UiT
THE ARCTIC
UNIVERSITY
OF NORWAY

Department of Computer Science

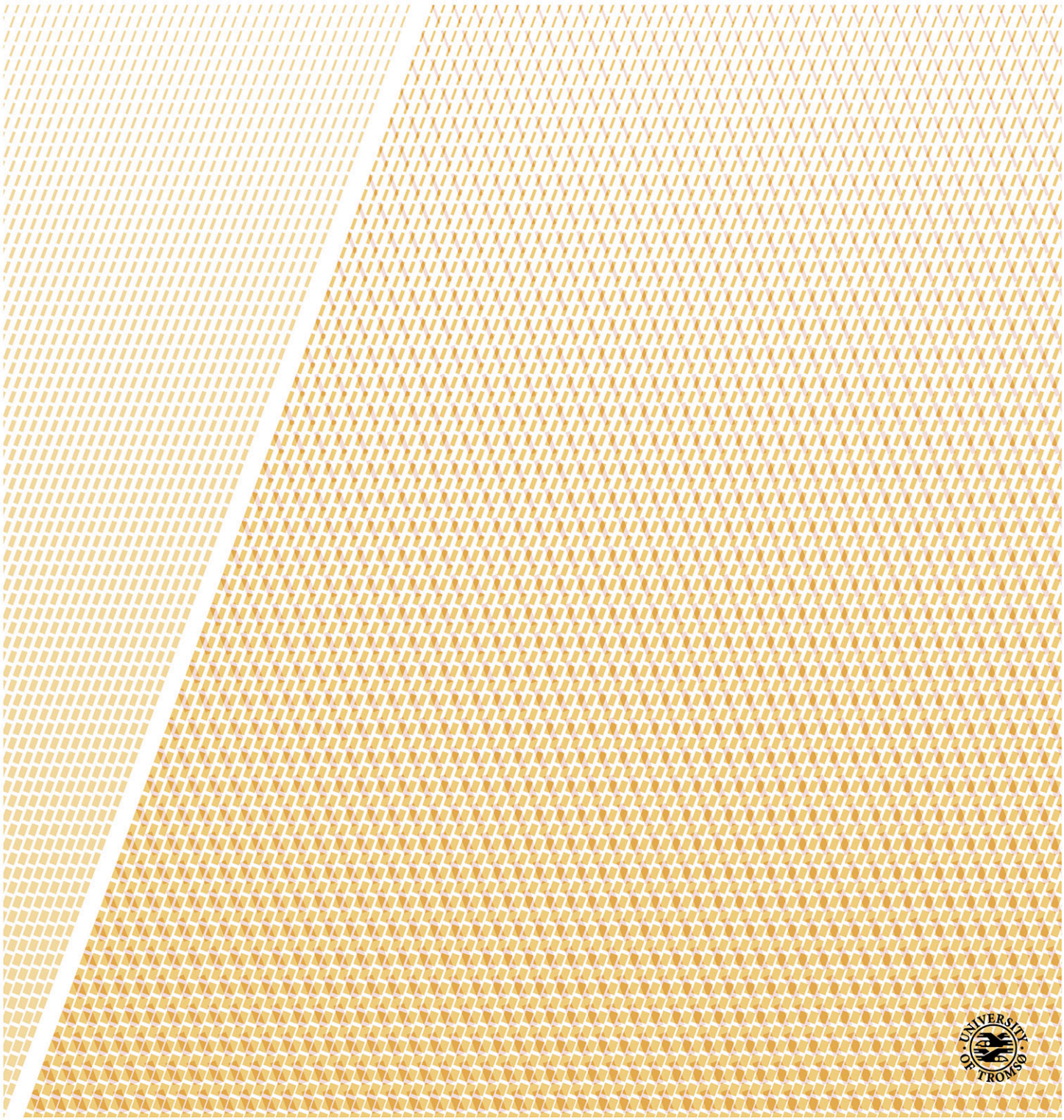
Faculty of Science and Technology

Modeling Energy Consumption of Computing Systems: from Homogeneous to Heterogeneous Systems

—

Vi Ngoc-Nha Tran

A dissertation for the degree of Philosophiae Doctor – September 2018



MODELING ENERGY CONSUMPTION OF COMPUTING SYSTEMS: FROM HOMOGENEOUS TO HETEROGENEOUS SYSTEMS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE FACULTY OF SCIENCE AND TECHNOLOGY
OF UiT THE ARCTIC UNIVERSITY OF NORWAY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Vi Ngoc-Nha Tran

Abstract

Nowadays, reducing energy consumption and improving energy efficiency of computing systems become ones of the main research topics in computer science. In order to improve energy efficiency, it is important to understand how computing systems consume energy and to characterize their energy consumption when running applications. Power and energy models are the essential tools to provide the prediction of the power and energy consumption of computing systems and insight into how they consume power and energy.

Devising models which can provide an accurate prediction of energy consumption requires the detailed understanding of the underlying platform and the communication and computation patterns of the considered application. Therefore, it is challenging to build accurate power and energy models that can be used for general devices and general applications.

This thesis addresses the above challenge by developing three approaches of devising power and energy models, varying from homogeneous systems including one type of devices (e.g., CPU, GPU, Ultra Low Power embedded system) to heterogeneous systems including several types of devices with different architectures.

- The thesis developed new fine-grained power models supporting architecture-application co-design by considering both platform and application properties. The models were trained and validated with data from a set of micro-benchmarks and application kernels on Movidius Myriad, an ultra-low power embedded system. The model predicted power consumption within 12% deviation from the real power consumption. We also proposed and validated a framework predicting when to apply race-to-halt (RTH) strategy to a given application.

- The thesis devised ICE, new energy complexity models for parallel (multi-threaded) algorithms that were validated on real multicore platforms and applicable to a wide range of parallel algorithms. We presented two case studies using the complexity models to characterize and compare the energy consumption of sparse matrix-vector multiplication and matrix multiplication kernels according to the three aspects: different algorithms, different input matrix types and different platforms. The experimental results regarding which algorithm consumes more energy with different inputs on different platforms confirmed the prediction by the new

models. The study also provided the platform parameters of the ICE models for eleven platforms including HPC, accelerator and embedded platforms to improve the model usability and accuracy.

- The thesis proposed REOH, the holistic tuning approach to choose the most energy-efficient configurations for heterogeneous systems including several types of devices with different architectures (e.g., CPUs, GPUs). REOH uses probabilistic network to predict the most energy-efficient configuration (i.e., which platform and its setting) of a heterogeneous system for running a given application. Based on the REOH approach, we developed an open-source energy-optimizing runtime framework for selecting an energy efficient configuration of a heterogeneous system for a given application at runtime.

Acknowledgments

This thesis work is only possible with the support of several people to whom I am deeply grateful. First and foremost, I would like to express my sincere gratitude to my advisor Phuong H. Ha for his guiding, support and encouragement during my PhD study.

I am also thankful to my second advisor Otto Anshus and Alexander Horsch for the helpful comments and discussions on the work presented in this thesis.

I would like to thank my thesis committee members: Per Stenström, Magnus Jahre and Randi Karlsen for the invaluable comments and suggestions to improve my thesis work.

I am grateful to the staffs of the Department of Computer Science including Svein Tore Jensen, Jan Fuglesteg, Maria Hauglann, Kai-Even Nilssen, Ken-Arne Jensen and Jon Ivar Kristiansen for their administrative and technical supports. A great help has been done to facilitate my research work.

I am also grateful to the AGC members: Ibrahim Umar, Saeed Shariati, Pradeep Kumar and Tommy Oines for all the discussions and the good time in our lab.

I would also like to thank everyone who worked with me in the EU EXCESS project, especially Christoph Kessler and Brendan Barry from whom I received a huge amount of constructive input on the deliverable works.

I own a deep gratitude to my family: my parents, my sister, my husband, and son for their unconditional love and support. They are the motivation for me to become a better person every day.

List of Papers and Reports

This thesis is based on the work described in the following publications and reports.

Chapter 3 revises the publications:

- [77]: Vi N.N. Tran, Brendan Barry, and Phuong Ha. Power models supporting energy-efficient co-design on ultra-low power embedded systems. In 2016 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS), pages 39-46, 2016.
- [80]: Vi N.N. Tran, B. Barry and Phuong Ha. RTHpower: Accurate fine-grained power models for predicting race-to-halt effect on ultra-low power embedded systems. In 2016 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), pages 155-156, 2016.

Chapter 4 revises the publication:

- [78]: Vi N.N. Tran and Phuong Ha. Ice: A general and validated energy complexity model for multi- threaded algorithms. In 2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS), pages 1041-1048, 2016.

