Thesis for The Degree of Licentiate of Engineering

Energy Efficient Task Mapping and Resource Management on Multi-Core Architectures

Jing Chen

Division of Computer and Network Systems
Department of Computer Science & Engineering
Chalmers University of Technology
Gothenburg, Sweden, 2022

Energy Efficient Task Mapping and Resource Management on Multi-Core Architectures

Jing Chen

Department of Computer Science & Engineering
Division of Computer and Network Systems
Chalmers University of Technology and Gothenburg University
Gothenburg, Sweden

This thesis has been prepared using LaTeX.
Printed by Chalmers Reproservice,
Gothenburg, Sweden 2022.

"It's not what happens to you, but how you react to it that matters."
- Epicurus

Abstract

Reducing energy consumption of parallel applications executing on chip multiprocessors (CMPs) is important for green computing. Hardware vendors have been developing a variety of system features to support energy efficient computing, for example, integrating asymmetric core types on a single chip referred to as static asymmetry and supporting dynamic voltage and frequency scaling (DVFS) referred to as dynamic asymmetry.

A common parallelization scheme to exploit CMPs is task parallelism, which can express a wide range of computations in the form of task directed acyclic graphs (DAGs). Existing studies that target energy efficient task scheduling have demonstrated the benefits of leveraging DVFS, particularly per-core DVFS. Their scheduling decisions are mainly based on heuristics, such as task criticality, task dependencies and workload sizes. To enable energy efficient task scheduling, we identify multiple crucial factors that influence energy consumption - varying task characteristics, exploitation of intra-task parallelism (task moldability), and task granularity - which we collectively refer to as task heterogeneity. Task heterogeneity and architecture asymmetry features together complicate the task scheduling problem, since the most energy efficient configuration of resource allocation and frequency setting varies with each task. Our analysis shows that leveraging task heterogeneity in conjunction with static and dynamic asymmetry provides significant opportunities for energy reduction.

This thesis contributes two scheduling techniques - ERASE and STEER - that target different scenarios. ERASE focuses on fine-grained tasking and in environments where DVFS is not under user control. It leverages the insights of task characteristics, task moldability, and instantaneous task parallelism detection for guiding scheduling decisions. ERASE comprises four modules: online performance modeling, power profiling, core activity tracing and a task scheduler. Online performance modeling and power profiling provide runtime with execution time and power predictions. Core activity tracing offers the instantaneous task parallelism and the task scheduler combines these information to enable the energy predictions and dynamically determine the best resource allocation for each task during runtime. STEER focuses on environments where DVFS is under user control and where the platform comprises multiple asymmetric cores grouped into clusters. STEER explores how much energy could be potentially saved by leveraging static asymmetry, dynamic asymmetry and task heterogeneity in conjunction. STEER comprises two predictive models for performance and power predictions, and a task scheduler that utilizes models for energy predictions and then identifies the best resource allocation and frequency settings for tasks. Moreover, it applies adaptive scheduling techniques based on task granularity to manage DVFS overheads, and coordinates the cluster frequency settings to reduce interference from concurrent running tasks on cluster-based architectures.

The evaluation on an NVIDIA Jetson TX2 shows that ERASE achieves 10% energy savings on average compared to the state-of-the-art DVFS-based schedulers and can adapt to external DVFS changes, and STEER consumes 38% less energy on average than both the state-of-the-art and ERASE.

Keywords: Energy Consumption, Task Scheduling, Resource Management, Predictive Models, Dynamic Voltage-Frequency Scaling (DVFS), Runtime

Acknowledgment

First and foremost, I would like to express my deep and sincere gratitude to my advisor, Associate Professor Miquel Pericàs, for giving me the opportunity to study abroad and providing invaluable guidance and support during my studies. His great knowledge, vision and motivation have deeply inspired and taught me the methodology to carry on the research and present research works clearly.

I would also like to thank my co-advisor Dr. Madhavan Manivannan for his insightful feedback and enthusiasm. His plentiful research experience that shared with me has encouraged me during my research studies and daily life. I am also grateful to Dr. Mustafa Abduljabbar and Dr. Bhavishya Goel for their research collaboration and great ideas.

Thanks to Professor Per Stenström who has been my examiner. I would like to say thanks to my, past and present, colleagues and friends at Chalmers, Pirah, Soniar, Nadja, Hao, Yuchong, Nufail, Mahmoud, Mehrzad, Waqar, Alexandra, Albin, Prajith, Pedro, Monica, Arne, Rolf and many others who created a nice and friendly work environment.

Finally, I would like to give special thanks to my family for their unconditional support. I am also extremely grateful to my life partner Franz for accompanying all I ever needed.

This research has been funded by the European Union Horizon 2020 research and innovation programme under grant agreement No.780681 (https://legato-project.eu/). This research has also received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No.956702 (https://eprocessor.eu). The JU receives support from the European Union's Horizon 2020 research and innovation programme and Spain, Sweden, Greece, Italy, France, Germany. The computations were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC), partially funded by the Swedish Research Council through grant agreement No.2018-05973 (https://www.vr.se/).

