Improving the Accuracy of Energy Predictive Models for Multicore CPUs Using *Additivity* of Performance Monitoring Counters

Arsalan Shahid(✉), Muhammad Fahad, Ravi Reddy Manumachu, and Alexey Lastovetsky

School of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland
{arsalan.shahid,muhammad.fahad}@ucdconnect.ie,
{ravi.manumachu,alexey.lastovetsky}@ucd.ie

Abstract. Energy predictive modelling using performance monitoring counters (PMCs) has emerged as the leading mainstream approach for modelling the energy consumption of an application. Modern computing platforms such as multicore CPUs provide a large set of PMCs. The programmers, however, can obtain only a small number of PMCs (typically 3–4) during an application run due to the limited number of hardware registers dedicated to storing them. Therefore, selection of a reliable subset of PMCs as predictor variables is crucial to the prediction accuracy of online energy models. State-of-the-art methods for selecting the PMCs are largely based on their correlation with energy consumption.

Recently, *Additivity* is introduced as a property of PMCs that appears to have significant impact on the accuracy of energy predictive models. It is based on an experimental observation that energy consumption of serial execution of two applications is equal to the sum of the energy consumption of those applications when they are run separately. In this work, we demonstrate how the accuracy of energy predictive models based on three popular techniques (Linear regression, Random forests, and Neural networks) can be improved by selecting PMCs based on a property of *additivity*.

Keywords: Performance monitoring counters · Energy consumption · Energy modelling · Multicore CPU · Energy predictive models

1 Introduction

Energy is now a first-class design constraint along with performance in all computing settings. It is a critical limitation for battery-operated mobile systems. Energy-proportional designs [1] in servers are crucial to the operational efficiency of data centres. According to a 2010 DOE Office of Science report [3], it is the leading concern for High Performance Computing (HPC) system designs.

V. Malyshkin (Ed.): PaCT 2019, LNCS 11657, pp. 51–66, 2019.
https://doi.org/10.1007/978-3-030-25636-4_5

Energy consumption in computing contributes nearly 3% to the overall carbon footprint and is now a serious environmental concern [24].

Energy efficiency in computing is driven by innovations in hardware represented by the micro-architectural and chip-design advancements, and software that can be grouped into two categories: (a). System-level energy optimization, and (b). Application-level energy optimization. System-level optimization methods aim to maximize energy efficiency of the environment where the applications are executed using techniques such as DVFS (dynamic voltage and frequency scaling), Dynamic Power Management (DPM), and energy-aware scheduling. Application-level optimization methods use application-level parameters and models to maximize the energy efficiency of the applications.

Accurate measurement of energy consumption during an application execution is key to energy minimization techniques at software level. There are three popular approaches to providing it: (a). System-level physical measurements using external power meters, (b). Measurements using on-chip power sensors, and (c). Energy predictive models.

While the first approach is known to be accurate, it can only provide the measurement at a computer level and therefore lacks the ability to provide fine-grained component-level decomposition of the energy consumption of an application. This is a serious drawback. Consider, for example, a computer consisting of a multicore CPU and an accelerator (GPU or Xeon Phi), which is representative of nodes in modern supercomputers. While it is easy to determine the total energy consumption of a hybrid application run that utilizes both the processing elements (CPU and accelerator) using the first approach, it is difficult to determine their individual contributions. This decomposition is critical to energy models, which are key inputs to data partitioning algorithms that are critical building blocks for optimization of the application for energy. Without the ability to determine accurate decomposition of the total energy consumption, one has to employ an exhaustive approach (involving huge computational complexity) to determine the optimal data partitioning that optimizes the application for energy.

The second approach has no definitive research works proving its accuracy.

The third approach of energy predictive modelling emerged as the pre-eminent alternative. The existing models predominantly use performance monitoring counters as predictor variables for modelling energy consumption. Performance monitoring counters are special-purpose registers provided in modern microprocessors to store the counts of software and hardware activities. We will use the acronym PMCs to refer to software events, which are pure kernel-level counters such as *page-faults*, *context-switches*, etc. as well as micro-architectural events originating from the processor and its performance monitoring unit called the hardware events such as *cache-misses*, *branch-instructions*, etc. They have been developed primarily to aid low-level performance analysis and tuning. While remarkably PMCs have not been used for performance modelling, they have been speedily adopted for energy predictive modelling and have come to dominate its landscape over the years. The energy predictive models are, however, trained

and validated using system-level physical measurements of energy consumptions of the training and test applications. The most common approach proposing an energy predictive model is to determine the energy consumption of a hardware component based on linear regression of the performance events occurring in the hardware component during an application run. The total energy consumption is then calculated as the sum of these individual energy consumptions. Therefore, this approach constructs component-level models of energy consumption and composes them using summation to predict the energy consumption during an application run.

