the utilization rate for a given number of active cores as well as frequency.
2) The power consumption of the processor is given by varying both the utilization rate as well as frequency for a given number of active cores.
3) The generic power consumption of the processer is expressed in terms of the utilization rate, frequency as well as the number of active cores.

From aforementioned steps, we notice that the proposed approach is based on three parameters: number of active (i.e. executing instructions) cores, the frequency as well as the utilization rate of the underlying investigated multi-core processor. Hence, the derived power consumption models are expressed in terms of the above-mentioned three metrics, which can be easily extracted by the monitoring framework of any computing system (e.g. server). The major advantage of the proposed methodology is that we provide power consumption models both at core and processor level. In order to demonstrate the accuracy of the derived models from our proposed methodology, we carry out experimental analysis on Intel quad- and hexa-core processors. The obtained results assess the correctness of the proposed methodology.

The rest of this paper is organized as follows: In Section II, we introduce the relevant steps of deriving power consumption prediction models for multi-core processors using the proposed methodology. In Section III, we present the experimental analysis by first introducing the setup configuration, used workload and modeling tools and then give the obtained results. Section IV presents the related work and the paper is concluded in Section V.

II. Methodological Approach

In this section, we introduce the proposed methodological approach in order to devise power consumption prediction models for multi-core processors. The corresponding approach takes into account the fact that the *scheduler* of the operating system distributes the workload of the processor evenly among the cores. Note that such an assumption is already implemented for Linux operating systems with kernel 2.6 or higher [11].

The considered methodology deems the following metrics:

1) The utilization rate L of the processor expressed in percentage (e.g. 100% denotes full utilization).
2) The frequency $f \in [f_{min}, f_{max}]$ of the processor given in GHz, where f_{min} and f_{max} represent respectively the minimum and maximum frequency specified by means of the P-states.
3) The number of active (i.e. executing instructions) cores n.

Based on the above-mentioned three criteria, the following steps are proposed to derive power consumption estimation models empirically:

Step 1: Compute the power consumption by fixing the number of active cores $n = 1$ as well as frequency $f = f_{min}$ and varying with increments of 10% the utilization rate[1] L. We denote such a model in the form of $P_{n,f}(L)$ such that:

$$P_{1,f_{min}}(L) \leq P_{n,f}(L) \leq P_{t,f_{max}}(L), \tag{1}$$

where t denotes the total number of cores.

Step 2: Repeat *Step 1* for all possible values of frequencies until $f = f_{max}$. Note that when f reaches the maximum frequency, we have as many power consumption models $P_{n,f}(L)$ as the total number of P-states where each such model is expressed in terms of utilization rate L.

Step 3: Transform the one-dimensional models $P_{n,f}(L)$ obtained in *Steps 1 and 2* into a two-dimensional model expressed in terms of frequency f and utilization rate L. We denote such a model in the form of:

$$P_n(f, L). \tag{2}$$

Step 4: Increment n of *Step 1* by one and repeat *Steps 1–4* until the number of active cores n reaches t. Note that when $n = t$, we have as many power consumption models $P_n(f, L)$ as the total number of cores where each such model is expressed in terms of frequency f and utilization rate L.

Step 5: Transform the two-dimensional models $P_n(f, L)$ obtained in Step 4 into a three-dimensional model expressed in terms of number of active cores n, frequency f and utilization rate L. We denote such a model in the form of:

$$P(n, f, L). \tag{3}$$

It is worthwhile to note that the three parameters of Equation (3), to estimate the power consumption of a multi-core processor, can be easily extracted from the monitoring system of any given server.

[1]It is also possible to take any percentage increment. In this paper, we chose increments of 10% since we noticed that such an increment represents a reasonable trend for power consumption behavior.

III. Experimental Analysis

In this section, we present the experimental analysis performed by adopting the methodology of Section II. To this end, we first introduce the setup configuration (e.g. hardware characteristic), used benchmarks and modeling tools. Then we give the outcome of the carried out observations with a detailed accuracy analysis of the proposed approach.

A. Setup Configuration

The observations were performed by considering quad- and hexa-core processors from Intel corporation. The tested quad-core processor is *Xeon L5420* [12] possessing two P-states: P_0 at 2.5 GHz and P_1 at 2.0 GHz. The investigated hexa-core processor is *Xeon X5650* [13] having 9 P-states: P_0 at 2.667 GHz, P_1 at 2.533 GHz, P_2 at 2.4 GHz, P_3 at 2.267 GHz, P_4 at 2.133 GHz, P_5 at 2.0 GHz, P_6 at 1.867 GHz, P_7 at 1.733 GHz and P_8 at 1.6 GHz. Table I gives the hardware characteristics of the corresponding processors.

For monitoring the power consumption of a processor, ZES Zimmer Electronic Systems LMG500 [14] power meter was used. The corresponding device takes 10 samples per second. Each observation ran for 30 minutes without altering the utilization rate and the corresponding *average power* was obtained. The experiments were carried out on Linux Ubuntu 12.04 operating system.

