SLOPE: Towards Accurate and Reliable Energy Predictive Modelling using Performance Events on Modern Computing Platforms

PhD Transfer Report

Arsalan Shahid
arsalan.shahid@ucdconnect.ie

Student Number
16203221

PhD Start Date
01/09/2016

Supervisor
Dr. Alexey Lastovetsky

Last DSP Meeting
14/01/2018

University College Dublin

March 13, 2018

Contents

Abstract

Energy is now a first-class design constraint along with performance in all computing settings. Energy predictive modelling based on performance events or performance monitoring counters (PMCs) is the leading method used for prediction of energy consumption during an application execution and the fundamental building block for application-level energy optimization techniques. Modern hardware processors provide a large set of PMCs. Determination of the best subset of PMCs for energy predictive modeling is a non-trivial task given the fact that all the PMCs can not be determined using a single application run. Several techniques have been devised to address this challenge. While some techniques are based on a statistical methodology, some use expert advice to pick a subset (that may not necessarily be obtained in one application run) that, in experts' opinion, are significant contributors to energy consumption. However, the existing techniques have not considered a fundamental property of predictor variables that should have been applied in the first place to remove PMCs unfit for modeling energy. I address this oversight in this report by proposing a novel selection criterion for PMCs called *additivity*, which can be used to determine the subset of PMCs that can potentially be used for reliable energy predictive modeling. It is based on the experimental observation that the energy consumption of a serial execution of two applications is the sum of energy consumptions observed for the individual execution of each application.

I further study the *composability* of well-known energy predictive models based on *additivity* of PMCs. I show that the accuracy of a linear energy predictive models is greatly triggered by the number of additive PMCs used as predictor variable in them.

Finally I conclude and present the current state and future research plan of my PhD program.

1 Introduction

1.1 High-Performance Computing *(HPC)* Platforms & Challenges

Modern Computing Platforms are evolving with increased complexities and highly shared resources. Since the origin, HPC community has always been concerned with increasing the platform performance for executing an application or set of applications. Figure 1 shows the top supercomputing platforms and their increased performance (in petaflops) over a period of time. It can be seen that *U.S. summit* will be the fastest supercomputing platform after 2018 and China plans to release first exascale machine by the end of 2020. It should be noted that latest HPC platforms have become highly heterogeneous owing to tight integration of multicore CPUs and accelerators (such as GPUs, Intel Xeon Phis, or FPGAs) to maximize the dominant objectives of performance and energy efficiency.

Evolution of Performance Modelling for HPC Platforms: A Bird's-Eye View

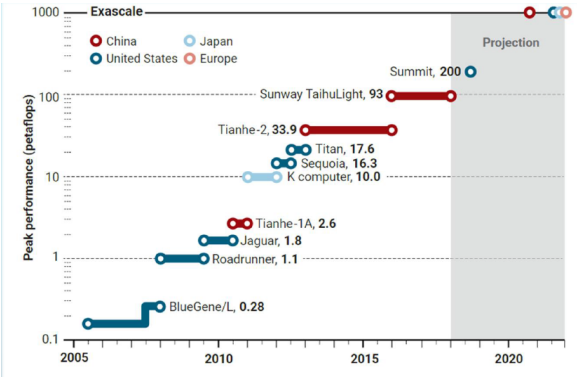

Figure 1: Evolution of HPC Platforms for Performance

For more than three decades prior to mid-2000s, computer users came to expect performance doubling every 18 months due to Moore's law and Dennard scaling. Both clock rate and power increased rapidly. This is the era of homogeneous and heterogeneous clusters of single-core processors. However, by 2004, computer designers hit the power wall caused by problems stemming from increasing power consumption and increasing power density (amount of power dissipated per unit area, which represents the heat dissipation). The power problem was caused primarily by the breakdown of Dennard scaling, a scaling model whereby the power density of a transistor based processor of a unit area remains constant due to voltage and current scaling down with the length of the transistor. Up until 2004, moving to a smaller transistor process meant frequency could be increased for no increase in heat dissipation. The breakdown of Dennard scaling meant frequency scaling was no longer economical. The chip fabrication industry turned to multicore CPU architectures to address this problem of increased power consumption and power density. Frequency scaling was abandoned in favour of multiple processors per chip. Round about 2001, general purpose computing on GPUs became practical with the appearance of programmable shaders and floating point support. Therefore, performance efficiency of the HPC system has been the researched widely by the community.

I briefly study the evolution of performance models and researches that have attempted to realistically capture the real-life behavior of applications executing on these platforms for performance maximization.

The simplest models used positive constant numbers and different notions such as normalized processor speed, normalized cycle time, task computation time, average execution time, etc., to characterize the speed of an application [1], [2], [3]. The common aspect of these models is that the performance of a processor is assumed to have no dependence on the size of the workload. We call them the constant performance models (CPMs).

The most advanced load balancing algorithms use functional performance models (FPMs), which are application-specific. The FPMs rep-

resent the speed of a processor by continuous function of problem size but satisfying some assumptions on its shape [4], [5]. The assumptions require them to be smooth enough in order to guarantee that optimal solutions minimizing the computation time are always load balanced. The FPMs capture accurately the real-life behaviour of applications executing on nodes consisting of uniprocessors (single-core CPUs).

However, modern HPC platforms have complex nodal architectures with highly hierarchical arrangement and tight integration of processors where resource contention and NUMA are inherent. On such platforms, the performance profiles of real-life scientific applications are not smooth and may deviate significantly from the shapes that allowed traditional and state-of-the-art load balancing algorithms to find optimal solutions.

Lastovetsky et al. [6] study the drastic deviations in the performance profiles for a real-life scientific application. The authors propose an optimization technique reusing an advanced performance model of computation (FPM) but using novel load distribution to minimize the computation time of the application. Lastovetsky et al. [7] illustrate in depth these variations in performance and energy profiles of two widely known and highly optimized scientific routines, OpenBLAS DGEMM [8] and FFTW [9] on a modern multicore Intel Haswell CPU platform. They explain the limitations of the FPM-based load balancing algorithms proposed in [10], [11], [12], [13], [14], [15], [16], [17], [18]. They propose novel model-based methods and algorithms for minimization of time and energy of computations for the most general performance and energy profiles of data parallel applications executing on homogeneous multicore clusters. Unlike load balancing algorithms, optimal solutions found by these algorithms may not load-balance an application. The new model-based methods proposed in [6], [7], however, can not be used for optimization of data-parallel applications on HPC platforms with hybrid nodes for maximization of performance since they are designed for homogeneous clusters, i.e., cluster of identical processors. There have also been attempts to profile performance of a computing system by using high-level parameters such as DRAM activity, data stream etc. such as roofline model [19].

1.2 *Energy*: A Recent & Big Challenging Area

Increased performance of HPC systems comes at the cost of energy consumption and with such a tremendous boost in performance of these modern systems, tons of watts of power is being consumed by HPC and supercomputing centers. Tianhe-2, the 33.9-petaflop, 3.12-million processor machine by China supercomputing center consumes around 17.8 MW of power and it is equivalent to powering 13501 households, approximately.

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 [20] in servers are crucial to the operational efficiency of data centres. According to a 2010 DOE Office of Science report [21], it is the leading concern for HPC system designs. Energy consumption in computing contributes nearly 3% to the overall carbon footprint and is now a serious environmental concern [22].

While microarchitectural and chip-design advancements have been the

leading providers of energy savings, application-level energy optimization strategies such as DVFS (dynamic voltage and frequency scaling), Dynamic Power Management (DPM), power-aware scheduling and, most importantly, improvements due to algorithmic innovations, are serious contenders and are also prolifically researched.

Accurate measurement of energy consumption during an application execution is key to several application-level energy minimization techniques. There are two dominant approaches to providing it: a). Physical measurements using external power meters or on-chip power sensors, and b). 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.

Energy predictive software modelling

Energy predictive modelling emerged as the pre-eminent alternative. The existing energy predictive models predominantly use performance events as predictor variables for modelling energy consumption. Performance events or 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 predictions (as explained earlier), over the years, they have been speedily adopted for energy predictive modelling and have come to dominate its landscape. These models are, however, trained using physical measurements of energy consumptions of the examples. 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.

To summarize, physical measurements and energy predictive models are two dominant approaches to determine the energy consumption during an application run. Energy predictive modelling emerged as the preeminent alternative to physical measurements providing fine-grained decomposition of energy consumption during an application run.

Evolution of PMCs as the dominant parameters in energy predictive models

We now take a brief tour through the beginnings and consequent firm establishment of performance events as the dominant parameters in energy predictive models. While presenting this brief history, we also review notable models that predict energy consumption based on utilization or activity factors estimated from performance events.