List of Publications

Appended publications

This thesis is based on the following publications:

[I] Jing Chen, Madhavan Manivannan, Mustafa Abduljabbar, and Miquel Pericàs
"ERASE: Energy Efficient Task Mapping and Resource Management for Work Stealing Runtimes"
Published in ACM Transactions on Architecture and Code Optimization (TACO), 2022.

[II] Jing Chen, Madhavan Manivannan, Bhavishya Goel, Mustafa Abduljabbar, and Miquel Pericàs
"STEER: Asymmetry-aware Energy Efficient Task Scheduler for Cluster-based Multicore Architectures"
Under review.

Other publications

The following publications are not included in the thesis.

[a] Jing Chen, Madhavan Manivannan, Mustafa Abduljabbar, and Miquel Pericàs
"Towards an Energy Aware Task Scheduler for Asymmetric Architectures"
Published in 12th Nordic Workshop on Multi-Core Computing (MCC), 2019, Karlskrona, Sweden

[b] Jing Chen, Pirah Noor Soomro, Mustafa Abduljabbar, Madhavan Manivannan, and Miquel Pericàs
"Scheduling Task-parallel Applications in Dynamically Asymmetric Environments"
Published in 49th International Conference on Parallel Processing - ICPP Workshops SRMPDS, 2020, Edmonton, AB, Canada

Contents

Abstract **v**

Acknowledgement **vii**

List of Publications **ix**

1 Introduction **1**
1.1 Background 1
1.2 Related Work 2
1.3 Problem Statements 3
1.4 Contributions 4

2 Summary of the Papers **5**
2.1 Paper I - Summary 5
2.2 Paper II - Summary 7

3 Conclusions and Future Work **11**

Bibliography **13**

Paper I **17**

Paper II **47**

Chapter 1

Introduction

1.1 Background

Reducing energy consumption is vital for achieving green computing, as it plays a key role in reducing the environmental impact of technology and promoting sustainability. In the context of high performance computing (HPC) systems, conserving energy is not only crucial to lower electricity bills and cooling cost for saving environment resources but also a big step forward to the exascale supercomputing era. In addition, reducing energy in mobile devices translates to longer battery life for improving system reliability and enhancing friendly user experiences.

Hardware vendors have been exploring and integrating a variety of energy efficient features on modern chip multi-processor (CMP) systems. Dynamic voltage and frequency scaling (DVFS) is a well-known technique that introduces the *dynamic asymmetry* and offers great promise to significantly reduce power consumption by adapting both voltage and frequency of the system to suit various workloads [1]. In order to provide more opportunities for energy efficient execution, CMPs are designed by composing of multiple core types (single-ISA) with different micro-architectures onto a single chip. These architectures can provide diverse energy-performance capabilities for different workloads. This is referred to as *static asymmetry* due to its fixed feature at design time. Core-clustering paradigm is being adopted for organizing such architectures, in which cores of the same type are grouped into clusters [2–9]. For power management, these cluster-based platforms often support per-cluster DVFS, where cores in the same cluster must operate at the same voltage-frequency level.

A common parallelization scheme to exploit such platforms is task-parallelism, which has been implemented in several production runtime systems, e.g. Cilk [10], TBB [11], StarPU [12], OpenMP's explicit tasks [13]. With task parallelism, parallel applications can be expressed as task Directed Acyclic Graphs (DAGs), where nodes denote tasks and edges denote task dependencies. A DAG implicitly reveals the inter-task parallelism, i.e. the independent tasks that can be executed in parallel. In particular, by exposing fine-grained task parallelism, it allows programmers to express applications that can scale to larger CMPs. However, we identify that the intrinsic *task heterogeneity* feature exhibits mul-

tiple contributing factors that can influence the energy consumption of running a task-based application. These factors include various task characteristics (i.e. task-aware mapping to appropriate resources), and potential intra-task parallelism (task moldability) exploitation by running a single task on multiple resources to reduce resource oversubscription and make use of idle resources, and task granularity (size) in relation to the DVFS timing overheads.

Overall, static asymmetry, dynamic asymmetry and task heterogeneity together form a complex task scheduling problem, since the best configurations for energy savings diverge. Therefore, it is crucial that a scheduling scheme can be developed to identify the configuration that consumes the least energy per task. Solving such a multi-dimensional problem is non-trivial and requires to search for a large space to figure out the best task mapping and resource management decisions. To improve energy efficiency, runtime schedulers should consider the impact of task heterogeneity and exploit the available hardware tuning knobs (core asymmetry and runtime DVFS). Unfortunately, contemporary production runtime systems lack systematic energy-aware task scheduling methods. Instead they mainly focus on improving performance and scalability [14]. Therefore, in this thesis, we propose two scheduling techniques to fulfill the gap and address the energy efficient task scheduling problem when running task-based applications on cluster-based platforms with different environment setting scenarios.