We focus in this work on energy predictive modelling using PMCs. Modern computing platforms such as multicore CPUs provide a large set of PMCs. The most popular tools that can be used to gather the values of the PMCs for a platform include Likwid [25], PAPI [18], Intel PCM [11], and Linux *perf* [19]. The programmers, however, can obtain only a small number of PMCs (typically 3–4) during an application run due to the limited number of hardware registers dedicated to storing them. Consider, for example, the Intel Haswell server whose specification is shown in Table 1. *Likwid* tool provides 167 PMCs for this platform. To obtain the values of the PMCs for an application, the application must be executed about 53 times since only a limited number of PMCs can be obtained in a single application run.

Table 1. Specification of the Intel Haswell and Intel Skylake multicore CPUs

Technical Specifications	Intel Haswell Server	Intel Skylake Server
Processor	Intel E5-2670 v3 @2.30 GHz	Intel Xeon Gold 6152
OS	CentOS 7	Ubuntu 16.04 LTS
Micro-architecture	Haswell	Skylake
Thread(s) per core	2	2
Cores per socket	12	22
Socket(s)	2	1
NUMA node(s)	2	1
L1d cache/L11 cache	32 KB/32 KB	32 KB/32 KB
L2 cache	256 KB	1024 KB
L3 cache	30720 KB	30976 KB
Main memory	64 GB DDR4	96 GB DDR4
TDP	240 W	140 W
Idle Power	58 W	32 W

Since only 3–4 PMCs can be collected in a single application run, selecting such a reliable subset as predictor variables is crucial to the prediction accuracy of online energy models.

We classify techniques for selecting the PMCs into following four categories:

- Techniques that consider all the PMCs offered by a tool for a platform with the goal to capture all possible contributors to energy consumption. To the best of our knowledge, we found no research works that adopt this approach.
- Techniques that are based on a statistical methodology such as correlation, principal component analysis (PCA) etc. [15,28].
- Techniques that use expert advice or intuition to pick a subset (that may not necessarily be determined in one application run) and that, in experts' opinion, is a dominant contributor to energy consumption [8].
- Techniques that select parameters with physical significance based on fundamental laws such as energy conservation of computing [21].

Shahid et al. [21] introduced a new property of PMCs that appear to have significant impact on the accuracy of energy predictive models. It is based on an experimental observation that dynamic energy consumption of serial execution of two applications is equal to the sum of the dynamic energy consumption of those applications when they are run separately. The property, therefore, is based on a simple and intuitive rule that if the parameter is intended for a linear predictive model, the value of a PMC for a serial execution of two applications should be equal to the sum of its values obtained for the individual execution of each application. The PMC is branded as *non-additive* on a platform if there exists an application for which the calculated value differs significantly from the value observed for the application execution on the platform. The use of *non-additive* PMCs in a model impairs its prediction accuracy. The authors show by employing a detailed statistical experimental methodology on a modern Intel Haswell multicore server CPU that while many PMCs are potentially *additive*, a considerable number of PMCs are not. Some of the *non-additive* PMCs are widely used in energy predictive models as key predictor variables.

In this work, we study how the criterion of *additivity* can be used to select PMCs to improve the accuracy of the following types of models: Linear regression (*LR*), Random forests (*RF*), and Neural networks (*NN*). We observe that a large number of energy predictive models in the literature (Sect. 3) is based on these three methods. In a linear regression, we solve a linear model by estimating the regression coefficients. The *RF* is a decision tree based non-linear model build by constructing many linear boundaries. A linear transfer function is used to train our *NN*. *Additivity* property has been envisioned to be useful for selection of PMCs to use as predictor variables in linear energy predictive models. In this paper, we first validate it using detailed experimental evaluation on two modern multicore platforms: (1). Intel Haswell and (2). Intel Skylake. We further investigate its applicability on non-linear modelling techniques such as *RF* and *NN*. We analyze these techniques in terms of the PMCs employed in them and make sure that they appear as *additive* linear parameters. We demonstrate that *additivity* is highly applicable to non-linear methods that employ linear functions for composition of models.

We perform three classes of experiments: Class A, Class B, and Class C. For Class A, we use a dual-socket Intel Haswell multicore server (Table 1).

We select six PMCs which are common in the state-of-the-art models [4,8,14,27] and which are highly correlated with dynamic energy consumption. We build three sets of models. The first set, ($\{LR1, LR2, ..., LR6\}$, contains linear regression models (LRS). The second set, $\{RF1, RF2, ..., RF6\}$, contains random forest models (RFS). The third set, $\{NN1, NN2, ..., NN6\}$, contains neural network models (NNS). In each set, the models contain decreasing number of *non-additive* PMCs. Consider, for example, the first set. Model LR1 employs all the selected PMCs as predictor variables. Model LR2 is based on five most *additive* PMCs. Model LR3 uses four most *additive* PMCs and so on until Model LR6 containing the highest *additive* PMC.

The predictions of the models are compared with system-level physical measurements using power meters ([9]), which we consider to be the ground truth. Our results show that the removal of *non-additive* PMCs improves the average prediction accuracy of LR from 31.2% to 18.01%. Similarly, the average prediction accuracy for RF is improved from 38% to 24%, and for NN from 30% to 24%.

