Improving Energy Efficiency and Security of DVFS-Enabled Clouds

by

Jianzhou Mao

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
August 6, 2022

Keywords: Cloud computing, DVFS, Energy optimization, Security overhead,
Multi-objective, NSGA-II-SER.

Approved by

Xiao Qin, Chair, Alumni Professor of Computer Science and Software Engineering
Saad Biaz, Professor of Computer Science and Software Engineering/.'
Tao Shu, Associate Professor of Computer Science and Software Engineering
Richard Chapman, Associate Professor of Computer Science and Software Engineering

Abstract

Since modern data centers have been significantly scaling up in capacity in past decades, it is demanding to curtail energy consumption of virtual-machine-powered data centers. Cloud computing has radically changed the landscape of computing, storage, and communication infrastructures and services. Cloud computing's benefits encompass on-demand capacity, low cost of ownership, and flexible pricing. In the first part of this dissertation I propose a frequency-aware management strategy, which controls dynamic power and static power of processors running virtual machines in data centers. Unlike existing dynamic voltage and frequency scaling schemes, my strategy simply incorporates frequency requirements rather than task execution times. This salient feature is practical because task execution times in a raft of real-world applications are unknown in a priori. I build a frequency-aware model to derive an optimal frequency ratio that minimizes processors' energy consumption. With my model in place, the energy efficiency of a datacenter can be maximized by adjusting the processor's frequency to meet the optimal frequency ratio. I design a management approach to judiciously adjust frequency ratio to conserve energy without violating the frequency requirements imposed by virtual machines. After analyzing the correlations between frequency ratio and energy consumption, I show that a small static power proportion gives rise to high energy-saving performance. The results demonstrate that my model lays out a solid theoretical foundation catering to the development of power management software in DVFS-enabled clouds.

Besides the energy consumption, security issues coupled with resource allocations in cloud computing remain a challenging problem to be tackled by the industry and academia. While moving towards the concept of on-demand services and resource pooling in a distributed computing environment, security is a major obstacle for this new dreamed vision

of computing capability. In the second part of the dissertation study, I articulate novel energy-aware scheduling policies customized for virtual machines running on clouds, in which service-level agreements (SLAs) are fulfilled. After addressing security concerns in cloud computing, I advocate for a research roadmap towards future security-aware energy management in clouds. I propose a high-level design for a security- and frequency-ware DVFS model or SF-DVFS, which orchestrates security services, security overhead analysis, and DVFS control green cloud computing systems. I delve into the main technical challenges associated with the proposed SF-DVFS model. To solve this multi-objective problem, I design a Secure and Economical DVFS-enabled Scheduling Policy with NSGA-II-SER(Non-dominated Sorting Genetic Algorithm II with Security and Energy Requirement) for Clouds.

Blockchain is an ideal privacy protection technology characterized by decentralization, transparency, data security, and system autonomy. As the last project in this dissertation research, I navigate leading-edge energy saving and privacy protection techniques for clouds. Next, I investigate privacy controls in blockchain systems. Inspired by modern blockchain and cloud computing techniques, I elaborate on a research roadmap towards future energy-aware privacy protection mechanisms in clouds. In a case study, I design a blockchain-based VM consolidation framework accompanied by the DVFS (Dynamic Voltage and Frequency Scaling) technique to offer energy savings and privacy controls in clouds. I expect that the roadmap will open up potentials to develop energy-efficient blockchain-based cloud computing platforms.

Acknowledgments

First, I am deeply indebted to Dr.Xiao Qin for all aspects of my research, career, and life matters. I will never forget the moment I walked into Dr.Qin's office first time, I was encouraged by his big smile, and Dr.Qin helped me to plan the long-term study scheme at Auburn University with patience and enthusiasm. I will never forget the night my labmates and I were invited to his house for dinner and games: as a young student who had just arrived in a new country on the other side of the world, it truly gave me tremendous mental support during the tough time. I will never forget the weekly meetings with Dr.Qin, from inspiring my idea to improving my writing skill, he put all his efforts into helping me at every step of this research. Without Dr.Qin's guidance, patience, knowledge, and endless support, this dissertation research would have never been possible. I will never forget the tons of hours he used to help me to analyze my future career and assist me in job hunting. I cannot imagine having a better advisor, and will always remember and benefit from what he has taught me.

I would also like to thank my committee members, Dr.Saad Biaz, Dr.Tao Shu, and Dr.Richard Chapman, their advice and assistance has been very helpful, and their contribution is greatly appreciated. I would like to extend my sincere thanks to Dr.Song-yul Choe, one of the most responsible university readers I have ever known, who reviewed this dissertation word by word and offered lots of detailed and valuable suggestions.

I was fortunate to collaborate with many talented researchers during my Ph.D. I would like to thank Xiaopu Peng, Chaowei Zhang, Ting Cao, Dr.Yi Zhou, and Tathagata Bhattacharya for working closely with me and giving me enormous help on research.

My years at Auburn would have been different without all the friends, with their smiles, discussions, meals, and others. I am also grateful to the friendly local families in Auburn,

who invited me to their house for the Christmas party, and taught me horse riding and shooting on their farm without asking for anything in return.

Last and most, I would like to express my heartfelt appreciation to my parent for all the love and support and for helping me be the best version of myself. Thank you mom and dad, for everything I can imagine.

This dissertation research is supported by the U.S. National Science Foundation under Grants IIS-1618669, OAC-1642133, CCF-0845257.

To my parents

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this dissertation research, I will present novel approaches to enhancing energy efficiency and security of computing clouds. I start off the dissertation with background knowledge of data centers, cloud computing, and virtualization techniques. Perhaps most importantly, this Chapter highlights the motivations for the three research thrusts - frequency-aware management, security-aware management, and energy-aware privacy controls.

More specifically, this chapter is organized as follows. I introduce the background of data centers, cloud computing, and virtualization technique in Section 1.1. Then, I elaborate on the motivations and basic ideas for frequency-aware management strategies in Section 1.2. Next, I share the motivations for security-aware energy management in clouds and energy-aware privacy controls for clouds in Section 1.4 and Section 1.4, respectively. After that, Section 1.5 concludes the contributions of this dissertation research. Last, but not least, I show the organization of this dissertation in Section 1.6.

1.1 Data Centers, Cloud Computing, and Virtualization Techniques

1.1.1 Data Centers

In our expanding digital world, data is changing the way I live, work, and entertain. International Data Corporation or IDC speculates that the aggregated data in the world will grow from 33 zettabytes in 2018 to 175ZB by 2025 at a significant annual growth rate of 61% [119]. To meet the accommodate such a massive amount of data, the scale of datacenters is snowballing to reach an unbelievable level. The global data-center market is estimated to exceed $174 billion by 2023, growing at an annual rate of approximately 4% during the

forecast period. The largest companies such as Facebook, Google, Amazon, and Microsoft are focusing on the development of modular and hyperscale data-center construction facilities [3].

1.1.2 Cloud Computing

Cloud-based datacenters have become a new trend of the enterprise data repository replacing traditional datacenters. Cloud computing offers promising benefits such as enhanced scalability, efficiency, flexibility of business operations, and to name just a few. A growing number of large companies have chosen to utilize cloud datacenters (e.g., Microsoft Clouds and Amazon Web Services) to store enormous amount of data. IDC predicts that 49% of the world's stored data will reside in public cloud environments in year 2025 [119]. A report published by Cisco forecasts that 94% of all workloads and computing will be placed in cloud datacenters in 2021 [64]. More specifically, overall workloads and compute instances in datacenters will be doubled (i.e., 2.3-fold) from 2016 to 2021. When it comes to computing clouds, the workloads are expected to expand by a factor of 2.7-fold during the same period.

1.1.3 Virtualization Techniques

With the advanced virtualization technologies deployed in data centers, cloud infrastructures become a predominant computing platform (see, for example, Amazon Elastic Compute Cloud (EC2) [148] and Microsoft Azure [152]). Virtual computation environments furnish on-demand and elastic computation and storage capabilities, thereby facilitating large-scale data analysis and big-data applications. In modern virtualization techniques, resources residing in physical machines are partitioned into individual virtual machines (VMs), which isolates one application from the counterparts running on the other VMs. Multiple VMs assigned to one physical machine share resources on the same machine. One or more applications may run on a virtual machine; in contrast, a large-scale application can make use of enormous resources across multiple virtual machines.

1.2 Motivations and Basic Ideas for Frequency-aware Management Strategies

The performance of cloud computing platforms has been significantly growing in the past decade. With the dramatic evolution in computing capacity, the power consumption of datacenters customized for cloud computing is skyrocketing. In the first part of this dissertation study, I propose to seamlessly integrate DVFS (i.e., Dynamic Voltage and Frequency Scaling) scheduling with virtual machine management to effectively conserve energy consumption in datacenters.

My frequency-aware scheduler presented in Chapter 3 aims to minimize CPU energy cost in virtual-machine-enabled datacenters by configuring the most appropriate CPU frequency ratios according to workload conditions. Four emerging trends below strongly motivate us to contrive my frequency-aware DVFS scheduling algorithm.

- Virtual-machine-based cloud computing platforms are technical underpinnings for modern datacenters in the future (see Section 1.2.1).
- Large-scale datacenters have a pressing demand to be economically and environmentally friendly (see Section 1.2.2).
- The dynamic voltage and frequency scaling technique or *DVFS* is one of the most practical energy-saving technologies (see Section 1.2.3).
- Quality of service (i.e., QoS) opens a door for optimizing energy efficiency without violating service-level agreements (see Section 1.2.4).

I detail the above four motivations followed by the basic idea behind my novel frequency-aware management design presented in Section 1.2.5.

1.2.1 Virtual Machines

Virtual machines are a driving force behind the wide adoption of cloud-based datacenters. Virtual machines - software-based machine emulations - facilitate flexible, cost-effective,

and on-demand computing environments. Virtual machines make it feasible to save a lot of energy by increasing workloads and compute instance density. The workload and compute instance density of cloud servers is expected to grow from 8.8 in 2016 to 13.2 by 2021. In comparison, the workload and compute instance density in traditional data center servers will merely increase from 2.4 in 2016 to 3.8 in the same period [64]. Moreover, virtual machines are positioned to support legacy software and computing environments [93].

1.2.2 Economically and Environmentally Friendly Datacenters

The energy consumption of these large-scale datacenters is truly tremendous. A long-term goal of this study is to devise resource management tools for future economically and environmentally friendly datacenters. Servers, core IT components in datacenters, are major players contributing to high energy cost (e.g., more than 26%) [57]. CPUs, main memory, and disk I/Os are key contributors to energy cost in servers. The energy cost of CPU, in turn, occupies the highest proportion in the server (i.e., 42%) [18]; in some scenarios, CPUs account for up to 70% of total energy consumption in servers [19]. Thus, conserving CPU energy is indispensable in tackling the datacenter energy problem. In a handful of prior studies [105], CPU energy saving policies have been investigated without considering main memory or I/Os and; therefore, I followed this research methodology to place the CPU energy efficiency under the microscope.

1.2.3 Dynamic Voltage and Frequency Scaling

VM consolidation and DVFS (Dynamic Voltage and Frequency Scaling) are two popular energy conservation methods adopted in data centers. The VM consolidation technique reduces the number of physical servers by migrating VMs and shutting down idle servers to save energy consumption [158][69]. Unlike VM consolidation, DVFS is beneficial when servers exhibit low idle power consumption, high utilization, and large VM migration overhead. DVFS enables processors to consume less power by adjusting to the most appropriate frequency and

supplied voltage. Almost all processors are built in CMOS circuit and; therefore, evidence shows that energy consumption of CPU is approximately proportional to frequency and the square of the voltage. Decreasing voltage and processor frequency will, of course, scale down computing performance and stretch executing times. This performance problem is well addressed by incorporating the concept of quality of service (a.k.a., QoS) while adjusting voltage and CPU frequency. The popularity of the DVFS technique motivates us to explore an optimization strategy to enhance the energy efficiency of DVFS-endabled datacenters.

When deploying the DVFS technique, one has to be aware of static power consumption that is incurred by leakage current in CMOS devices. More often than not, DVFS inevitably extends execution times at the cost of increased static power consumption. I advocate a holistic solution that orchestrates both dynamic power and static power management. More specifically, I propose a model that takes into account static and dynamic power consumption. With the proposed model in place, I conduct an experiment to evaluate the energy efficiency of servers running multiple virtual machines in datacenters.