1.2 Related Work

Existing studies have demonstrated the benefits of exploiting dynamic asymmetry (DVFS) for energy reduction. Prior works [15–17] propose to use per-core DVFS throttling on symmetric architectures for improving energy efficiency. The work in [18] explores both static asymmetry (i.e. different core micro-architectures) and dynamic asymmetry (i.e. per-core DVFS) on multi-core processors. The methodologies presented in [19–21] address the scheduling problem on symmetric cluster-based platforms, while others [22, 23] further explore the techniques applied on the combination of cluster-based static asymmetry and dynamic asymmetry of cluster-level DVFS.

However, these proposals have several limitations. Firstly, they overlook the negative impact of DVFS reconfiguration overheads from fine-grained tasking. With fine-grained tasks that execute in the order of microseconds, DVFS reconfiguration overheads are quite large and can offset the energy benefits obtained from frequency throttling. Secondly, in a multi-user system it is common that DVFS control is restricted to the kernel [24], the system administrator [25], or power management frameworks such as GEOPM [26]. Therefore, the reliance on DVFS for energy efficient execution precludes their applicability in environment settings where the DVFS is not under the control of the application (i.e. externally controlled). Thirdly, with DVFS enabled from user space, deploying per-core DVFS schedulers on platforms with cluster-based DVFS is inefficient, since the scaling action of a single core may result in destructive interference on other cores in the same cluster and offset the energy benefits. Finally, some works address frequency throttling in cluster-based architectures without considering the impact on energy when leveraging

static asymmetry in conjunction with DVFS. Moreover, the majority of prior works [15,16,18,19,22,23] rely on heuristics, such as workload size, task criticality assignation and task dependencies to guide scheduling decisions. However, they lack the consideration of the holistic impacts of task heterogeneity on energy consumption. Overall, none of them fully explores the potential energy impacts when leveraging static asymmetry, dynamic asymmetry and task heterogeneity in conjunction.

1.3 Problem Statements

Context 1: Applications often expose fine-grained task parallelism for achieving better scalability and load balancing in multi-core and many-core architectures [27, 28]. With fine-grained tasking, the execution time of tasks are in the order of microseconds, which makes the DVFS reconfiguration overheads non-negligible. Therefore, leveraging per-task DVFS is impractical in this case. Moreover, it is common that in a multi-user OS, many processes share a subset of cores, and DVFS control is commonly not under control of the application but restricted to the kernel, the system administrator, or power management frameworks like GEOPM. Consequently, it is crucial to design an adaptive energy efficient task scheduling technique that applies to fine-grained tasking with low overheads and can be reactive to externally controlled DVFS.

Problem 1: *How can we reduce energy consumption of running fine-grained tasking parallel applications on multi-core platforms with externally controlled DVFS?*

Challenges: To answer the question, there are two major challenges that need to be addressed in this context: (1) In order to make the energy efficient task mapping decisions under given environment settings, identifying the best resource assignment per task, i.e. (cluster, number of cores), requires the accurate predictions of task execution time and power consumption; (2) The proposed technique needs to quickly detect and adapt to external DVFS changes and is able to recompute the best resource allocation with low overheads.

Context 2: With DVFS enabled from user space, in conjunction with static symmetry and task heterogeneity, a larger design space needs to be searched for identifying the best configurations. Furthermore, managing DVFS on fine-grained task schedules while minimizing the negative impacts from DVFS reconfiguration is a challenge. In addition, on platforms with cluster-level DVFS, it is possible that multiple tasks are scheduled to run on the same cluster and different frequencies are selected to achieve energy savings. Consequently, coordinating cluster frequency tuning is crucial since it can otherwise lead to destructive DVFS tuning interference and DVFS reconfiguration serialization, thereby becoming a performance bottleneck and offsetting the energy benefits.

Problem 2: *How to reduce energy consumption by leveraging static asymmetry, dynamic asymmetry, task heterogeneity in conjunction when running task-based applications on multi-core platforms that feature cluster-level DVFS?*

Challenges: To answer the question, there are three major challenges that need to be addressed in this context: (1) Predicting the best resource allocation and frequency setting for a task involves exploring a three-dimensional search

space, i.e. (cluster, number of cores, frequency). Additionally, the complexity is exacerbated by the need to estimate energy consumption with low runtime overhead and with reasonable accuracy; (2) The design of adaptive scheduling techniques for various task granularities to manage DVFS negative overheads; (3) The necessary frequency coordination method to mitigate the detrimental interference impacts on energy consumption from concurrent running tasks.

1.4 Contributions

This thesis is based on two papers. *Paper I* addresses the first problem and the main contributions are:

- We propose ERASE: an energy efficient task scheduler that combines power profiling, performance modeling and core activity tracing for energy efficient mapping (i.e. choosing the cluster) and resource management (i.e. selecting the number of cores per task). The proposal exploits the insights of task moldability, task-type awareness and instantaneous task parallelism detection for guiding scheduling decisions to reduce the total energy consumption of parallel applications.
- We describe how to integrate ERASE on top of work stealing runtimes, using the XiTAO [29] runtime for the prototype implementation.
- We compare ERASE to state-of-the-art scheduling techniques on top of the runtime and the evaluation shows that ERASE achieves up to 31% energy savings and outperforms the state-of-the-art by 44% on average.