B. Workload Generator

In order to carry out observations on investigated multi-core processors, a custom benchmark was developed. This benchmark generates workload (i.e. utilization rate) simultaneously on multiple cores of a processor by performing simple arithmetic (e.g. addition, multiplication, etc.) and trigonometric (e.g. sine, cosine, etc.) operations, followed by slots of sleeping phases. The ratio of computing instructions to the sleeping period determines the corresponding utilization rate by taking into account the processor's P-state. It is worth pointing out that there exists a synthetic load generator for Linux systems called *lookbusy* [15]. However, the major drawback of such a tool is its lack of ability to generate workload on multiple cores simultaneously. In other words, it generates synthetic load on *all cores* of the processor. Consequently, such a tool does not serve for our modeling purposes, especially the one introduced in Section II.

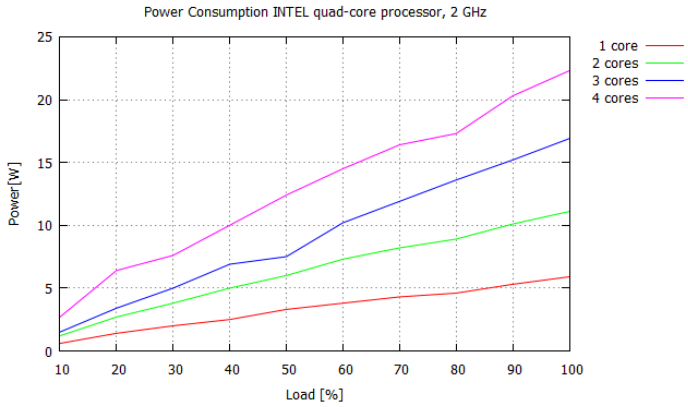
C. Modeling Tools

Numerous observations need to be carried out to satisfy the different steps proposed in Section II. The outcome of all those observations is a huge set of values that relates power consumption to the corresponding conducted steps.

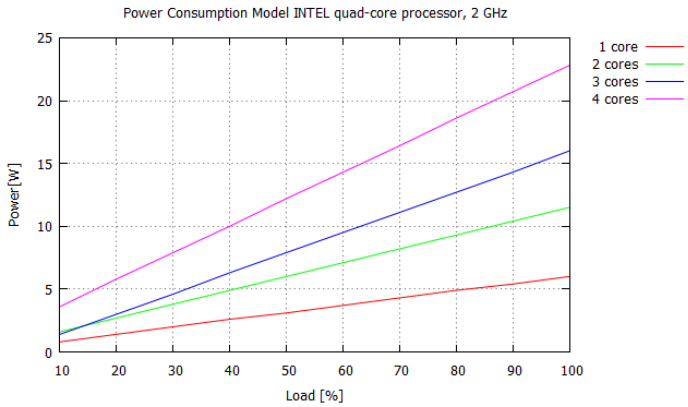
To derive power consumption prediction models, the tool *Gnuplot* [16] was used. To this end, the obtained measured power consumption values that correspond to each step of Section II were provided to this tool which on its turn computed the *best-fitting* functions for Equations (1), (2), and (3). In order to assess the accuracy of the proposed methodology, we used another modeling tool called Eureqa [17]. For this purpose, *all* the obtained measured power consumption values were fed to this tool which on its turn computed the best-fitting overall function.

CPU type	Frequency [GHz]	Cache [MB]	Number of P-states
Intel Xeon L5420 (4 cores)	2.0 (min) – 2.5 (max)	12 (L2)	2
Intel Xeon X5650 (6 cores)	1.6 (min) – 2.667 (max)	12 (L2)	9

TABLE I. INVESTIGATED PROCESSORS' CHARACTERISTICS

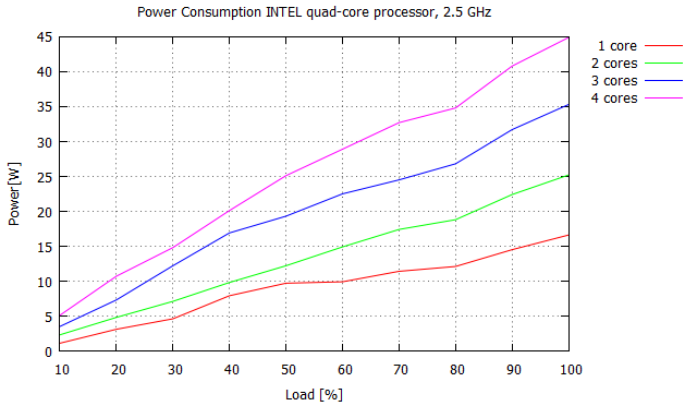


(a) Measured using power meter of [14]

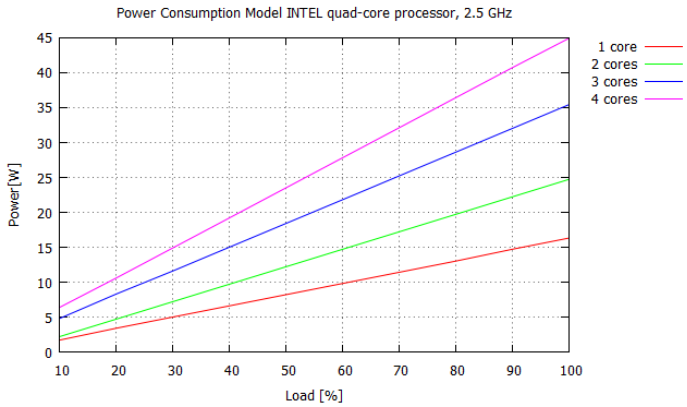


(b) Estimated using Equation (16)

Fig. 1. Power consumption for Intel quad-core processor of 2.0 GHz



(a) Measured using power meter of [14]



(b) Estimated using Equation (16)

Fig. 2. Power consumption for Intel quad-core processor of 2.5 GHz

D. Obtained Results

In this section, we give the derived power consumption prediction models both for Intel quad- and hexa-core processors.