In the late 1990s, architects began studying in earnest architecture-level power models in simulators similar to how performance is studied in cycle-level architecture simulators. The Cacti tool [23] originally written to study latencies of caches in detail, subsequently provided their dynamic power and leakage power models. In 2000, whole-processor power simulators, SimplePower [24] and Wattch [25], appeared. SimplePower provided detailed dynamic power models of integer ALU and other architectural units in an in-order pipelined processor; the Wattch tool focused on an out-of-order super-scalar pipeline. While both simulators used analytical methods for modeling power, IBM's PowerTimer [26] used empirical techniques. It predicted power consumption of an architectural unit based on measured power consumption of similar unit in an existing microprocessor and scaling it appropriately; taking into account variations in size and design. However, simulators had some drawbacks, chief among them being their speed of exploration, and their lack of rapid adaptability and sustainability to fast-changing hardware architecture landscape. An appealing alternative route was allowed by direct physical measurements of power consumption using power meters. Although power meters provided the total power consumption of a system, one major challenge that remained to be addressed was the accurate decomposability of power consumption at fine-grained component level of granularity. To address this challenge, the performance events based approaches estimating power consumption of architectural units based on their activity factors were invented and eventually became the core of current energy predictive models. The accelerated adoption of this approach was made possible by the simultaneous provision of hardware performance counters (almost akin to standardization) by all the major hardware vendors, and also availability of lightweight tools providing portable APIs to determine the PMCs.

One of the first models correlating PMCs to energy values was developed by Bellosa et al. [27]. Their model is based on events such as integer operations, floating-point operations, memory requests due to cache misses, etc., which they believed to be strongly correlated with energy consumption. Icsi et al. [28] propose an elaborate methodology to determine component-level power estimates from the access rates of the components,

which are based on PMCs. Li et al. [29] propose power models for the operating system (OS) based on their observations of strong correlation between instructions per cycle (IPC) and OS routine power. Lee et al. [30] adopt a statistically rigorous approach to derive regression models using performance events to predict power. A linear model that is based on the utilization of CPU, disk, and network is proposed in [31]. A more complex power model (Mantis) is proposed in [32] relying on the utilization metrics of CPU, disk, and network components and PMCs for memory. Fan et al. [33] propose a simple linear model that correlates power consumption of a single-core processor with its utilization. Energy profiling for applications using CPU and disk activity is the subject in [34]. Rivoire et al. [35], [36] study and compare five full-system real-time power models using a variety of machines and benchmarks. Four of these models are utilization-based whereas the fifth includes CPU PMCs in the model parameter set along with the utilizations of CPU and disk. 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 (especially the memory activity). They also question the generality of their PMC-based model since the PMCs used in their model parameter set may not have the same essence and hence not portable across different architectures (Intel, AMD, etc). Singh et al. [37] develop per-core power models based on multiple linear regression using PMCs. Powell et al. [38] use a linear regression model to estimate activity factors and power for a large number of micro-architectural structures using a small number of PMCs. Goel et al. [39] derive per-core power models using PMC values and temperature readings. A linear model that takes into account CPU utilization and I/O bandwidth is described in [40] to predict power consumption of a server. Basmadjian et al. [41] construct a power model of a server as a summation of power models of its components, the processor (CPU), memory (RAM), fans, and disk (HDD). Bircher et al. [42] propose an iterative modeling procedure to predict power using PMCs. They use PMCs that trickle down from the processor to other subsystems such as CPU, disk, GPU, etc and PMCs that flow inward into the processor such as Direct Memory Access (DMA) and I/O interrupts. Dargie et al. [43] use the statistics of CPU utilization to model the relationship between the power consumption of multicore processor and workload quantitatively. They demonstrate that the relationship is quadratic for single-core processor and linear for multicore processors.

However, there have been few attempts that have critically examined and highlighted the poor prediction accuracy of performance events for energy predictive modeling.

Economou et al. [32] highlight the fundamental limitation, which is the inability to obtain all the PMCs simultaneously or in one application run. They also mention the lack of PMCs to model energy consumption of disk I/O and network I/O. McCullough et al. [44] evaluate the competence of predictive power models for modern node architectures and show that linear regression models show prediction errors as high as 150%. They suggest that direct physical measurement of power consumption should be the preferred approach to tackle the inherent complexities posed by modern node architectures. Hackenberg et al. [45] present a study of

Table 1: Specification of the Intel Haswell multicore CPU

Technical Specifications	**Intel Haswell Server**
Processor	Intel E5-2670 v3 @2.30GHz
OS	CentOS 7
Micro-architecture	Haswell
Thread(s) per core	2
Cores per socket	12
Socket(s)	2
NUMA node(s)	2
L1d cache	32 KB
L11 cache	32 KB
L2 cache	256 KB
L3 cache	30720 KB
Main memory	64 GB DDR4
Memory bandwidth	68 GB/sec
TDP	240 W
Idle Power	58 W

various power measurement strategies, which includes *Intel RAPL* [46]. *Intel RAPL* uses a software modeling approach that uses PMCs to predict energy consumption. They report that the accuracy of *RAPL* depends on the type of workload and is quite poor for workloads that use the hyper-threading feature. They also report that the accuracy is poor for applications with small execution times and becomes better only for applications with longer execution times since the predictions are energy averages. O'Brien et al. [47] survey predictive power and energy models focusing on the highly heterogeneous and hierarchical node architecture in modern HPC computing platforms. Using a case study of PMCs, they highlight the poor prediction accuracy and ineffectiveness of models to accurately predict the dynamic power consumption of modern nodes due to the inherent complexities (contention for shared resources such as Last Level Cache (LLC), NUMA, and dynamic power management).

Challenges of using PMCs For energy modelling

Modern hardware processors provide a large set of PMCs. Consider the Intel Haswell multicore server CPU whose specification is shown in Tab. 1. On this server, the PAPI tool [48] provides 53 hardware performance events. The Likwid tool [49], [50] provides 167 PMCs. This includes events for uncore and micro-operations (μops) of CPU cores specific to Haswell architecture that are not provided by PAPI. However, all the PMCs can not be determined using a single application run since only a limited number of registers is dedicated to collecting them. For example, to collect all the Likwid PMCs for a single runtime configuration of an application on the server, the application must be executed 53 times. It must be also pointed out that energy predictive models based on PMCs are not portable across a wide range of architectures. While a model based on either Likwid PMCs or PAPI PMCs may be portable across Intel and AMD architectures, it will be unsuitable for GPU architectures.

Therefore, there are three serious constraints that pose difficult challenges to employing PMCs as predictor variables for energy predictive

modeling. First, there is a large number of PMCs to consider. Second, tremendous programming effort and time are required to automate and collect all the PMCs. This is because all the PMCs can not be collected in one single application run. Third, a model purely based on PMCs lacks portability. We now explore the techniques employed to select a subset of PMCs to be used as predictor variables for energy predictive modeling. I now present a brief survey of them.

O'Brien et al. [47] survey the state-of-the-art energy predictive models in HPC and present a case study demonstrating the ineffectiveness of the dominant PMC-based modeling approach for accurate energy predictions. In the case study, they use 35 carefully selected PMCs (out of a total of 390 available in the platform) in their linear regression model for predicting dynamic energy consumption. [42], [51], [52] select PMCs manually, based on in-depth study of architecture and empirical analysis. [53], [39], [54], [55], [56], [57], [58] select PMCs that are highly correlated with energy consumption using Spearman's rank correlation coefficient (or Pearson's correlation coefficient) and principal component analysis (PCA). [42], [57], [59] use variants of linear regression to remove PMCs that do not improve the average model prediction error.

From the survey, I can classify the existing techniques into three categories. The first category contains techniques that consider all the PMCs with the goal to capture all possible contributors to energy consumption. To the best of our knowledge, I found no research works that adopt this approach. This could be due to several reasons: a) Gathering all PMCs requires huge programming effort and time; b) Interpretation (for example, visual) of the relationship between energy consumption and PMCs is difficult especially when there are large number of PMCs; c) Dynamic or runtime models must choose PMCs that can be gathered in just one application run; d) Typically, simple models (those with less parameters) are preferred over complex models not because they are accurate but because simplicity is considered a desirable virtue.

The second category consists of techniques that are based on a statistical methodology. The last category contains 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.

In this report, I address the challenges already discussed with PMCs based energy predictive models with the aim to build reliable end accurate energy predictive models.

Energy consumption of a serial execution of two applications is the sum of energy consumptions observed for the individual execution of each application. The same condition should be fulfilled by the PMCs if they are predictor variables of energy in linear models. Hence, I feel a need to address this fundamental property of predictor variables (PMCs) that should have been considered in the first place to remove PMCs unfit for modeling energy. I propose a novel selection criterion for PMCs called *additivity* (publised and available online @ [60]), which can be used to determine the subset of PMCs that can potentially be used for reliable energy predictive modeling. I define a *compound application* to represent a serial execution of a combination of two or more individual applications.