Paper II addresses the second problem and the main contributions are:

- We show that considerable energy savings can be achieved by leveraging static asymmetry, dynamic asymmetry and task heterogeneity in conjunction. Accordingly, we propose STEER, a task scheduling framework that exploits these features to predict the best energy-saving configuration for each task.
- We propose a performance model and a power model to predict the impact of varying the core type, the number of cores and the frequency, that are not limited by the availability of performance counters.
- We develop heuristics to (1) manage DVFS overheads by applying adaptive scheduling techniques for varying task granularities, and (2) mitigate DVFS interference from concurrent task execution on cluster-based multicore architectures.

The rest of the thesis is organized as follows. In Chapter 2, a summary of each paper is presented. Finally, Chapter 3 concludes the thesis, and discusses some possible future research directions.

Chapter 2

Summary of the Papers

2.1 Paper I - Summary

Parallel applications often rely on work stealing schedulers in combination with fine-grained tasking to achieve high performance and scalability. Random work stealing is a well-known approach for scheduling task-parallel applications [30, 31] that has been implemented in several production runtimes, such as Cilk [10], TBB [11] and OpenMP's explicit tasks [13]. However, reducing the total energy consumption in the context of work stealing runtimes is still challenging, particularly when using asymmetric architectures with different types of CPU cores. This is because the two principles on which random work stealing is based, namely the work-first principle and random victim selection, are neither aware of task characteristics nor of the cores' performance and energy profiles.

A common approach for energy savings in work stealing runtimes in prior works [15, 16, 18, 19] involves leveraging DVFS, wherein the throttling is carried out based on factors like task parallelism, stealing relations, task criticality and workload sizes. However, the reliance on DVFS limits their applicability due to several reasons. First, studies have shown that DVFS transition delay is around 100 microseconds [16, 18, 32, 33], thereby the DVFS switching overheads are non-negligible for fine-grained tasks that execute in the order of microseconds. Second, per-core DVFS control utilized in [15, 16, 18] precludes their techniques applicability on cluster-based architectures. With cluster-level DVFS, multiple tasks attempting frequency changes within the same cluster will result in destructive interference, since the decision taken to reduce energy consumption of a task mapped to a specific core can affect concurrently running tasks on the same cluster. More importantly, it is common that in a multiuser OS, DVFS is most often not under the control of the application, but is externally controlled by the Linux kernel [24], the system administrator [25], or power management frameworks such as GEOPM [26]. Consequently, an energy efficient runtime designed to be reactive to both given static frequency settings and externally controlled DVFS has the potential to be a more general solution in this case.

In Paper I, we present ERASE - an energy efficient task scheduler to address the problem of reducing the total energy consumption when running task-based applications on multi-core platforms wherein the frequency throttling is externally controlled.

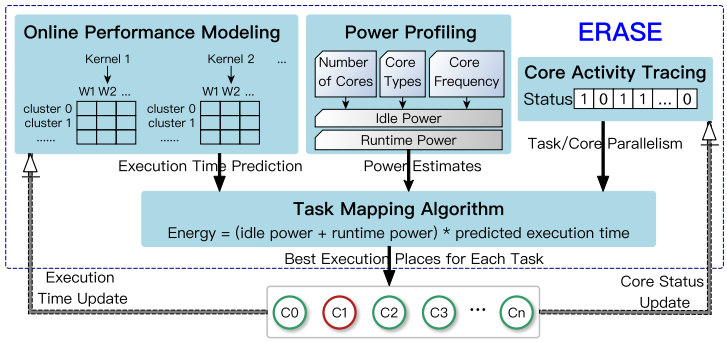

Figure 2.1: An overview of ERASE comprising four modules. C_x denotes the core id, W_y denotes the possible resource widths. In status, "1" denotes that C_x is in active state while "0" denotes the core is in sleep state.

The total energy consumption of running a DAG on a multi-core platform consists of two components: the energy consumed for running each task and the energy consumed in the idle periods due to the dependencies between tasks. To reduce the energy consumed by each task, runtime needs to evaluate the energy consumption of all different resource configurations (cluster, number of cores) and identify the one that consumes the least energy for the task. Hence, it is crucial that the runtime can predict the execution time and the power consumption and thereby the energy consumption to facilitate the configuration selection. In work stealing runtimes, idle cores that continuously attempt stealing without success lead to energy waste. The problem of reducing energy consumption during idle periods requires the runtime to be able to detect the cores' instantaneous utilization and dynamically put idle cores to sleep. The challenge is to determine the sleep duration such that energy consumption during the period is minimized with minimal performance impact.

In a nutshell, ERASE reduces energy consumption of executing a task DAG by attempting to execute each task with the lowest possible energy consumption. It leverages the insights of task moldability (i.e. intra-task parallelism) and task-type awareness to figure out the appropriate resource allocation, i.e. choosing the cluster and selecting the number of cores (resource width), for each task. Meanwhile, it adaptively puts idle cores to sleep state by applying an exponential back-off sleeping strategy. Figure 2.1 provides an overview of ERASE, which comprises four essential modules: online performance modeling, power profiling, core activity tracing and a task mapping algorithm.