1.2.4 Quality of Service

To build energy-efficient datacenters, it is prudent to make a good tradeoff between energy efficiency and performance. A practical energy-saving method is to conserve energy while meeting Service Level Agreements (a.k.a., SLA requirements) [113]. Improving the energy efficiency of datacenters maximizes profits by significantly cutting back energy consumption. It is conventional wisdom to incorporate execution time and deadlines into QoS requirements of submitted tasks. In a raft of real-world scenarios (e.g., web crawlers), task execution times are unknown a priori. This constraint motivates us to pilot a novel model that does not rely on task execution times and deadlines.

1.2.5 Basic Idea of My New Frequency-aware Management Approach

In my design, I take into account frequency requirements as part of SLAs. In cloud computing platforms like Google Cloud and Microsoft Azure, users submit the required number of virtual-machine instances accompanied by frequency requirements. Since cloud-computing platform cost is proportional to the number of VMs as well as required frequency, users strive to make the most appropriate tradeoff between performance and cost.

I propose a new Qos model, where minimum frequency requirements is a core component. In doing so, users have no obligation to specify execution times and deadlines in QoS; rather, frequency requirements can straightforwardly serve the purpose. My energy conservation scheme fosters energy efficiency while striving to meet frequency requirements rather than deadlines. QoS requirements will be guaranteed if configured frequencies are the same or higher than the specified minimum frequencies.

Compared with existing solutions, my approach has the following three salient strengths. First of all, frequency ratios are used as a metric to specify QoS requirements and energy consumption rather than the execution time parameter. My model derives optimal energy savings for servers without estimating execution times or specifying deadlines. Second, my model offers optimal frequency ratios customized for various processors in accordance to their static power proportions, thereby maximizing the processors' energy efficiency. Last, my frequency-aware DVFS model, being practical in nature, is readily to be adopted by power management systems. Given a server's static power and max dynamic power, the model can govern a power manager to achieve high energy efficiency.

1.3 Motivations for Security-Aware Energy Management in Clouds

In the second part of the dissertation research, I are focusing on research thrusts in security-aware energy management for cloud computing infrastructures. My research is inspired by the following three trends.

- Cloud computing is an effective technology that delivers interesting services to customers over the Internet.
- There is a pressing demand to build energy-efficient clouds housed in large-scale data centers.
- Building trustworthy cloud environments remains a challenging issue.

1.3.1 Hyper-Scale Data Centers

In a drastically expanding digital world, big data are changing the way we live, work, and entertain. International Data Corporation or IDC speculates that the aggregated data around the world will grow from 33 zettabytes in 2018 to 175ZB by 2025 at a significant annual growth rate of 61% [119]. To accommodate such a massive amount of data, the scale of data centers is demanded to snowball to reach an unprecedented and unbelievable level. The global data center market is estimated to exceed $174 billion by 2023, representing an annual growth rate of approximately 4% during the forecast period. To meet such pressing demands, the largest technology companies such as Facebook, Google, Amazon, and Microsoft are focusing on the development of modular and hyper-scale data center construction facilities [3].

1.3.2 Energy-Efficient Data Centers

The energy consumption of these large-scale datacenters is truly tremendous. For example, the global data center power market size will hit the bar of $10.77 billion by year 2025, expanding at an annual rate of 6.9%, even faster than that of the datacenter market [1]. Globally, Power consumption of data centers is close to 416 terawatts, representing three percent of all electricity generated on the planet. In other words, data center energy consumption around the world accounts for 40 percent more than all the energy consumed

in the United Kingdom [100]. Nowadays, over 80% of the world's energy is still being generated by fossil fuels [5], which could lead to CO2 emissions and other global environmental problems like global warming.

1.3.3 Trustworthy Cloud Environments

Cloud computing offers services with scalable resources in a protected view. Although cloud features are well understood from a business point of view, building trustworthy cloud environments remains a challenging issue. Cloud computing has increasingly gained in popularity among individual users and organizations, but recently raised security issues demand new solutions. For example, organizations have a dire need for secure infrastructures when data are transferred to and managed at remote locations.

It is conventional wisdom to handle big data in local storage systems, where data processing, movement, and management are carried out in local domains. More often than not, security measures developed by cloud service providers are transparent to the public and, for this reason, some enterprise users hesitate to rely on cloud services and infrastructure to store and process digital assets [81][73].

1.4 Motivations for Energy-Aware Privacy Controls for Clouds

I navigate energy-aware privacy preserving techniques in realms of centralized and decentralized computing systems. I start my investigation by focusing on cloud data centers, the backbone of cloud infrastructure platforms supporting large-scale data processing and storage, followed by blockchain techniques that safeguard energy transactions in a distributed network. This study is inspired by the following two motivations - (1) privacy-aware energy-efficient data storage (see Section 1.4.1) and (2) blockchain-based privacy preserving techniques for clouds (see Section 1.4.2).

1.4.1 Privacy-aware Green Data Storage

BP or British Petroleum forecasts that global energy demand continues to grow in the predictable future, driven by increasing prosperity and living standards [6]. Moreover, ExxonMobil predicts that global energy demand will rises by 20 percent to 2040 and; during the same time period, the electricity consumption will rise by 60 percent [4]. The trend to electrify buildings, factories, cars, and buses, along with smart appliances, spurs the pressing need for more electricity everywhere. Constructing energy-efficient data centers catering to cloud computing aims to address the concerns of increasing electricity demands. Cloud data center is energy friendly by the virtue of the on-demand deployment of resources through cutting-edge virtualization technology. Cloud users are enabled to swiftly allocate computing power and resources according to dynamically changing needs at any time without maintaining the underlying physical structure of computing platforms. Growing evidence demonstrates that cloud computing platforms are slated to conserve energy by providing information resources in a pay-as-you-go model [145].

The virtualization technology in cloud data centers brings forth versatility and reliability to cloud services. To facilitate virtual machine (VM) management in cloud data centers, system administrators make use of shared data storage to handle VMs' data in uniform storage space. The concept of shared data storage is implemented by adopting a centralized structure, where all physical nodes are connected to a centralized storage unit such as network-attached storage (NAS) [146]. Even it is convenient to build high-end centralized storage systems, a centralized structure is prone to data leakage of VMs running in cloud data centers when access privileges of some nodes are comprised [65].

1.4.2 Blockchains and Privacy Protections

The preceding discussions emphasize the importance of decentralization, one trait commonly associated with blockchain [168]. Blockchain is characterized by decentralization, transparency, data security, and system autonomy. It has been applied widely in areas such

as finance, education and employment, culture and entertainment, public service, information security, healthcare, supply chain, and internet of things. Moreover, blockchain gains its popularity in the energy sectors [23] thanks to blockchain's underpinning characteristics such as anonymity, decentralization, transparency, and reliability.

Despite the observable benefits of using blockchain in a diversity of areas including the energy sectors, privacy concerns are restricting blockchain's applications. For instance, users may need to disclose private energy demand data to a third-party in order to schedule the use of shared energy resources. This process may reveal sensitive personal data such as working patterns, number of occupants, and vacation periods. On the other hand, the privacy-related attacks should not be overlooked. Representative privacy concerns include linking attacks, which utilize open information recorded in blocks and obtain privacy from linking the information with other datasets. Moreover, in the arena of privacy preserving methods, I confront the following challenges. First, more times than not, attackers are capable of obtaining privacy with inaccurate data. Thus, any features pertaining to privacy control ought to be hidden to prevent privacy leakage. Second, most existing differential privacy schemes are inadequate for accurately recording energy trading operations because a noised record stored in blocks results in a malfunction of a transaction ledger. Hence, there is an urgent demand for designing adoptable privacy-preserving mechanisms catering for blockchain-enabled energy applications.

1.5 Contributions

This section summarizes the main contributions of this dissertation research, which embraces three technical underpinnings. The contributions of each research component are articulated in Section 1.5.1, Section 1.5.2, and Section 1.5.3, repectively.

- Frequency-aware management in clouds.
- Security-aware energy management in clouds.

- Energy-aware privacy controls for clouds.
- I apply the novel model to measure the energy efficiency of virtual-machine-powered servers in data centers.

1.5.1 Contributions for Frequency-aware Management

I design a QoS-base strategy that adjusts frequency ratios to optimize energy efficiency. Next, I build a model that offer optimal frequency ratios customized for various processors in accordance to static power proportions. Then, I investigate the intriguing features of the frequency-ratio-based DVFS model. Finally, I apply the novel model to measure the energy efficiency of virtual-machine-powered servers in data centers. The contributions are tabulated in the list below.

- A QoS-base strategy adjusts frequency ratios to optimize energy efficiency.
- A new model offers optimal frequency ratios customized for various processors in accordance to static power proportions.
- Investigating the features of the frequency-ratio-based DVFS model.
- Applying the developed model for gauging the energy efficiency of data centers.

1.5.2 Contributions for Security-Aware Energy Management

I discuss the development of security overhead models for various security services, followed by proposing an idea to incorporate security and frequency awareness into the context of qualify of service (QoS). Next, I develop a security-aware energy management system for cloud computing environments. My novel energy management system, which is expected to achieve high security and energy efficiency in clouds, seamlessly integrates the security services, a security overhead model, and the security- and frequency-aware DVFS model. Then, I articulate an approach to translating security and energy requirements into a

multi-objective optimization problem that can be solved by the genetic algorithm *NSGA-II*. Finally, I implement the novel algorithm to enhance energy efficiency of computing servers where the prescribed security requirements are met. The contributions are tabulated in the list below.

- An improved security- and frequency-aware QoS model converts security overhead incurred in security-sensitive applications into frequency requirements
- The security-aware energy management system design achieves high security and energy efficiency in clouds by the virtue of a seamless integration of security services, a security overhead model, and the security- and frequency-aware DVFS model.
- Treating the security and energy requirements are defined in a format of a multi-objective optimization problem to be solved by the genetic algorithm NSGA-II
- The developed model is applied to optimize the energy efficiency and security strengths of data centers.

1.5.3 Contributions for Energy-aware Privacy Controls

Among all the energy-saving and privacy protection schemes for cloud computing, I shed bright a light on blockchain-based virtual-machine consolidation combining DVFS to offer energy savings and privacy protection in clouds. In this last part of the dissertation study, I investigate three connected research issues: (1) energy-aware privacy protection services, (2) energy-efficient blockchains, and (3) blockchain-enabled energy management modules in clouds. I design a blockchain-based virtual-machine consolidation framework accompanied by the DVFS (Dynamic Voltage and Frequency Scaling) technique to conserve energy consumption and privacy controls in clouds. This system combines virtual machine (VM) migrations with the DVFS technique to further improve energy efficiency. To protect data during VM migrations and data movement, I propose to make use of blockchain-enabled resource allocation to offer a transparent and trustworthy service on clouds. I promote

the blockchain technique as an advanced decentralized structure to avoid privacy leakage during VM migrations while guarding data against malicious tampering. My designed energy management system is expected to achieve high privacy and energy efficiency in clouds by orchestrating the blockchain, VM consolidation and frequency-aware DVFS model.

- Virtual machine (VM) migrations is combined with the DVFS technique to improve energy efficiency.
- Blockchain-enabled resource allocation is proposed to protect data during VM migrations and data movement.

1.6 Dissertation Organization

This dissertation is organized as follows. In the next Chapter, Chapter 2, related work reported in the literature is comprehensively and extensively reviewed. In Chapter 3 I describe a new frequency-aware QoS model, in which QoS requirements are represented using CPU frequencies rather than deadlines. The concept of frequency ratio accompanied by the frequency-aware DVFS model and the analysis of the DVFS model with respect to frequency ratio are proposed and detailed in this chapter. Moreover, I discuss the sample usages and applicability of the proposed frequency-aware model and conclude this paper with future work.

I kick off Chapter 4 with a research roadmap by presenting the concepts of security services and strengths. After I discuss the development of security overhead models for various security services, I propose an idea to incorporate security and frequency awareness into the context of qualify of service (QoS). In this chapter, I also present a security- and frequency-aware DVFS model, referred to as *SF-DVFS*, for cloud computing systems.

In Chapter 5, I dive into three connected research issues centered around the privacy-aware energy management in clouds: (1) energy-aware privacy protection services, (2) energy-efficient blockchains, and (3) blockchain-enabled energy management modules in

clouds. My novel energy management system is expected to achieve high privacy and energy efficiency in clouds by seamlessly integrating the blockchain, VM consolidation and frequency-aware DVFS model.

The last chapter (Chapter 6), I conclude this dissertation by summarizing an array of major research contributions. More importantly, I discuss future research directions from various perspectives that have not yet been fully addressed in Chapter 4 and Chapter 5.