1) Quad-core: By applying Steps 1–3 of Section II, we derive the following models when a single core is active:

$$P_{1,2.0}(L) = 0.05691L + 0.24 \quad (4)$$

$$P_{1,2.5}(L) = 0.16212L + 0.173 \quad (5)$$

$$P_1(f, L) = (0.21f - 0.363)L - 0.133f + 0.506 \quad (6)$$

When two cores are active and by applying Steps 1–3 of Section II, we derive the following models:

$$P_{2,2.0}(L) = 0.107L + 0.519 \quad (7)$$

$$P_{2,2.5}(L) = 0.25L - 0.286 \quad (8)$$

$$P_2(f, L) = (0.286f - 0.464)L - 1.613f + 3.746 \quad (9)$$

By applying Steps 1–3 of Section II, we derive the following models when three cores are active:

$$P_{3,2.0}(L) = 0.17L - 0.186 \quad (10)$$

$$P_{3,2.5}(L) = 0.336L + 1.466 \quad (11)$$

$$P_3(f, L) = (0.332f - 0.493)L + 3.306f - 6.799 \quad (12)$$

When all cores are active and by applying Steps 1–3 of Section II, we derive the following models:

$$P_{4,2.0}(L) = 0.208L + 1.539 \quad (13)$$

$$P_{4,2.5}(L) = 0.43L + 2.106 \quad (14)$$

$$P_4(f, L) = (0.444f - 0.681)L + 1.133f - 0.726 \quad (15)$$

Based on Equations (6), (9), (12) and (15) as well as by taking into account Step 5 of Section II, the following generic power

Active Cores	1	2	3	4
Max error [%]	61	37	39	35
Avg error [%]	9	4	8	6

TABLE II. ERROR ANALYSIS OF EQUATION (16) FOR INTEL QUAD-CORE PROCESSOR

Active Cores	1	2	3	4
Max error [%]	15	5	7	7
Avg error [%]	4	2	5	3

TABLE III. ERROR ANALYSIS OF EQUATION (16) FOR INTEL QUAD-CORE PROCESSOR WITH UTILIZATION RATES ABOVE 50%

consumption prediction model is derived for quad-core Intel processors:

$$\begin{aligned} P(n,f,L) = {} & (0.075nf + 0.131f - 0.098n - 0.255)L \\ & - 2.248n^3f + 16.693n^2f - 35.817nf + 21.239f \\ & + 5.067n^3 - 37.299n^2 + 79.665n - 46.926 \end{aligned} \tag{16}$$

Figs. 1 and 2 illustrate the power consumption of Intel quad-core processor possessing 2.0 GHz and 2.5 GHz of frequency respectively. Comparing the measured power consumption with the estimated one obtained by Equation (16), we highlight the following:

1) In general, 86% of the numbers have an error less than 10%, with an average error of 9% (see Table II).
2) Above 50% of utilization rate, 98% of the numbers have an error less than 10%. Furthermore, the maximum error is about 15% which rises up at 10% of utilization rate when one core is active (see Table III).
3) For all observations, the maximum error is about 61% which rises up at 10% of utilization rate when one core is active (see Table II).

Tables II and III give a detailed analysis of the results obtained from the model of Equation (16).

In order to assess the accuracy of both the model given by Equation (16) as well as the proposed methodology of Section II, we provided all the numbers obtained from all the carried out observations to the Eureqa modeling tool of Section III-C. The resulting model is given by:

$$P(n,f,L) = 2.16f + 0.114nfL - 0.174nL - 3.98 \tag{17}$$

Comparing the measured power consumption with the estimated one obtained by Equation (17), we highlight the following:

1) In general, 72% of the numbers have an error less than 10%, with an average error of 19% (see Table IV).
2) Above 50% of utilization rate, 84% of the numbers have an error less than 10%. Furthermore, the maximum error is about 29% which rises up at 10% of utilization rate when one core is active (see Table V).
3) For all observations, the maximum error is about 128% which rises up at 10% of utilization rate with a single active core (see Table IV).

Active Cores	1	2	3	4
Max error [%]	128	56	36	27
Avg error [%]	19	9	7	6

TABLE IV. ERROR ANALYSIS OF EQUATION (17) FOR INTEL QUAD-CORE PROCESSOR

Active Cores	1	2	3	4
Max error [%]	29	6	13	10
Avg error [%]	12	3	3	4

TABLE V. ERROR ANALYSIS OF EQUATION (17) FOR INTEL QUAD-CORE PROCESSOR WITH UTILIZATION RATES ABOVE 50%

Tables IV and V give a detailed analysis of the results obtained from the model of Equation (17). From the above-mentioned analysis, we can assert the correctness of our proposed modeling methodology and confirm the high accuracy of the derived model of Equation (16).