Online performance modeling adopts a history-based model, it continuously monitors task execution during runtime and updates look-up tables. Thus, the module can not only provide performance predictions for incoming tasks by referring to corresponding look-up table entries but also quickly detect the external frequency changes by comparing to previous table entry records. Power profiling provides the power estimates with respect to different resource configurations (i.e. number/type of cores) for given core frequencies. The module groups tasks into three representative types: compute-bound, memory-bound and cache-intensive. By profiling a set of microbenchmarks, we compute the arithmetic intensity (AI) values and employ the k-NN algorithm to cluster them into the three groups. During runtime, we compute AI of a task and map the task to one of the groups where the average power consumption of

the group is utilized as the reference. Core activity tracing continuously tracks the activities (i.e. work stealing attempts) and status (i.e. active or sleep) of each core and infers the instantaneous task parallelism, which gives the task mapping algorithm a hint for attributing power consumption to concurrently running tasks accurately. Finally, the task mapping algorithm integrates the aforementioned information and guides scheduling decision for each task based on the energy estimates of running the task on different resource configurations.

We evaluate the effectiveness of ERASE by comparing against several state-of-the-art scheduling techniques on an asymmetric platform (NVIDIA Jetson TX2) and a symmetric platform (a dual-socket 16-core per socket Intel Xeon Gold 6130 node). The results indicate that ERASE provides significant energy savings in comparison, across a range of benchmarks and different system environment settings. More importantly, ERASE is capable of quickly adapting to the external frequency changes with low overheads.

2.2 Paper II - Summary

In order to improve energy efficiency, hardware vendors have been integrating a variety of system features on multi-core platforms. DVFS is a well-known technique that represents the *dynamic asymmetry* feature and offers great promise to significantly reduce power consumption by adapting both voltage and frequency of the system with respect to various workloads [1]. In addition, CPUs that are being composed of multiple core types with different micro-architectures onto a single chip, provide diverse energy-performance capabilities for different workloads, which is a demonstration of *static asymmetry* due to its fixed feature at design time.

Core-clustering paradigm [2] is adopted for organizing such architectures to reduce the hardware design complexity [3–9]. In such designs, cores of the same type are clustered together to share common resources like last level cache and memory controller. For power management, these cluster-based platforms often support per-cluster DVFS, where cores in the same cluster must operate at the same voltage-frequency level.

Besides static and dynamic asymmetry, energy efficient scheduling can benefit from considering task characteristics, which motivates us to understand the contributing factors to energy consumption in order to make energy efficient task schedules on such platforms. These factors include task placement (i.e. task mapping to appropriate resources), and potential intra-task parallelism (task moldability) exploitation by running a single task on multiple resources to reduce resource oversubscription and make use of idle resources, and task granularity (size) in relation to the DVFS timing overheads. Collectively, we refer to these three aspects, namely task characteristics, task moldability and task granularity, as *task heterogeneity*.

Existing studies [15–23] either explore the energy benefits of levering dynamic asymmetry in isolation or the combination of static asymmetry and dynamic asymmetry without considering the impact of task heterogeneity. Moreover, the proposals that use per-core DVFS [15–18] have limited applicability on cluster-based architectures. In Paper II, we show that leveraging task heterogeneity in conjunction with static asymmetry and dynamic asymmetry

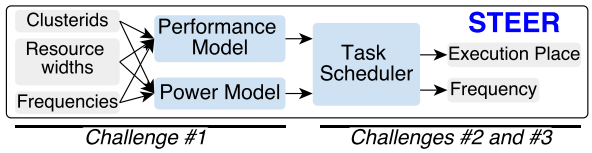

Figure 2.2: An Overview of STEER framework.

can provide significant opportunities for energy reduction and accordingly propose a task scheduling framework STEER.

In a nutshell, STEER reduces the energy consumption of the entire DAG by running each task with the lowest energy consumption possible through identifying the best execution place and frequency setting. However, there are three major challenges that need to be addressed to enable the energy efficient schedules on platforms with cluster-level DVFS. First, predicting the best execution place and frequency setting for a task involves exploring a three-dimensional search space, i.e. (cluster, resource width, frequency). The complexity is exacerbated by the need to estimate energy consumption with low runtime overhead and with reasonable accuracy. Second, it is necessary to design adaptive scheduling techniques for various task granularity, particularly for fine-grained tasks to exploit DVFS and reduce DVFS negative impacts simultaneously. Finally, with concurrent running tasks, it is possible that different frequencies are selected for energy reduction. This fact makes frequency coordination crucial for mitigating the detrimental energy impacts on the concurrent running tasks and reducing the overall energy consumption.

To address these aforementioned challenges, STEER leverages two predictive models to help the runtime accurately predict the impact of different execution place and frequency settings on the execution time and power consumption. Then a task scheduler is developed to leverage the models to predict the energy consumption, determine the best execution place and frequency setting that consumes the least modeled energy and schedule tasks for execution. Figure 2.2 provides an overview of the essential components in STEER.