We find no PMC to be *additive* for all categories of applications within a tolerance of 5%. For Class B and Class C experiments, we use a single-socket Intel Skylake server (Table 1) to study the application specific energy predictive models. We choose two highly optimized scientific kernels offered by Intel math kernel library (MKL): (a). Fast Fourier transform (FFT) and (b). Dense matrix-matrix multiplication application (DGEMM). We identify a set of nine most *additive* PMCs (PA) common for both the applications and a set of nine PMCs that are *non-additive* (PNA) but which are used in state-of-the-art energy predictive models. For Class B, we build three models, {LR-A,RF-A,NN-A}, based on PA and three models, {LR-NA,RF-NA,NN-NA}, based on PNA. We show that the models based on PA demonstrate notably better prediction accuracy.

For Class C, since only four PMCs can be collected in a single application run, we compose two sets of PMCs, $PA4$ and $PNA4$. $PA4$ contains four highly energy correlated PMCs selected from PA, and $PNA4$ contains four most correlated PMCs selected from PNA. Models that use $PA4$ demonstrate noteworthy improvement in average prediction accuracy in comparison with models composed using $PNA4$. We also observed that higher correlation with energy when applied to *non-additive* PMCs does not improve their prediction accuracy. The models based on $PNA4$ perform even worse than those based on PNA.

We conclude, therefore, that correlation with dynamic energy consumption alone is not sufficient to provide good prediction accuracy but should be combined with methods such as *additivity* that take into account the physical significance of the parameters originating from fundamental laws such as energy conservation of computing.

To summarize, the main contribution of this work is a study of the impact of *additivity* on the accuracy of mainstream PMCs-based energy predictive modelling techniques.

The rest of this paper is organized as follows. Section 2 present the terminology related to power and energy followed by related work in Sect. 3. Section 4

explains the *additivity* criterion of PMCs and its implications for energy predictive models. In Sect. 5, we present our experimental methodology including setup and design of the three classes of experiments. Section 5 presents the experimental results. Finally, Sect. 6 concludes the paper.

2 Terminologies

There are two types of power consumptions in a component: dynamic power and static power. Dynamic power consumption is caused by the switching activity in the component's circuits. Static power or idle power is the power consumed when the component is not active or doing work. From an application point of view, we define dynamic and static power consumption as the power consumption of the whole system with and without the given application execution. From the component point of view, we define dynamic and static power consumption of the component as the power consumption of the component with and without the given application utilizing the component during its execution.

There are two types of energy consumptions, static energy and dynamic energy. We define the static energy consumption as the energy consumption of the platform without the given application execution. Dynamic energy consumption is calculated by subtracting this static energy consumption from the total energy consumption of the platform during the given application execution. If P_S is the static power consumption of the platform, E_T is the total energy consumption of the platform during the execution of an application, which takes T_E seconds, then the dynamic energy E_D can be calculated as, $E_D = E_T - (P_S \times T_E)$.

In this work, we consider only the dynamic energy consumption. We describe the rationale behind using dynamic energy consumption in the section 1 of supplemental [22].

3 Related Work

This section presents a brief literature survey of some important tools widely used to obtain PMCs, notable research on energy predictive models, and research works that provide a critical review of PMCs.

Tools to obtain PMCs. Perf [19] can be used to gather the PMCs for CPUs in Linux. PAPI [18] and Likwid [25] allow obtaining PMCs for Intel and AMD microprocessors. *Intel PCM* [11] gives PMCs of core and uncore components of an Intel processor.

Notable Energy Predictive Models for CPUs. Initial Models correlating PMCs to energy values include [6,10,12,13]. Events such as integer operations, floating-point operations, memory requests due to cache misses, component access rates, instructions per cycle (IPC), CPU/disk and network utilization, etc. were believed to be strongly correlated with energy consumption. Simple linear models have been developed using PMCs and correlated features to predict energy consumption of platforms. Rivoire et al. [20] study and compare five full-system

real-time power models using a variety of machines and benchmarks. They report that PMC-based model is the best overall in terms of accuracy since it accounted for majority of the contributors to system's dynamic power. Other notable PMC-based linear models are [2,8,23,26]. Manila [15] construct a densely populated multi-dimensional space of PMCs and predict the energy consumption of platform using a nearest neighborhood search algorithm. Zhuo et al. [28] present a PMC-based energy consumption models for task characteristics in cloud data center using regression algorithms.

Critiques of PMCs for Energy Predictive Modelling. Some attempts where poor prediction accuracy of PMCs for energy predictive modeling has been critically examined include [5,7,16,17]. Researchers highlight the fundamental limitation to obtain all the PMCs simultaneously or in one application run and show that linear regression models give prediction errors as high as 150%.

4 *Additivity* of PMCs

The property of *additivity* is based on a simple and intuitive rule that if a PMC is intended as a parameter in a linear term of the energy predictive model then its value for a compound application should be equal to the sum of its values for the executions of the base applications constituting the compound application. It is based on the experimental observation that the dynamic energy consumption of a serial execution of two applications is the sum of dynamic energy consumptions observed for the individual execution of each application.