2) Hexa-core: By applying Steps 1–3 of Section II, we derive the following models when a single core is active:

$$P_{1,f}(L) = xL + y \tag{18}$$

$$\begin{aligned} P_1(f,L) = {} & (-0.003f^3 + 0.039f^2 - 0.092f + 0.184)L \\ & - 0.27f^3 + 1.456f^2 - 2.562f + 1.278 \end{aligned} \tag{19}$$

where f, x, and y of Equation (18) are given in Table VI.

When two cores are active and by applying Steps 1–4 of Section II, we derive the following models:

$$P_{2,f}(L) = xL + y \tag{20}$$

$$\begin{aligned} P_2(f,L) = {} & (0.1f^3 - 0.583f^2 + 1.175f - 0.646)L \\ & - 5.718f^3 + 36.515f^2 - 78.154f + 56.992 \end{aligned} \tag{21}$$

where f, x, and y of Equation (20) are given in Table VII.

f	x	y
1.6	0.127	-0.193
1.733	0.129	-0.259
1.867	0.131	-0.0733
2.0	0.137	-0.259
2.133	0.145	-0.519
2.267	0.142	0.233
2.4	0.153	-0.233
2.533	0.164	-0.579
2.667	0.167	0.239

TABLE VI. PARAMETERS OF EQUATION (18) WHEN ONE CORE IS ACTIVE

f	x	y
1.6	0.155	1.973
1.733	0.147	1.599
1.867	0.169	0.993
2.0	0.175	0.833
2.133	0.176	1.246
2.267	0.185	0.926
2.4	0.197	0.36
2.533	0.209	0.573
2.667	0.239	-0.219

TABLE VII. PARAMETERS OF EQUATION (20) WHEN TWO CORES ARE ACTIVE

f	x	y
1.6	0.188	2.246
1.733	0.18	2.073
1.867	0.194	1.886
2.0	0.223	0.266
2.133	0.248	-0.36
2.267	0.264	-0.82
2.4	0.291	-1.219
2.533	0.289	-0.413
2.667	0.295	0.833

TABLE VIII. PARAMETERS OF EQUATION (22) WHEN THREE CORES ARE ACTIVE

f	x	y
1.6	0.235	0.5
1.733	0.227	1.32
1.867	0.254	0.16
2.0	0.268	1.146
2.133	0.296	-0.326
2.267	0.322	0.026
2.4	0.355	-0.633
2.533	0.372	-0.306
2.667	0.394	-0.453

TABLE IX. PARAMETERS OF EQUATION (24) WHEN FOUR CORES ARE ACTIVE

By applying Steps 1–3 of Section II, we derive the following models when three cores are active:

$$P_{3,f}(L) = xL + y \tag{22}$$

$$P_3(f,L) = (-0.321f^3 + 2.035f^2 - 4.105f + 2.86)L + 16.936f^3 - 101.756f^2 + 196.821f - 121.5 \tag{23}$$

where f, x, and y of Equation (22) are given in Table VIII.

When four cores are active and by applying Steps 1–3 of Section II, we derive the following models:

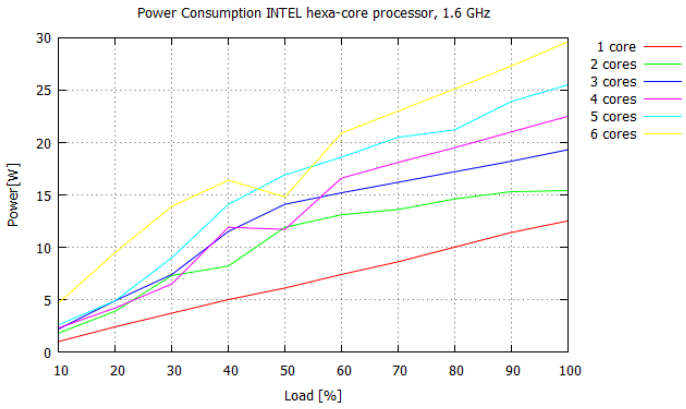
$$P_{4,f}(L) = xL + y \tag{24}$$

f	x	y
1.6	0.255	1.686
1.733	0.276	1.526
1.867	0.316	-0.579
2.0	0.319	1.133
2.133	0.358	0.479
2.267	0.394	-0.373
2.4	0.419	0.079
2.533	0.454	-0.299
2.667	0.496	-1.239