Chapter 5 revises the publication:

- [79] Vi N.N. Tran, Tommy Oines, Alexander Horsch and Phuong Ha. REOH: Using Probabilistic Network for Runtime Energy Optimization of Heterogeneous Systems. In 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS), pages to appear.

Workshops and Technical Reports:

- Vi N.N. Tran, Phuong Ha. ICE: A General and Validated Energy Complexity Model for Multithreaded Algorithms. The 9th Nordic Workshop on Multi-core Computing (MCC '16)
- Yosandra Sandoval, Dennis Hoppe, Dmitry Khabi, Micheal Gienger, ChristophKessler, Lu Li, Usman Dastgeer, Vi N.N Tran, Ibrahim Umar, Phuong Ha, Philippas Tsigas, Anders Gidenstam, Ivan Walulya, Paul Renaud-Goud. EXCESS: Execution Models for Energy-Efficient

Computing Systems. fEEDBACk Workshop Energy Efficient Distributed and Parallel Computing (fEEDBACk '15)

- [35]: Phuong Ha, Vi N.N. Tran, Ibrahim Umar, Philippas Tsigas, Anders Gidenstam, Paul Renaud-Goud, Ivan Walulya, and Aras Atalar. D2.1 Models for energy consumption of data structures and algorithms. Technical Report FP7-611183 D2.1, EU FP7 Project EXCESS, August 2014.

- [32]: Phuong Ha, Vi N.N. Tran, Ibrahim Umar, Aras Atalar, Anders Gidenstam, Paul Renaud-Goud, and Philippas Tsigas. D2.2 White-box methodologies, programming abstractions and libraries. Technical Report FP7-611183 D2.2, EU FP7 Project EXCESS, February 2015.

- [34]: Phuong Ha, Vi N.N. Tran, Ibrahim Umar, Aras Atalar, Anders Gidenstam, Paul Renaud-Goud, Philippas Tsigas, and Ivan Walulya. D2.3 power models, energy models and libraries for energy- efficient concurrent data structures and algorithms. Technical Report FP7-611183 D2.3, EU FP7 Project EXCESS, February 2016.

- [33]: Phuong Ha, Vi N.N. Tran, Ibrahim Umar, Aras Atalar, Anders Gidenstam, Paul Renaud-Goud, Philippas Tsigas, and Ivan Walulya. D2.3 d2.4 report on the final prototype of programming abstractions for energy-efficient inter-process communication. Technical Report FP7-611183 D2.4, EU FP7 Project EXCESS, August 2016.

- [44]: Christoph Kessler, Lu Li, Usman Dastgeer, Philippas Tsigas, Anders Gidenstam, Paul Renaud- Goud, Ivan Walulya, Aras Atalar, David Moloney, Phuong Ha Hoai, and Vi N.N. Tran. D1.1 Early validation of system-wide energy compositionality and affecting factors on the EXCESS plat- forms. Technical Report FP7-611183 D1.1, EU FP7 Project EXCESS, April 2014.

- [43]: Christoph Kessler, Lu Li, Usman Dastgeer, Rosandra Cuello, Oskar Sjostrom, Phuong Ha Hoai, and Vi N.N. Tran. D1.3 Energy-tuneable domain-specific language/library for linear system solving. Technical Report FP7-611183 D1.3, EU FP7 Project EXCESS, February 2015.

- [45]: Dmitry Khabi, Vi N.N. Tran, and Ivan Walulya. D5.4 Report on the first evaluation results and discussion. Technical Report FP7-611183 D5.4, EU FP7 Project EXCESS, August 2015.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Along with performance optimization, energy efficiency is one of the main concerns of computing systems. Reducing energy consumption of computing systems, varying from homogeneous systems such as embedded systems, CPUs, GPUs to heterogeneous systems including different devices with different architectures becomes one of the top challenges in computer science.

Significant efforts have been focused on architectural energy-saving techniques. To further reduce the energy consumption of future computing systems, the co-design of software and hardware considering both applications and systems is essential to exploit both software and hardware energy-saving techniques [42].

One of the key research directions to improve energy efficiency is to understand how much energy a computing system consumes and characterize their energy consumption. By characterizing the energy consumption of computing systems, researchers and practitioners can design and implement new approaches to reduce the energy consumed by a certain algorithm on a specific platform.

The energy and power consumption of computing systems can be either measured by integrated sensors or external multi-meters or estimated by models. Energy and power measurement equipment and sensors are not always available and can be costly to deploy and set up. Therefore, energy and power models are the alternative and convenient methods to estimate the energy consumption of an application on a computing system [67]. Devising power and energy models is also crucial to gain insights into how a computer system consumes power and energy.

1.1 Research Questions

1.1.1 Research Question 1

Significant efforts have been devoted to devising power and energy models of computing systems, resulting in several seminal papers in the literature, such as [41, 53, 55, 10, 19, 18, 46, 47, 39, 63, 73]

modeling power of architectures or applications.

Jacobson et al. [41] proposed accurate power modeling methodologies for POWER-family processors while GPUWattch and McPAT are robust power models for GPUs and CPUs. Alonso et al. [10] proposed energy models for three key dense-matrix factorizations. Roofline model of energy [19, 18] considers both algorithmic and platform properties. However, the Roofline model does not consider the number of cores running applications as a model parameter (i.e., coarse-grained models). Theoretical models by Korthikanti et al. [47, 46] were based on strong theoretical assumptions and are not yet validated on real platforms. Koala model [73] requires the system supported dynamic voltage and frequency scaling (DVFS) and short frequency switching delay in order to gain energy saving from its methodology. However, only two x86-based platforms among 10 validated platforms gained energy saving results which are presented in the paper. Imes et al. [39] provided a portable approach to make real-time decision and run the chosen configuration to minimize energy consumption. However, the approach requires systems supporting hardware resource (e.g., model-specific register) to expose energy data to the software during run-time. Mishra et al. [63] used a probabilistic modeling approach to find the most energy-efficient configuration by combining online and offline machine-learning approaches. This approach requires a significant amount of data collected to feed to its probabilistic network.

Recently, novel and specific-purpose systems such as ultra-low power (ULP) embedded systems have become popular in the scientific community and industry, especially in media and wearable computing. ULP embedded systems have different architectures from the general-purpose architectures (e.g., CPU and GPU). As a result, the approach to model the power of ULP systems needs to be customized for their architecture. ULP systems can achieve low energy per instruction down to a few pJ [9]. Alioto [9] mentioned that techniques such as pipe-lining, hardware replication, ultra-low-voltage memory design, and leakage-reducing make a system ultra-low power. In order to model ULP systems where energy per instruction can be as low as few pJ, more accurate fine-grained approaches are needed. For instance, the dynamic power P^{dyn} of operations in Table 3.2, which is as low as 13 mW, cannot be measured by using the prior coarse-grained approaches [19, 18].

For embedded systems which has real-time constraint and limited energy supply, two of the most popular strategies to reduce the energy consumption are Dynamic Voltage and Frequency Scaling (DVFS) [51] and race-to-halt (RTH) (i.e, systems run at higher frequency to finish as soon as possible, and then put certain hardware parts to sleep to save energy) [13]. These two techniques are explained in Chapter 2. For new embedded systems which do not support DVFS features such as Movidius Myriad [40], RTH is one of the remaining choices for saving energy. RTH theory is used to let the CPU work at the highest performance levels then go back to a low energy-draw state. The process is repeated multiple times during program execution. In fact, Myriad supports a power management feature to power on/off individual cores. However, to the best of our knowledge, there is no fine-grained power model that supports investigating the trade-off between performance and

energy consumption on ULP embedded systems and whether the RTH strategy that is widely used in high-performance computing (HPC) systems is still applicable to ULP embedded systems.

The first part of this thesis work investigates the modeling methodology to answer the research question: *"RQ1: How to accurately model and estimate the power and energy consumptions and support energy-efficient co-design of ultra-low power embedded systems?"*

1.1.2 Research Question 2

The models which are able to estimate absolute values of power and energy consumption from RQ1 however, requires a significant detailed understanding of the targeted platform and its components to develop a set of micro-benchmarks. For other domains such as algorithm design, the absolute values of energy consumption estimation are not required. Instead, an analysis tool to provide an understanding of how an algorithm consumes energy as the input grows is more essential. In the next work of this thesis, we aim to provide the understanding of how an algorithm consumes energy via energy complexity models.

Understanding the energy complexity of algorithms is crucially important to improve the energy efficiency of algorithms [82, 81, 83, 49] and reduce the energy consumption of computing systems [80, 77, 50].