We now present a test to determine if a PMC is *non-additive* or potentially *additive*. It comprises of two stages. A PMC must pass both stages to be pronounced *additive* for a given compound application on a given platform.

In the first stage, we determine if the PMC is deterministic and reproducible.

In the second stage, we examine how the PMC of the compound application relates to its values for the base applications. At first, we collect the values of the PMC for the base applications by executing them separately. Then, we execute the *compound* application and obtain its value of the PMC. Typically, the core computations for the compound application consist of the core computations of the base applications programmatically placed one after the other.

If the PMC of the *compound* application is equal to the sum of the PMCs of the base applications (with a tolerance of 5.0%), we classify the PMC as potentially *additive*. Otherwise, it is *non-additive*.

For each PMC, we determine the maximum percentage error. For a *compound* application, the percentage error (averaged over several runs) is calculated as follows:

$$Error(\%) = (|\frac{(\overline{e_{b1}} + \overline{e_{b2}}) - \overline{e_c}}{\overline{e_{b1}} + \overline{e_{b2}}}|) \times 100 \quad (1)$$

where $\overline{e_c}, \overline{e_{b1}}, \overline{e_{b2}}$ are the sample means of predictor variables for the compound application and the constituent base applications respectively. The maximum percentage error is then calculated as the maximum of the errors for all the *compound* applications in the experimental testsuite.

We automated the determination of a PMC's *additivity* using a tool called *AdditivityChecker* (see section 3 of the supplemental [22]).

5 Experimental Results

The experiments are carried out on two modern multicore platforms: (1). an Intel Haswell based dual-socket server and (2). an Intel Skylake based single-socket server. The specifications for both are given in Table 1. We choose a diverse set of benchmarks in our test suite (section 4 of supplemental [22]) with highly memory bound and compute bound scientific computing applications such as DGEMM and FFT from Intel math kernel library (MKL), scientific applications from NAS Parallel benchmarking suite, Intel HPCG, *stress*, non-optimized and non-scientific applications. Apart from reducing bias, one other reason to compose a diverse test suite is to have a range of PMCs for different executions of applications on the platform.

For an application execution, we measure the following: (1). the dynamic energy consumption, (2). the execution time and (3). PMCs. The dynamic energy consumption of the platform is provided by WattsUp pro power meter and the readings are obtained programatically using a detailed statistical methodology employing HCLWattsUp API [9]. The power meters are periodically calibrated using an ANSI C12.20 revenue-grade power meter, Yokogawa WT210. To ensure the reliability of our results, we follow a statistical methodology where a sample mean for a response variable is obtained from several experimental runs. We follow a strict statistical methodology to ensure the reliability of our experiments (see section 3 of supplemental [22]).

We use *Likwid* package [25] to obtain the PMCs. It offers 164 PMCs and 385 PMCs on Intel Haswell and Intel Skylake platform, respectively. We eliminate PMCs with counts less than or equal to 10. The eliminated PMCs have no significance on modeling the dynamic energy consumption of our platform since they are non-reproducible over several runs of the same application on our platform.

The reduced set contains 151 PMCs for Intel Haswell and 323 for Intel Skylake. The collection of all of them takes a huge amount of time since only four PMCs can be obtained in a single application run. This is because of a limited number of hardware registers dedicated for storing them. We also notice that some PMCs can only be collected individually or in sets of two or three for single execution of an application. Therefore, we observe that each application must be executed about 53 and 99 times on Intel Haswell and Intel Skylake platform, respectively, to collect all the PMCs.

We select three predictive models for our experiments: (1). Linear Regression Model (LR), (2). Random Forest (RF), and (3). Neural Networks (NN). We explain them in detail in section 1 of supplemental [22]. In all these models, PMCs appear as parameters in linear terms, and therefore must be *additive*.

We now divide our experiments into three classes, class A, class B and class C, as follows:

1. Class A: we show the improvements in the average prediction accuracy of the three modeling techniques by the *additivity* of PMCs. A diverse set of applications (see section 4 of supplemental [22]) on a dual socket Intel Haswell multicore server is used in these experiments.
2. Class B: we study the impact of the *additivity* of PMCs on prediction accuracy of application-specific energy predictive models. Two highly memory bound and compute bound scientific computing applications such as DGEMM and FFT from Intel MKL, are used in these experiments.
3. Class C: we compare the accuracy of two four parameter models. Both models employ subsets of parameters from the original selected set. The only difference is that one subset include higher energy correlated parameters, and the other contains the most *additive* parameters.