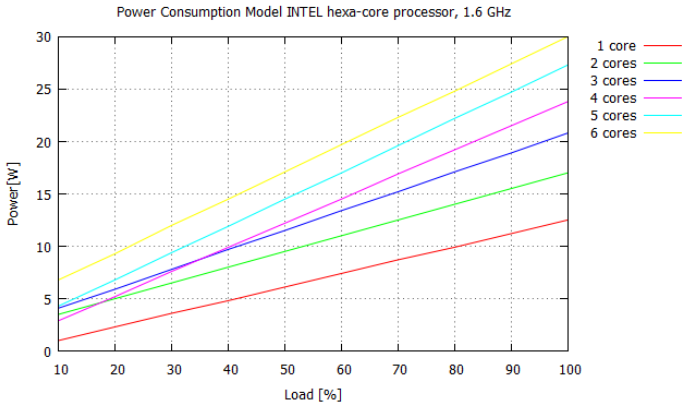
TABLE X. PARAMETERS OF EQUATION (26) WHEN FIVE CORES ARE ACTIVE

f	x	y
1.6	0.26	4.16
1.733	0.29	3.84
1.867	0.33	2.99
2.0	0.36	2.89
2.133	0.38	3.74
2.267	0.43	2.65
2.4	0.46	3.06
2.533	0.48	3.54
2.667	0.54	2.91

TABLE XI. PARAMETERS OF EQUATION (28) WHEN SIX CORES ARE ACTIVE



(a) Measured using power meter of [14]



(b) Estimated using Equation (30)

Fig. 3. Power consumption for Intel hexa-core processor of 1.6 GHz

$$P_4(f,L) = (-0.207f^3 + 1.39f^2 - 2.882f + 2.137)L + 6.524f^3 - 41.83f^2 + 86.606f - 57.557 \tag{25}$$

where f, x, and y of Equation (24) are given in Table IX.

By applying Steps 1–3 of Section II, we derive the following models when five cores are active:

$$P_{5,f}(L) = xL + y \tag{26}$$

$$P_5(f,L) = (0.046f - 0.237f^2 + 0.59f - 0.272)L - 8.229f^3 + 52.8332 - 113.5f + 81.838 \tag{27}$$

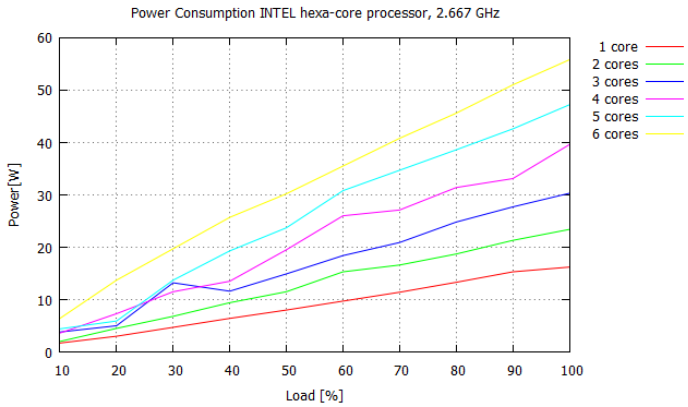
where f, x, and y of Equation (26) are given in Table X.

When all cores are active and by applying Steps 1–3 of Section II, we derive the following models:

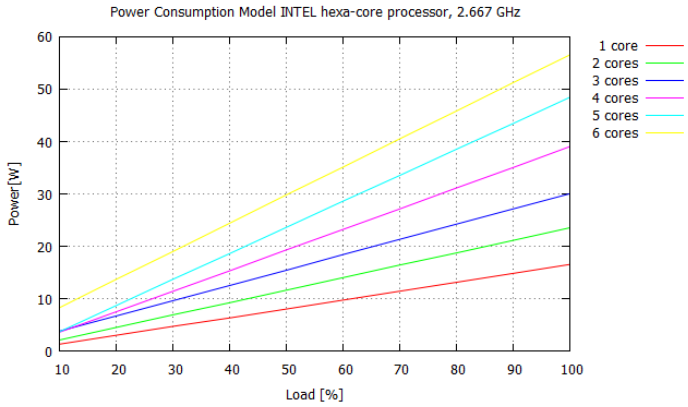
$$P_{6,f}(L) = xL + y \tag{28}$$

$$P_6(f,L) = (0.095f - 0.583f^2 + 1.42f - 0.91)L - 5.022f^3 + 34.103f^2 - 76.636f + 60.118 \tag{29}$$

where f, x, and y of Equation (28) are given in Table XI.



(a) Measured using power meter of [14]



(b) Estimated using Equation (30)

Fig. 4. Power consumption for Intel hexa-core processor of 2.667 GHz

Based on Equations (19), (21), (23), (25), (27) and (29) as well as by taking into account Step 5 of Section II, the following generic power consumption prediction model is derived for hexa-core Intel processors:

$$\begin{aligned} P(n,f,L) = {} & ((0.000084n^7 - 0.0099n^5 + 0.679n^3 - 3.523n^2 \\ & + 6.217n - 3.366)f^3 + (-0.00049n^7 - 0.0593n^5 \\ & - 4.091n^3 + 21.313n^2 - 37.697n + 20.457)f^2 \\ & + (0.00095n^7 - 0.114n^5 + 8.017n^3 - 41.966n^2 \\ & + 74.48n - 40.51)f - 0.00059n^7 + 0.0723n^5 \\ & - 5.122n^3 + 26.969n^2 - 48.051n + 26.316)L \\ & + (-0.00536n^7 + 0.625n^5 - 41.719n^3 \\ & + 213.259n^2 - 371.895n + 199.464)f^3 \\ & + (0.03n^7 - 3.645n^5 + 246.776n^3 - 1268.33n^2 \\ & + 2221.98n - 1195.34)f^2 + (-0.057n^7 \\ & + 6.861n^5 - 471.49n^3 + 2441.85n^2 - 4306.16n \\ & + 2326.43)f + 0.0335n^7 - 4.134n^5 + 291.173n \\ & - 1524.39n^2 + 2714.61n - 1476 \end{aligned} \quad (30)$$