The individual applications are also termed as *base applications*. A linear predictive energy model is consistent if and only if its predictor variables are additive in the sense that the vector of predictor variables for a compound application is the sum of vectors for the individual execution of each application. The *additivity* criterion, therefore, is based on simple and intuitive rule that the value of a PMC for a compound application is equal to the sum of its values for the executions of the base applications constituting the compound application.

A PMC is branded *non-additive* on a platform if there exists a compound application for which the calculated value significantly differs from the value observed for the application execution on the platform (within a tolerance of 5.0%). The use of a *non-additive* PMC as a predictor variable in a model renders it inconsistent and therefore unreliable. I study the *additivity* of PMCs offered by two popular tools, *Likwid* and *PAPI*, by employing a detailed statistical experimental methodology on a modern Intel Haswell multicore server CPU. I observe that all the Likwid PMCs and PAPI PMCs are reproducible. However, I show 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.

Based on the presented property of additivity, I further study and present a mathematical and experimental study of the prediction accuracy of the most popular class of energy predictive models, which are based on linear regression and employ performance events as predictor variables. I postulate a simple and intuitive property called *composability* of models that has been heretofore unstudied and which I believe is essential for their reliable design.

1.3 Organization of Report

The rest of the report is structured as follows. Section 2 details the terminology that has been used throughout in this report. In Section 3, I present a tour of previous researches as related work. Section 4 presents the research objective and questions that I am addressing during my PhD. Section 5 illustrates the results and experimental findings and section 6 conclude this report. Finally, in section 7, I summarize my current achievements and a tentative future plan for PhD program.

2 Terminology

This section describes the various terms related to energy predictive models used in this work.

2.1 Static and Dynamic Energy Consumption

There are two types of energy consumptions, static energy and dynamic energy. Static energy consumption is defined 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. That is, 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,

$$E_D = E_T - (P_S \times T_E) \tag{1}$$

I only consider only the dynamic energy consumption because static energy consumption is a constant (inherent property) of a platform and will remain same for different application configurations. Let us now elucidate using two examples from published results the importance of using dynamic energy consumption.

In our *first example*, consider a model that reports predicted and measured total energy consumption of a system to be 16500J and 18000J, respectively. It would report the prediction error to be 8.3%. However, if it is known that the static energy consumption of the system is 9000J, then the actual prediction error (based on dynamic energy consumptions only) would be 16.6%, i.e., the double.

In our *second example*, consider two different energy prediction models (M_A and M_B) with same prediction errors of 5% for an application execution on two different machines (A and B) with same total energy consumption of 10000J. One would consider both the models to be equally accurate. But supposing it is known that the dynamic energy proportions for the machines are 30% and 60%. Now, the true prediction errors (using dynamic energy consumptions only) for the models would be 16.6% and 8.3% respectively. Therefore, the second model M_B should be considered more accurate than the first.

The dynamic energy consumption in the energy function is calculated by subtracting the static energy consumption from the total energy consumption during the execution of the application measured using Watts Up power meter. The static energy consumption is calculated by multiplying the idle power of the platform (without application execution) with the execution time of the application.

3 Related Work

This section is divided into two parts. In the first part, I present tools widely used to obtain PMCs. I also survey notable research on selection of PMCs for power and energy modeling from a large set supplied by a tool. In second part, I present notable energy predictive models in CPU based platforms and accelerators.

3.1 Tools to Determine PMCs

PAPI [48] provides a standard API for accessing PMCs available on most modern microprocessors. It provides two types of events, *native events* and *present events*. *Native events* correspond to PMCs native to a platform. They form the building blocks for *present events*. A *preset event* is mapped onto one or more native events on each hardware platform.

While *native events* are specific to each platform, *preset events* obtained on different platforms can not be compared.

Likwid [49] provides command-line tools and an API to obtain PMCs for both Intel and AMD processors on the Linux OS.

For Nvidia GPUs, CUDA Profiling Tools Interface (*CUPTI*) [61] can be used for obtaining the PMCs. *Intel PCM* [62] is used for reading PMCs of core and uncore (which includes the QPI) components of an Intel processor. *Perf* [63] also called *perf_ events* can be used to gather the PMCs for CPUs in Linux.

3.1.1 Techniques for Selection of PMCs for Energy Predictive Modeling

All the models surveyed in this section are linear energy predictive models.

Singh et al. [53] use PMCs provided by AMD Phenom processor. They divide the PMCs into four categories and rank them in the increasing order of correlation with power using the Spearman's rank correlation. Then they select the top PMC in each category (four in total) for their energy prediction model.

Goel et al. [39] divide PMCs into event categories that they believe capture different kinds of microarchitectural activity. The PMCs in each category are then ordered based on their correlation to power consumption using the Spearman's rank correlation.The PMCs with less correlation are then investigated by analyzing the accuracy of several models that employ them.

Kadayif et al. [64] present a PMC-based model for predicting energy consumption of programs on a UltraSPARC platform. The platform provides 30 different PMCs. However, they use only eight and do not specify how they have selected them.

Lively et al. [54] employ 40 PMCs in their predictive model. They use an elaborate statistical methodology to select PMCs. They compute the Spearman's rank correlation for each PMC and remove those below a threshold. They compute the principal components (PCA) of the remaining PMCs and select those with the highest PCA coefficients. Bircher et al. [42] employ an iterative linear regression modeling process where they add a PMC at each step and stop until desired average prediction error is achieved.

Song et al. [55] select a group of PMCs (for their energy model of Nvidia Fermi C2075 GPU) that are strongly correlated to power consumption based on the Pearson correlation coefficient.

Witkowski et al. [56] use PMCs provided by the *Perf* tool for their model. They use the correlation (Pearson correlation coefficient) between a PMC and the measured power consumption and select those PMCs, which have high correlation coefficients. Although they find that the PMCs related to DRAM have a low correlation with power consumption, they still use them since these variables signify intensity of DRAM operations, which contribute significantly to power consumption.

Gschwandtner et al. [51] deal with the problem of selecting the best subset of PMCs on the IBM POWER7 processor, which offers over 500

different PMCs. They first manually select a medium number of hardware counters that they believe are prominent contributors to energy consumption. Then they empirically select a subset from their initial selection. Jarus et al. [57] use PMCs provided by the *Perf* tool for their models. The PMCs employed differ for different models and are selected using two-stage process. In the first stage, PMCs that are correlated 90% or above are selected. In the second stage, stepwise regression with forward selection is used to decide the final set of PMCs.

Haj-Yihia et al. [52] start with a set of 23 PMCs (offered by Likwid) based on expert knowledge of the Intel architecture. Then they perform linear regression iteratively where they drop PMCs (one by one) that do not impact the average prediction error of their model.

Wu et al. [58] use the Spearman correlation coefficient and PCA to select the subset of PMCs, that are highly correlated with power consumption. Chadha et al. [59] select a particular PMC from the list of PAPI PMCs available for their platform and check if it fits well with linear regression model. If it does, they select it as a key parameter for their modeling and experimental study. Otherwise, they skip it.

3.2 Notable Energy Predictive Models

3.2.1 Energy Predictive Models for CPUs and accelerators

Bertran et al. [65] present a power model that provides per-component power breakdown of a multicore CPU. Their model is based on activity factors obtained from PMCs for various components in a multicore CPU. Basmadjian et al. [66] report that summation of power consumptions of all active cores to derive the total power consumption is inaccurate and take into account resource sharing in their power prediction model for multicore processors.

Rotem et al. [46] present a software power model, which eventually became *RAPL*, in Intel Sandybridge. This model predicts the energy consumption of core and uncore components (QPI, LLC) based on some PMCs (which are not disclosed).

McPAT [67] is an integrated power, area, and timing modeling framework for multithreaded, multicore, and manycore architectures. It supports estimation of power consumption for various components in a multiprocessor, which includes shared caches, integrated memory controllers, in-order and out-of-order processor cores, and networks-on-chip. However, McPAT has known limitations in power estimation, which were reported in [68]. Haj-Yihia et al. [52] present a linear regression model for Intel Skylake processors based on PMCs. They selected the PMCs which are popular in well-known energy and power models. Lastovetsky et al. [69] present an application-level energy model where the dynamic energy consumption of a processor is represented by a function of problem size. Unlike PMC-based models that contain hardware-related PMCs and do not consider problem size as a parameter, this model takes into account highly non-linear and non-convex nature of the relationship between energy consumption and problem size for solving optimization problems of

data-parallel applications on homogeneous multicore clusters for energy.