Performance Model: To predict the execution time and reduce the modeling overheads, STEER adopts a hybrid approach by first sampling a limited number of possible settings and utilizing a model to predict performance for the rest of settings. The performance model utilizes memory-boundness (MB) as a measure of the fraction of execution time that does not scale with core frequency. Therefore, when changing frequency, the computation fraction (1-MB) will scale proportionally with frequency, while the memory bound fraction (MB) will remain independent of the frequency. Consequently, the equation for predicting execution time based on MB at a different frequency ($f0$) can be expressed as below:

$$Time_{f0} = Time_f \times (MB + (1 - MB) \times \frac{f}{f0}) \tag{2.1}$$

The performance model relies on knowing $Time_f$ and MB at frequency f to predict the execution time for the task when running at other frequencies. $Time_f$ can be obtained by timing task execution inside the runtime. We obtain a second sample at a different frequency ($f0$) and derive MB as follows:

$$MB = \frac{\frac{Time_{f0}}{Time_f} - \frac{f}{f0}}{1 - \frac{f}{f0}} \tag{2.2}$$

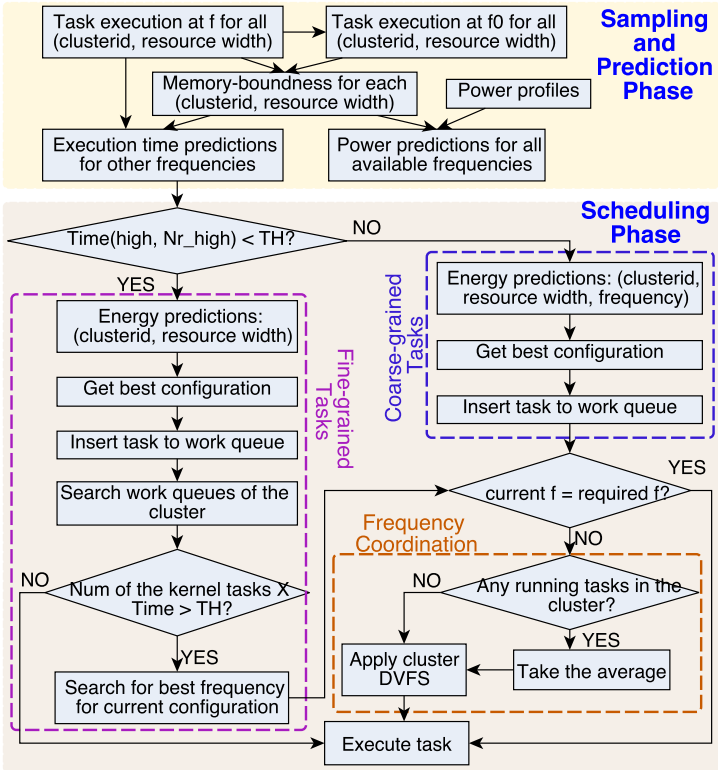

Figure 2.3: The work flow of STEER scheduler.

Once MB is estimated by sampling at two different frequencies, the execution time predictions for all other frequencies are obtained using equation 2.1.

Power Model: The power model adopts an offline characterization approach, where look-up tables are constructed for power references during runtime. We profile a set of training benchmarks (NAS parallel benchmark suite in this work) to record the power consumption and execution time on different execution place and frequency settings. We then calculate the MB values and cluster these benchmarks into groups according to their MB values. Estimating the power consumption for a task at runtime involves mapping the task into one of the groups given its MB value and accessing the corresponding table entry.

Task Scheduler: The task scheduler in STEER essentially comprises two phases as shown in Figure 2.3. In sampling and prediction phase, the scheduler performs execution time sampling of tasks to compute MB values and enable performance and power prediction at different execution place and frequency settings. In scheduling phase, the scheduler utilizes the performance and power consumption prediction information to enable energy efficient task scheduling. For coarse-grained tasks, the scheduler iterates all possible settings and identifies the one that consumes the least modeled energy and then performs the task mapping and frequency throttling accordingly. In the case of fine-grained tasks, DVFS throttling overheads are large enough to offset any benefit. Therefore, STEER adaptively adjusts scheduling policy once fine-grained tasks are detected. Specifically, fine-grained tasks are composed together with other

tasks that are instances of the same kernel and can be viewed as a single coarse-grain task, such that DVFS throttling can be performed for this entire group of tasks for further energy savings. To address the challenge of frequency coordination in cluster-based platforms, STEER develops a heuristic to mitigate the problem by tuning the frequency of the cluster to the arithmetic average predicted frequency of each of the individual tasks running on the cluster.

We evaluate the effectiveness of STEER by comparing it to multiple state-of-the-art schedulers on an asymmetric cluster-based platform NVIDIA Jetson TX2. The evaluation across a range of benchmarks shows that STEER achieves 53% energy reduction on average compared to the baseline scheduler greedy random work stealing and 38% energy reduction on average than both the state-of-the-art approach and ERASE. Furthermore, the proposed performance model and power model are not limited by the availability of performance counters and achieve 95% and 90% prediction accuracy, respectively.