However, there are no analytic models for multithreaded algorithms that are both applicable to a wide range of algorithms and comprehensively validated yet (cf. Table 1.1). The existing *parallel* energy models are either theoretical studies without validation or only applicable for specific algorithms. Modeling energy consumption of *parallel* algorithms is difficult since the energy models must take into account the complexity of both parallel algorithms and parallel platforms. The algorithm complexity results from parallel computation, concurrent memory accesses and inter-process communication. The platform complexity results from multicore architectures with a deep memory hierarchy.

The existing models and their classification are summarized in Table 1.1 by three aspects: i) ability to analyze the energy complexity of parallel algorithms (i.e. Energy complexity analysis for parallel algorithms), ii) applicability to a wide range of algorithms (i.e., Algorithm generality), and iii) model validation (i.e., Validation). To the best of our knowledge, the energy model that covers all three aspects: Energy complexity analysis for parallel algorithms, Algorithm generality and Validation is missing.

The second study of this thesis answers the energy complexity question: *"RQ2: Given two parallel algorithms A and B for a given problem, how to identify which algorithm consumes less energy analytically?"*

Table 1.1: Energy Model Summary

Study	**Energy complexity analysis for parallel algorithms**	**Algorithm generality**	**Validation**
LEO [63]	No	General	Yes
POET [39]	No	General	Yes
Koala [73]	No	General	Yes
Roofline [19, 18]	No	General	Yes
Energy scalability [46, 47]	Yes	General	No
Sequential energy complexity [70]	No	General	Yes
Alonso et al. [10]	Yes	Algorithm-specific	Yes
Malossi et al. [62]	Yes	Algorithm-specific	Yes

To the best of our knowledge, the ICE model is the first *validated model that supports* energy complexity analysis for *general multithreaded algorithms.*

1.1.3 Research Question 3

So far, both the research questions RQ1 and RQ2 addresses the energy modeling questions for accurate models and complexity models conducted on homogeneous systems including one type of devices (e.g., embedded systems, CPU or GPU). Modeling the energy consumption of applications running on heterogeneous systems including different types of devices are more complex and challenging. In the next modeling approach, we want to estimate the energy consumption of an application running on heterogeneous systems and identify the system configurations to run the application to achieve the most energy efficiency.

The factors that have impacts on the application performance, energy-efficiency and its optimization strategies are algorithm design, implementation (i.e., control flow, memory types, memory access pattern, and instruction count), and its execution configuration [24]. When an application runs on a heterogeneous system, one of the strategies to reduce energy consumption is to run the application with an appropriate system configuration.

Several attempts [60, 92, 38, 63, 17, 6, 65, 61, 29, 58, 85] have been made to find the best configurations to run an application to achieve energy efficiency. However, available tuning approaches are mostly conducted for homogeneous systems while little research considers heterogeneous systems including several platform components (e.g., CPUs and GPUs) with different types of processing units and different architectures.

Table 1.2 summarizes the studies to optimize energy efficiency by choosing an appropriate configuration of computing systems for a given application. Table 1.2 lists the related works according to the four aspects: the optimization goal (i.e, Optimization), whether the optimization object is

configuration or code variant (i.e., Object), whether the targeted system is homogeneous or heterogeneous (i.e., System), and whether the approach is applicable to general or specific applications (i.e., Application). The details of the related work are described in Section 5.5.

The main goal of existing tuning approaches is to improve energy-efficiency. However, the existing models are mostly built for homogeneous systems, which has only one type of devices such as GPU [17, 6, 65, 29, 61, 85] or CPU [38, 92, 63]. There are also a set of studies [72, 91, 90] for heterogeneous systems (i.e., APUs) but they mainly focus on improving performance instead of energy-efficiency.

The existing heterogeneous approaches in the Table 1.2 are either for specific applications (i.e., iterative applications that can be divided to several iterations where execution time of the next iteration can be predicted based on the current iteration) [58, 59] or for finding a heterogeneous balance of datacenter [30] where the configuration at datacenter level is a mix of CPUs or microprocessors.

Among the available tuning approaches, probabilistic model-based approaches have their advantages of not requiring prior knowledge on the targeted application or the throughout understanding of system components like other approaches [65, 29]. By finding the similarity between a targeted application from sampling data and previously observed applications from training data, it can quickly provide the accurate estimation of energy consumption for the targeted application.

The previous probabilistic model-based approaches are only applicable to homogeneous systems (i.e., CPUs). Heterogeneous systems have complex structures containing different platform architectures (e.g., CPUs, GPUs, FPGAs, ASICs) where each platform has its own sets of settings and methods to change its configurations. Applying the probabilistic model-based approach [63] on each individual platform of a heterogeneous system requires the analysis of the available settings and a new configuration data for each platform. In the other words, it requires separated sets of training and sampling data, and separated runs of prediction for each platform. This results in more sampling runs than doing one prediction for a heterogeneous system with only one whole set of training and sampling data. Therefore, the probabilistic model based approaches for heterogeneous systems requires the analysis of the available settings of all included platforms within a heterogeneous system and finding the setting equivalence of one platform to another platform. The third part of this thesis aims to address the research question: *"RQ3: How to identify the most energy-efficient system configurations (i.e., platform and its setting) of a heterogeneous system containing platforms with different architectures to run the application?"*

1.2 Research Contributions

This thesis tackles the above three research questions by investigating and developing the three modeling approaches:

- Accurate Power Models Supporting Energy Efficient Co-design for Ultra-low Power Embedded Systems

Table 1.2: Auto-Tuning Framework

Study	**Optimization**	**Object**	**System**	**Application**
OSKI [84]	Time	Code variant	Homogeneous (i.e., CPU)	Specific (i.e., Sparse kernels)
Nitro [64]	Time	Code variant	Homogeneous	General
PowerCap [92]	Timeliness Energy-efficiency	Configuration	Homogeneous (i.e., CPU)	General
POET [38]	Energy-efficiency	Configuration	Homogeneous (i.e., CPU)	General
LEO [63]	Time Energy-efficiency	Configuration	Homogeneous (i.e., CPU)	General
HPC runtime framework [17]	Energy-efficiency	Configuration	Homogeneous (i.e., CPU)	General
GPU models [6]	Power	Configuration	Homogeneous (i.e., GPU)	General
CRISP [65]	Energy	Configuration	Homogeneous (i.e., GPGPU)	General
MPC [61]	Energy-efficiency	Configuration	Homogeneous (e.g., GPGPU)	General
GreenGPU [58, 59]	Energy-efficiency	Workload division Frequency	Heterogeneous (e.g., CPU and GPU)	Specific (i.e., Iterative applications)
GPGPU DVFS models [29]	Energy-efficiency	Configuration	Homogeneous (i.e., GPGPU)	General
GPGPU SVR models [85]	Energy-efficiency	Configuration	Homogeneous (i.e., GPGPU)	General
Market mechanism [30]	Service quality Energy-efficiency efficiency	High-level configurations (i.e., Datacenters)	Heterogeneous (e.g., CPUs and microprocessors) (e.g., CPU and GPU)	General

- Energy Complexity Models for Multithreaded Algorithms
- Runtime Energy Optimization for Heterogeneous Systems

In the remaining of this section, the brief descriptions of solutions and results to each of the three modeling approaches are described. The full details of the three modeling approaches can be found in Chapters 3, 4 and 5, respectively.

1.2.1 RQ1: Accurate Power Models Supporting Energy Efficient Co-design for Ultra-low Power Embedded Systems

In order to estimate the absolute power consumption of an application on ULP embedded system and investigate RTH strategy, we propose new RTHpower models which support architecture-application co-design by considering both platform and application properties. The RTHpower models are application-general since they characterize applications by their arithmetic intensity [87] which can be extracted from any application. The RTHpower models are also practical since they are built and validated on Movidius platform using application kernels. The main contributions of this modeling approach are three-fold as follows:

- We propose new application-general fine-grained power models (namely, RTHpower) that provide insights into how a given application consumes power and give hints to investigate the trade-offs between performance and power consumption on ULP embedded systems. The RTHpower models support co-design on ULP systems by considering three parameter groups: platform properties, application properties (e.g., arithmetic intensity and scalability) and execution settings (e.g., the number of cores executing a given application) (cf. Section 3.2).
- We validate the new RTHpower models on an ultra-low power embedded system, namely Movidius Myriad. The models are trained and validated with power data from different sets of micro-benchmarks, two computation kernels from Berkeley dwarfs [12] and one data-intensive kernel from Graph500 benchmarks [74]. The three chosen application kernels are dense matrix multiplication (Matmul), sparse matrix vector multiplication (SpMV) and breadth first search (BFS). The model validation has percentage error at most 8.5% for micro-benchmarks and 12% for application kernels (cf. Section 3.3).
- We investigate the RTH strategy on an ultra-low power embedded platform using the new RTHpower models. We propose a framework that is able to predict when to and when not to apply the RTH strategy in order to minimize energy consumption. We validate the framework using micro-benchmarks and application kernels. From our experiments, we show real scenarios when to use RTH and when not to use RTH. We can save up to 61% energy for dense matrix multiplication, 59% energy for SpMV by using RTH and up to 5% energy for BFS by not using RTH (cf. Section 3.4).