Chapter 2

Literature Review

In the past decade, the scale of cloud data centers snowball to reach an unprecedented and unbelievable level, a growing number of large companies have chosen to utilize cloud datacenters (e.g., Microsoft Clouds and Amazon Web Services) to store enormous amount of data. The challenges of cloud datacenters can be categorized into two camps, namely, energy consumption and security preservation. In this chapter, I present a diversity of previous research studies that are closely related to this dissertation - the energy saving and privacy preservation in cloud data centers. I start off this chapter with the power model and stricture of cloud datacenter. Then, I focus on the existed energy-saving methods in cloud-based datacenters, and the sample methods include energy-aware hardware/software techniques as well as the popular DVFS scheme (Dynamic Voltage and Frequency Scaling). After that, research projects of task scheduling in cloud computing are surveyed and introduced. Finally, I reviewed a bevy of privacy protection strategies have been raised to protect data by considering the security issues of cloud data centers.

More specifically, this chapter is organized as follows. I introduce various power models of cloud datacenters in Section 2.1. Then, I discuss one of the major differences between cloud datacenter and traditional datacenter–virtual machines in Section 2.2. Next, classify leading-edge energy saving techniques from the following perspectives of infrastructure, hardware techniques, and software in section 2.3. Moreover, I classify in Section 2.4 existing real-time task models followed by scheduling schemes for clouds. Last, but not least, Section 2.5 summarizes the security issues in cloud computing, and Section 2.6 shows the related solutions for privacy protections.

2.1 Power Modeling of Cloud Datacenters

Prior studies have been focused on energy-consumption monitoring and management of datacenters from a global perspective [156]. In order to monitor and manage cloud severs in a much more meticulous view, researchers built up models to gauge the energy consumption of hosts. For example, Roy *et al.* [124] demonstrated that server energy consumption is the summation of CPU and RAM energy consumption. Additionally, Jain *et al.* [67] divided CPU and RAM into the accumulation of data and instruction features, respectively. Tudor *et al.* [144] incorporated I/O energy consumption into an energy-consumption model. Song *et al.* [138] replaced the previous I/O energy consumption with disk and network card energy consumption.

In addition to the above cumulative models, a handful of practical models shed light on the relationship between CPU utilization and total energy consumption of severs. For instance, Inozahy *et al.* [41] discovered the energy consumption models of severs under various CPU frequencies. Following-up studies (see, for example, [42][49]) investigated that energy consumption is linearly proportional to CPU frequency. On this basis, Ang *et al.* [141] raised two concerns about two dependent model parameters, which rely on specific server power to the model. Such a design made the energy consumption model fairly complicated. In contrast, linear-like models are popular and practical thanks to the advantages of simplicity and high accuracy. Therefore, in this study I adopt the linear-like modeling method to build a new model for DVFS-enabled clouds.

2.2 Virtual Machines and Load Balancing

2.2.1 Load Balancing for Virtual Machines

The purpose of load balancing is to evenly distribute computing workloads across multiple computing resources to maximize overall system performance. Load balancing aims to achieve an array of objectives, including (1) optimizing resource usage, (2) maximizing

throughput, (3) minimizing response time, and (4) avoiding the overload of any resource. VM-based load balancing is implemented through live VM migrations in data centers, where a primarily concern is to optimize the usage of physical computing resources by migrating virtual machines from heavily loaded physical machines (PMs) to those with least workload. By dynamically adjusting the locations of VMs, one may optimize various objective functions to provide superb cloud services. Sample objective functions include, but are not limited to, improving performance, boosting system security, minimizing failure impact, and reducing energy consumption.

Fig. 2.1 illustrates a classic load balancing architecture in a cloud computing platform. All user requests are submitted to the load balancing module, which is responsible for dispatching requests to virtual machines to optimize resource utilization and energy efficiency.

Load balancing plays a critical role in guaranteeing the service-level agreements (SLAs) of applications in cloud computing. The increasing workload of applications in virtual machines may trigger overloaded utilization in one resource or more (e.g., CPU, memory, I/O and network bandwidth) on physical machines. More often than not, an overloaded physical machine degrades application performance of all the VMs running on the PM. Consequently, unbalanced loads inevitably impose an adverse impact on the finish times of batch applications and the response times of interactive applications. To eliminate the potential bottleneck, one has to migrate excess load from overloaded physical machines to under-utilized ones in computing clouds.

2.2.2 Challenges in Load Balancing

It is arguably true that load balancing techniques powered by VM migrations confront the following challenges.

- **Overhead.** It is prudent to quantify the amount of overhead involved in deploying a load balancing system. Load balancing overhead entails VM migration cost and communication cost. For example, load of each physical machine ought to be periodically

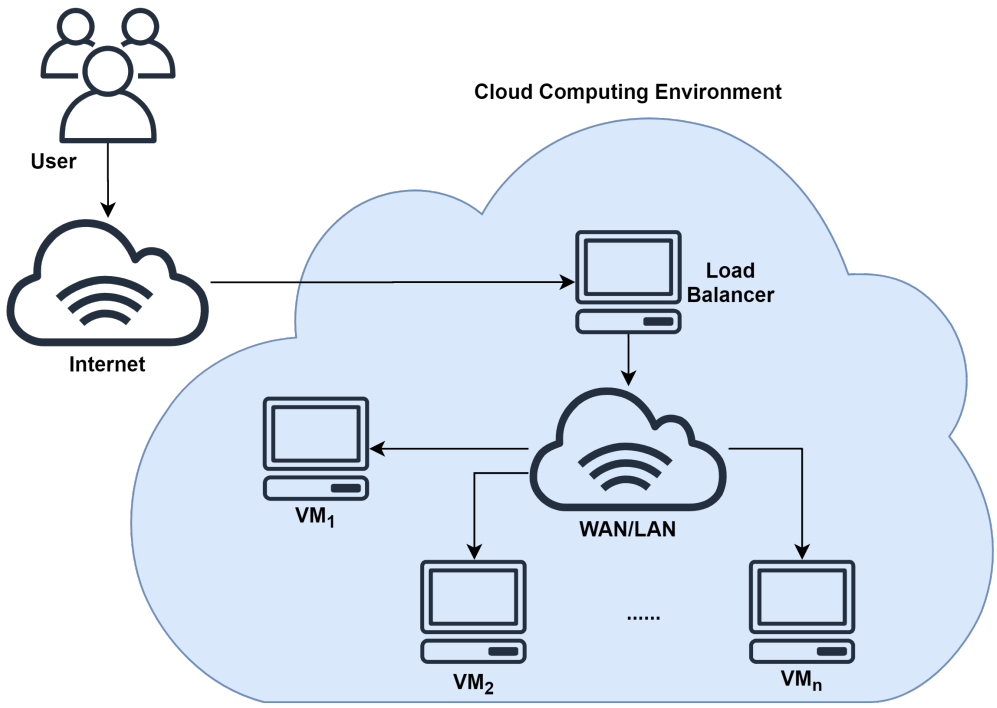


Figure 2.1: A load balancing architecture for clouds, where all user requests scheduled and dispatched by the load balancing module to optimize resource utilization and energy efficiency.

collected by a load balancing mechanism, which pays the communication cost to monitor load across multiple PMs. A well-designed load balancing algorithm should reduce such an overhead.

- **Prediction.** Due to the dynamic changes of application workload in VMs, it is inefficient to make migration decisions merely based on the current status of the system. An ideal load balancing algorithm should be equipped with the capacity to accurately predicting workload to orchestrate VM management prior to any sharp changes in future load. Such proactive approaches avert making last-minute load-balancing decisions, which are in some cases too late.

- **Performance.** Various performance metrics are introduced to assess the efficiency of cloud computing systems. Performance of a computing cloud can be measured from the perspectives of system throughput as well as user experience and satisfaction. Given performance requirements prescribed by end users, computing clouds are responsible for

ensuring such requirements defined as quality of service (QoS). Modern load balancing mechanisms seek to boost overall system performance while meeting QoS requirements.

2.3 Green Cloud Data Centers

Cloud computing has radically changed the landscape of computing, storage, and communication infrastructures. With strong interest and investment from the industry and government, cloud computing infrastructures are being increasingly patronized by both organizations and individuals. With increasing energy prices and data center scaling, high energy consumption has become an impediment to the development of cloud-computing environments. Reducing operational costs of data centers should be achieved through boosting energy efficiency. As such, optimizing the energy efficiency of data centers supporting cloud computing has captured much attention. A flood of intriguing studies have been recently conducted to facilitate the development of energy-efficient data centers. As shown in Fig. 2.2, I classify leading-edge energy saving techniques from the following perspectives of infrastructure, hardware techniques, and software solutions.

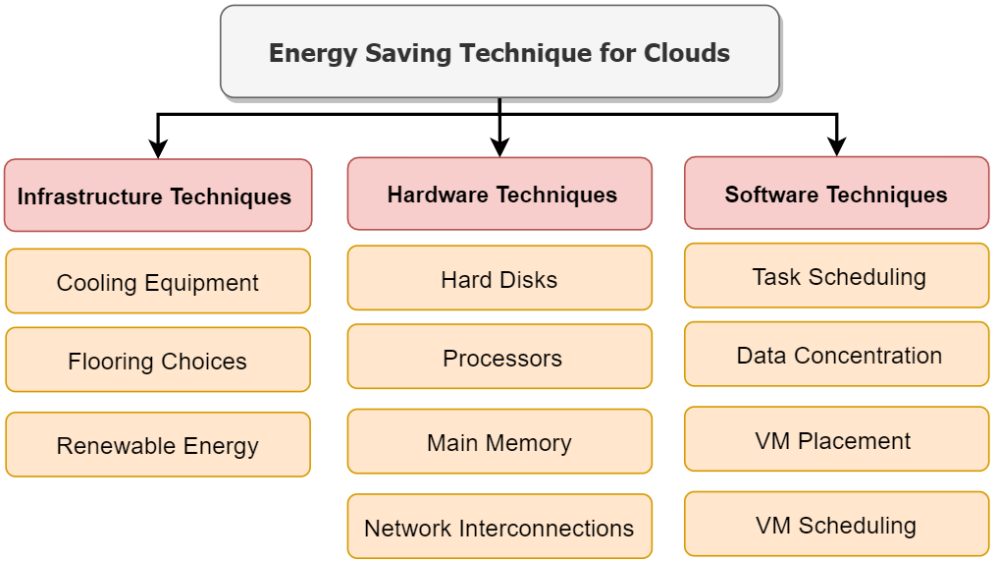


Figure 2.2: Commonly adopted energy conservation techniques for clouds.

2.3.1 Energy-Efficient Infrastructure Techniques

Infrastructure techniques intend to curb energy consumption by building green data centers. The infrastructural energy conservation techniques entail cooling equipment [109], flooring choices [71], and renewable energy sources [167].

With regard to the cooling principle, existing cooling solutions fall into three camps, namely, air-cooling, liquid-cooling or free-cooling schemes. Thanks to low operational cost accompanied by simple maintenance, the air-cooling technology is the most conventional way of cooling down large-scale data centers [109]. Unlike air-cooling solutions, liquid cooling is one of the most prominent and practical methods to be directly or indirectly implemented in data centers. An indirect liquid-cooling system embraces a heat dissipation process where heat sources and liquid coolants contact indirectly [82]. In contrast, liquid coolant in a direct liquid-cooling method directly contacts electronic devices, where dielectric fluid offers electrical insulation [133]. Furthermore, a raft of data centers leverage the free-cooling technology to conserve cooling cost by the virtue of natural free cooling sources [164]. For instance, Facebook constructed a naturally cooled data center in northern Sweden in 2013 [147].

Apart from the above versatile cooling techniques, a growing number of green data centers adopt flooring techniques with perforated tiles and a raised floor plenum for cool air intake [71]. The application of renewable sources of energy (e.g., solar, geothermal, wind, hydro power) unequivocally and considerably cut back energy consumption in data centers. For example, Zhang *et al.* devised *GreenWare*, a novel middle-ware system that conducts dynamic request dispatching to enhance the percentage of renewable energy powering a distributed data center [167]. *GreenWare* fully utilizes renewable energy sources while meeting the desired cost budget for cloud service providers.

2.3.2 Energy-Aware Hardware Techniques

Acquiring a diversity of energy-efficient hardware components makes it possible and desirable to construct green data centers with high energy efficiency. Energy-efficient hardware

techniques deployed in modern data centers include hard disks [137], processors [75], main memory [35], and network interconnections [10].

When it comes to data storage systems, solid-state disks and multi-speed disks are proved to be capable of trimming energy consumed by disks [137]. The dynamic voltage frequency scaling (DVFS) technique [75] is a popular technique, which is a feasible solution to conserve energy consumption of DVFS-enabled CPU and main memory - key underpinnings in computing servers. It is evident that DVFS enables processors and main memory to consume less power by electing the most appropriate frequency and supplied voltage. For example, Garg *et al.* developed the near-optimal energy-efficient scheduling algorithms, where DVFS is employed to decrease carbon emission by scaling down CPU frequency and optimizing cloud providers' profits [50]. Speaking of energy-aware network interconnects for data centers, an array of network architectures have been proposed and customized for data centers. Representative techniques include, but not limited to, energy proportional networks [8], networks based on the elastic tree topology [59] and the *Proteus* network [134].