5.1 *Class A*: Improving Prediction Accuracy of Energy Predictive Models Using *Additivity*

We conduct the Class A experiments on the dual-socket Intel Haswell multicore server (see Table 1). We choose six PMCs (X_1 to X_6 in Table 2), which are widely used in energy predictive models. We build a dataset of 277 points as *base* applications by executing the applications from our test suite with different problem sizes. This dataset is used to train the models. We build a test dataset containing points for 50 *compound* applications which are composed up of serial executions of *base* applications. Each point contains the dynamic energy consumption and PMCs for the execution of an application. We apply *additivity* test with allowed error percentage of 5% and found no PMC to be *additive*. We list the PMCs and their *additivity* error percentages in Table 2.

Table 2. List of selected PMCs for modelling with their *additivity* test errors (%).

Selected PMCs	*Additivity* test error (%)
X_1: IDQ_MITE_UOPS	13
X_2: IDQ_MS_UOPS	37
X_3: ICACHE_64B_IFTAG_MISS	36
X_4: ARITH_DIVIDER_COUNT	80
X_5: L2_RQSTS_MISS	14
X_6: UOPS_EXECUTED_PORT_PORT_6	10

We build three sets of models, LRS = {LR1, LR2, LR3, LR4, LR5, LR6}, RFS = {RF1, RF2, RF3, RF4, RF5, RF6}, and NNS = {NN1, NN2, NN3, NN4, NN5, NN6}. In each set, the models contain decreasing number of *non-additive* PMCs. Consider, for example, the first set. Model LR1 employs all the selected PMCs as predictor variables. Model LR2 is based on five most *additive* PMCs. PMC X_4 is removed because it has the highest *non-additivity*. Model LR3 uses

Table 3. Linear predictive models (LR1-LR6) using zero intercepts and positive coefficients with their minimum, average, and maximum prediction errors.

Model	PMCs	Coefficients	Percentage prediction errors (min, avg, max)
LR1	$X_1, X_2, X_3, X_4, X_5, X_6$	3.83E−09, 3.67E−10, 5.30E−07, 0, 5.56E−08, 0	(6.6, 31.2, 61.9)
LR2	X_1, X_2, X_3, X_5, X_6	3.83E−09, 3.67E−10, 5.30E−07, 0, 5.56E−08	(6.6, 31.2, 61.9)
LR3	X_1, X_3, X_5, X_6	3.75E−09, 5.34E−07, 5.58E−08, 0	(2.5, 25.3, 62.1)
LR4	X_1, X_5, X_6	4.00E−09, 5.59E−08, 0	(2.5, 23.86, 100.3)
LR5	X_1, X_6	4.60E−09, 1.46E−09	(2.5, 18.01, 89.45)
LR6	X_6	1.60E−09	(2.5, 68.5, 90.5)

Table 4. Random forest (RF) regression based energy predictive models (RF1-RF6) with their minimum, average, and maximum prediction errors.

Model	PMCs	Percentage prediction errors (min, avg, max)
RF1	$X_1, X_2, X_3, X_4, X_5, X_6$	(2.78, 37.8, 185.4)
RF2	X_1, X_2, X_3, X_5, X_6	(2.5, 30.4, 199.6)
RF3	X_1, X_3, X_5, X_6	(2.5, 30.02, 104)
RF4	X_1, X_5, X_6	(2.5, 23.68, 59.3)
RF5	X_1, X_6	(2.5, 43.4, 174.4)
RF6	X_6	(2.5, 57.7, 172.1)

four most *additive* PMCs and so on until Model LR6 containing the highest *additive* PMC, which is X_6.

We compare the predictions of the models with system-level physical measurements using HCLWattsUp, which we consider to be the ground truth. The minimum, average, and maximum percentage prediction errors for the models in the sets LRS, RFS, and NNS are given in Tables 3, 4 and 5.

Table 5. Neural Networks based energy predictive models (NN1-NN6) with their minimum, average, and maximum prediction errors.

Model	PMCs	Percentage prediction errors (min, avg, max)
NN1	$X_1, X_2, X_3, X_4, X_5, X_6$	(2.5, 30.31, 192.3)
NN2	X_1, X_2, X_3, X_5, X_6	(2.5, 26.32, 201.2)
NN3	X_1, X_3, X_5, X_6	(2.5, 24.14, 160.1)
NN4	X_1, X_5, X_6	(2.5, 24.06, 180.3)
NN5	X_1, X_6	(2.5, 40.21, 202.45)
NN6	X_6	(2.5, 45.05, 180.5)

Since we are modelling dynamic energy consumption, the linear models in Table 3 are built using penalized linear regression using *R programming* interface

that forces the coefficients to be non-negative. All the models also have zero intercept. One can see that the accuracy of the models improves as we remove the highest *non-additive* PMCs one by one until Model $LR5$, which exhibits the least average prediction error of 18.01%. We observe that $LR6$ has the worst average prediction error of 68.5% due to poor linear fit.

Table 4 shows the same trend for random forest models in RFS until Model $RF4$, which has the least average prediction error of 23.68%. Table 5 also shows the same trend for neural network models in NNS until Model $NN4$ with the least average prediction error of 24.06%.