1.2.2 RQ2: Energy Complexity Models for Multithreaded Algorithms

The energy complexity model ICE proposed in this modeling approach is for general multithreaded algorithms and validated on three aspects: different algorithms for a given problem, different input types and different platforms. The proposed model is an analytic model which characterizes both algorithms (e.g., representing algorithms by their *work*, *span* and *I/O* complexity) and platforms (e.g., representing platforms by their static and dynamic energy of memory accesses and computational operations). By considering *work*, *span*, and *I/O* complexity, the new ICE model is applicable to any multithreaded algorithms.

Since the new ICE energy model focuses on analyzing the energy complexity of algorithms, the model does not give the estimation of absolute energy consumption. The new model, instead, provides the algorithm designers with the understanding of how an algorithm consumes energy and give insight into how to choose one algorithm over the others for different input types and platforms. The new ICE model is designed for analyzing the energy *complexity* of algorithms and therefore the model does not provide the estimation of absolute energy consumption. Hence, the details of underlying systems (e.g., runtime and architectures) are abstracted away to keep the ICE model simple and suitable for complexity analysis. O-notation represents an *asymptotic upper-bound* on energy complexity.

In this work, the following contributions have been made.

- Devising a new general energy model ICE for analyzing the energy complexity of a wide range of multithreaded algorithms based on their *work*, *span* and *I/O* complexity (cf. Section 4.2). The new ICE model abstracts away possible *multicore platforms* by their static and dynamic energy of computational operations and memory access. The new ICE model complements previous energy models such as energy roofline models [19, 18] that abstract away possible *algorithms* to analyze the energy consumption of different multicore platforms.

- Conducting two case studies (i.e., SpMV and matmul) to demonstrate how to apply the ICE model to find energy complexity of parallel algorithms. The selected parallel algorithms for SpMV are three algorithms: Compressed Sparse Column(CSC), Compressed Sparse Block(CSB) and Compressed Sparse Row(CSR)(cf. Section 4.3). The selected parallel algorithms for matmul are two algorithms: a basic matmul algorithm and a cache-oblivious algorithm (cf. Section 4.4).

- Validating the ICE energy complexity model with both data-intensive (i.e., SpMV) and computation-intensive (i.e., matmul) algorithms according to three aspects: different algorithms, different input types and different platforms. The results show the precise prediction on which validated SpMV algorithm (i.e., CSB or CSC) consumes more energy when using different matrix input types from Florida matrix collection [23] (cf. Section 4.5.6). The results also show the precise prediction on which validated matmul algorithm (i.e., basic or cache-oblivious) consumes more

energy (cf. Section 4.5.7). The model platform-related parameters for 11 platforms, including x86, ARM and GPU, are provided to facilitate the deployment of the ICE model. Moreover, the ICE models can also be applied to theoretical exascale systems and enable their energy complexity analysis.

1.2.3 RQ3: Using Probabilistic Network for Runtime Energy Optimization of Heterogeneous Systems

This study proposes holistic tuning approach based on probabilistic network to predict the most energy-efficient configuration of heterogeneous systems for a given application. Based on the application communication and computation patterns (i.e., Berkeley dwarfs [12], we choose the Rodinia benchmarks [4] for the experiments and devise a training data set. The objectives when choosing the benchmarks are to devise a training data set that covers a wide range of application patterns and characteristics.

In this modeling approach, we propose a way to unify the configurations of different platforms on a heterogeneous system in order to perform the prediction only once as compared to the previous approach for homogeneous systems. This way we save energy of the sampling runs. Even though we evaluate our probabilistic model-based approach (i.e., REOH) on a system containing CPU and GPU only, REOH is general for heterogeneous systems which contain any architectures (e.g., CPUs, GPUs, FPGAs, ASICS) where we can identify and change their configurations (i.e., the combination of number of cores, memory and frequency) in runtime.

We also provide an open-source energy-optimizing runtime framework to choose which configuration of a heterogeneous system to run a given application at runtime. Even though the open-source is for the experimented system including only one CPU and one GPU, the code is available and can be adjusted to heterogeneous systems containing other types of platforms as long as changing platform configurations during runtime is supported.

This study is for applications that run on one platform (e.g., CPU or GPU) at a time. The application has different executable files for different platforms (e.g., CPU or GPU) that can be chosen during runtime. For example, Rodinia benchmarks suite [4] supports programming models such as OpenCL which can provide different executable files of the same benchmark. This approach, however, can also apply to applications that can be divided to several phases. Each phase is wrapped in an executable file and can be considered as one application in REOH approach. Therefore, each phase of such applications only runs on one platform but the whole execution with different phases runs on several platforms.

The contributions of this study are as follows.

- Devise a new holistic tuning approach for heterogeneous systems using a probabilistic modeling approach, which is called REOH. In this study, we propose a method to unify the configurations

of different platform types (e.g., CPU and GPU), consider the total energy of both static and dynamic energy and devise a training data set containing 7074 samples by running a selected set of 18 applications based on the knowledge of application patterns from Berkeley dwarfs on a total of 393 system configurations.

- Validate the REOH approach on a heterogeneous system consisting of CPU and GPU, showing that REOH approach achieves the close energy consumption (i.e., within 5% different) to the optimal energy consumption by the brute-force approach when choosing the most energy-efficient system configuration for the applications while saving 17% number of sampling runs than the existing probabilistic network approaches [63].

- Develop an open-source energy-optimizing runtime framework for selecting an energy efficient configuration of a heterogeneous system for a given application at runtime. The framework takes as the input the executable files that the users want to run on a targeted heterogeneous system. Then the framework will choose an appropriate configuration of the targeted heterogeneous system to run the executable files energy-efficiently. This tool is provided as an open source for scientific research purposes.

1.3 Thesis Roadmap

The content of this thesis is organized as follows. Chapter 2 explains the background and important concepts mentioned in this thesis. The details of the three modeling approaches are reported in Chapter 3, 4 and 5. Chapter 3 describes the power models that provide the exact power estimation to support energy efficient co-design on ultra-low-power embedded systems. Chapter 4 presents the energy-complexity models to analyze the energy consumption of multithreaded algorithms. Chapter 5 explains the runtime energy optimization approach and framework to predict the most energy-efficient configurations for heterogeneous systems. Chapter 6 concludes the thesis and discusses the future work.

Chapter 2

Background

In this chapter, we give the descriptions of the concepts that the thesis work concerns. First, we explain the general concepts related to energy modeling including power, energy, energy efficiency, and the roles of energy models in Section 2.1. Second, the energy and power management techniques (i.e., DVFS and RTH) discussed in RQ1 are introduced in the Section 2.2. Then, the concepts related to parallel computing (i.e., multithreaded algorithms and application patterns) and used in RQ2 are described in Section 2.3. Finally, the concepts of homogeneous and heterogeneous computing systems mentioned in RQ3 are explained in Section 2.4.

2.1 Energy Modeling

2.1.1 Power, Energy and Energy Efficiency

Power in science is defined as the rate at which work is done per unit time and usually measured in watts. Power can be defined as $P = \frac{W}{T}$, with P denotes power, W denotes work and T denotes time.

Energy is measured in watt-hour (Wh) when the power of one watt running for one hour. Energy is defined as $E = P \times T$ where T is the period of time a power runs for.

Energy efficiency, according to the EU Energy Efficiency Directive, means "the ratio of output of performance, service, goods or energy, to input of energy" [76]. Examples of the mentioned output can be thermal comfort in a building; transport service of persons or of information as a service; and a smart phone as a good.

Since energy cost has increased dramatically and negatively impact the economy and ecology [93], improving energy efficiency is clearly a research emphasis.

2.1.2 Energy Models

For mobile and portable embedded systems, power and energy consumption is a major design constraint, where efficient power management affects the lifetime of battery. For high performance computing system, performance is also affected by energy-aware design.