I now survey few notable research works that have proposed energy predictive models for accelerators such as GPU, Xeon Phi, and FPGA. Hong et al. [70] propose an energy prediction model for an Nvidia GPU similar to the PMC-based unit power prediction approach of [28]. Nagasaka et al. [71] present a statistical approach that uses GPU performance counters exposed for CUDA applications to predict power consumption of GPU kernels. Song et al. [55] propose power and energy prediction models that employ a configurable, back-propagation, artificial neural network (BP-ANN). The parameters of the BP-ANN model are ten carefully selected PMCs of a GPU. The values of these PMCs are obtained using the CUDA Profiling Tools Interface (CUPTI) [61] during the application execution. Shao et al. [72] construct an instruction-level energy model of a Xeon Phi processor. Khatib et al. [73] propose a linear instruction-level model to predict dynamic energy consumption for soft processors in FPGA. The model considers both inter-instruction effects and the operand values of the instructions.

4 Research Objective & Questions

Energy is now a mainstream challenge along with performance in all computing systems. Energy-proportional designs in servers are crucial to the operational efficiency of data centres. Chip designers and developers have been mainly concerned with providing energy saving equipment. However, the application level energy optimization techniques such as dynamic frequency and voltage scaling, core switching, cache line locking, clock and power gating, dynamic power management, scheduling, and algorithmic developments etc., have caught much attention in research community as they are serious contenders towards saving energy.

Accurately being able to capture energy consumption during an application execution is key to several application-level energy minimization techniques. Two most dominant approaches to measure energy consumption of a computing machine include 1). physical measurements and 2). software models. The first approach includes hardware measurements through power-meters. However, physical measurements are only capable to capture overall energy consumption of a system. 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. Hence, they are not helpful for energy optimization at component level. In addition to this, 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.

In such a scenario, energy predictive software modelling has emerged as an alternative solution for fine grain decomposition of energy consumption. The existing energy predictive models are predominantly based on performance events, i.e., specific purpose memory storing elements to track hardware activity. They have been developed primarily to aid

low-level performance analysis and tuning. While remarkably PMCs have not been used for performance modelling, over the years, they have been speedily adopted for energy predictive modelling and have come to dominate its landscape. These models are, however, trained using physical measurements of energy consumptions of the examples. These models are, however, trained using physical measurements of energy consumptions of the examples. 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.

My main research objective is to build reliable and accurate energy predictive models for modern and complex computing machines. For this, I study PMC based energy modelling as they have been widely adopted and considered strong contenders for modelling energy consumption.

I will now list research questions that I have answered during my PhD and also I will state a few open research questions which I plan to answer in future during my PhD. RQ is used to denote research question in this section.

As majority of models that use PMCs as a predictor variables are linear, I first study them. While PMC based energy models has attracted vast community, there are a few researches who doubt the accuracy of PMC based energy models (also mentioned in section 3). They also report high errors of predictions in PMC based models. This motivate me to explore if it is the case in reality or not? For answering this, after a comprehensive background research (see section 2) and on the basis of my own experimental analysis, I found the present energy predictive linear models based on PMCs are not reliable and they show unavoidable inaccuracy. Hence, my first research question which I answered is:

RQ 1: Why are the PMC based linear energy predictive models not reliable and accurate?

For my quest to answering this question, I first studied the limitations of PMC based models (presented in section 1). As one of the limitation of PMCs include the restriction to access a limited number of events in a single application run, it requires a huge programming effort to collect all the PMCs for a particular workload. And, there exist no open-source tool to collect all the PMC for an application.

Problem: Need of an open-source software tool to automate the process of collection of PMCs.
For this, I wrote SLOPE-PMC [74], which includes two software wrappers for automatic collection of all the PMCs for an application, i.e., SLOPE-PMC-LIKWID and SLOPE-PMC-PAPI.

Moving forward, to studying the PMC based energy predictive models, I have to generate data containing all PMCs for a number of applications. And, there is no such database that is reliable and openly available to use. I generated a data set with all PMCs available on our platform for 215 applications. For the sake of understanding, each data point can be

considered as a vector of PMCs and energy consumption for particular application. The process of collecting this data is not trivial. It took me 8086 hours (approximately) to obtain this data-set. However, for better analysis, there is a need to generate data having thousands of data points rather than 215 points. For extending the data-set, I decided to simulate the existing data points.

It is an experimental observation and a fact that energy consumption of a serial execution of two applications is the sum of energy consumptions observed for the individual execution of each application.

I decided to combine the vector of PMCs for different applications to have a huge amount of simulated data-points via artificial applications.

For a linear energy predictive model, each individual predictor variable (in my case, a PMC) must be additive. That is, to check if value of a PMC for a compound application (serial execution of two individual applications called the base applications) is equal to the sum of its values for the executions of the base applications constituting the compound application.

To proceed further, I came up with a few other questions; answering to which led me to answer the fundamental research question (i.e., RQ 1). I will now mention all of them as follows:

- Are all the PMCs available on modern computing platforms additive?

  With comprehensive suite of applications, I found many of the PMCs on modern platforms as non-additive and presented a property of *additivity* as a selection criteria for linear PMC based analytical models. An initial list of potentially additive PMCs is published in [60]. Studying further the additivity of PMCs, I formulated another research question given below:

- Are there different number of additive PMCs for different classes of applications (compute bound or memory bound)?

  For answering this, I choose highly optimized applications from both classes and check the additivity of PMCs for them. I found that there are a number of PMCs which may be additive for one class of applications and non-additive for the other. This bring up another research question given below:

- Are there any additive PMCs in common for both class of applications?

  After a detailed experimental analysis, I finally found no PMC that is commonly additive for both application classes. Another important research question that I answered is given as follows:

- How the PMCs have evolved from single core to multicore complex architectures in terms of additivity?

  I answered this question via a comprehensive set of experimental analysis (results shown in section 5). Another research question that I have answered after this is:

- How much inaccuracy the presence of non-additive PMCs in a model can be expected in state-of-the-art models?

I studied the composability of existing energy predictive models (results shown in section 4). In the current stage of my PhD, I along with my team are formalizing the linear energy predictive models mathematically by integrating the assumed characteristics of PMC based models and the property of additivity.

Hence, from the above mentioned study, I conclude that one of the reasons for the inaccuracy in PMC based linear energy predictive models is the use of non-additive PMCs in them. There maybe some other reasons for the inaccuracy other than this, but we demonstrate (in section 5) that using only one or two highly additive PMC can lead to a more reliable model with desirable accuracy.

Furthermore, I conclude that general PMC based linear models can not be reliable and accurate because in general there are no PMCs that are found to be additive for all class of applications. They may have been reliable in single core systems because of more additive nature of PMCs in them but with modern computing systems where a lot of computing resources are shared, the linear PMC based models can not predict energy accurately. However, application specific linear energy predictive modelling using PMCs is possible.

After studying the linear energy predictive models and presenting a novel selection criteria (additivity) for PMCs, the next open research questions for my future work are given below:

RQ 2: Are there any other unstudied properties of PMCs like additivity that non-linear energy predictive models should possess?

The accuracy of on-chip sensors for measuring the energy consumption at component level also need to be checked. Therefore, one of the research question that I will answer in future is as follows:

RQ 3: How accurate are the on-chip hardware sensors when it comes to to estimate fine-grain decomposition of energy consumption?

RQ 4: Is there possibility to have a general non-linear PMC based energy predictive model?

RQ 5: Are there any other parameters other than PMCs that are more reliable and can be used as predictor variables in models for more accurate estimation of energy consumption?

RQ 6: Why PMCs have not been used for performance estimation of platforms over the years?

Apart from the mentioned questions and problems, there may rise other research questions in future as I progress to explore non-linear models.

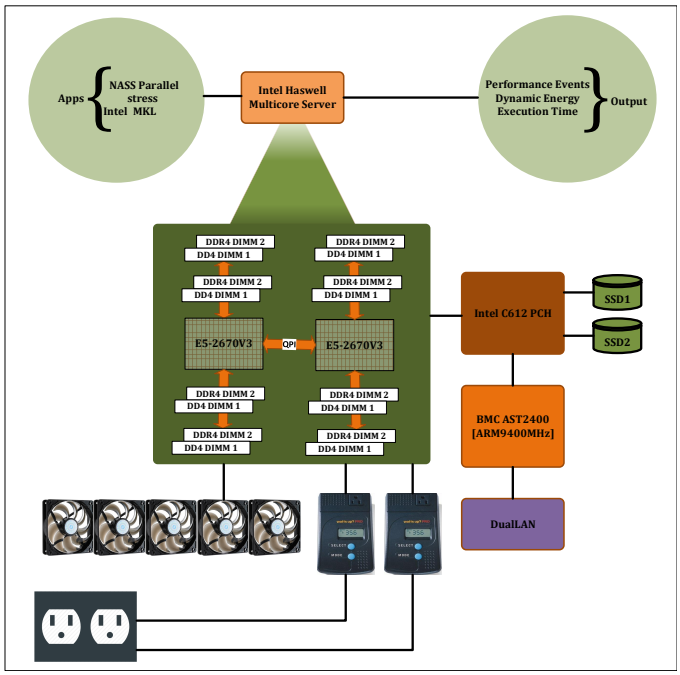

Figure 2: Experimental workflow to determine the PMCs on the Intel Haswell server [60]

5 Experimental Results

5.1 Additivity of Likwid and PAPI PMCs

In this section, I present our experimental results for the *additivity* of PMCs. We used two popular tools, i.e., Likwid and PAPI to extract the PMCs.