2.3.3 Energy-aware Software Techniques

A wide range of software techniques were designed to conserve energy consumption in clouds. Such cutting-edge software solutions include task scheduling [38], data concentration [114], virtual machine (VM) placement [106] and scheduling [89].

Dong *et al.* developed a greedy task-scheduling policy (the most efficient-server-first scheduling) to enhance energy efficiency of servers deployed in data centers. This scheduling scheme shortens the average task response time while minimizing the energy expenditure of servers [38]. Pinheiro *et al.* devised the popular data concentration (PDC) technique, a promising technique that migrates frequently accessed data to a small subset of disks [114]. The overarching goal of PDC is to skew the load towards a few of the disks, allowing the other disks to be transitioned to a low-power energy-saving mode [114]. VM placement, consisting of VM migration and consolidation, is one of the most common approaches to

achieving high energy efficiency by dynamically scaling down the size of running clusters. With the help of virtualization, energy consumed by cluster computing infrastructures are immensely reduced by applying energy-aware VM migrations and consolidations [106]. In the arena of VM management, VM scheduling is an outstanding energy-saving method in cloud environments. For example, Li *et al.* proposed GRANITE - a holistic virtual machine scheduling algorithm being capable of minimizing total energy consumption in a data center [89]. GRANITE embraces an elaborate thermal model, which is adept at analyzing the temperature distribution of airflow and processors [89].

2.3.4 Dynamic Voltage and Frequency Scaling

Dynamic voltage and frequency scaling or DVFS is a widely adopted energy-conservation technique for clusters housed in data centers. Much attention has been paid toward DVFS-based task schedulers that minimize the total energy consumption without violating deadlines [162]. Aydin and Yang [16] showed that task scheduling on multiprocessors to minimize energy consumption is an NP-Hard problem in the strong sense. As such, a heuristic EDF (i.e., Earliest Deadline First) scheduler was devised to minimize CPU energy of multiprocessors. For example, Wu *et al.* [155] applied DVFS to reduce the energy consumption of a server during the course of idle or light workload. Unlike the existing DVFS techniques that are focused on dynamic CPU power, my model pay heed on CPU static power consumption due to CMOS circuits.

The utilization-based DVFS scheme devised by Arroba *et al.* [15] is adept at making excellent trade-offs between energy consumption and performance degradation. Lin *et al.* proposed a two-tier algorithm to solve the local chip mapping problem with polynomial time complexity. Pahlevan *et al.* investigated an energy proportionality-aware dynamic allocation method (EPACT) [112] - a novel dynamic VM allocation scheme combining DVFS and VM consolidation. It is assumed in this study that the optimal utilization is a fix value for all the servers (e.g., 50% when satisfying QoS). My solution is similar to the above studies in the

way that all the approaches address the concerns of static power consumption. Nevertheless, there are the following two stark differences between the aforementioned utilization-based DVFS techniques and my method. First, the former one assumes that task execution times are specified in a priori, whereas ours orchestrates frequency requirements. My solution is beneficial for non-stopping services (e.g., web crawlers and streaming applications). Second, unlike the fixed optimal utilization in the existing study (see [112][91]), the optimization criteria in ours are configurable. It is evident that my method can be readily customized for a raft of servers.

2.4 Scheduling in Clouds

At the heart of a cloud computing platform that orchestrates a diversity of virtualized resources, scheduling mechanisms become a vital component to optimize resource utilization. A client may leverage multiple virtualized computing resources to accomplish tasks submitted to clouds. An overarching goal of task scheduling is to slate tasks running on computing clouds to achieve specific objectives. Sample objectives include minimizing response time, maximizing performance, reducing energy consumption, and improving system security, to name just a few.

Fig. 2.3 depicts a scheduling architecture designed for cloud computing platforms, in which scheduling mechanisms and security-service optimization modules are fully integrated. Similar architectures can be found in the literature (see, for example, [7][96]). In the architecture illustration, a cloud is depicted in a dotted box. Cloud users dynamically submit a wide range of tasks to the scheduler, which oversees virtualized resources in the cloud. After scheduling decisions are made by the scheduler, tasks are dispatched to corresponding virtual machines. As a part of the scheduling mechanism, a monitor periodically keeps track of the utilization of the virtual machines as well as physical machines in the cloud. Apart from scheduling tasks, the scheduler is in charge of launching appropriate security services for input and output data of tasks to fulfill user requirements.

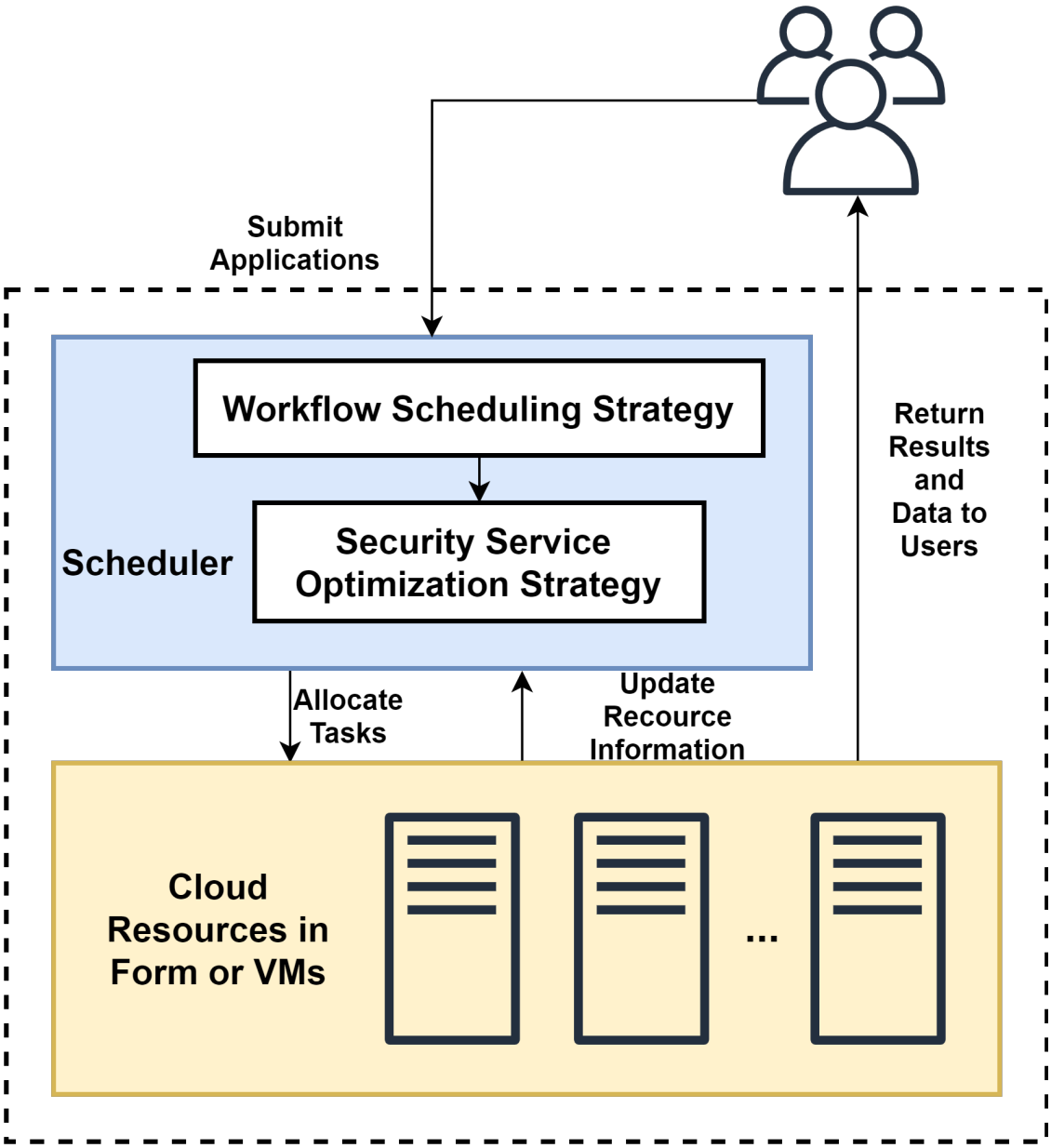


Figure 2.3: The scheduling architecture for cloud computing platforms, which embrace scheduling mechanisms and security-service optimization modules.

2.4.1 Task Models for Real-time Computing

Before reviewing real-time scheduling algorithms in Section 2.4.2, I briefly go through the background knowledge of tasks models that lay a solid foundation for task scheduling research. When it comes to real-time tasks, corresponding *QoS* requirements ought to be satisfied when conserving system energy consumption. Task models become an underpinning component of energy-efficient schedulers for real-time tasks [87]. For instance, Wang *et al.* [149] constructed a model, in which tasks have various priorities coupled with task-precedence constraints. In some cases, a group of tasks are represented in the form of a weighted directed acyclic graph or DAG.

Real time tasks embrace deadlines, which are specified in the format of QoS requirements. Missing deadlines is treated as a failure or an error for the real-time tasks submitted to clouds. The ability to satisfy deadlines (a.k.a., timing constraints) of real-time tasks is an overarching goal to be achieved by schedulers managing virtualized resources in cloud computing environments.

As conventional schedulers, real-time schedulers customized for clouds aim to make good trade-off among multiple factors such as scheduling complexity, real-time performance, energy efficiency, and security [157]. Real-time tasks ought to be carried out by clouds correctly and in a timely fashion. Evidence shows that obtaining a minimal schedule for a set of real-time tasks running in multiprocessor systems is an NP-hard problem [171]

In another model, an array of tasks share a common deadline [160][76]; these tasks have diversified worst case execution time or WCET. Aydin and Yang built a periodic task model, for which a scheduling algorithm improves energy efficiency [16]. With the tasks models in place, numerous scheduling algorithms were implemented for saving energy of computing systems running real-time applications [125][175][115]. In contrast to the existing task models, ours incorporates frequency requirements rather than relying on task execution times. My model addresses a critical issue in a raft of real-time applications, where task execution times are unknown in a priori.

2.4.2 Real-Time Scheduling

The timeliness of to real-time applications is a key toward high quality of service (QoS) on clouds. Virtual machines can be handled as tasks from the perspective of real-time scheduling. Therefore, I use terms virtual machines and tasks interchangeably throughout this manuscript.

For hard real-time applications, the timeliness measures the system capability of guaranteeing deadlines specified by users. In the realm of cloud computing, timeliness is referred to as a performance metric that entails the sum of utility or benefits obtained by real-time tasks or services [97].

Real time tasks embrace deadlines, which are specified in the format of QoS requirements. Missing deadlines is treated as a failure or an error for the real-time tasks submitted to clouds. The ability to satisfy deadlines (a.k.a., timing constraints) of real-time tasks is an overarching goal to be achieved by schedulers managing virtualized resources in cloud computing environments.

As conventional schedulers, real-time schedulers customized for clouds aim to make good trade-off among multiple factors such as scheduling complexity, real-time performance, energy efficiency, and security [157]. Real-time tasks ought to be carried out by clouds correctly and in a timely fashion. Evidence shows that obtaining a minimal schedule for a set of real-time tasks running in multiprocessor systems is an NP-hard problem [171]. Unsurprisingly, real-time schedulers are unable to deliver deterministic response times, which are an important metric gauged for system robustness analysis. Security-sensitive real-time tasks running on clouds must be protected against cyber-security threats, which make the design of resource management systems for clouds a grand challenge. To address the aforementioned challenging issues, I will pilot a security- and frequency-aware DVFS model (SF-DVFS) to incorporate security services and energy management in a computing cloud. Please refer to my roadmap elaborated in Section 4.3.2 for a detailed research plan on FS-DVFS.

2.4.3 Energy-aware Scheduling

In the past decade, high energy consumption in cloud-based data centers has motivated the research community to develop energy-efficient techniques, among which a growing number of energy-aware scheduling algorithms offer impressive energy savings to computing clusters on clouds [174][166]. Generally speaking, energy-efficient scheduling approaches can be categorized into two camps, namely, DVFS-based (Dynamic Voltage and Frequency Scaling) and VM-based (Virtual Machine) techniques.