Reducing energy consumption of computing systems has become one of the main research topics. Reducing energy consumption can be gained by thermal-aware hardware design or power-aware software design or the combination of both [76]. Energy-aware hardware design involves various levels from different hardware components such as memory hierarchies, interconnects and processor architecture, etc. Energy-aware software design also involves various levels, from operating systems to compiler and applications layers.

In order to improve energy efficiency and reduce the energy cost of computing systems, we need to understand how a computer system consumes energy when running different workloads. This understanding requires analysis tools to estimate how much energy a system consumes. Analysis tools can be performance or energy counter which are not always available. Modeling power and energy consumption is another alternative approach to estimate power and energy consumption. The models not only provide the estimation of power and energy cost, but also the understanding of how computing systems consume power and energy and the insight into how to reduce them.

2.2 Energy and Power Management Techniques

Traditionally, the power consumption of a CMOS integrated circuit is accounted by dynamic power and static leakage power consumption [51]. The dynamic power consumption is computed by Equation 2.1, where C is the capacitance of the transistor gates, f is the operating frequency and V is the operating voltage.

$$P = C \times f \times V^2 + P_{static} \tag{2.1}$$

2.2.1 Dynamic Voltage and Frequency Scaling

Dynamic voltage and frequency scaling (DVFS) reduce the operating frequency or the operating voltage of the processors in order to consume less power. In frequency scaling, the processor clock rate is reduced so that the processor consumes less power at the expense of reduced performance. When a frequency is reduced, the number of instructions run by processors per unit of time is reduced and therefore, performance decreases. In dynamic voltage scaling, the operating voltage is reduced so that the power consumption is also reduced. Frequency scaling and dynamic voltage scaling often work in conjunction since adjusting the operating frequency is related to the operating voltage. Voltage scaling is more advantageous because power consumed by a processor is directly proportional to the square of voltage values as in Equation 2.1.

Since there is static leakage power consumption, the reduced performance from reducing frequency or voltage increases static energy consumption. Therefore, DVFS is usually used when the workload is not CPU-bound. Previous research has proposed to use DVFS to reduce the energy consumption of processors [73, 88]. However, the energy advantage of using DVFS are diminishing in modern architectures due to several factors such as better memory performance, advanced idle/sleep modes and complexity of multi-core processors [51].

2.2.2 Sleep states/ Race-to-halt

Race-to-halt is another power management approach where workloads are run as fast as possible to finish earlier, then some parts of the hardware (e.g., processor, caches, DRAM) are put into sleep states or its lowest operating frequency to save energy. This process can repeat multiple times during a workload execution. That means the systems runs with its highest setting to finish the task, and then wait for another job without being halt. Race-to-halt aims to reduces the static leakage energy.

DVFS is usually used for memory-bound workload while Race-to-halt is used for CPU-bound workload. However, which power management approach is better depends on both workload patterns and the underlying hardware.

2.3 Parallel computing

A parallel computing system is a system containing and using multiple processors simultaneously to solve a computational problem by splitting a computing task into several subtasks and assign each processor (e.g., CPU or core) to solve each subtask.

In the scope of parallel computing, there are important concepts that are mentioned in this thesis, including multithreaded algorithms, application patterns, data-intensive and computation-intensive applications.

2.3.1 Multithreaded Algorithms

Multithreaded algorithms are algorithms that are designed for a computing system with multiple processors (e.g., CPU or core) and a shared memory. Multithreaded computation can be modeled by a computation DAG (Directed Acyclic Graph) represented by $G = (V, E)$ where V is a set of nodes represented for operations/instructions and E is a set of edges represented for the dependencies of the nodes [21]. Along with the definition of DAG, there are concepts of two metrics: **work** and **span**, which are the indications of the theoretical efficiency of a multithreaded algorithm. The **work** is the total time to execute the whole computation on one processor. The **span** is the time to execute the longest or the critical path in the DAG. The parallelism of the multithreaded computation is computed as the ratio of its **work** to its **span**.

2.3.2 Application Patterns

Asanovic et al.[12] have introduced classes (dwarfs) of computational methods which captures computation and communication common patterns of applications. They are the most common patterns in diverse sets of domains such as machine learning, graphics, database, etc. The classes are defined by the similarity in computation and data movement. Each dwarf is the high level of abstractions across a class of applications. The dwarfs and their example applications are as below:

- Dense Linear Algebra (E.g., Body Tracking, Kmeans)
- Sparse Linear Algebra (E.g., Support vector machines, quadratic programming)
- Spectral Methods (E.g., spectral clustering, FFT)
- N-Body Methods (E.g., Molecular dynamics)
- Structured Grids (E.g., GemsFDTD, Maxwell EM)
- Unstructured Grids (E.g., Belief propagation, Global illumination)
- Map Reduce (E.g.,Monte Carlo, Ray tracer)
- Combinational Logic (E.g., Hashing, IP Packet, Route Lookup)
- Graph Traversal (E.g., Bayesian networks, Decision trees)
- Dynamic Programming (E.g., Query optimization, SPEC Integer: Go)
- Backtrack and Branch+Bound (E.g., Kernel regression, 2D Path finding library)
- Construct Graphical Models (E.g., Hidden Markov models, Viterbi Decode)
- Finite State Machine (E.g., EEMBC Networking: QoS, SPECT Integer: text processing (perlbench))

Understanding whether the dwarfs are limited by computation or by memory is essential to make use of the architecture. This insight also helps to develop future architectures.

The applications can also be classified as data-intensive or compute-intensive. The applications considered as data-intensive when a limit factor of CPU power is the amount of data, the complexity of data and its changing speed [37]. An example of data-intensive application is sparse matrix multiplication which has a high demand for data transfer from memory. Compute-intensive applications are applications demanding high computation such as matrix multiplication.

2.4 Computing Systems

This section discusses the definitions of homogeneous and heterogeneous systems.

2.4.1 Homogeneous Systems

According to Lastovetsky et.al [8], there are three types of homogeneity:

- Homogeneous machine: a hardware whose each processor ensure the same storage presentation and guarantees the same results of operations on floating-point numbers.
- Homogeneous network: a collection of homogeneous machines where the communication layer among all processors ensures the exact transmittal of the floating-point values.
- Homogeneous computing environment: a platform where the softwares on each processor ensure the same storage representation and the same results of operations on floating-point numbers.

2.4.2 Heterogeneous Systems

Heterogeneous systems refer to the systems that include different types of computational units or processors and do not satisfy the homogeneity. E.g., the differences can come from unlike instruction set architectures, communication layer among processors, operation systems or compilers. The combinations of many different kinds of hardware and software aim to solve computation problems more efficiently. Heterogeneous systems exploit the advantages of each included hardware by using specialized processing capabilities for particular tasks and increases their performance and energy efficiency. Heterogeneous systems have more complex architectures and therefore, is more challenging to understand and model their performance and energy consumption.

Chapter 6

Conclusion

The energy consumed by worldwide computing systems increases annually and becomes a major concern in information technology society. In order to tackle this issue, the scientific community and industry have proposed several approaches to reduce the energy consumption of computing systems. Modeling energy consumption of applications running on computing systems providing the understanding of how applications consume energy and the insight into how to improve its energy efficiency.

This thesis presents three modeling approaches for energy consumption of computing systems varying from homogeneous to heterogeneous systems. The three approaches complement each other by targeting different types of computing systems such as homogeneous systems (e.g., embedded system, CPU or GPU) and heterogeneous systems (e.g., containing both CPU and GPU) and accomplishing different research objectives such as estimating absolute energy values, analyzing energy complexity of multithreaded algorithms and choosing the most energy-efficient configurations in runtime.

In the first study, we propose new application-general fine-grained power models (namely, RTHpower) that are able to investigate the trade-offs between performance and power consumption on ULP embedded systems. The RTHpower models consider both platform and application properties. We validate the new RTHpower models on Movidius Myriad, an ultra-low-power embedded system by developing different sets of micro-benchmarks and three application kernels such as dense matrix multiplication (Matmul), sparse matrix vector multiplication (SpMV) and breadth first search (BFS). We investigate the RTH strategy on an ultra-low power embedded platform using the new RTHpower models. We propose and validate a framework to predict when to use the race-to-halt (RTH) strategy to minimizes energy consumption for a given application.

In the second study, we devise a new general energy model ICE to provide an analysis tool to identify the energy complexity of a wide range of multithreaded algorithms on high-performance platforms based on their *work*, *span* and *I/O* complexity. We conduct two case studies (i.e., SpMV

and matmul) to demonstrate how to apply the ICE model to find energy complexity of parallel algorithms. The validation results show the precise prediction regarding which validated SpMV algorithm (i.e., CSB or CSC) consumes more energy when using different matrix input types from Florida matrix collection. The results also show the precise prediction on which validated matmul algorithm (i.e., basic or cache-oblivious) consumes more energy.