The experiments are performed on the Intel Haswell multicore CPU platform (specifications given in Tab. 1). I used diverse range of applications (both compute-bound and memory-bound) in our testsuite composed of NAS parallel benchmarking suite (NPB), Intel math kernel library (MKL), HPCG [75], and *stress* [76]. The experimental workflow is shown in Fig. 2 where the internals of the server are shown in great detail.

For each run of a application in our testsuite, I measure the following: 1) Dynamic energy consumption, 2) Execution time, and 3) PMCs. The dynamic energy consumption and the application execution time are obtained using the HCLWattsUp interface [77]. I would like to mention that the output variables (or response variables) in the performance and energy predictive models, i.e., energy consumption and execution time, are *additive*. I confirm this via thorough experimentation and therefore I will not discuss them hereafter. The details of experimental methodology in which we use statistical analysis are given in [60].

Table 2: Additivity Test for Intel MKL, DGEMM and FFT

APP	Total PMCs	Additive	Non-Additive	Maximum Range of Error (%)					
				5-10	10-20	20-40	40-100	100-500	200-1000
DGEMM	151	27	126	43	51	21	6	1	4
FFT	151	5	146	7	50	41	40	1	4

After applying additivity test, we found that a number of PMCs that have been used in well-known energy and power models are non-additive. our initial study (using a limited set of applications) reveals the list of potentially *additive* PMCs which are given in Arsalan et al. [60].

In-Depth Study for Additivity of Likwid PMCs

I choose *Intel MKL DGEMM* as an example of compute-bound application. For the base applications, I select a range of problem sizes (2000×2000 to 10000×10000). A compound application is written using a pair of base applications. I execute all the compound applications one after the another and determine the maximum of absolute percentage errors for a PMC.

We, then, choose *Intel MKL FFT* as the memory-bound application. For the base applications, I select a range of problem sizes (7000×7000 to 20000×20000).

In Table 2, I show the number of non-additive PMCs with the ranges of prediction error for both applications.

My next study was to check if there exist any PMC in common for both class of applications that is additive. After analysis, I found no PMCs in common that satisfies additivity criteria.

5.1.1 Evolution of Additivity of PMCs from Single-core to Multicore Architectures

In this section, I study the evolution of *additivity* of PMCs from single-core to multicore platforms.

First, I design experiments using Intel MKL DGEMM by selecting a range of large problem sizes (16000×16000 to 30000×30000) that stresses our platform for a reasonable amount of time. I also designed many compound applications from the given range of problem sizes.

By completely dedicating the server for few days, I launch the experiments by binding the applications to specific CPU-core configurations (from one core of each socket to 12 cores with an increment of 4). That is, I perform additivity test for the applications (both base and compound) that run on four different core configurations of machine (1-core, 4-core, 8-core and 12-core). For each core and application configuration, I note the maximum of errors for each PMCs and count the number of non-additive PMCs observed for each core setting. it should be noted that all the applications are bind to both memory banks of the server.

I then repeat the same experiments with Intel MKL FFT (for large problem sizes: 7000×7000 to 21000×21000) and Naive MV (for problem

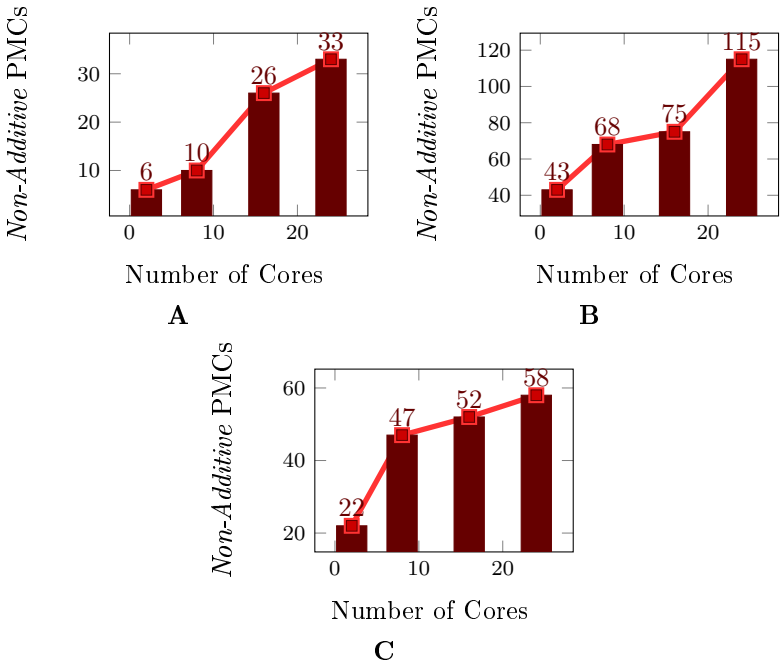

Figure 3: **Increase in number of *non-additive* PMCs with cores. (A)** shows *non-additive* PMCs for Intel MKL DGEMM. **(B)** shows *non-additive* PMCs for Intel MKL FFT. **(C)** shows *non-additive* PMCs for naive matrix-vector multiplication.

sizes: 30000 to 35000)

In Figure 3, I show the rise of non-additivity in PMCs with the advancements of cpu-cores and more use of shared resources for application execution in multicore platforms. The results are presented with respect to total number of cores on each socket to which an application is pinned. As one socket has 12 cores, X-axis stretches to total 24 cores. It can be seen that for each application on single-core configuration there are least number of non-additive PMCs; which make it easy to understand that emergence of complex computing systems causes the PMCs to possess non-additive and sometimes non-deterministic nature.

5.2 Composability of Energy Predictive Models

In this section, I present a study of the composability of models based on non-additive nature of several well-known PMCs used in energy predictive models. I select PMCs which are most commonly used to in energy predictive models from published research [28, 29, 37, 52, 53, 78, 79]. These are listed in Table 3. Having real dynamic energy measurements from *HCLWattsUp* interface against PMCs' counts from *Likwid* for highly optimized *base applications*, I build four linear models and tested their

Table 3: **Correlation of PMCs with Dynamic Energy Consumption. (A.)** List of selected PMCs for modeling. **(B.)** Correlation matrix showing relationship of dynamic energy (E_D) with PMCs. *1* denotes maximum 100% correlation and *0* for no correlation. X_1,X_2,X_5 and X_6 are highly correlated (over 90%) with E_D

X_1: *IDQ_MITE_UOPS*
X_2: *ICACHE_MISSES*
X_3: *IDQ_MS_UOPS*
X_4: *ARTH_DVDR_UOP*
X_5: *L2_RQSTS_CODE**
X_6: *DRAM_CLOCK**

A

	E_D	X_1	X_2	X_3	X_4	X_5	X_6
E_D	1	0.95	0.97	0.56	-0.45	0.97	0.93
X_1	0.95	1	0.97	0.38	-0.63	0.98	0.88
X_2	0.97	0.97	1	0.44	-0.54	0.99	0.90
X_3	0.56	0.38	0.44	1	0.39	0.44	0.68
X_4	-0.45	-0.63	-0.54	0.39	1	-0.55	-0.25
X_5	0.97	0.98	0.99	0.44	-0.55	1	0.91
X_6	0.93	0.88	0.90	0.68	-0.25	0.91	1

B

accuracy for different *compound applications* (i.e., *a*, *b*, ..., *g*) designed from *base applications*. These models are explained as below:

- *Model A:* Linear model with all six selected PMCs as predictor variables for dynamic energy estimation.
- *Model B:* Linear model having only highly correlated PMCs shown in Table 3 as predictor variables for dynamic energy estimation.
- *Model C:* Linear model composed of only one additive PMC from the list of selected PMCs as predictor variable for dynamic energy estimation.
- *Model D:* Linear model using two most additive PMCs from the list of selected PMCs as predictor variable for dynamic energy estimation.