DVFS-based Scheduling

Recall that DVFS-based schemes strive to make good trade-offs between energy consumption and performance in processors, which are a major player in reducing power consumption of data centers. For example, Garg *et al.* developed the near-optimal energy-efficient scheduling algorithms, where DVFS is employed to minimize carbon emission by scaling down CPU frequency while maximizing profits of cloud providers [50]. Fettes *et al.* designed practical scheduling policies, which seamlessly integrate DVFS and the virtual-machines consolidation scheme to make cloud-based data centers energy efficient [43]. Maroulis *et al.* applied DVFS to curb the energy consumption of MapReduce applications running on computing clusters [101]. Suleiman *et al.* merged the thermal-aware approach and DVFS in a smart way to offer power management in data centers [140]. Duan *et al.* devised an algorithm to judiciously tune CPU frequency in accordance with QoS requirements [40]. In this algorithm, a prediction method was incorporated to adapt CPU frequency by jointly considering QoS and available slack time. Consequently, the novel scheduler is capable of reducing energy consumption in heterogeneous Hadoop clusters. Similarly, Ibrahim *et al.* mixed the DVFS and machine learning approaches to slash energy consumption in network-on-chips systems (NoCs) [63].

Virtual-Machine-based Scheduling

A tremendous effort in building energy-aware schedulers over the past several years has concentrated on dynamical consolidation of virtual machines. A vast majority of such scheduling algorithms aim to manage virtual machines according to dynamic system workload, thereby cutting back the number of physical hosts so that idle hosts are switched off to conserve energy. Recently developed scheduling strategies leverage live migrations of virtual machines to support multiple fields, including scientific workflows and real-time tasks. For example, Xu *et al.* designed an energy-aware resource allocation method to allocate virtual machines in support of scientific workflow executions [159]. After proposing a novel rolling-horizon scheduling architecture for real-time tasks running on clouds, Zhu *et al.* implemented an energy-aware scheduling algorithm called *EARH* for real-time, aperiodic, independent tasks [172].

A wide range of scheduling algorithms was designed to conserve energy consumption in clouds by the virtue of virtual-machine migrations and consolidation. For instance, Khazaei *et al.* proposed a scheduling technique to minimize service delay in clouds by lowering transmission and processing times through virtual-machine migrations [122]. After investigating a way of dynamically consolidating tasks to boost resource utilization and to reduce energy consumption, Hsu *et al.* presented an energy-aware task consolidation (ETC) method to optimize energy efficiency in clouds [61]. To take uncertain-ties into account, Chen *et al.* employed proactive and reactive algorithms to mitigate adverse impacts of uncertainties on scheduling quality of cloud-based data centers [28].

2.4.4 Cloud-aware Scheduling

Online Scheduling

Much attention has been paid to towards online scheduling of multiple tasks and jobs. For example, Shin et al. extended the conservative back-filling algorithm by utilizing the

earliest deadline first and the largest weight first policies to schedule real-time jobs [132]. Ge *et al.* dived into a genetic algorithm based task scheduler, which manages waiting tasks through a genetic algorithm with a goal of balancing load [52]. Liu and Han proposed an online scheduler allowing virtual machines to obtain extra CPU shares when blocked by I/O interrupts, thereby curtailing energy-efficiency losses caused by I/O intensive tasks [92].

Scheduling for Multi-processors

When cloud computing platforms are fueled by multi-processor systems, scheduling algorithms are focused on enhancing the overall performance of multi-processor systems. For instance, Dorronsoro.et al. presented a two-level strategy for scheduling large workloads on multicore distributed systems, taking into account their total execution time and energy consumption [39]. Kwok and Ahmad devised an array of optimal static algorithms to schedule task graphs with random parameters for multiple homogeneous processors [80]. Similarly, Mohamed and Awadalla proposed multi-processor-based scheduling approaches, namely the modified list scheduling heuristic (MLSH) and the hybrid genetic algorithm (GA) [108].

Performance-aware scheduling

Performance-aware scheduling solutions were deployed to optimize system performance measured in terms of response time, makespan, and completion time. Please refer to [103] and [163] for the comprehensive surveys on task and resources scheduling policies that are intended to speed up system performance of clouds. For example, Tang *et al.* designed a self-adaptive scheduling algorithm for jobs running on MapReduce-based computing clusters [142]. This algorithm dynamically decides the start time of each reduce task according to the corresponding job's context such as task completion time and map tasks' output size. Gan *et al.* implemented a genetic simulated annealing algorithm to optimize the makespan of a set of tasks. In this approach, simulated annealing is used to optimize each offspring yielded by the genetic algorithm [47]. Furthermore, an improved genetic algorithm was

developed to apply the outputs of Max-Min and Min-Min as initial solutions to schedule independent tasks [79]. Zuo *et al.* proposed a multi-objective ant colony algorithm to address the task scheduling problem. The focal point of this multi-objective algorithm is to minimize makespans by incorporating user-budget costs as constraints during the course of task scheduling [176].

2.5 Security Issues in Cloud Computing

Fig. 2.4 summarizes the five major data security issues to be addressed in the arena of cloud computing. A risk of data misuse is likely to occur when resources are shared among multiple organizations. To avert such a risk, it is prudent to secure storage infrastructures along with processed and archived data. Data protection, a vital and challenging feature of cloud computing, keeps any potential security threats at bay. Authentication, authorization, and access control services are devised for to enhance data security in clouds.

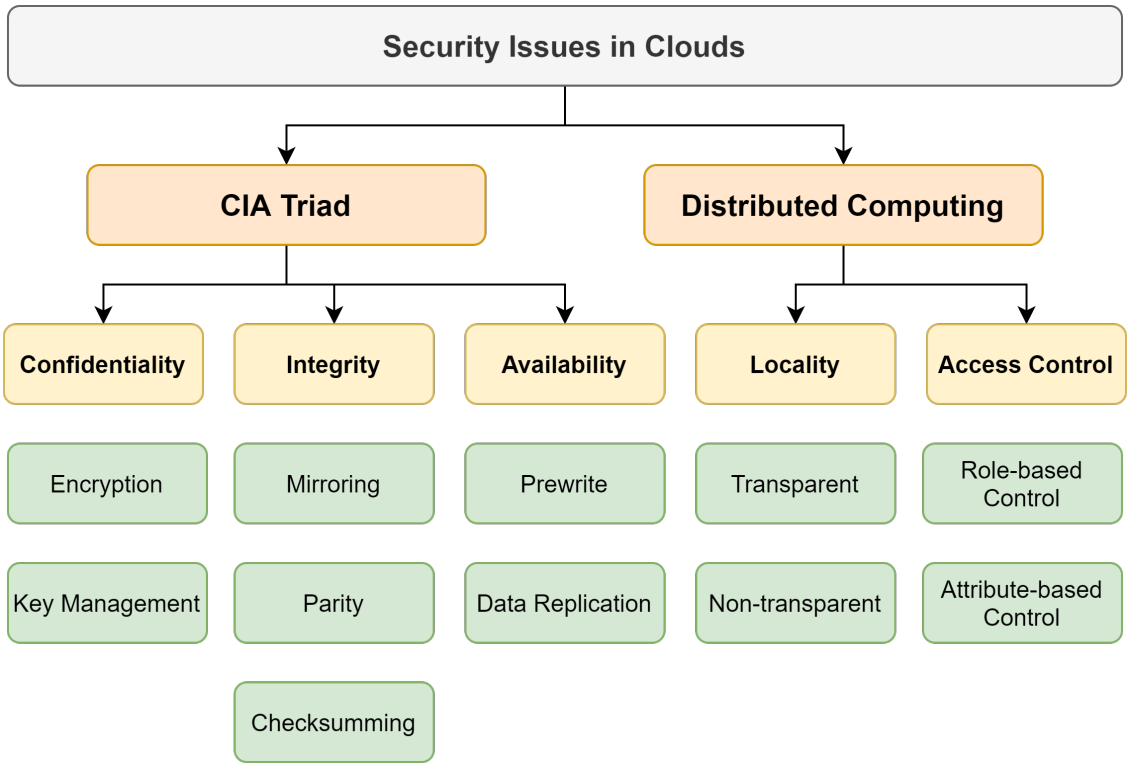


Figure 2.4: Five major data security issues to be addressed in the arena of cloud computing.

2.5.1 Confidentially, Integrity, and Availability

Confidentiality, *integrity*, and *availability*, which are known to as the CIA triad, are the three critical properties of data centers. Confidentiality ensures that data owned by cloud service consumers should not be revealed to unauthorized parties under any circumstance [15]. Various encryption techniques [116] and key management [85] mechanisms are deployed to ensure high confidentiality of cloud services. Data integrity entails confidence that data stored in and transferred to/from clouds are not fiddled by unauthorized users. Data integrity can be detected by modern techniques like mirroring, parity, or checksumming at either the file or the block levels [135]. Data availability implies that data should be readily accessed by users without any delay or denial of service when the users issue requests. A handful of leading solutions are available to achieve high data availability. For example, data replication [54] and prewrite operation [95] are two common practices to furnish high data availability to cloud computing systems.

2.5.2 Distributed Computing

When it comes to distributed computing in clouds, two security challenges to be tackled are *locality* and *access*. Nowadays, data tends to be distributed across multiple regions, where pinpointing the location of data is non-trivial. When data are moved or migrated to/from one geographic location into another, the laws and regulations governing the data may change. Consequently, cloud service providers must be compliant with data privacy laws according to the geographic locations. This emerging challenge is referred to as *locality* issues of data security in cloud computing environments. Such a data locality issue is handled by clouds in two fashions. On one hand, cloud service providers make data locality transparent to end users. On the other hand, users are in full control of data locations to meet prescribed security requirements. I refer to the former one as transparent data locality and the later one as non-transparent data locality. A benefit of the transparent approach is that users

can easily access their data without being aware of the locations of data. In contrast, non-transparent location policies enable a cloud user to elect desired service locations to safeguard data with respect to locality.

Access control is regarded as a second security issue in distributed computing over clouds. In an organization where computing platforms are outsourced to clouds, members of the organization are authorized to manage a portion of data in accordance with access policies. Such data may not be retrieved or modified by the other members of the organization in the distributed computing environments. Most leading-edge access control techniques applied to cloud computing fall into two camps, namely, role-based and attribute-based schemes. For example, Zhou *et al.* designed a role-based encryption scheme to enforce access control policies for encrypted data stored in public clouds [169]. Yang *et al.* developed a time-domain attribute-based access control scheme, which allows a group of users to securely share videos in clouds [161].

2.6 Privacy Protections in Cloud Computing

The key benefits of cloud computing, from the cloud provider's perspective, include resource consolidation, uniform management, and cost-effective operation. When it comes to cloud computing users, cloud computing's benefits encompass on-demand capacity, low cost of ownership, and flexible pricing. The sharing and consolidation features that bring forth such benefits inevitably introduce potential security and privacy concerns. Security and privacy issues arising from illegal and unethical use of data as well as disclosure of confidential information can tremendously hinder users' willingness to participate in cloud-based services. Recognizing such security concerns, a growing research and development efforts in the industry and academia have been devoted to preserving the cloud's data privacy. Domingo-Ferrer *et al.* classified the privacy protection techniques into three categories (see Fig. 2.5), namely, (1) data splitting mechanisms, (2) data anonymization methods, and (3) cryptographic techniques [37].

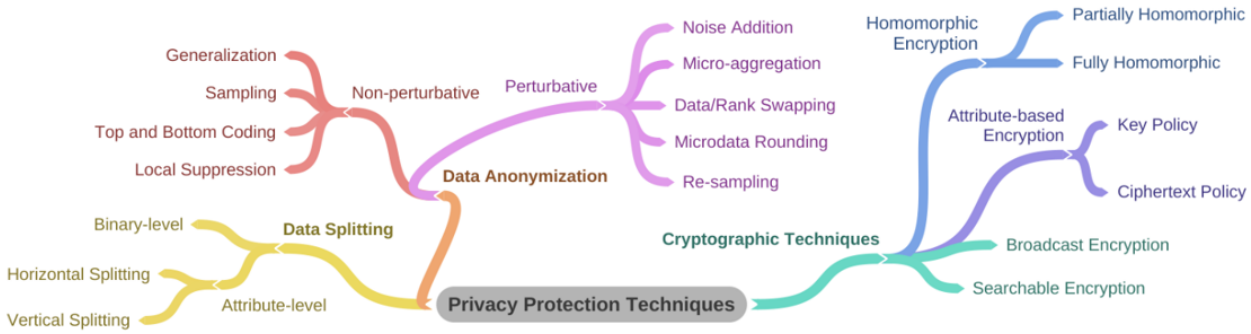


Figure 2.5: Commonly adopted privacy protection techniques for clouds.