In the third study, we develop REOH, a new holistic tuning approach for heterogeneous systems. The approach uses a probabilistic network, a machine learning technique to predict energy consumption of an application on all possible configurations of the heterogeneous systems. In order for REOH to provide the energy estimation on heterogeneous systems, we propose a method to unify the configurations of different platform types (e.g., CPU and GPU) and devise a training data set with a set of applications based on the knowledge of application characteristics from Berkeley dwarfs. REOH can predict the energy consumption of all possible configurations of a heterogeneous system and identify the most energy-efficient configuration. REOH approach has its energy consumption close to the optimal energy consumption by the Brute Force approach while saving the number of sampling runs by running one prediction for the whole heterogeneous system instead of running separate predictions for every individual device in the heterogeneous system. Based on the approach, we also develop an energy-optimizing runtime framework as an open-source that is able to select an energy-efficient configuration of a heterogeneous system to run a given application at runtime.

6.1 Future Work

In this thesis, a machine learning technique (e.g., probabilistic network) has been used for modeling energy consumption of heterogeneous systems. For future computing systems containing more complex architectures, modeling energy consumption of large-scale systems becomes more challenging. Therefore, machines learning techniques are essential to be able to learn from available energy data to predict the energy consumption of such large-scale systems and suggests suitable system configurations to achieve the most energy efficiency. The accuracy of the modeling approaches can also be improved by identifying the most suitable techniques in a given context.

One of our future directions is to apply different machine learning techniques to model energy consumption, identify the most energy-efficient configuration and develop a more portable runtime framework. The probabilistic network approach used in this thesis requires a training data set obtained in advance for each considered system. When changing the underlying system, the training data set need to be collected again. This reduces the portability of the approach. In the context where energy training data can not be obtained in advance, investigating how to estimate energy consumption in runtime by using other machine learning techniques (e.g. reinforcement learning) is potential to improve both energy-efficiency and approach applicability.

Moreover, with heterogeneous systems, an application can be run coordinately by a task scheduler on multiple platforms simultaneously in the same execution. The modeling approaches presented in this thesis can be further developed to support a runtime scheduler to distribute the tasks of applications to different platforms in a heterogeneous system. By increasing the utility of each individual device in a heterogeneous system, we aim to reduce the static energy consumption and improve their energy efficiency.

Appendix A

Paper I

Appendix B

Paper II

Appendix C

Paper III

Bibliography

[1] Movidius vision processing unit. http://www.movidius.com/solutions/vision-processing-unit (retrieved on Sept. 23, 2015).

[2] Poski: Parallel optimized sparse kernel interface. http://bebop.cs.berkeley.edu/poski (retrieved on Nov. 17, 2015).

[3] Regression diagnostics - https://www.mathworks.com/help/stats/regstats.html.

[4] Rodinia:accelerating compute-intensive applications with accelerators - http://www.cs.virginia.edu/ skadron/wiki/rodinia/index.php/ rodinia:accelerating_compute-intensive_applications_with_accelerators.

[5] x2fx - convert predictor matrix to design matrix - https://www.mathworks.com/help/stats/x2fx.html.

[6] Y. Abe, H. Sasaki, S. Kato, K. Inoue, M. Edahiro, and M. Peres. Power and performance characterization and modeling of gpu-accelerated systems. In *2014 IEEE 28th International Parallel and Distributed Processing Symposium*, pages 113–122, May 2014.

[7] Nir Ailon, Ragesh Jaiswal, and Claire Monteleoni. Streaming k-means approximation. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 10–18. Curran Associates, Inc., 2009.

[8] Jack J. Dongarra Alexey L. Lastovetsky. *HighPerformance Heterogeneous Computing*. Wiley Online Library, 2009.

[9] M. Alioto. Ultra-low power vlsi circuit design demystified and explained: A tutorial. *IEEE Transactions on Circuits and Systems I: Regular Papers, vol.59, no.1, pp.3-29, Jan. 2012*, 59(1):3–29, Jan 2012.

[10] P Alonso, M F Dolz, R Mayo, and E S Quintana-Orti. Modeling power and energy consumption of dense matrix factorizations on multicore processors. *Concurrency Computat.*, 2014.

[11] Lars Arge, Michael T. Goodrich, Michael Nelson, and Nodari Sitchinava. Fundamental parallel algorithms for private-cache chip multiprocessors. In *Procs of the Twentieth Annual Symp on Parallelism in Algorithms and Architectures*, SPAA '08, pages 197–206, 2008.

[12] Krste Asanovic, Ras Bodik, Bryan Christopher Catanzaro, Joseph James Gebis, Parry Husbands, Kurt Keutzer, David A. Patterson, William Lester Plishker, John Shalf, Samuel Webb Williams, and Katherine A. Yelick. The landscape of parallel computing research: A view from berkeley. *Technical Report No. UCB/EECS-2006-183, University of California, Berkeley*, 2006.

[13] Muhammad Ali Awan and Stefan M. Petters. Race-to-halt energy saving strategies. *Journal of Systems Architecture*, 60(10):796 – 815, 2014.

[14] MichaelA. Bender, GerthStoelting Brodal, Rolf Fagerberg, Riko Jacob, and Elias Vicari. Optimal sparse matrix dense vector multiplication in the i/o model. *Theory of Computing Systems*, 47(4):934–962, 2010.

[15] Keren Bergman, Shekhar Borkar, Dan Campbell, William Carlson, William Dally, Monty Denneau, Paul Franzon, William Harrod, Jon Hiller, Sherman Karp, Stephen Keckler, Dean Klein, Robert Lucas, Mark Richards, Al Scarpelli, Steven Scott, Allan Snavely, Thomas Sterling, R. Stanley Williams, Katherine Yelick, Keren Bergman, Shekhar Borkar, Dan Campbell, William Carlson, William Dally, Monty Denneau, Paul Franzon, William Harrod, Jon Hiller, Stephen Keckler, Dean Klein, Peter Kogge, R. Stanley Williams, and Katherine Yelick. Exascale computing study: Technology challenges in achieving exascale systems, 2008.

[16] Aydin Buluç, Jeremy T. Fineman, Matteo Frigo, John R. Gilbert, and Charles E. Leiserson. Parallel sparse matrix-vector and matrix-transpose-vector multiplication using compressed sparse blocks. In *Procs of the Twenty-first Annual Symp on Parallelism in Algorithms and Architectures*, SPAA '09, 2009.

[17] G. L. T. Chetsa, L. Lefrvre, J. M. Pierson, P. Stolf, and G. Da Costa. A runtime framework for energy efficient hpc systems without a priori knowledge of applications. In *2012 IEEE 18th International Conference on Parallel and Distributed Systems*, pages 660–667, Dec 2012.

[18] Jee Choi, Marat Dukhan, Xing Liu, and Richard Vuduc. Algorithmic time, energy, and power on candidate hpc compute building blocks. In *Proceedings of the 2014 IEEE 28th International Parallel and Distributed Processing Symposium*, IPDPS '14, pages 447–457, Washington, DC, USA, 2014.

[19] Jee Whan Choi, Daniel Bedard, Robert Fowler, and Richard Vuduc. A roofline model of energy. In *Proceedings of the 2013 IEEE 27th International Symposium on Parallel and Distributed Processing*, IPDPS '13, pages 661–672, Washington, DC, USA, 2013.

[20] Henry Cook, Miquel Moreto, Sarah Bird, Khanh Dao, David A. Patterson, and Krste Asanovic. A hardware evaluation of cache partitioning to improve utilization and energy-efficiency while preserving responsiveness. In *Proceedings of the 40th Annual International Symposium on Computer Architecture*, ISCA '13, 2013.

[21] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 2009.

[22] Andrew Corrigan, Fernando Camelli, Rainald Löhner, and John Wallin. Running unstructured grid cfd solvers on modern graphics hardware. In *19th AIAA Computational Fluid Dynamics Conference*, number AIAA 2009-4001, June 2009.

[23] Timothy A. Davis and Yifan Hu. The university of florida sparse matrix collection. *ACM Trans. Math. Softw.*, 38(1):1:1–1:25, 2011.

[24] Wu-chun Feng, Heshan Lin, Thomas Scogland, and Jing Zhang. Opencl and the 13 dwarfs: A work in progress. In *Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering*, ICPE '12, pages 291–294, New York, NY, USA, 2012. ACM.