It can be seen that improvements in average prediction accuracy due to *additivity* are less for RF and NN models compared to linear models where we are certain that *additivity* is crucial. The maximum prediction error percentages for RF and NN models are particularly bad. We will investigate in our future work how *additivity* can be used to reduce the maximum error percentage for the three types of models. One can see, however, that the average prediction error percentages of the best RF and NN models are close to the average prediction accuracy of the best linear model suggesting that the RF and NN models exhibit a relationship close to linearity.

5.2 *Class B*: Impact of *Additivity* on the Prediction Accuracy of Application-specific Energy Predictive Models

In this section, we study the accuracy of application specific energy predictive models built using *LR*, *RF*, and *NN* techniques. We choose a single-socket Intel Skylake server (Table 1) for the experiments. We found no PMC to be *additive* within tolerance of 5% for the application suite (see section 4 of supplemental [22]). However, we discover that some PMCs are highly *additive* for two highly optimized scientific kernels: Fast Fourier Transform (FFT) and Dense Matrix-Multiplication application (DGEMM), from Intel Math Kernel Library (MKL).

We build a dataset of 50 *base* applications using different problem sizes for DGEMM and FFT and apply the *additivity* test. The range of problem sizes for DGEMM is 6500×6500 to 20000×20000, and for FFT is 22400×22400 to 29000×29000. We select this range because of reasonable execution time (>3 s) of the applications. We also build a dataset of 30 *compound* applications from these *base* applications.

The *Additivity* test based on the two datasets reveals that there are a number of PMCs which are commonly *additive* for both applications. We select nine PMCs that are highly *additive* with *additivity* test errors of less than 1%. We also select nine PMCs which are *non-additive* for both the applications but which have been employed as predictor variables in energy predictive models given in literature (Sect. 3). We check the correlation of all PMCs with dynamic energy consumption. The selected PMCs with their correlations are given in Table 6.

We denote the set of *additive* PMCs by *PA* and *non-additive* PMCs by *PNA*. We build a dataset containing 801 points representing DGEMM and FFT for a range of problem sizes from 6400×6400 to 38400×38400 and 22400×22400 to 41536×41536, respectively, with a constant step sizes of 64. We record the

dynamic energy consumption and the selected PMCs (Table 6) for each application. We split the dataset into training and test datasets. Training dataset contains 651 points used to train the three energy predictive models. Test dataset contains 150 points.

We build two linear models, {LR-A,LR-NA}, two random forest models, {RF-A,RF-NA}, and two neural network models, {NN-A,NN-NA}. The models {LR-A,RF-A,NN-A} are trained using PMCs belonging to *PA* and the models {LR-NA,RF-NA,NN-NA} are trained using PMCs belonging to *PNA*. Table 7a show the prediction error percentages of the models. One can see that the models based on *PA* have better average prediction accuracy than the models based on PNA.

Table 6. *Additive* and *non-additive* PMCs highly correlated with dynamic energy consumption. 0 to 1 represents positive correlation of 0% to 100%.

	Additive PMCs	Correlation
$X1$	UOPS_RETIRED_CYCLES_GE_4_UOPS_EXEC	0.992
$X2$	FP_ARITH_INST_RETIRED_DOUBLE	0.993
$X3$	MEM_INST_RETIRED_ALL_STORES	0.870
$X4$	UOPS_EXECUTED_CORE	0.993
$X5$	UOPS_DISPATCHED_PORT_PORT_4	0.870
$X6$	IDQ_DSB_CYCLES_6_UOPS	0.981
$X7$	IDQ_ALL_DSB_CYCLES_5_UOPS	0.972
$X8$	IDQ_ALL_CYCLES_6_UOPS	0.993
$X9$	MEM_LOAD_RETIRED_L3_MISS	−0.112
	Non-additive PMCs	
$Y1$	ICACHE_64B_IFTAG_MISS	0.960
$Y2$	CPU_CLOCK_THREAD_UNHALTED	0.600
$Y3$	BR_MISP_RETIRED_ALL_BRANCHES	0.992
$Y4$	MEM_LOAD_L3_HIT_RETIRED_XSNP_MISS	−0.020
$Y5$	FRONTEND_RETIRED_L2_MISS	0.806
$Y6$	ITLB_MISSES_STLB_HIT	0.111
$Y7$	L2_TRANS_CODE_RD	0.860
$Y8$	IDQ_MS_UOPS	0.99
$Y9$	ARITH_DIVIDER_COUNT	0.986

5.3 *Class C*: Comparison of the Impact of Energy Correlation and *Additivity* of PMCs on the Accuracy of Energy Predictive Models

Since only four PMCs can be collected in a single application run, selection of such a reliable subset is crucial to the prediction accuracy of online energy models. The Intel Skylake server (Table 1) is used for the experiments. We use

Table 7. Prediction accuracies of *LR*, *RF*, and *NN* models. **(a)** Class B experiments using nine PMCs. **(b)** Class C experiments using four PMCs.