2.6.1 Data Splitting

Privacy-preserving data splitting protects data privacy by deploying multiple-CSPs-based (cloud service providers) architectures [17]. It is evident that data splitting minimizes information leakage through distributed data among an array of CSPs. This technique is proved to be a practical solution as long as the distributed CSPs have zero communication with one another.

A horde of data splitting mechanisms devised in the prior studies undertake data partitioning at the binary level. For instance, Zhang *et al.* implemented a scheme to split sensitive files into bits, which are reassembled to form numerous part-files before being uploaded to various cloud storage servers. After part-files are downloaded from multiple cloud servers, the part-files are concatenated to build an original file [165]. To strength the security protection for split chunks, Gai *et al.* mixed the byte-level data splitting technique with an encryption module [46]. The secure-efficient data distributions algorithm or *SED2* was proposed to spill data in a way of preventing sensitive information from leaking on clouds. The SED2 algorithm was realized through two underpinning algorithms, which are slated to efficiently encrypt and decrypt data [46]. Dev *et al.* assigned each file a privacy level in accordance with the sensitivity of the file's content [36]. Then, file fragments are created under the governance of a standalone RAID storage system, in which data with high privacy levels are stored in trustworthy locations [36].

Apart from the aforementioned binary-level strategies, innovative split methods built at the attribute level capture much attention. In this technique category, data splitting may be executed in a horizontal or vertical fashion. A horizontal format implies that sets of data records are separately stored, whereas a vertical layout indicates that sets of attributes are separately stored. In the case of horizontal data splitting, data sets are structured in a tabular format according to attributes. To achieve confidentiality at the record level, vertical chunks are comprised of all data records on a single attribute. Aggarwal *et al.* designed an approach to decomposing a dataset into two privacy-preserving vertical fragments [9]. With the deployment of the graph-coloring techniques, the proposed decomposition algorithm cuts back the data querying cost. In case sensitive attribute pairs require more than two chunks to preserve data privacy, an encryption module will be incorporated [9]. Ganapathy *et al.* investigated a solution based on two fragments coupled with a data encryption service [48]. The three additional heuristics were developed to shorten query time by applying the greedy hill-climbing algorithm. In this study, the time complexity of the proposed data splitting solution was articulated [48].

2.6.2 Data Anonymization Methods

A key advantage of anonymized data over encrypted data and data splitting is rooted in ease of data processing. In the realm of *data anonymization*, masking is merely performed at the data storage phase. More times than not, data anonymization are implemented by linear or quasi-linear algorithms. In this type of approaches, anonymized data's query, being transparent, does not incur hefty overhead for clouds. Original data of a micro data set are manipulated to originate new data, which are applicable for statistical analysis. In doing so, the confidentiality of respondents is enforced. Masking methods in turn are be divided into two camps, depending on the effect on original data (see Fig. 2.6).

Because Non-perturbative methods have no intent to alter data, these methods yield partial suppressions or reductions of detail in an original dataset. Representative non-perturbative masking methods are sampling, generalization, top and bottom coding, and local suppression. A sampling scheme masks a sample of original files rather than publishing the original files [154]. If an intruder identifies a unique record in released file (sample), the intruder will be unsure about the data uniqueness in the original file. Generalization, which is referred to as global recording, forms a new less specific attribute by combining several more specific categorical attributes [127]. Top and bottom coding approaches can be envisioned as special cases of generalization techniques, where a new category is forged by gleaning values that are above (top coding) or below (bottom coding) a threshold [32]. A local suppression mechanism aims to suppress the values of individual attributes to increase a set of records supporting quasi-identifiers. Chen *et al.* explored a local suppression method to build a customized privacy model for trajectory data anonymization [30].

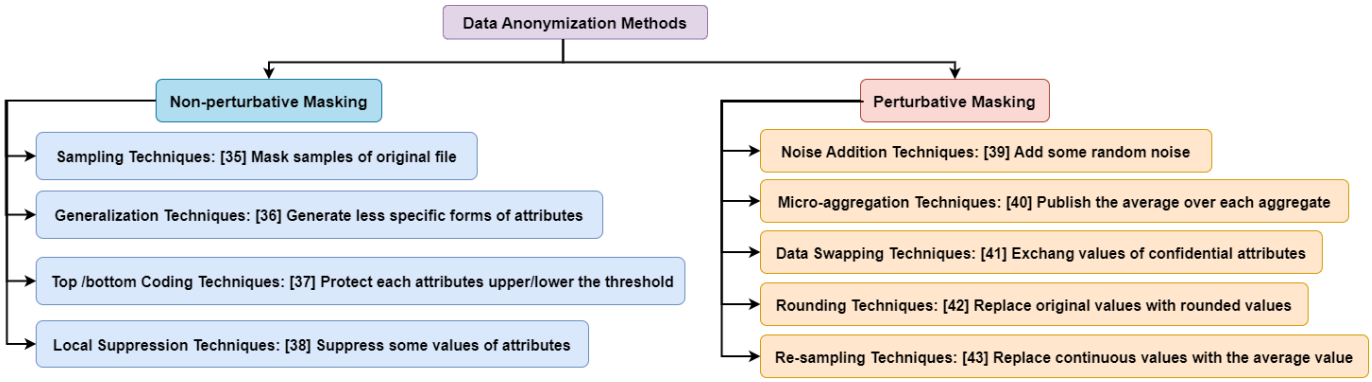


Figure 2.6: Nine data anonymization methods for clouds.

A perturbative masking technique manipulates a dataset in a way that respondents' privacy is preserved to a certain degree. It is noteworthy that such an approach aims to protect statistical properties of the dataset. Representatives of leading-edge perturbative masking methods are noise addition, micro-aggregation, data/rank swapping, microdata rounding, and re-sampling. A noise addition scheme intends to mask an original dataset by injecting random noise [31]. A micro-aggregation solution groups individual tuples into small

aggregates of a fixed dimension k, where an average over each aggregate rather than individual values is published [139]. Data/rank swapping techniques are adroit at transforming databases by switching values of confidential attributes across stored records [123]. Rounding techniques substitute rounded values for the original values of attributes [110]. The key idea behind re-sampling schemes is to exchange the values of a continuous attribute with an average value derived from a set of samples gleaned from original data [102].

2.6.3 Cryptographic Techniques

Cryptography is one of the predominant building blocks deployed to address privacy concerns in the clouds. For example, growing evidence indicates that homomorphic encryption [120], attribute-based encryption [126], broadcast encryption [58], searchable encryption [70] are adopted to offer remote secure computation solutions for clouds. These diversity of techniques furnish fine-grained access controls in cloud storage.

In homomorphic cryptosystems, encryption functions are a homomorphism that protects group operations by the virtue of ciphertexts. With homomorphic encryption algorithms in place, one is enabled to perform computations on ciphertexts without decrypting data in advance, thereby preserving data privacy. After Rivest *et al.* introduced homomorphism in 1978 [120], all the homomorphism schemes are classified into two categories, namely, partially homomorphic encryption and fully homomorphic encryption. A partially homomorphic encryption solution supports merely one type of operation repeatedly running for unlimited times [72]. In contrast, a fully homomorphic encryption allows an unlimited number of operations performed on encrypted data, where output data range within a given ciphertext space [53][14].

Attribute-based encryption approaches are proved to be a practical and promising technique. These solutions cater to facilitate encrypted fine-grained access controls for outsourced data. An attribute-based encryption or *ABE* applies public-key encryption where the secret key of a user as well as ciphertext rely on attributes. The decryption of a ciphertext becomes

feasible only if the attribute set of the user key matches the ciphertext's attributes [126]. Given an access policy, I group these solutions into two camps - (1) key policy attribute-based encryption schemes or KP-ABE and (2) ciphertext-policy attribute-based encryption schemes or CP-ABE. A KP-ABE scheme employs an attribute set to model encrypted data and to construct an access policy in private key [118]. On the flip side, ciphertext in CP-ABE is associated with an access policy, whereas secret keys are associated with attributes [88]. In this case, data owners are enabled to elect users who have the privilege to decode. If the policy ought to be frequently managed, the CP-ABE schemes will be flexible because the data owner can readily update the ciphertext's access structure.

Now let us introduce broadcast encryption and searchable encryption schemes - two popular cryptographic techniques. Broadcast encryption solutions allow a broadcaster to encrypt messages and to transmit the messages to any subset of authorized users. Given a broadcast encryption scheme, a broadcasting sender is positioned to encrypt a message by combining receivers' public identities in the subset coupled with system parameters. In doing so, only a dynamically changing privileged subset of users are able to decode encrypted messages [58]. Searchable encryption techniques equip data users with the capability to securely search over encrypted data using keywords without having to decrypt the data. The overall objective of a searchable encryption system is to combine confidentiality with respect to cloud providers with a powerful search functionality. Kamara *et al.* made use of the multicore architecture to realize a searchable encryption scheme [70]. Such a multicore-based implementation makes the searchable encryption module highly scalable. This novel solution offers sublinear search time by the virtue of a tree-based multimap data structure per keyword, which is referred to as red-black trees [70].

Chapter 3

A Frequency-aware Management Strategy for DVFS-Enabled Clouds

It is demanding to curtail energy consumption of virtual-machine-powered data centers, because modern data centers have been significantly scaling up in capacity in past decades. In this chapter, we propose a frequency-aware management strategy, which controls dynamic power and static power of processors running virtual machines in data centers. Unlike existing dynamic voltage and frequency scaling schemes, my strategy simply incorporates frequency requirements rather than task execution times. This salient feature is practical, because task execution times in a raft of real-world applications are unknown in a priori. We build a frequency-aware model, which is adept at deriving an optimal frequency ratio that minimizes processors' energy consumption. With my model in place, the energy efficiency of a data center can be maximized by adjusting the processor's frequency to meet the optimal frequency ratio. We design a management approach to judiciously adjust frequency ratio to conserve energy without violating the frequency requirements imposed by virtual machines. After analyzing the correlations between frequency ratio and energy consumption, we show that a small static power proportion gives rise to high energy-saving performance. The results demonstrate that my model lays out a solid theoretical foundation catering to the development of power management software in DVFS-enabled clouds.

The remainder of this chapter is organized as follows. Section 3.1 describes a new frequency-aware QoS model, in which QoS requirements are represented using CPU frequencies rather than deadlines. The concept of frequency ratio accompanied by the frequency-aware DVFS model are proposed in Section 3.2 and Section 3.3, respectively. The analysis of the DVFS model with respect to frequency ratio can be found in Section 3.4. Section 3.5

discusses the sample usages and applicability of the proposed frequency-aware model. We conclude this chapter with my achievements in Section 3.6.

3.1 Frequency-Aware QoS

Recall that the DVFS conserves energy consumption by lowering processor frequency. Scaling down processor frequency inevitably increases task execution times, which may incur violations in service level agreements or SLA. In this section, we shed some light on the concept of worst case execution time in Section 3.1.1. Next, we discuss a way of applying my model to satisfy QoS requirements in Section 3.1.2.

3.1.1 Worst Case Execution Time

From the perspective of QoS modeling, virtual machines are treated as standalone tasks running on servers. As such, we use terms virtual machines and tasks interchangeably throughout this section. Table 3.1 lists the notation used throughout this chapter.

Table 3.1: Symbol and Annotation

Symbol	Annotation
f_c^{opt}	optimal energy-saving frequency for processor c applied my model.
$f_{i,c}^{req}$	frequency requirement of virtual machine vm_i on processor c
$f_{V,c}^{req}$	frequency requirement of virtual machine set V on processor c. V is the set of virtual machine running on processor c.
$f_{V,c}^{conf}$	frequency configured for virtual machine set V on processor c
Γ_i	total execution requirements of virtual machine vm_i.
$\gamma_{i,j}$	execution requirements of the j th segment of virtual machine vm_i.
P_c^{sta}	static power of processor c.
P_c^{dmax}	max dynamic power power of processor c.
r_c	frequency ratio of processor c.
f_c^{max}	max frequency level of processor c.
E_c^{opt}	the optimal energy consumption of processor c.
E_C^{opt}	the optimal energy consumption of processor set C.

In real-time task modeling, it is a conventional wisdom to adopt WCET (i.e., Worst case excution-time) and deadlines (i.e., upper timing bound) to characterize QoS requirements of real-time applications [153]. Fig. 3.1 shows a traditional QoS model that shed light on the relationships among CPU frequency, WCET time, and deadlines. Task τ_1 and τ_2, sharing the same deadline, have different WCET requirements measured in clock cycles.