[25] Malte Forster and Jiri Kraus. Scalable parallel amg on ccnuma machines with openmp. *Computer Science - Research and Development*, 26(3-4):221–228, 2011.

[26] Matteo Frigo, Charles E. Leiserson, Harald Prokop, and Sridhar Ramachandran. Cache-oblivious algorithms. In *Procs of the 40th Annual Symp on Foundations of Computer Science*, FOCS, 1999.

[27] Matteo Frigo and Volker Strumpen. The cache complexity of multithreaded cache oblivious algorithms. In *Procs of the Eighteenth Annual ACM Symp on Parallelism in Algorithms and Architectures*, SPAA '06, pages 271–280, 2006.

[28] John R. Gilbert, Cleve Moler, and Robert Schreiber. Sparse matrices in matlab: Design and implementation. *SIAM J. Matrix Anal. Appl.*, 13(1):333–356, January 1992.

[29] J. Guerreiro, A. Ilic, N. Roma, and P. Tomas. Gpgpu power modeling for multi-domain voltage-frequency scaling. In *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 789–800, Feb 2018.

[30] Marisabel Guevara, Benjamin Lubin, and Benjamin C. Lee. Market mechanisms for managing datacenters with heterogeneous microarchitectures. *ACM Trans. Comput. Syst.*, 32(1):3:1–3:31, February 2014.

[31] P. Ha, V. Tran, I. Umar, P. Tsigas, A. Gidenstam, P. Renaud-Goud, I. Walulya, and A. Atalar. Models for energy consumption of data structures and algorithms. Technical report, EU FP7 project EXCESS deliverable D2.1 (http://www.excess-project.eu), 2014.

[32] Phuong Ha, Vi Tran, Ibrahim Umar, Aras Atalar, Anders Gidenstam, Paul Renaud-Goud, and Philippas Tsigas. D2.2 White-box methodologies, programming abstractions and libraries. Technical Report FP7-611183 D2.2, EU FP7 Project EXCESS, February 2015.

[33] Phuong Ha, Vi Tran, Ibrahim Umar, Aras Atalar, Anders Gidenstam, Paul Renaud-Goud, Philippas Tsigas, and Ivan Walulya. D2.3 d2.4 report on the final prototype of programming abstractions for energy-efficient inter-process communication. Technical Report FP7-611183 D2.4, EU FP7 Project EXCESS, August 2016.

[34] Phuong Ha, Vi Tran, Ibrahim Umar, Aras Atalar, Anders Gidenstam, Paul Renaud-Goud, Philippas Tsigas, and Ivan Walulya. D2.3 power models, energy models and libraries for energy-efficient concurrent data structures and algorithms. Technical Report FP7-611183 D2.3, EU FP7 Project EXCESS, February 2016.

[35] Phuong Ha, Vi Tran, Ibrahim Umar, Philippas Tsigas, Anders Gidenstam, Paul Renaud-Goud, Ivan Walulya, and Aras Atalar. D2.1 Models for energy consumption of data structures and algorithms. Technical Report FP7-611183 D2.1, EU FP7 Project EXCESS, August 2014.

[36] M.D. Hill and M.R. Marty. Amdahl's law in the multicore era. *Computer*, 41(7):33–38, 2008.

[37] P. Hoai Ha, N.-N. Tran, Vi, I. Umar, P. Tsigas, A. Gidenstam, P. Renaud-Goud, I. Walulya, and A. Atalar. D2.1 Models for energy consumption of data structures and algorithms. *ArXiv e-prints*, January 2018.

[38] C. Imes, D. H. K. Kim, M. Maggio, and H. Hoffmann. Poet: a portable approach to minimizing energy under soft real-time constraints. In *21st IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 75–86, April 2015.

[39] C. Imes, D.H.K. Kim, M. Maggio, and H. Hoffmann. Poet: a portable approach to minimizing energy under soft real-time constraints. In *Real-Time and Embedded Technology and Applications Symposium (RTAS), 2015 IEEE*, pages 75–86, April 2015.

[40] Mircea Horea Ionica and David Gregg. The movidius myriad architecture's potential for scientific computing. *Micro, IEEE*, 35(1):6–14, Jan 2015.

[41] H. Jacobson, A. Buyuktosunoglu, P. Bose, E. Acar, and R. Eickemeyer. Abstraction and microarchitecture scaling in early-stage power modeling. In *IEEE 17th International Symposium on High Performance Computer Architecture (HPCA), 2011*, pages 394–405, Feb 2011.

[42] Chao Jin, Bronis R de Supinski, David Abramson, Heidi Poxon, Luiz DeRose, Minh Ngoc Dinh, Mark Endrei, and Elizabeth R Jessup. A survey on software methods to improve the energy efficiency of parallel computing. *The International Journal of High Performance Computing Applications*, 31(6):517–549, 2017.

[43] Christoph Kessler, Lu Li, Usman Dastgeer, Rosandra Cuello, Oskar Sjöström, Phuong Ha Hoai, and Vi Tran. D1.3 Energy-tuneable domain-specific language/library for linear system solving. Technical Report FP7-611183 D1.3, EU FP7 Project EXCESS, February 2015.

[44] Christoph Kessler, Lu Li, Usman Dastgeer, Philippas Tsigas, Anders Gidenstam, Paul Renaud-Goud, Ivan Walulya, Aras Atalar, David Moloney, Phuong Ha Hoai, and Vi Tran. D1.1 Early validation of system-wide energy compositionality and affecting factors on the EXCESS platforms. Technical Report FP7-611183 D1.1, EU FP7 Project EXCESS, April 2014.

[45] Dmitry Khabi, Vi Tran, and Ivan Walulya. D5.4 Report on the first evaluation results and discussion. Technical Report FP7-611183 D5.4, EU FP7 Project EXCESS, August 2015.

[46] V.A. Korthikanti and Gul Agha. Analysis of parallel algorithms for energy conservation in scalable multicore architectures. In *International Conference on Parallel Processing, 2009. ICPP '09.*, pages 212–219, Sept 2009.

[47] Vijay Anand Korthikanti and Gul Agha. Towards optimizing energy costs of algorithms for shared memory architectures. In *Proceedings of the 22nd Annual ACM Symposium on Parallelism in Algorithms and Architectures*, 2010.

[48] Vladimir Kotlyar, Keshav Pingali, and Paul Stodghill. A relational approach to the compilation of sparse matrix programs. Technical report, 1997.

[49] Pardeep Kumar, Andrei Gurtov, and Phuong H. Ha. An efficient authentication model in smart grid networks. In *Procs of the 15th Int Conf on Information Processing in Sensor Networks*, pages 65:1–65:2, 2016.

[50] J. Lagravire, J. Langguth, M. Sourouri, P. H. Ha, and X. Cai. On the performance and energy efficiency of the pgas programming model on multicore architectures. In *Procs of the Int Workshop on Optimization of Energy Efficient HPC & Distributed Systems (OPTIM-HPCS)*, pages 800–807, 2016.

[51] Etienne Le Sueur and Gernot Heiser. Dynamic voltage and frequency scaling: The laws of diminishing returns. In *Proceedings of the 2010 International Conference on Power Aware Computing and Systems*, HotPower'10, pages 1–8, Berkeley, CA, USA, 2010. USENIX Association.

[52] Victor W. Lee, Changkyu Kim, Jatin Chhugani, Michael Deisher, Daehyun Kim, Anthony D. Nguyen, Nadathur Satish, Mikhail Smelyanskiy, Srinivas Chennupaty, Per Hammarlund, Ronak Singhal, and Pradeep Dubey. Debunking the 100x gpu vs. cpu myth: An evaluation of throughput computing on cpu and gpu. In *Proceedings of the 37th Annual International Symposium on Computer Architecture*, ISCA '10, pages 451–460, New York, NY, USA, 2010. ACM.

[53] Jingwen Leng, Tayler Hetherington, Ahmed ElTantawy, Syed Gilani, Nam Sung Kim, Tor M. Aamodt, and Vijay Janapa Reddi. Gpuwattch: Enabling energy optimizations in gpgpus. In *Proceedings of the 40th Annual International Symposium on Computer Architecture*, ISCA '13, pages 487–498, New York, NY, USA, 2013. ACM.

[54] Lu Li and Christoph Kessler. Validating energy compositionality of GPU computations. In *HIPEAC Workshop on Energy Efficiency with Heterogeneous Computing (EEHCO)*, 2015.