Chapter 3

Conclusions and Future Work

The growing impact of energy on operational cost and reliability becomes a strong motivation for reducing energy consumption in parallel computing. Multi-core platforms are equipped with features to enable energy efficient computing, such as DVFS and core-clustering paradigm of integrating asymmetric cores into clusters on a single chip. To exploit the capability of such multi-core platforms, one of the most generalized programming models is to employ task parallelism and represent parallel applications as task DAGs. This leads to the investigation of task scheduling and resource management techniques for reducing energy consumption.

This thesis proposes two task scheduling techniques for addressing the energy reduction problem in different contexts. Paper I targets energy reduction when running fine-grained tasking applications on multi-core platforms wherein the DVFS is externally controlled. The scheduler leverages the insights of task moldability and task-type awareness to figure out the appropriate resource allocation for each task. Moreover, it can quickly detect external frequency changes and adaptively schedule tasks with low overheads. Paper II aims to design an integrated scheduling strategy that can effectively manage multiple forms of asymmetry (incl. static asymmetry, dynamic asymmetry and task heterogeneity), while targeting cluster-based multi-core architectures. The proposed framework leverages two predictive models for predicting the execution time and power consumption and a task scheduler for determining the execution place and frequency settings that consume the least energy and schedule tasks for execution. Furthermore, it applies adaptive scheduling techniques on various task granularity to manage the DVFS overheads' impact on energy, and coordinates the frequency settings to reduce frequency throttling interference within clusters.

There are several interesting research directions for future research. Firstly, one direction would be to evaluate the adaptivity of the proposed scheduler when targeting other crucial metrics, such as exploring performance-energy trade-offs (best performance given an energy budget and least energy given a performance constraint). It is still a challenging question in the context of work stealing runtimes. For example, when targeting EDP, performance has a

larger emphasis therefore it is important to tune the work stealing attempts across clusters for improving performance while still keeping lowest energy possible. Secondly, another interesting direction would be to explore the overall energy reduction problem by taking the memory usage and the workload distributions among CPU and accelerators into account. For example, compute-intensive tasks require higher core frequency with lower memory frequency, while memory-intensive tasks observe the opposite trend. Thus, tuning the CPU and memory frequencies according to task characteristics would be helpful in reducing the overall energy consumption. In addition, managing the task allocation for utilizing CPU and accelerators computing resources efficiently is vital for reducing the total energy consumption. More research is necessary towards using task allocation algorithms and other asymmetry features provided by platforms together to determine the execution place for a task (i.e. CPU - core types, number of resources or accelerators - number of resources or both) and the suitable voltage or frequency for CPU and accelerators. Finally, energy efficient task schedules involve energy consumption estimates thereby requiring accurate predictive models. However, it may not be necessary to pursue the development of complicated performance and power models, since it can introduce overheads. Thus, it would be interesting to explore the model accuracy and cost-performance trade-off, i.e. how accurate the predictive models need to be to figure out the best configurations with low overhead.

Bibliography

[1] T. Simunic, L. Benini, A. Acquaviva, P. Glynn, and G. De Micheli, "Dynamic voltage scaling and power management for portable systems," in *Proceedings of the 38th annual Design Automation Conference*, 2001, pp. 524–529.

[2] T. Odajima, Y. Kodama, M. Tsuji, M. Matsuda, Y. Maruyama, and M. Sato, "Preliminary performance evaluation of the fujitsu a64fx using hpc applications," in *2020 IEEE International Conference on Cluster Computing (CLUSTER)*, 2020, pp. 523–530.

[3] B. Jeff, "Advances in big.little technology for power and energy savings improving energy efficiency in high-performance mobile platforms," 2012.

[4] "ODROID XU3," https://developer.arm.com/solutions/graphics-and-gaming/development-platforms/odroid-xu3.

[5] "Nvidia jetson," https://developer.nvidia.com/buy-jetson.

[6] "Mediatek x20 development board," https://www.96boards.org/product/mediatek-x20/.

[7] E. Rotem, Y. Mandelblat, V. Basin, E. Weissmann, A. Gihon, R. Chabukswar, R. Fenger, and M. Gupta, "Alder lake architecture," in *2021 IEEE Hot Chips 33 Symposium (HCS)*, 2021, pp. 1–23.

[8] S. Shankland, "iphone xs a12 bionic chip is industry-first 7nm cpu," https://www.cnet.com/news/iphone-xs-a12-bionic-chip-is-industry-first-7nm-cpu/, September 2018.

[9] R. Ritchie, "Apple a14 bionic explained — from ipad air to iphone 12," https://www.imore.com/apple-a14-bionic-explained-ipad-air-iphone-12, September 2020.

[10] M. Frigo, C. E. Leiserson, and K. H. Randall, "The Implementation of the Cilk-5 Multithreaded Language," in *Proceedings of SIGPLAN 1998*, Jun. 1998.

[11] G. Contreras and M. Martonosi, "Characterizing and improving the performance of intel threading building blocks," in *2008 IEEE International Symposium on Workload Characterization*. IEEE, 2008, pp. 57–66.