These two tasks have different execution time under the same CPU frequency. If no DVFS energy-saving technology is deployed, server will run at the its full capacity using the max frequency level f^{max}. In this case, τ_1 and τ_2 finish at $FinTime1$ and $FinTime2$, which must be earlier than the deadline. The last sub-figure in the bottom of Fig. 3.1 depicts that DVFS scales the CPU frequency from f^{max} down to f_α. If τ_1 and τ_2 both run on this server concurrently, each task's execution time is extend. Nevertheless, it is mandatory to make finish time $FinTimeNew$ earlier than the deadline.

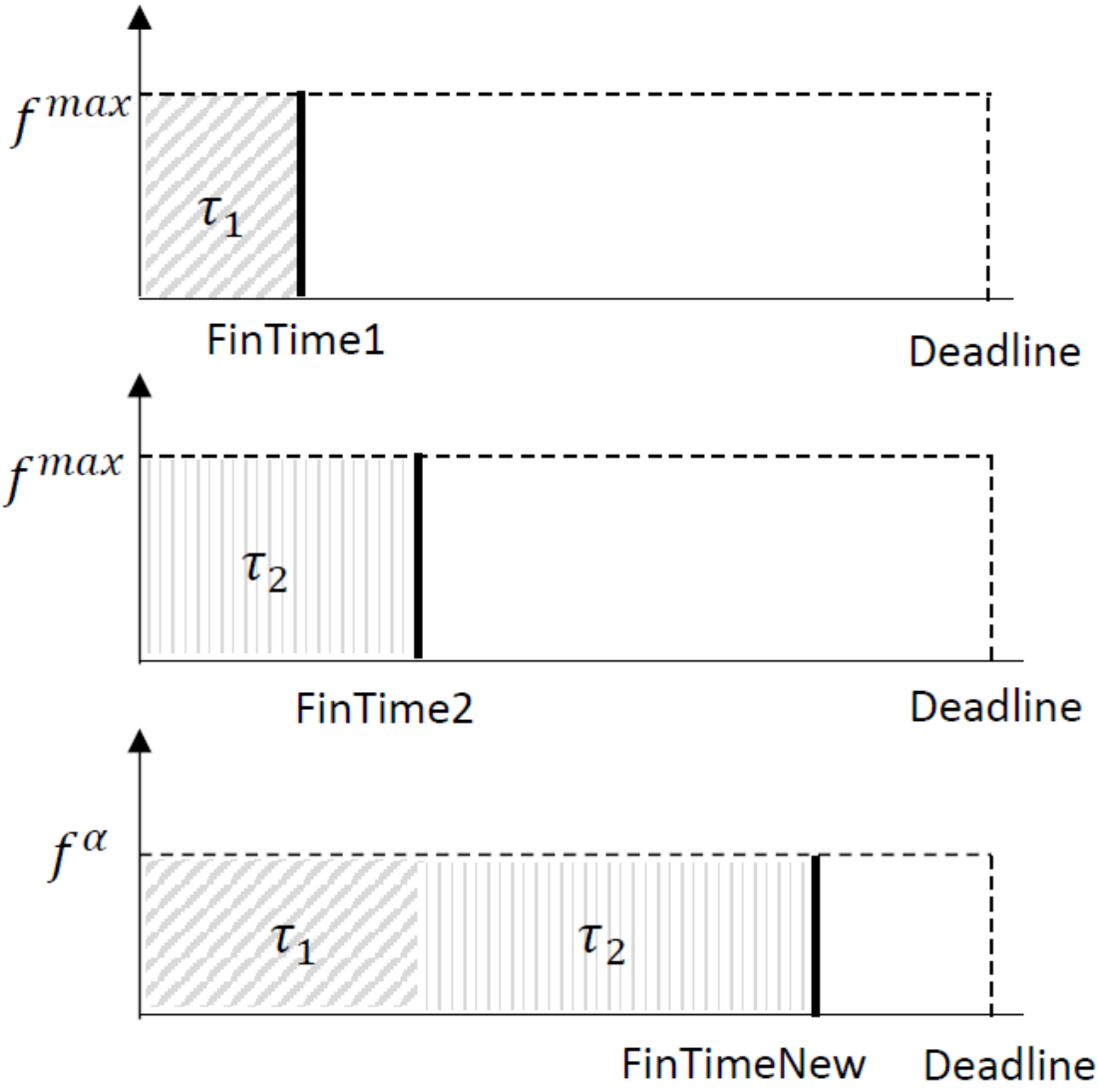


Figure 3.1: Traditional Time-aware Qos Model

3.1.2 Frequency-Aware QoS Modeling

Apart from WCET and deadlines, CPU frequency is an exceptional and practical parameter to capture the QoS requirement of real-time tasks. Giving memory and I/O resources, task execution times largely depends on processors and the elected CPU frequency level. Therefore, it is arguably feasible to transfer a time-aware requirement into a frequency-aware requirement.

Fig. 3.2 depicts two types of real-time tasks coexisting in my frequency-aware QoS modeling. For the traditional real-time tasks (see the left-hand side of Fig. 3.2), real-time tasks should entail timing constraints like deadlines and WCET values, and we transfer the time-aware requirements to the frequency-aware requirements (e.g., minimum frequency requirements). When it comes to non-time-sensitive tasks like long-running applications. (see the right-hand side of Fig. 3.2), non-time-sensitive applications merely pass on frequency requirements to the DVFS-enabled system.

In my frequency-aware QoS model, we propose to maintain a processor frequency higher than the summation of the minimum frequency requirements of all the tasks running on the processor. Fig. 3.3 shows my frequency-aware Qos model for traditional tasks, in which task τ_1 and τ_2 have the same settings as those in Fig. 3.1. Frequency f_1^{min} and f_2^{min} are the minimum frequency that averts SLA violations in τ_1 and τ_2 governed by DVFS. As the execution time of τ_1 and τ_2 extend to the max time task allowed, f_1^{min} and f_2^{min} are no doubt lower than f^{max}. When τ_1 and τ_2 both run on this processor, f_{sum}^{min} is the new minimum frequency requirement that is higher than f_1^{min} and f_2^{min}. In this scenario, the most appropriate energy-saving frequency f^{α} is higher than f_{sum}^{min}. As such, we choose f_{α} as the processor frequency; Otherwise, we should use the sum of the minimum frequency requirements as the processor frequency. In a nutshell, this model allows servers to undertake real-time tasks using frequency requirements rather than traditional WCET and deadlines.

In real-world scenarios, there might be the lack of specified requirements in terms of deadline or WCET in submitted tasks. This phenomenon is specially true when it comes to

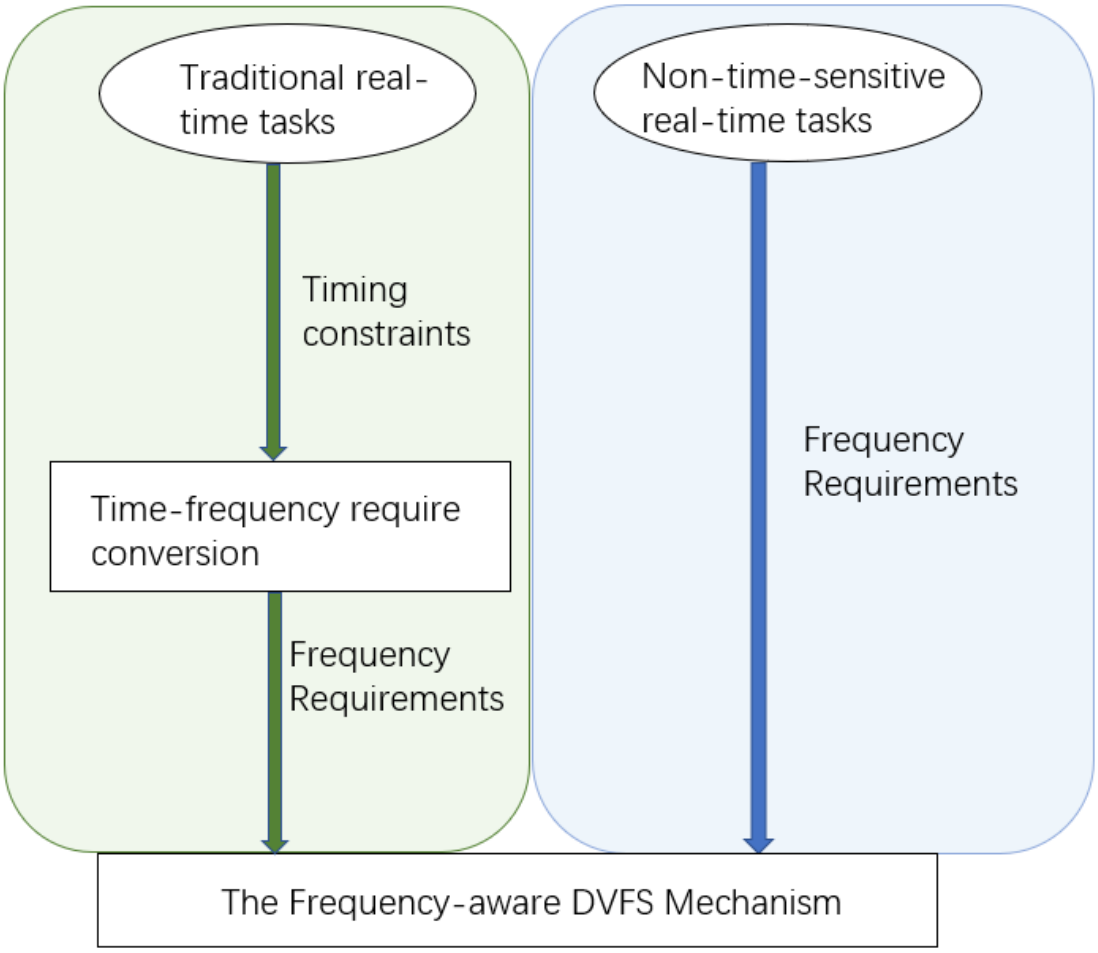


Figure 3.2: Frequency-aware modeling for time-sensitive and non-time-sensitive real-time tasks running in DVFS enabled systems.

long-running tasks like crawlers and web services. For such non-time-sensitive tasks, a system can render QoS by satisfying their minimum frequency requirements rather than WCET or deadlines (see Fig. 3.2). After taking into account minimum frequency requirements, my model facilitates the comparison between the sum of tasks' minimum frequency requirements running on the same processor and the best energy-saving frequency of that CPU. Then, we specify a higher one to avoid potential SLA violations.

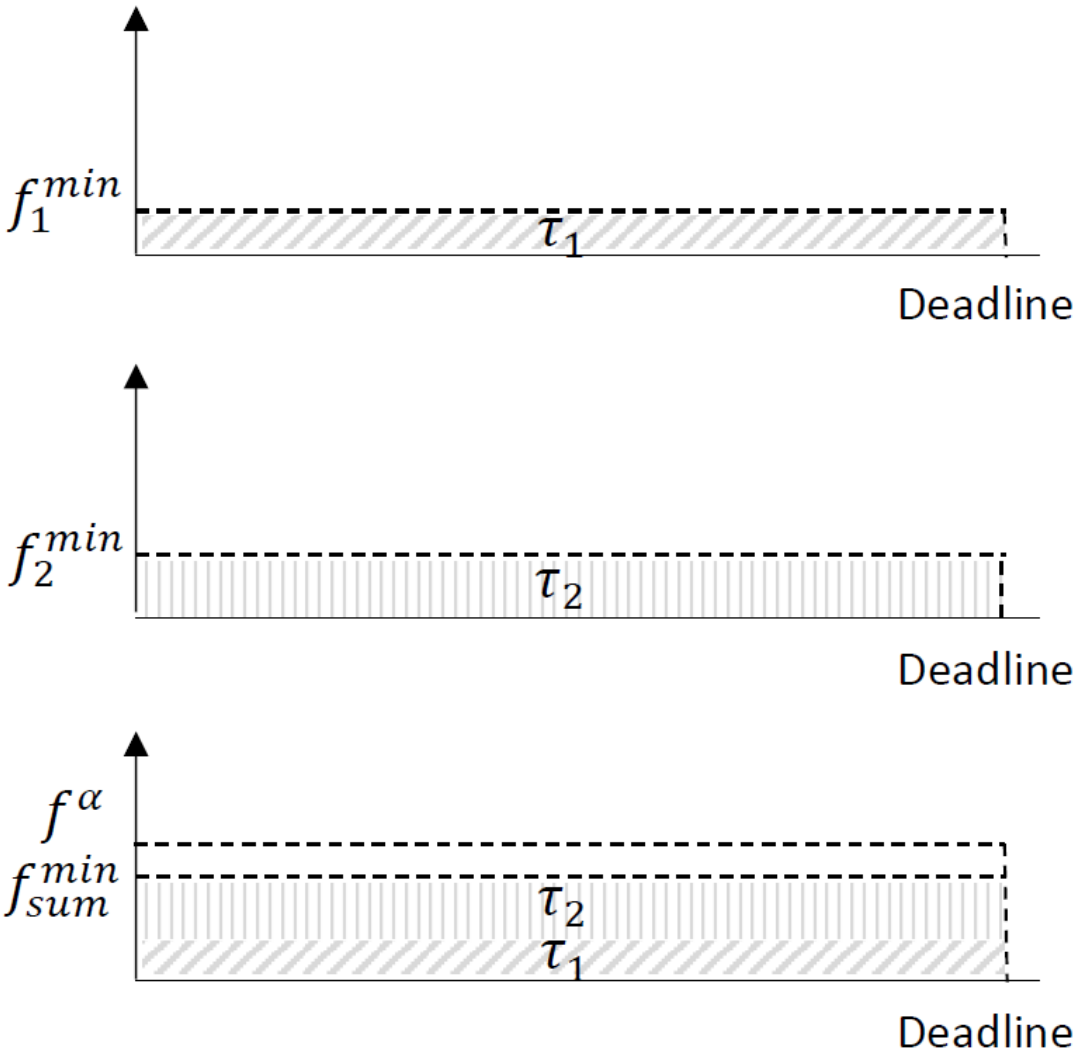


Figure 3.3: Our frequency-aware Qos model for traditional real-time tasks

We consider multiple independent aperiodic virtual machines running on a group of processors $C = \{c_1, c_2, ..., c_m\}$. For simplicity without the loss of generality, we investigate a