Figs. 3 and 4 illustrate the power consumption of Intel hexa-core processor possessing 1.6 GHz and 2.667 GHz of frequency respectively. Comparing the measured power consumption with the estimated one obtained by Equation (30), we highlight the following:

1) In general, 81% of the numbers have an error less than 10%, with an average error of 10% (see Table XII).
2) Above 50% of utilization rate, 91% of the numbers have an error less than 10%. Furthermore, the maximum error is about 20% which rises up at 10% of utilization rate when two cores are active (see Table XIII).
3) For all observations, the maximum error is about 106% which rises up at 10% of utilization rate when four cores are active Table XII).

Active Cores	1	2	3	4	5	6
Max error [%]	30	91	87	106	66	63
Avg error [%]	4	10	10	8	8	7

TABLE XII. ERROR ANALYSIS OF EQUATION (30) FOR INTEL HEXA-CORE PROCESSOR

Active Cores	1	2	3	4	5	6
Max error [%]	15	20	18	12	15	16
Avg error [%]	2	5	4	3	4	1

TABLE XIII. ERROR ANALYSIS OF EQUATION (30) FOR INTEL HEXA-CORE PROCESSOR WITH UTILIZATION RATES ABOVE 50%

Tables XII and XIII give a detailed analysis of the results obtained from the model of Equation (30).

In order to assess the accuracy of both the model given in Equation (30) as well as the proposed methodology of Section II, we provided all the numbers obtained from all the carried out observations to the Eureqa modeling tool of Section III-C. The resulting model is given by:

$$\begin{aligned} P(n,f,L) = {} & 19.7 + 3.19f^2 + 0.175n^2 + 0.00077L^2 \\ & + 0.042nfL - 1.56n - 15f - 0.000329nL^2 \end{aligned} \quad (31)$$

Comparing the measured power consumption with the estimated one obtained by Equation (31), we highlight the following:

1) In general, 74% of the numbers have an error less than 10%, with an average error of 19% (see Table XIV).
2) Above 50% of utilization rate, 91% of the numbers have an error less than 10%. Moreover, the maximum error is about 29% which rises up at 10% of utilization rate when two cores are active (see Table XV).
3) For all observations, the maximum error is about 201% which rises up at 10% of utilization rate when one core is active (see Table XIV).

Active Cores	1	2	3	4	5	6
Max error [%]	201	53	40	74	65	25
Avg error [%]	19	10	8	8	8	4

TABLE XIV. ERROR ANALYSIS OF EQUATION (31) FOR INTEL HEXA-CORE PROCESSOR

Tables XIV and XV give a detailed analysis of the results obtained from the model of Equation (31).

From the aforementioned analysis, we can assert the correctness of our proposed modeling methodology and confirm

Active Cores	1	2	3	4	5	6
Max error [%]	22	29	27	13	10	21
Avg error [%]	5	7	5	4	3	2

TABLE XV. ERROR ANALYSIS OF EQUATION (31) FOR INTEL HEXA-CORE PROCESSOR WITH UTILIZATION RATES ABOVE 50%

the high accuracy of the derived models. Also, significant errors are noticeable at low utilization rates (e.g. 10% with one active core), whereas for high utilization rates (i.e. above 50%) the derived models have an inaccuracy of at most 10%. Such an error rate for power consumption prediction models is reasonable given the fact that in current datacenters, most of the time the servers are highly utilized due to virtualization and consolidation concepts.

IV. RELATED WORK

In literature, several approaches were proposed to compute the power consumption of processors which can be classified into two classes: hardware- and software-level.

Computing the power consumption directly at the hardware-level is realized either by measuring the CPU-cycles [4], or the circuit [5] or even at the register-transfer-level [6]. Approaches [19] and [4] take into account the CPU-cycles where the former concentrates on each functional unit (e.g. adders, ALU, shifter, register files) and the latter on the activity factor and the capacitance. Furthermore, [19] has power tables and technology dependent switch capacitance tables for the different functional units, whereas [4] tries to model the activity factor and the capacitance based on circuits and transistor sizes. Both approaches need deep information of the underlying hardware and are not independent of the used transistors (respectively processors). Looking at the circuit-level, the authors of [5] propose to simulate current and power in VLSI circuits, such as CMOS circuits. This event driven approach furthermore models computations on this circuit and estimates the power consumption based on these computations. Again, these power consumption models are focused on one special circuit with regard to the modeled computations. Hence, information of the underlying hardware and the effects of the carried out computations need to be taken into account. The authors of [6] propose an approach at the register-transfer-level which tries to minimize the power consumption of CMOS circuits. To this end, the power consumption is computed directly at the circuit-level. To minimize and optimize the power consumption, specific characteristics of the circuits are exploited, e.g. the threshold voltage of the transistor. Consequently, for this approach detailed information of the hardware is needed, e.g. the voltage. All these hardware-based models require the monitoring of low level activities directly at the processor, e.g. at transistor-level. Furthermore, they all need detailed information of the transistors, e.g. the voltage. These two key requirements lead to a large complexity, since modern processors consist of billions of transistors and the information needed for these models are complex to fetch or even not available. Whereas these facts are a major drawback of such models, the advantage is a high accuracy since the models are directly at the hardware level.