[55] Sheng Li, Jung Ho Ahn, R.D. Strong, J.B. Brockman, D.M. Tullsen, and N.P. Jouppi. Mcpat: An integrated power, area, and timing modeling framework for multicore and manycore architectures. In *Microarchitecture, 2009. MICRO-42. 42nd Annual IEEE/ACM International Symposium on*, 2009.

[56] Xu Liu and John Mellor-Crummey. A tool to analyze the performance of multithreaded programs on numa architectures. In *Proceedings of the 19th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, PPoPP '14, 2014.

[57] Lu Li, Christoph Kessler. MeterPU: A Generic Measurement Abstraction API Enabling Energy-tuned Skeleton Backend Selection. In *Proc. International Workshop on Reengineering for Parallelism in Heterogeneous Parallel Platforms (REPARA-2015) at ISPA-2015*, volume 3, pages 154–159. IEEE, 2015.

[58] K. Ma, X. Li, W. Chen, C. Zhang, and X. Wang. Greengpu: A holistic approach to energy efficiency in gpu-cpu heterogeneous architectures. In *2012 41st International Conference on Parallel Processing*, pages 48–57, Sept 2012.

[59] Kai Ma, Yunhao Bai, Xiaorui Wang, Wei Chen, and Xue Li. Energy conservation for gpucpu architectures with dynamic workload division and frequency scaling. *Sustainable Computing: Informatics and Systems*, 12:21 – 33, 2016.

[60] Alberto Magni, Christophe Dubach, and Michael F. P. O'Boyle. A large-scale cross-architecture evaluation of thread-coarsening. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '13, pages 11:1–11:11, New York, NY, USA, 2013. ACM.

[61] A. Majumdar, L. Piga, I. Paul, J. L. Greathouse, W. Huang, and D. H. Albonesi. Dynamic gpgpu power management using adaptive model predictive control. In *2017 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pages 613–624, Feb 2017.

[62] A. Cristiano I. Malossi, Yves Ineichen, Costas Bekas, Alessandro Curioni, and Enrique S. Quintana-Orti. Systematic derivation of time and power models for linear algebra kernels on multicore architectures. *Sustainable Computing: Informatics and Systems*, 7:24 – 40, 2015.

[63] Nikita Mishra, Huazhe Zhang, John D. Lafferty, and Henry Hoffmann. A probabilistic graphical model-based approach for minimizing energy under performance constraints. In *Proceedings of the 20th International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '15. ACM, 2015.

[64] S. Muralidharan, M. Shantharam, M. Hall, M. Garland, and B. Catanzaro. Nitro: A framework for adaptive code variant tuning. In *2014 IEEE 28th International Parallel and Distributed Processing Symposium*, pages 501–512, May 2014.

[65] Rajib Nath and Dean Tullsen. The crisp performance model for dynamic voltage and frequency scaling in a gpgpu. In *Proceedings of the 48th International Symposium on Microarchitecture*, MICRO-48, pages 281–293, New York, NY, USA, 2015. ACM.

[66] G. Ofenbeck, R. Steinmann, V. Caparros, D.G. Spampinato, and M. Puschel. Applying the roofline model. In *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2014*, pages 76–85, March 2014.

[67] Anne-Cecile Orgerie, Marcos Dias de Assuncao, and Laurent Lefevre. A survey on techniques for improving the energy efficiency of large-scale distributed systems. *ACM Comput. Surv.*, 46(4):47:1–47:31, 2014.

[68] Franz Pernkopf, Robert Peharz, and Sebastian Tschiatschek. *Introduction to Probabilistic Graphical Models*, volume 1, chapter 18, pages 989–1064. Elsevier, 2014.

[69] Phitchaya Mangpo Phothilimthana, Tikhon Jelvis, Rohin Shah, Nishant Totla, Sarah Chasins, and Rastislav Bodik. Chlorophyll: Synthesis-aided compiler for low-power spatial architectures. In *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '14, 2014.

[70] Swapnoneel Roy, Atri Rudra, and Akshat Verma. An energy complexity model for algorithms. In *Procs of the 4th Conf on Innovations in Theoretical Computer Science*, ITCS '13, pages 283–304, 2013.

[71] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd edition, 2003.

[72] J. Shen, A. L. Varbanescu, Y. Lu, P. Zou, and H. Sips. Workload partitioning for accelerating applications on heterogeneous platforms. *IEEE Transactions on Parallel and Distributed Systems*, 27(9):2766–2780, Sept 2016.

[73] David C. Snowdon, Etienne Le Sueur, Stefan M. Petters, and Gernot Heiser. Koala: A platform for os-level power management. In *Proceedings of the 4th ACM European Conference on Computer Systems*, EuroSys '09, pages 289–302, New York, NY, USA, 2009. ACM.

[74] T. Suzumura, K. Ueno, H. Sato, K. Fujisawa, and S. Matsuoka. Performance characteristics of graph500 on large-scale distributed environment. In *IEEE International Symposium on Workload Characterization (IISWC), 2011*, pages 149–158, 2011.

[75] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-15(1):116–132, Jan 1985.

[76] SYMPRAXIS TEAM. Assessing the employment and social impact of energy efficiency. Technical report, Cambridge Econometrics, 2015.

[77] V. N. N. Tran, B. Barry, and P. H. Ha. Power models supporting energy-efficient co-design on ultra-low power embedded systems. In *2016 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation (SAMOS)*, pages 39–46, July 2016.

[78] V. N. N. Tran and P. H. Ha. Ice: A general and validated energy complexity model for multithreaded algorithms. In *2016 IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS)*, pages 1041–1048, Dec 2016.

[79] V. N.N. Tran, T. Oines, A. Horsch, and P. H. Ha. Reoh: Using probabilistic network for runtime energy optimization of heterogeneous systems. In *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*, 2018.

[80] Vi Ngoc-Nha Tran, Brendan Barry, and Ha. Rthpower: Accurate fine-grained power models for predicting race-to-halt effect on ultra-low power embedded systems. In *Proceedings of the 17th IEEE International Symposium on Performance Analysis of Systems and Software*, ISPASS 16, 2016.

[81] Ibrahim Umar, Otto J. Anshus, and Phuong H. Ha. Effect of portable fine-grained locality on energy efficiency and performance in concurrent search trees. In *Proceedings of the 21st ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, PPoPP '16, 2016.

[82] Ibrahim Umar, Otto J. Anshus, and Phuong H. Ha. Greenbst: An energy-efficient concurrent search tree. In *Proc. of the 22nd Intl. European Conf. on Parallel and Distributed Computing (Euro-Par 16)*, page pages to appear., 2016.

[83] Ibrahim Umar, Otto Johan Anshus, and Phuong Hoai Ha. Deltatree: A locality-aware concurrent search tree. In *Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, SIGMETRICS '15, 2015.

[84] Richard Vuduc, James W Demmel, and Katherine A Yelick. Oski: A library of automatically tuned sparse matrix kernels. In *Institute of Physics Publishing*, 2005.

[85] Qiang Wang and Xiaowen Chu. Gpgpu power estimation with core and memory frequency scaling. *SIGMETRICS Perform. Eval. Rev.*, 45(2):73–78, October 2017.

[86] S. Williams, L. Oliker, R. Vuduc, J. Shalf, K. Yelick, and J. Demmel. Optimization of sparse matrix-vector multiplication on emerging multicore platforms. In *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing, 2007. SC '07.*, pages 1–12, Nov 2007.

[87] Samuel Williams, Andrew Waterman, and David Patterson. Roofline: An insightful visual performance model for multicore architectures. *Commun. ACM*, 52(4):65–76, 2009.

[88] F. Yao, A. Demers, and S. Shenker. A scheduling model for reduced cpu energy. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 374–382, Oct 1995.

[89] Kathy Yelick. Cs 267 parallel matrix multiplication, Sept 2004.

[90] F. Zhang, B. Wu, J. Zhai, B. He, and W. Chen. Finepar: Irregularity-aware fine-grained workload partitioning on integrated architectures. In *2017 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*, pages 27–38, Feb 2017.

[91] F. Zhang, J. Zhai, B. He, S. Zhang, and W. Chen. Understanding co-running behaviors on integrated cpu/gpu architectures. *IEEE Transactions on Parallel and Distributed Systems*, 28(3):905–918, March 2017.

[92] Huazhe Zhang and Henry Hoffmann. Maximizing performance under a power cap: A comparison of hardware, software, and hybrid techniques. In *Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '16, pages 545–559, New York, NY, USA, 2016. ACM.

[93] Albert Y. Zomaya and Young Choon Lee. *Energy Efficient Distributed Computing Systems.* Wiley-IEEE Computer Society Pr, 1st edition, 2012.