Figure 4 presents a comparison between the dynamic energy estimated by models with real hardware power-meter measurements along with the absolute errors %. It can be seen that maximum prediction errors are reported for *model A* that uses all selected PMCs. The prediction error slightly reduces for *model B* but not too much. Instead of using all selected PMCs in modelling dynamic energy, I observe a noticeable reduction in dynamic energy predictions when only one highly additive PMC is used as a predictor variable. It is further evident from results obtained from *model C* that average prediction error drops to almost half than model A predictions. I then used two most additive PMCs from the list of our selected PMCs to build a *model D* which reasonably estimate dynamic energy consumption for compound applications in our case. It can be seen that there is a huge reduction in maximum prediction error of *Model A* and *Model D*, i.e., 135 % to 47 %, respectively.

Having said earlier that PMCs are unfit for modelling dynamic energy consumption of a platform in general, based on our analysis, I conclude that only highly additive PMCs can yield a composable model for estimating a platform's dynamic energy consumption; if it executes specific set of applications, only. The use of more and more non-additive PMCs

will make a model non-composable and hence unfit to be used as a tool for estimating dynamic energy consumption.

6 Conclusion

Energy consumption in modern computing platforms is a big and challenging area of research. Performance events (PMCs) are now dominant predictor variables for modeling energy consumption. In this report, I discuss a novel selection criteria to select a subset of PMCs out of a large set called as *additivity*. The PMCs that pass the additivity test are considered to be fit to modelling dynamic energy consumption. I tested the PMCs obtained from two most popular tools based on this criteria to for their additivity and provided a list of PMCs that can potentially be used as predictor variables for energy predictive modelling to achieve considerable accuracy. Following this, I classified the additive and non-additive PMCs based on applications nature and then studied the evolution of additivity of PMCs with the complexities in CPU architecture.

Furthermore, this report present the composability of well-known energy predictive models using PMCs. Based on established research, I conclude that the general use of PMCs for energy predictive linear models is totally in-accurate in modern computing platforms.

In future, I will formalize the linear energy predictive models by integrating the property of additivity and adding the assumptions of PMCs based models in mathematical form. Furthermore, I plan to study non-linearity of PMCs for their use in general non-linear and functional models.

7 Current state and future plans

In this section, I highlight the current state and and tentative future plans of my PhD program.

- **Current Achievements**
  1. Literature review of previous researches completed.
  2. Presented the problem for energy modelling using PMCs in Second NESUS PhD Symposium held in conjunction with Winter School at Calabria, Italy [Feb 20, 2017 to Feb 23, 2017].
  3. Developed SLOPE-PMC [74], i.e., an open-source software for automated collection of PMCs on Intel Haswell CPU platform.
  4. Published a novel property (Additivity) for selection of suitable PMCs for energy modelling.
     - **Shahid, A.**, M. Fahad, R. R. Manumachu, and A. Lastovetsky, "Additivity: A Selection Criterion for Performance Events for Reliable Energy Predictive Modeling", Journal of Supercomputing Frontiers and Innovations, vol. 4, issue 4, pp. 50-65, 12/2017
  5. Presented the property of Additivity for selection of PMCs for linear regression based energy predictive modelling in Third NE-

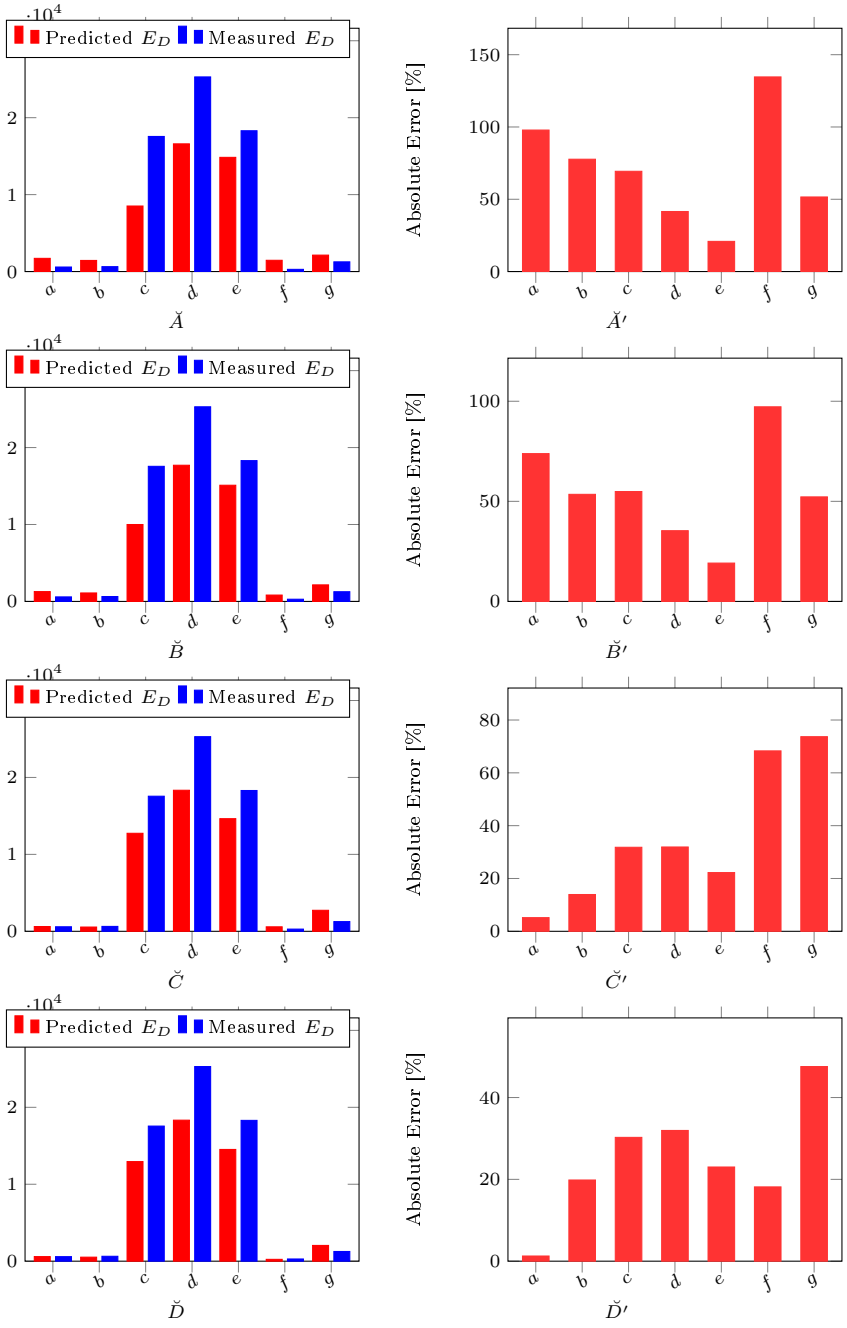

Figure 4: Prediction accuracy of the four linear energy predictive models. Predicted E_D is obtained from model and WattsUp Pro power-meter provides measured E_D. Comparison of measured and predicted $Energy_D$ from *model A*, *model B*, *model C* and *model D* is shown in $\check{A}$, $\check{B}$, $\check{C}$ and $\check{D}$, respectively. $\check{A}'$, $\check{B}'$, $\check{C}'$ and $\check{D}'$ shows the absolute errors (%) in predictions. The average prediction errors from $\check{A}$, $\check{B}$, $\check{C}$ and $\check{D}$ are 70.5%, 50%, 35.3% and 24.5%, respectively.

SUS PhD Symposium held in conjunction with Winter School at Zagreb, Croatia [January 22, 2017 to January 25, 2017].

- **Future plans**
  1. Formalize the assumptions behind the linear energy predictive models and integrate the additivity criteria to extend PMC based models for *composability*.
  2. Develop an open-source tool to check the additivity of PMCs for given application set.
  3. Study the suitability of PMCs for non-linear energy predictive modelling.
  4. Develop AI based techniques for functional energy predictive modelling.
  5. Explore other techniques for reliable and accurate energy predictive modelling apart from PMCs.
  6. Check the accuracy of on-chip sensors for measuring energy consumption.
  7. Publish the conducted research in high-rated conferences and journals.
  8. Completion of PhD thesis by ($\sim$ May, 2020)

Acknowledgement

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

References

[1] M. Cierniak, M. J. Zaki, and W. Li, "Compile-time scheduling algorithms for a heterogeneous network of workstations," *The Computer Journal*, vol. 40, no. 6, pp. 356–372, 1997.

[2] O. Beaumont, V. Boudet, F. Rastello, and Y. Robert, "Matrix multiplication on heterogeneous platforms," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 12, no. 10, pp. 1033–1051, 2001.