Going one step further in abstracting from the hardware-level, software-level power consumption models were proposed. These models abstract from the underlying hardware and predict the power consumption based on executed instructions [20], [8], [21], and [22] or functions [9] and [23]. The authors of [23] follow the macro-modeling approach. This approach needs an accurate lower-level power model of the processor. Taking this lower-level model, a macro-model for the executed function is constructed. This constructed macro-model is made up of various parameters which form the complexity of the executed functions. In [9], a power database per core is used. Such a power database stores the power information of the built in library functions and basic instructions per core. Using a tracing tool, execution information of the investigated functions are gathered. Together with the power database, the average power consumption per function can be estimated. The approach of [22] investigates power consumption models on dual-core mobile processors. Again, a tracing tool is needed to find an energy profiling report of the investigated software. Consequently, the power distribution on each core is modeled. In [21] and [20], an instruction-level power estimation is presented based on current measurement. Both approaches need tracing tools that give information of the executed instructions. Furthermore, they need to measure the current directly at the processor level. Whereas [20] performs an infinite loop of instructions and takes the average of the measured power as the cost per instruction, [21] also takes into account inter-instruction effects. [8] also starts with the investigation of the current when performing an infinite loop with instructions. Furthermore, this approach also presents statistical analysis of the measured data. Summarizing the software-level models, the complexity of the above mentioned hardware-level models is reduced. But this advantage also leads to two major drawbacks. These software-level models are much more inaccurate than the hardware-level ones. Furthermore, such models depend on tracing tools which determine all instructions and functions of an application. If such a tracing tool is unable to determine all information regarding the instructions and functions, such models cannot deliver an accurate power consumption estimation.

Also in the literature, there were proposed models which are based on performance monitoring counters such as [24] [25] and [26]. Performance counters are registers built in processors that store the amount of hardware-level activities within the processor. Such activities can be monitored by event counters [27]. [25] uses the approach of [4] and performance counters to estimate the power consumption on the fly. Since the available performance counters do not reflect the needed power relevant performance counters, heuristics are used to compute the relevant counters from the existing performance counters. A more granular approach is presented in [24] and [26]. This approach defines power components within the processor. Each component has its own performance counters. Using microbenchmarks, activity ratios for each power component are computed, taking into account the mentioned performance counters at each component. In the next step, a weight is assigned to each power component. Hence, the overall power consumption is estimated by the weighted sum of all activity ratios of the several power components. The major drawback of models based on performance monitoring counters is that they are unable to provide all the information needed to get an accurate power consumption prediction since processors do not allow to monitor of all events.

To overcome all the drawbacks arising from the existing

models, namely the dependency on performance monitoring counters and tracing tools, and furthermore to avoid the complexity of the hardware-based models, in this paper we propose power consumption prediction models that are only based on the frequency, the utilization rate and the number of loaded cores. Those three parameters can be easily extracted from the monitoring system of any computing system, which is the main advantage of our model over the aforementioned ones.

V. Conclusion

Energy-efficient computing is becoming more and more relevant for datacenters due to their high energy usage and hence increasing energy costs. Energy-aware optimization policies have been proposed recently, to save energy usage of datacenters, that have the aim of consolidating workloads and turning off unutilized resources (e.g. idle servers) [18]. However, in order that such policies can take the most suitable decisions, they need to be guided by accurate models that estimate the power consumption of servers.

In this paper, we presented a generic methodology in order to derive empirical power consumption prediction models for multi-core processors. Based on the proposed methodology, the devised models were expressed in terms of three parameters that can be easily extracted from any monitoring system: the number of active cores, frequency and utilization rate. In order to assess the accuracy of our approach, an experimental analysis was carried out on Intel quad- and hexa-core processors. The results showed correctness of the approach as well as the high accuracy of the corresponding derived models.

Among the lessons learnt, we highlight the following:

1) As we previously claimed in [10], we reconfirm the fact that the power consumption of a multi-core processor is not the pure summation of the power consumption of its constituent cores.
2) The power consumption at core- or processor-level is almost linearly dependent on the utilization rate for any given frequency.
3) The power consumption is not linearly dependent on the frequency for any given utilization rate.

As a future perspective, it would be interesting to introduce a new methodology to devise power consumption models for the case where scheduler of the operating system does not fairly distribute the workload among the cores of the processor.

Acknowledgment

The research leading to these results was supported by the European Community's 7^{th} Framework Programme in the context of the ALL4Green project.