Model	PMCs	Prediction Errors (%) [Min, Avg, Max]
LR-A	*PA*	(0.005, 35.32, 225.5)
LR-NA	*PNA*	(0.449, 85.61, 4039)
RF-A	*PA*	(.0001, 29.39, 157.4)
RF-NA	*PNA*	(0.004, 36.90, 1682)
NN-A	*PA*	(0.001, 15.43, 104.2)
NN-NA	*PNA*	(0.003, 21.04, 170.3)

(a)

Model	PMCs	Prediction Errors (%) [Min, Avg, Max]
LR-A4	$PA4$	(0.024, 25.12, 87.25)
LR-NA4	$PNA4$	(0.449, 85.61, 4039)
RF-A4	$PA4$	(0.005, 22.73, 207.7)
RF-NA4	$PNA4$	(0.035, 38.06, 1628)
NN-A4	$PA4$	(0.003, 11.46, 152.2)
NN-NA4	$PNA4$	(0.016, 21.32, 227.5)

(b)

PA and *PNA* from Class B experiments to build two sets of four most energy correlated PMCs. The first set PA4, $\{X1, X2, X4, X8\}$, is constructed using *PA* and the second set *PNA4*, $\{Y1, Y3, Y8, Y9\}$, using *PNA*.

We build two linear models, {LR-A4,LR-NA4}, two random forest models, {RF-A4,RF-NA4}, and two neural network models, {NN-A4,NN-NA4}. The models {LR-A4,RF-A4,NN-A4} are trained using PMCs belonging to *PA4* and the models {LR-NA4,RF-NA4,NN-NA4} are trained using PMCs belonging to PNA4. The training and test datasets are the same as those for Class B experiments.

Table 7b shows the prediction error percentages of the models. Model NN-A4 has the least average prediction error of 11.46%. We can see that models {LR-NA4,RF-NA4,NN-NA4} built using highly correlated but *non-additive* PMCs do not demonstrate any improvement in average prediction accuracy compared to models {LR-NA,RF-NA,NN-NA} based on nine *non-additive* PMCs.

The models based on *PA4* containing four most *additive* and highly correlated PMCs have better average prediction accuracy than the models based on the set of *non-additive* PMCs, PNA4.

We conclude, therefore, that correlation with dynamic energy consumption alone is not sufficient to provide good average prediction accuracy but should be combined with methods such as *additivity* that take into account the physical significance of the parameters originating from fundamental laws such as energy conservation of computing.

6 Conclusion

The ability of PMC-based predictive models to provide fine-grained decomposition of energy consumption during the execution of an application makes them ideal fundamental building blocks for several application-level energy optimization techniques. Modern computing platforms such as multicore CPUs provide a large set of PMCs. However, only a limited number of PMCs (typically 3–4) can be obtained during an application run. Therefore, selection of a reliable subset of 3–4 PMCs is crucial to the prediction accuracy of online energy predictive models. The existing techniques select the PMCs based on their correlation

with total energy consumption and construct models employing data analytical approaches such as linear regression, random forests, and neural networks. They do not consider the physical significance of a PMC parameter arising from fundamental laws such as energy conservation of computing.

In this work, we demonstrated how the accuracy of energy predictive models based on three popular techniques (Linear regression, Random forests, and Neural networks) can be improved by selecting PMCs based on a criterion of *Additivity*, which is derived from the application of energy conservation law for computing.

We showed that the removal of *non-additive* PMCs from the list of predictor variables in energy predictive models improved their accuracy. We illustrated that using highly *additive* PMCs resulted in notable improvements in the average prediction accuracy of application-specific models compared to application-specific models employing *non-additive* PMCs. Finally, we studied how a reliable subset of 3–4 PMCs can be constructed for employment in *online* energy predictive models. We showed that using correlation based PMC selection methods to *non-additive* PMCs do not improve the average prediction accuracy of energy models. We demonstrated that using highly correlated PMCs but which are also highly *additive* significantly improves the average prediction accuracy of the models.

In our future work, we will focus on theoretic framework explaining why additivity, which is based on a fundamental physical law of energy conservation, improves the prediction accuracy for the three types of models.

Acknowledgement. This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number 14/IA/2474.

References

1. Barroso, L.A., Hölzle, U.: The case for energy-proportional computing. Computer **12**, 33–37 (2007)
2. Basmadjian, R., Ali, N., Niedermeier, F., de Meer, H., Giuliani, G.: A methodology to predict the power consumption of servers in data centres. In: 2nd International Conference on Energy-Efficient Computing and Networking. ACM (2011)
3. DOE: The opportunities and challenges of exascale computing (2010). http://science.energy.gov/~/media/ascr//pdf/reports/Exascale_subcommittee_report.pdf
4. Dolz, M.F., Kunkel, J., Chasapis, K., Catalán, S.: An analytical methodology to derive power models based on hardware and software metrics. Comput. Sci.-Res. Dev. **31**(4), 165–174 (2016)
5. Economou, D., Rivoire, S., Kozyrakis, C., Ranganathan, P.: Full-system power analysis and modeling for server environments. In: In Proceedings of Workshop on Modeling, Benchmarking, and Simulation, pp. 70–77 (2006)
6. Fan, X., Weber, W.D., Barroso, L.A.: Power provisioning for a warehouse-sized computer. In: 34th Annual International Symposium on Computer architecture, pp. 13–23. ACM (2007)