[3] A. Kalinov and A. Lastovetsky, "Heterogeneous distribution of computations solving linear algebra problems on networks of heterogeneous computers," *J. Parallel Distrib. Comput.*, vol. 61, no. 4, Apr. 2001.

[4] A. L. Lastovetsky and R. Reddy, "Data partitioning with a realistic performance model of networks of heterogeneous computers," in *Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International*. IEEE, 2004, p. 104.

[5] A. Lastovetsky and R. Reddy, "Data partitioning with a functional performance model of heterogeneous processors," *International Journal of High Performance Computing Applications*, vol. 21, no. 1, pp. 76–90, 2007.

[6] A. Lastovetsky, L. Szustak, and R. Wyrzykowski, "Model-based optimization of EULAG kernel on Intel Xeon Phi through load imbalancing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 3, pp. 787–797, 2017.

[7] A. Lastovetsky and R. Reddy, "New model-based methods and algorithms for performance and energy optimization of data parallel applications on homogeneous multicore clusters," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1119–1133, 2017.

[8] OpenBLAS, "OpenBLAS: An optimized BLAS library," 2016. [Online]. Available: http://www.openblas.net/

[9] FFTW, "FFTW: A fast, free c FFT library," 2016. [Online]. Available: http://www.fftw.org/

[10] A. Lastovetsky and R. Reddy, "Data distribution for dense factorization on computers with memory heterogeneity," *Parallel Computing*, vol. 33, no. 12, Dec. 2007.

[11] A. Ilić, F. Pratas, P. Trancoso, and L. Sousa, "High-performance computing on heterogeneous systems: Database queries on CPU and GPU," *High Performance Scientific Computing with Special Emphasis on Current Capabilities and Future Perspectives*, pp. 202–222, 2010.

[12] D. Clarke, A. Lastovetsky, and V. Rychkov, "Dynamic load balancing of parallel computational iterative routines on highly heterogeneous HPC platforms," *Parallel Processing Letters*, vol. 21, no. 02, pp. 195–217, 2011.

[13] D. Clarke, A. L. Lastovetsky, and V. Rychkov, "Column-based matrix partitioning for parallel matrix multiplication on heterogeneous processors based on functional performance models," in *Euro-Par 2011: Parallel Processing Workshops*, ser. Lecture Notes in Computer Science, vol. 7155. Springer-Verlag, 2012.

[14] X. Liu, Z. Zhong, and K. Xu, "A hybrid solution method for CFD applications on GPU-accelerated hybrid HPC platforms," *Future Generation Computer Systems*, vol. 56, pp. 759–765, 2016.

[15] M. Radmanović, D. Gajić, and R. Stanković, "Efficient computation of galois field expressions on hybrid CPU-GPU platforms." *Journal of Multiple-Valued Logic & Soft Computing*, vol. 26, 2016.

[16] A. Ilic and L. Sousa, "Simultaneous multi-level divisible load balancing for heterogeneous desktop systems," in *Parallel and Distributed Processing with Applications (ISPA), 2012 IEEE 10th International Symposium on*. IEEE, 2012, pp. 683–690.

[17] J. Colaço, A. Matoga, A. Ilic, N. Roma, P. Tomás, and R. Chaves, "Transparent application acceleration by intelligent scheduling of shared library calls on heterogeneous systems," in *Parallel Processing and Applied Mathematics*. Springer, 2013, pp. 693–703.

[18] V. Cardellini, A. Fanfarillo, and S. Filippone, "Heterogeneous sparse matrix computations on hybrid GPU/CPU platforms." in *PARCO*, 2013, pp. 203–212.

[19] S. Williams, A. Waterman, and D. Patterson, "Roofline: an insightful visual performance model for multicore architectures," *Communications of the ACM*, vol. 52, no. 4, pp. 65–76, 2009.

[20] L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *Computer*, no. 12, pp. 33–37, 2007.

[21] DOE, "The opportunities and challenges of exascale computing," 2010. [Online]. Available: http://science.energy.gov/~/media/ascr/ /pdf/reports/Exascale_subcommittee_report.pdf

[22] L. Smarr, "Project greenlight: Optimizing cyber-infrastructure for a carbon-constrained world," *Computer*, vol. 43, no. 1, pp. 22–27, Jan 2010.

[23] N. Muralimanohar, R. Balasubramonian, and N. Jouppi, "Optimizing NUCA organizations and wiring alternatives for large caches with CACTI 6.0," in *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO 40. IEEE Computer Society, 2007.

[24] N. Vijaykrishnan, M. Kandemir, M. J. Irwin, H. S. Kim, and W. Ye, "Energy-driven integrated hardware-software optimizations using SimplePower," in *Proceedings of the 27th Annual International Symposium on Computer Architecture*, ser. ISCA '00. ACM, 2000.

[25] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A framework for architectural-level power analysis and optimizations," in *Proceedings of the 27th Annual International Symposium on Computer Architecture*, ser. ISCA '00. ACM, 2000.

[26] D. Brooks, M. Martonosi, J.-D. Wellman, and P. Bose, "Power-performance modeling and tradeoff analysis for a high end microprocessor," in *Proceedings of the First International Workshop on Power-Aware Computer Systems-Revised Papers*, ser. PACS '00. Springer-Verlag, 2001.

[27] F. Bellosa, "The benefits of event: driven energy accounting in power-sensitive systems," in *Proceedings of the 9th workshop on ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system*. ACM, 2000.

[28] C. Isci and M. Martonosi, "Runtime power monitoring in high-end processors: Methodology and empirical data," in *36th annual IEEE/ACM International Symposium on Microarchitecture*. IEEE Computer Society, 2003, p. 93.

[29] T. Li and L. K. John, "Run-time modeling and estimation of operating system power consumption," *SIGMETRICS Perform. Eval. Rev.*, vol. 31, no. 1, pp. 160–171, Jun. 2003.

[30] B. C. Lee and D. M. Brooks, "Accurate and efficient regression modeling for microarchitectural performance and power prediction," *SIGARCH Comput. Archit. News*, vol. 34, no. 5, pp. 185–194, Oct. 2006.

[31] T. Heath, B. Diniz, B. Horizonte, E. V. Carrera, and R. Bianchini, "Energy conservation in heterogeneous server clusters," in *10th ACM*

SIGPLAN symposium on Principles and practice of parallel programming (PPoPP). ACM, 2005, pp. 186–195.

[32] D. Economou, S. Rivoire, C. Kozyrakis, and P. Ranganathan, "Full-system power analysis and modeling for server environments," in *In Proceedings of Workshop on Modeling, Benchmarking, and Simulation*, 2006, pp. 70–77.

[33] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *34th Annual International Symposium on Computer architecture*. ACM, 2007, pp. 13–23.

[34] A. Kansal and F. Zhao, "Fine-grained energy profiling for power-aware application design," *ACM SIGMETRICS Performance Evaluation Review*, vol. 36, no. 2, p. 26, Aug. 2008.

[35] S. Rivoire, P. Ranganathan, and C. Kozyrakis, "A comparison of high-level full-system power models," in *Proceedings of the 2008 Conference on Power Aware Computing and Systems*, ser. HotPower'08. USENIX Association, 2008.

[36] S. Rivoire, "Models and metrics for energy-efficient computer systems. phd thesis." Stanford University, Stanford, California, 2008.

[37] K. Singh, M. Bhadauria, and S. A. McKee, "Real time power estimation and thread scheduling via performance counters," *SIGARCH Comput. Archit. News*, vol. 37, no. 2, pp. 46–55, Jul. 2009.

[38] M. D. Powell, A. Biswas, J. S. Emer, S. S. Mukherjee, B. R. Sheikh, and S. Yardi, "CAMP: A technique to estimate per-structure power at run-time using a few simple parameters," in *2009 IEEE 15th International Symposium on High Performance Computer Architecture*, Feb 2009, pp. 289–300.

[39] B. Goel, S. A. McKee, R. Gioiosa, K. Singh, M. Bhadauria, and M. Cesati, "Portable, scalable, per-core power estimation for intelligent resource management." Green Computing Conference, 2010 International, 2010-08-16 2010.

[40] H. Wang, Q. Jing, R. Chen, B. He, Z. Qian, and L. Zhou, "Distributed systems meet economics: pricing in the cloud," in *Proceedings of the 2nd USENIX conference on Hot topics in cloud computing*. USENIX Association, 2010.

[41] R. Basmadjian, N. Ali, F. Niedermeier, H. de Meer, and G. Giuliani, "A methodology to predict the power consumption of servers in data centres," in *2nd International Conference on Energy-Efficient Computing and Networking*. ACM, 2011.

[42] W. L. Bircher and L. K. John, "Complete system power estimation using processor performance events," *IEEE Transactions on Computers*, vol. 61, no. 4, pp. 563–577, Apr. 2012.