References

[1] G. Ghatikar, M.A. Piette, S.Fujita, A. McKane, J.H. Dudley and A. Radspieler, *Demand Response and Open Automated Demand Response Opportunities for Data Centers*, Lawrence Berkeley National Laboratory, 2010.

[2] X. Fan, W.D. Weber, and L.A. Barroso, *Power provisioning for a warehouse-sized computer*, In Proc. of the 34th annual Int'l Symposium on Computer Architecture, pp. 13–23, 2007.

[3] C. Lefurgy, K. Rajamani, F. Rawson, W. Felter, M. Kistler, and T.W. Keller, *Energy management for commercial servers*, Computer Journal, 36(12), pp. 39–48, 2003.

[4] D. Brooks, V. Tiwari, and M. Martonosi, *Wattch: a framework for architectural-level power analysis and optimizations*, In Proc. of the 27th Int'l Symp. on Computer Architecture, pp. 83–94, 2000.

[5] C.X. Huang, B. Zhang, A.C. Deng, and B. Swirski, *The design and implementation of PowerMill*, In Proc. of the Int'l Symp. on on Low Power Design , pp. 105–110, 1995.

[6] A.P. Chandrakasan and R.W. Brodersen, *Minimizing power consumption in digital CMOS circuits*, In Proc. of the IEEE, 83(4), pp 498–523, 1995.

[7] C.T. Hsieh, Q. Wu, C.S. Ding and M. Pedram, *Statistical sampling and regression analysis for RT-Level power evaluation*, In Proc. of Int'l Conf. on Computer-Aided Design, pp 583–588, 1996.

[8] J.T. Russell and M.F. Jacome, *Software power estimation and optimization for high performance, 32-bit embedded processors*, In Proc. of Int'l Conf. on Computer-Aided Design, pp 328–333, 1998.

[9] G. Qu and N. Kawabe, K. Usami, and M. Potkonjak, *Function-level power estimation methodology for microprocessors*, In Proc. of Design Automation Conference, pp810–813, 2000.

[10] R. Basmadjian and H. De Meer, *Evaluating and Modeling Power Consumption of Multi-Core Processors*, In Proc. of 3rd Int'l Conf. on Future Energy Systems (e-Energy 2012), 2012.

[11] http://www.ibm.com/developerworks/linux/library/l-completely-fair-scheduler/

[12] http://ark.intel.com/products/33929/Intel-Xeon-Processor-L5420-12M-Cache-2 50-GHz-1333-MHz-FSB

[13] http://ark.intel.com/products/47922/Intel-Xeon-Processor-X5650-12M-Cache-2 66-GHz-6_40-GTs-Intel-QPI

[14] http://www.zes.com/english/products/one-to-eight-channel-precision-power-ana html

[15] http://www.devin.com/lookbusy/

[16] http://www.gnuplot.info/

[17] http://creativemachines.cornell.edu/eureqa

[18] C. Dupont, G. Giuliani, F. Hermenier, T. Schulze, and A. Somov. *An Energy Aware Framework for Virtual Machine Placement in Cloud Federated Data Centres*, In Proceedings of the 3rd International Conference on Energy-Efficient Computing and Networking (e-Energy'12), ACM, Madrid, Spain, May 7–9, 2012.

[19] W. Ye, N. Vijaykrishnan, M. Kandemir, and M.J. Irwin. *The Design and Use of SimplePower: A Cycle-Accurate Energy Estimation Tool*, , 2000.

[20] V. Tiwari, S. Malik, and A. Wolfe. *Power Analysis of Embedded Software: A First Step Towards Software Power Minimization*, IEEE Transactions on VLSI Systems, 2:437–445, 1994.

[21] M. Lee, V. Tiwari, S. Malik, and M. Fujita. *Power Analysis and Minimization Techniques for Embedded DSP Software*, 1996.

[22] C-H. Hsu, J.J. Chen, and S.L. Tsao. *Evaluation and modeling of power consumption of a heterogeneous dual-core processor*, In Proceedings of the 13th International Conference on Parallel and Distributed Systems, ICPADS '07, pages 1–8, IEEE Computer Society, 2007.

[23] T.K. Tan, A. Raghunathan, G. Lakshminarayana, and N.K. Jha. *High-level Software Energy Macro-modeling*, In In Proc. Design Automation Conf,pages 605–610, 2001.

[24] R. Bertran, M. Gonzalez, X. Martorell, N. Navarro, and E. Ayguade. *Decomposable and responsive power models for multicore processors using performance counters*, In Proceedings of the 24th ACM International Conference on Supercomputing, ICS '10, pages 147– 158, 2010.

[25] R. Joseph and M. Martonosi. *Run-time Power Estimation in High-Performance Microprocessors*, , 2001.

[26] R. Bertran, M. Gonzalez, X. Martorell, N. Navarro, and E. Ayguade. *A Systematic Methodology to Generate Decomposable and Responsive Power Models for CMPs*, IEEE Transactions on Computers, 2012.

[27] R. Berrendorf and H. Ziegler. *PCL - The Performance Counter Library: a common interface to access hardware performance counters on microprocessors*, Technical Report FZJ-ZAM-IB-9816, Forschungszentrum Juelich, 1998.