7. Hackenberg, D., Ilsche, T., Schöne, R., Molka, D., Schmidt, M., Nagel, W.E.: Power measurement techniques on standard compute nodes: a quantitative comparison. In: 2013 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), pp. 194–204. IEEE (2013)
8. Haj-Yihia, J., Yasin, A., Asher, Y.B., Mendelson, A.: Fine-grain power breakdown of modern out-of-order cores and its implications on skylake-based systems. ACM Trans. Archit. Code Optim. (TACO) **13**(4), 56 (2016)
9. HCL: HCLWattsUp: API for power and energy measurements using WattsUp Pro Meter (2016). http://git.ucd.ie/hcl/hclwattsup
10. Heath, T., Diniz, B., Horizonte, B., Carrera, E.V., Bianchini, R.: Energy conservation in heterogeneous server clusters. In: 10th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP), pp. 186–195. ACM (2005)
11. IntelPCM: Intel® performance counter monitor - a better way to measure cpu utilization (2012). https://software.intel.com/en-us/articles/intel-performance-counter-monitor
12. Isci, C., Martonosi, M.: Runtime power monitoring in high-end processors: methodology and empirical data. In: 36th Annual IEEE/ACM International Symposium on Microarchitecture, p. 93. IEEE Computer Society (2003)
13. Kansal, A., Zhao, F.: Fine-grained energy profiling for power-aware application design. ACM SIGMETRICS Perform. Eval. Rev. **36**(2), 26 (2008)
14. Li, T., John, L.K.: Run-time modeling and estimation of operating system power consumption. In: ACM SIGMETRICS Performance Evaluation Review, vol. 31, pp. 160–171. ACM (2003)
15. Mair, J., Huang, Z., Eyers, D.: Manila: using a densely populated pmc-space for power modelling within large-scale systems. Parallel Comput. **82**, 37–56 (2019)
16. McCullough, J.C., Agarwal, Y., Chandrashekar, J., Kuppuswamy, S., Snoeren, A.C., Gupta, R.K.: Evaluating the effectiveness of model-based power characterization. In: Proceedings of the 2011 USENIX Conference on USENIX Annual Technical Conference. USENIXATC 2011. USENIX Association (2011)
17. O'Brien, K., Pietri, I., Reddy, R., Lastovetsky, A., Sakellariou, R.: A survey of power and energy predictive models in HPC systems and applications. ACM Comput. Surv. **50**(3), 37 (2017)
18. PAPI: Performance application programming interface 5.4.1 (2015). http://icl.cs.utk.edu/papi/
19. Perf Wiki: perf: Linux profiling with performance counters (2017). https://perf.wiki.kernel.org/index.php/Main_Page
20. Rivoire, S., Ranganathan, P., Kozyrakis, C.: A comparison of high-level full-system power models. In: Proceedings of the 2008 Conference on Power Aware Computing and Systems, HotPower 2008. USENIX Association (2008)
21. Shahid, A., Fahad, M., Reddy, R., Lastovetsky, A.: Additivity: a selection criterion for performance events for reliable energy predictive modeling. Supercomput. Front. Innovations **4**(4), 50–65 (2017)
22. Shahid, A., Fahad, M., Reddy Manumachu, R., Lastovetsky, A.: Supplemental: Improving the accuracy of energy predictive models for multicore cpus using Additivity of performance monitoring counters (2019). https://github.com/ArsalanShahid116/SLOPE-PMC/blob/master/PaCT-2019-Additivity-supplemental.pdf
23. Singh, K., Bhadauria, M., McKee, S.A.: Real time power estimation and thread scheduling via performance counters. SIGARCH Comput. Archit. News **37**(2), 46–55 (2009)

24. Smarr, L.: Project greenlight: optimizing cyber-infrastructure for a carbon-constrained world. Computer **43**(1), 22–27 (2010)
25. Treibig, J., Hager, G., Wellein, G.: LIKWID: a lightweight performance-oriented tool suite for x86 multicore environments. In: 2010 39th International Conference on Parallel Processing Workshops (ICPPW), pp. 207–216. IEEE (2010)
26. Wang, H., Jing, Q., Chen, R., He, B., Qian, Z., Zhou, L.: Distributed systems meet economics: pricing in the cloud. In: Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing. USENIX Association (2010)
27. Wang, S.: Ph.d thesis: Software power analysis and optimization for power-aware multicore systems (2014)
28. Zhou, Z., Abawajy, J.H., Li, F., Hu, Z., Chowdhury, M.U., Alelaiwi, A., Li, K.: Fine-grained energy consumption model of servers based on task characteristics in cloud data center. IEEE Access **6**, 27080–27090 (2018)