[43] W. Dargie, "A stochastic model for estimating the power consumption of a processor," *IEEE Transactions on Computers*, vol. 64, no. 5, 2015.

[44] J. C. McCullough, Y. Agarwal, J. Chandrashekar, S. Kuppuswamy, A. C. Snoeren, and R. K. Gupta, "Evaluating the effectiveness of model-based power characterization," in *Proceedings of the 2011 USENIX Conference on USENIX Annual Technical Conference*, ser. USENIXATC'11. USENIX Association, 2011.

[45] D. Hackenberg, T. Ilsche, R. Schöne, D. Molka, M. Schmidt, and W. E. Nagel, "Power measurement techniques on standard compute nodes: A quantitative comparison," in *Performance analysis of systems and software (ISPASS), 2013 IEEE international symposium on*. IEEE, 2013, pp. 194–204.

[46] E. Rotem, A. Naveh, A. Ananthakrishnan, E. Weissmann, and D. Rajwan, "Power-Management architecture of the intel microarchitecture Code-Named sandy bridge," *IEEE Micro*, vol. 32, no. 2, pp. 20–27, March 2012.

[47] K. O'Brien, I. Pietri, R. Reddy, A. Lastovetsky, and R. Sakellariou, "A survey of power and energy predictive models in HPC systems and applications," *ACM Computing Surveys*, vol. 50, no. 3, 2017.

[48] PAPI, "Performance application programming interface 5.4.1," 2015. [Online]. Available: http://icl.cs.utk.edu/papi/

[49] J. Treibig, G. Hager, and G. Wellein, "Likwid: A lightweight performance-oriented tool suite for x86 multicore environments," in *Parallel Processing Workshops (ICPPW), 2010 39th International Conference on*. IEEE, 2010, pp. 207–216.

[50] Likwid, "Architecture specific notes for intel haswell," 2017. [Online]. Available: https://github.com/RRZE-HPC/likwid/wiki/Haswell

[51] P. Gschwandtner, M. Knobloch, B. Mohr, D. Pleiter, and T. Fahringer, "Modeling CPU energy consumption of hpc applications on the IBM POWER7," in *Parallel, Distributed and Network-Based Processing (PDP), 2014 22nd Euromicro International Conference on*. IEEE, 2014, pp. 536–543.

[52] J. Haj-Yihia, A. Yasin, Y. B. Asher, and A. Mendelson, "Fine-grain power breakdown of modern out-of-order cores and its implications on skylake-based systems," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 13, no. 4, p. 56, 2016.

[53] K. Singh, M. Bhadauria, and S. A. McKee, "Real time power estimation and thread scheduling via performance counters," *ACM SIGARCH Computer Architecture News*, vol. 37, no. 2, pp. 46–55, 2009. [Online]. Available: http://doi.org/10.1145/1577129.1577137

[54] C. Lively, X. Wu, V. Taylor, S. Moore, H.-C. Chang, C.-Y. Su, and K. Cameron, "Power-aware predictive models of hybrid (mpi/openmp) scientific applications on multicore systems," *Computer Science-Research and Development*, vol. 27, no. 4, pp. 245–253, 2012. [Online]. Available: http://doi.org/10.1007/s00450-011-0190-0

[55] S. Song, C. Su, B. Rountree, and K. W. Cameron, "A simplified and accurate model of power-performance efficiency on emergent GPU ar-

chitectures," in *27th IEEE International Parallel & Distributed Processing Symposium (IPDPS)*. IEEE Computer Society, 2013, pp. 673–686.

[56] M. Witkowski, A. Oleksiak, T. Piontek, and J. Weglarz, "Practical power consumption estimation for real life HPC applications," *Future Gener. Comput. Syst.*, vol. 29, no. 1, Jan. 2013.

[57] M. Jarus, A. Oleksiak, T. Piontek, and J. Węglarz, "Runtime power usage estimation of HPC servers for various classes of real-life applications," *Future Generation Computer Systems*, vol. 36, 2014.

[58] X. Wu, V. Taylor, J. Cook, and P. J. Mucci, "Using Performance-Power modeling to improve energy efficiency of HPC applications," *Computer*, vol. 49, no. 10, pp. 20–29, 2016.

[59] M. Chadha, T. Ilsche, M. Bielert, and W. E. Nagel, "A statistical approach to power estimation for x86 processors," in *Parallel and Distributed Processing Symposium Workshops (IPDPSW), 2017 IEEE International*. IEEE, 2017, pp. 1012–1019. [Online]. Available: http://doi.org/10.1109/IPDPSW.2017.98

[60] A. Shahid, M. Fahad, R. Reddy, and A. Lastovetsky, "Additivity: A selection criterion for performance events for reliable energy predictive modeling," *Supercomputing Frontiers and Innovations*, vol. 4, no. 4, 2017.

[61] CUPTI, "Cuda profiling tools interface," 2017. [Online]. Available: https://developer.nvidia.com/cuda-profiling-tools-interface

[62] IntelPCM, "Intel® performance counter monitor - a better way to measure cpu utilization." 2012. [Online]. Available: https://software.intel.com/en-us/articles/intel-performance-counter-monitor

[63] P. Wiki, "perf: Linux profiling with performance counters," 2017. [Online]. Available: https://perf.wiki.kernel.org/index.php/Main_Page

[64] I. Kadayif, T. Chinoda, M. Kandemir, N. Vijaykirsnan, M. J. Irwin, and A. Sivasubramaniam, "vec: Virtual energy counters," in *Proceedings of the 2001 ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering*, ser. PASTE '01. ACM, 2001, pp. 28–31. [Online]. Available: http://doi.org/10.1145/379605.379639

[65] 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*. ACM, 2010, pp. 147–158.

[66] R. Basmadjian and H. de Meer, "Evaluating and modeling power consumption of multi-core processors," in *Future Energy Systems: Where Energy, Computing and Communication Meet (e-Energy), 2012 Third International Conference on*, May 2012, pp. 1–10.

[67] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, "The McPAT framework for multicore and manycore architectures: Simultaneously modeling power, area, and timing," *ACM Trans. Archit. Code Optim.*, vol. 10, no. 1, 2013.

[68] S. L. Xi, H. Jacobson, P. Bose, G.-Y. Wei, and D. Brooks, "Quantifying sources of error in McPAT and potential impacts on architectural studies," in *High Performance Computer Architecture (HPCA), 2015 IEEE 21st International Symposium on*. IEEE, 2015, pp. 577–589.

[69] A. Lastovetsky and R. Reddy, "New model-based methods and algorithms for performance and energy optimization of data parallel applications on homogeneous multicore clusters," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1119–1133, 2017.

[70] H. Hong, Sunpyand Kim, "An integrated GPU power and performance model," *SIGARCH Comput. Archit. News*, vol. 38, no. 3, 2010.

[71] H. Nagasaka, N. Maruyama, A. Nukada, T. Endo, and S. Matsuoka, "Statistical power modeling of GPU kernels using performance counters," in *International Green Computing Conference and Workshops (IGCC)*. IEEE, 2010.

[72] Y. S. Shao and D. Brooks, "Energy characterization and instruction-level energy model of Intel's Xeon Phi processor," in *Proceedings of the 2013 International Symposium on Low Power Electronics and Design*, ser. ISLPED '13. IEEE Press, 2013.

[73] Z. Al-Khatib and S. Abdi, "Operand-value-based modeling of dynamic energy consumption of soft processors in FPGA," in *International Symposium on Applied Reconfigurable Computing*. Springer, 2015, pp. 65–76.

[74] HCL, "SLOPE-PMC: Towards the automation of pmcs collection for intel based multicore platforms," 2017. [Online]. Available: https://git.ucd.ie/hcl/SLOPE/tree/master/SLOPE-PMC

[75] Intel Optimzed HPCG, "Overview of the intel optimized hpcg." [Online]. Available: https://software.intel.com/en-us/node/599524

[76] A. Waterland, "Stress," 2001. [Online]. Available: https://people.seas.harvard.edu/{\~}apw/stress/

[77] HCL, "HCLWattsUp: API for power and energy measurements using WattsUp Pro Meter," 2016. [Online]. Available: http://git.ucd.ie/hcl/hclwattsup

[78] M. F. Dolz, J. Kunkel, K. Chasapis, and S. Catalán, "An analytical methodology to derive power models based on hardware and software metrics," *Computer Science-Research and Development*, vol. 31, no. 4, pp. 165–174, 2016. [Online]. Available: http://doi.org/10.1007/s00450-015-0298-8

[79] S. Wang, *Software power analysis and optimization for power-aware multicore systems*. Wayne State University, 2014.