[12] "Documentation of starpu," https://files.inria.fr/starpu/doc/starpu.pdf.

[13] OpenMP Architecture Review Board, *OpenMP Application Program Interface. Version 5.0*, OpenMP Architecture Review Board Std., Nov 2018.

[14] M. A. H. Monil, "Dynamic adaptation techniques and opportunities to improve hpc runtimes," 2021.

[15] H. Ribic and Y. D. Liu, "Energy-efficient work-stealing language runtimes," in *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 513–528. [Online]. Available: https://doi.org/10.1145/2541940.2541971

[16] E. Castillo, M. Moreto, M. Casas, L. Alvarez, E. Vallejo, K. Chronaki, R. Badia, J. L. Bosque, R. Beivide, E. Ayguade, J. Labarta, and M. Valero, "Cata: Criticality aware task acceleration for multicore processors," in *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2016, pp. 413–422.

[17] B. Acun, K. Chandrasekar, and L. V. Kale, "Fine-grained energy efficiency using per-core dvfs with an adaptive runtime system," in *2019 Tenth International Green and Sustainable Computing Conference (IGSC)*, 2019, pp. 1–8.

[18] C. Torng, M. Wang, and C. Batten, "Asymmetry-aware work-stealing runtimes," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, 2016, pp. 40–52.

[19] H. Ribic and Y. Liu, "Aequitas: Coordinated energy management across parallel applications," 06 2016, pp. 1–12.

[20] R. A. Shafik, A. Das, S. Yang, G. Merrett, and B. M. Al-Hashimi, "Adaptive energy minimization of openmp parallel applications on many-core systems," in *Proceedings of the 6th Workshop on Parallel Programming and Run-Time Management Techniques for Many-Core Architectures*, ser. PARMA-DITAM '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 19–24. [Online]. Available: https://doi.org/10.1145/2701310.2701311

[21] Q. Chen, Y. Chen, Z. Huang, and M. Guo, "Wats: Workload-aware task scheduling in asymmetric multi-core architectures," in *2012 IEEE 26th International Parallel and Distributed Processing Symposium*, 2012, pp. 249–260.

[22] L. Costero, F. D. Igual, K. Olcoz, and F. Tirado, "Energy efficiency optimization of task-parallel codes on asymmetric architectures," in *2017 International Conference on High Performance Computing Simulation (HPCS)*, July 2017, pp. 402–409.

[23] M. Han, J. Park, and W. Baek, "Design and implementation of a criticality- and heterogeneity-aware runtime system for task-parallel applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 5, pp. 1117–1132, 2021.

[24] T. kernel development community, "Energy aware scheduling." [Online]. Available: https://www.kernel.org/doc/html/latest/scheduler/sched-energy.html

[25] D. Brodowski, "Cpu frequency and voltage scaling code in the linux(tm) kernel." [Online]. Available: https://www.kernel.org/doc/Documentation/cpu-freq/governors.txt

[26] J. Eastep, S. Sylvester, C. Cantalupo, B. Geltz, F. Ardanaz, A. Al-Rawi, K. Livingston, F. Keceli, M. Maiterth, and S. Jana, "Global extensible open power manager: A vehicle for hpc community collaboration on co-designed energy management solutions," in *High Performance Computing*, 2017.

[27] K. B. Wheeler, R. C. Murphy, and D. Thain, "Qthreads: An api for programming with millions of lightweight threads," in *2008 IEEE International Symposium on Parallel and Distributed Processing*, 2008, pp. 1–8.

[28] S. Kumar, C. J. Hughes, and A. Nguyen, "Carbon: Architectural support for fine-grained parallelism on chip multiprocessors," in *Proceedings of the 34th Annual International Symposium on Computer Architecture*, ser. ISCA '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 162–173. [Online]. Available: https://doi.org/10.1145/1250662.1250683

[29] T. C. Team, "XiTAO runtime," https://github.com/CHART-Team/xitao.git.

[30] R. D. Blumofe and C. E. Leiserson, "Scheduling multithreaded computations by work stealing," *Journal of the ACM*, vol. 46, no. 5, pp. 720–748, Sep. 1999.

[31] Q. Chen, Y. Chen, Z. Huang, and M. Guo, "WATS: Workload-aware task scheduling in asymmetric multi-core architectures," *Proceedings of the 2012 IEEE 26th International Parallel and Distributed Processing Symposium, IPDPS 2012*, 05 2012.

[32] S. Park, J. Park, D. Shin, Y. Wang, Q. Xie, M. Pedram, and N. Chang, "Accurate modeling of the delay and energy overhead of dynamic voltage and frequency scaling in modern microprocessors," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 5, pp. 695–708, 2013.

[33] R. Schöne, T. Ilsche, M. Bielert, M. Velten, M. Schmidl, and D. Hackenberg, "Energy efficiency aspects of the AMD zen 2 architecture," *CoRR*, vol. abs/2108.00808, 2021. [Online]. Available: https://arxiv.org/abs/2108.00808