set of n independent aperiodic virtual machines $V = \{vm_1, vm_2, ..., vm_n\}$ running on process c (i.e., $c \in C$). Each virtual machine is represented by a pair, namely, $vm_i = (a_i, f_i^{req})$, where a_i is the arrival time of virtual machine vm_i, f_i^{req} is the minimum frequency requirement of virtual machine vm_i. As f_{sum}^{min} is the summation of frequency requirements f_1^{min} and f_2^{min} in Fig. 3.3, the relation between overall frequency requirement and each virtual machine frequency requirement is:

$$f_{V,c}^{req} = \sum_{i \in V} f_{i,c}^{req}. \tag{3.1}$$

If $f_{V,c}^{conf} \geq f_{V,c}^{req}$ holds, then processor c is able to perform tasks without SLA violations. Therefore, after we obtain the summation of minimum frequency requirement on processor c, we compare the summation with the energy-saving frequency f_c^{opt} derived from DVFS algorithm (see Section 3.3). If the overall frequency requirement $f_{V,c}^{req}$ is lower than f_c^{opt}, we should elect the derived optimal energy-saving frequency. If $f_{V,c}^{req}$ is higher or equal to f_c^{opt}, we should guarantee the overall minimum frequency requirement in order to satisfy the Qos requirement of c performing virtual machine set V.

$$f_{V,c}^{conf} = \max(f_c^{opt}, f_{V,c}^{conf}). \tag{3.2}$$

3.2 Frequency Ratio Modeling

Nowadays, modern processors are powered by the CMOS technology. The energy consumption of a CMOS circuit is composed of dynamic energy consumption E^{dyn} and static energy consumption E^{sta} [74]:

$$E = E^{sta} + E^{dyn}. \tag{3.3}$$

Prior studies [173][160][51] confirmed that dynamic power P^{dyn} is computed as

$$P^{dyn} = C_l \cdot A \cdot V_{dd}^2 \cdot f, \tag{3.4}$$

where C_l is load capacitance, A is the percentage of active gates, V_{dd} is supply voltage, and f is processor frequency.

The voltages of CMOS processors affect processor frequencies. For example, decreasing voltages would enlarge circuit delay, changing the clock frequency [24]. The correlation between frequency and voltage is expressed as:

$$f = k \cdot \frac{(V_{dd} - V_{th})^2}{V_{dd}}, \tag{3.5}$$

where V_{th} - typically a small value - is a threshold voltage [25][60]. Approximately, there is a linear relation between frequency and voltage (i.e., $f \approx k \cdot V_{dd}$). Because Cl, A, k are constants, we consolidate these three constants as $\beta = C_l \cdot A/k$. Thus, the CPU dynamic power consumption is rewritten as

$$P^{dyn} = \beta \cdot f^3. \tag{3.6}$$

We use max dynamic power P^{dmax} to represent the dynamic power when the processor frequency is set to a maximum level f^{max}. Thus, we derive (3.7) from (3.6) as

$$P^{dyn} = P^{dmax} \cdot \frac{f^3}{(f^{dmax})^3}. \tag{3.7}$$

Combined with CPU static power P^{sta} - a constant irrelevant to the voltage and frequency, the total CPU power is represented in the format of frequency as

$$P = P^{sta} + P^{dmax} \cdot \frac{f^3}{(f^{max})^3}. \tag{3.8}$$

It should be noted that when using DVFS, not all frequencies ranging from 0 to f^{max} are available due to discrete frequency levels like $10\%f^{max}$, $20\%f^{max}$,$40\%f^{max}$, $100\%f^{max}$ [150]. We introduce *frequency ratio* r as a ratio between the current processor frequency f and the maximum frequency f^{max} held by a processor. Thus, we have $r = f/f^{max}$. The total CPU

power can obtained from frequency ratio r as

$$P = P^{sta} + P^{dmax} \cdot r^3. \tag{3.9}$$

3.3 Modeling Frequency-aware DVFS

Now we are in a position to present a DVFS model tailored for tasks issuing frequency requirements (see Fig. 3.2). If virtual machine vm_i is executed at frequency f_i in time t_i, Γ_i is the total number of clock cycles of vm_i. The energy consumption of virtual machine vm_i is derived from clock cycles Γ_i and frequency f_i as

$$E_i = P_i \cdot t_i = P_i \cdot \frac{\Gamma_i}{f_i}. \tag{3.10}$$

Power consumption is an instantaneous value depending on frequency in per time unit. We assume that virtual machine vm_i's execution consists of K_i segments, where the clock cycles of the jth segment ($1 \leq j \leq K_i$) is $\gamma_{i,j}$. Thus, the clock cycles of all the segments of virtual machine vm_i form a frequency set $\{\gamma_{i,1}, \gamma_{i,2}, ..., \gamma_{i,K_i}\}$, which determines the total execution requirements Γ_i of virtual machine vm_i. Thus, we have

$$\Gamma_i = \sum_{j=1}^{K_i} \gamma_{i,j}. \tag{3.11}$$

Let $f_{i,j}$ be the execution frequency for the jth execution segment of virtual machine vm_i. Energy consumption of the virtual machine can be obtained from the power, clock cycles, and frequency of each segment as

$$E_i = \sum_{j=1}^{k} P_{i,j} \frac{\gamma_{i,j}}{f_{i,j}}. \tag{3.12}$$

The *Lagrange* multiplier method is applied to minimize energy E_i [84]. In case where frequencies of vm_i during all its intervals are identical, the computation of (3.12) will be

simplified. Furthermore, the common execution frequency f_i is equal to the overall execution requirement of vm_i divided by the overall execution time of vm_i. Thus, we have

$$f_i = f_{i,1} = f_{i,2} = f_{i,3} = ... = f_{i,K}. \tag{3.13}$$

Hence, the optimal frequency of each segment is equal to the optimal solution for the entire virtual machine vm_i. Combining with the energy formula (see (3.10)), we derive energy consumption of processor c as

$$\begin{aligned} E_c &= \frac{\sum_{i \in T} \Gamma_i}{f_c}(P_c^{sta} \ + \ P_c^{dmax}(r_c)^3). \\ E_c &= \frac{\sum_{i \in T} \Gamma_i}{f_c^{max}}(\frac{P_c^{sta}}{r_c} \ + \ P_c^{dmax}(r_c)^2). \end{aligned} \tag{3.14}$$

(4.4) is a quadratic function with respect to frequency ratio r_c. To obtain the extreme value of the energy consumption, we take the derivative of E_c with respect to r_c

$$E_c' = \frac{\sum_{i \in T} \Gamma_i}{f_c^{max}} \ \cdot \ \frac{1}{(r_c)^2}(2 \ \cdot \ (r_c)^3 \ \cdot \ P_c^{dmax} \ - \ P_c^{sta}). \tag{3.15}$$

To investigate the trend of this quadratic function parabola, let us take the second derivative of E_c with respect to r_c

$$E_c'' = \frac{2 \ \cdot \ \sum_{i \in T} \Gamma_i}{f_c^{max}} \ \cdot \ (\frac{P_c^{sta}}{(r_c)^3} \ + \ P_c^{dmax}). \tag{3.16}$$

Since we have $E_c'' > 0$, it is verified that E_c is a convex function of r_c and the parabola opens upward. From the features of parabola, by setting the first derivation of E_c' to zero, we can obtain the most energy-efficient consumption E_c^{opt} at the optimal frequency ratio f_c^{opt}. The optimal frequency ratio r_c^{opt} is derived from P_c^{sta} and P_c^{dmax} as

$$r_c^{opt} = \sqrt[3]{\frac{P_c^{sta}}{2P_c^{dmax}}}. \tag{3.17}$$

Noticing that f is discrete, we choose the nearest r value to meet deadline requirements. The optimal energy consumption of processor c is E_c^{opt}, which can be calculated as follows.

$$E_c^{opt} = \frac{\sum_{i \in T} \Gamma_i}{f_c^{max}} (\frac{P_c^{sta}}{r_c^{opt}} + P_c^{dmax} \cdot (r_c^{opt})^2). \tag{3.18}$$

After optimizing energy consumption E_c^{opt} for a single processor (i.e., c), we derive optimal energy consumption E_C^{opt} for computing systems equipped with multiple processors (i.e., $C = \{c_1, ..., c_m\}$) in a datacenter. More specifically, energy E_C^{opt} is an accumulative value of the optimal energy consumption of all the processors in set C. Thus, the system level energy consumption can be expressed as:

$$E_C^{opt} = \sum_{c \in C} E_c^{opt}. \tag{3.19}$$

where E_c^{opt} on the right-hand side of the expression is obtained from (4.6).

It is noteworthy that in frequency-aware DVFS model, virtual machine energy consumption is independent of virtual machine execution time. Rather, the energy consumption is reliance to virtual machines' execution requirements and server hardware parameters (e.g., static power and max dynamic power.)

When the current frequency ratio is configured to r^{opt}, the energy consumption will be minimized to E^{opt}. There is a challenging roadblock to setup r to value r^{opt} due to the discrete value of processor frequency in the real world. If the current frequency ratio is less than r^{opt}, the energy consumption will be proportional to the frequency ratio. On the contrary, if the current frequency ratio is greater than r^{opt}, the energy consumption will be inversely proportional to the frequency ratio.

The optimal execution time of processor c is t_c^{opt} governed by the optimal frequency ratio r_c^{opt}. Hence, we express time t_c^{opt} as a function of frequency ratio r_c^{opt} in (3.20).

$$t_c^{opt} = \frac{\sum_{i \in T} \Gamma_i}{f_c^{max} \cdot r_c^{opt}}. \tag{3.20}$$

3.4 Analysis of Frequency Ratios

3.4.1 Energy Consumption and Frequency Ratio

Recall that in proposed frequency-aware DVFS model (see Section 3.3), optimal energy consumption derived from Eq. (4.6) is dependent of frequency ratio r. In this section, we place the spotlight on the trend and features of the optimization formula by applying the DVFS model to various theoretical cases. Such an analysis validates the versatility of my new model with frequency awareness.

Eq. (4.4) indicates the correlation between energy consumption and frequency ratio r. Fig. 3.4 depicts the normalized energy consumption E as a function of frequency ratio r of processors. This analysis is focused on a single processor. Nevertheless, the findings can be readily applied to scenarios where a computer system is orchestrating multiple processors.

In order to make the trends plotted in Fig. 3.4 broadly representative, we introduce a parameter called *static power proportion* in lieu of a server's power data. Eq. (4.4) suggests that the most vital parameters in the DVFS model is server static power P^{sta} and max dynamic power P^{dmax} - two constants. We simplify these two parameters by defining static power proportion λ to gauge static power P^{sta} as a percentage of the total power consumption in max frequency level. Thus, we have $\lambda = P^{sta} \ / \ (P^{dmax} \ + \ P^{sta})$. More often than not, a server's static power proportion λ is anywhere between 15% and 35 %. We categorize all types servers in terms of λ values, thereby allowing us to capture the energy consumption features of a wide range of general cases.

Fig. 3.4 shows five parabolas of the function expressed in Eq. (4.4) in a variety of cases. The red line represents a server's energy consumption when the server keeps running on at

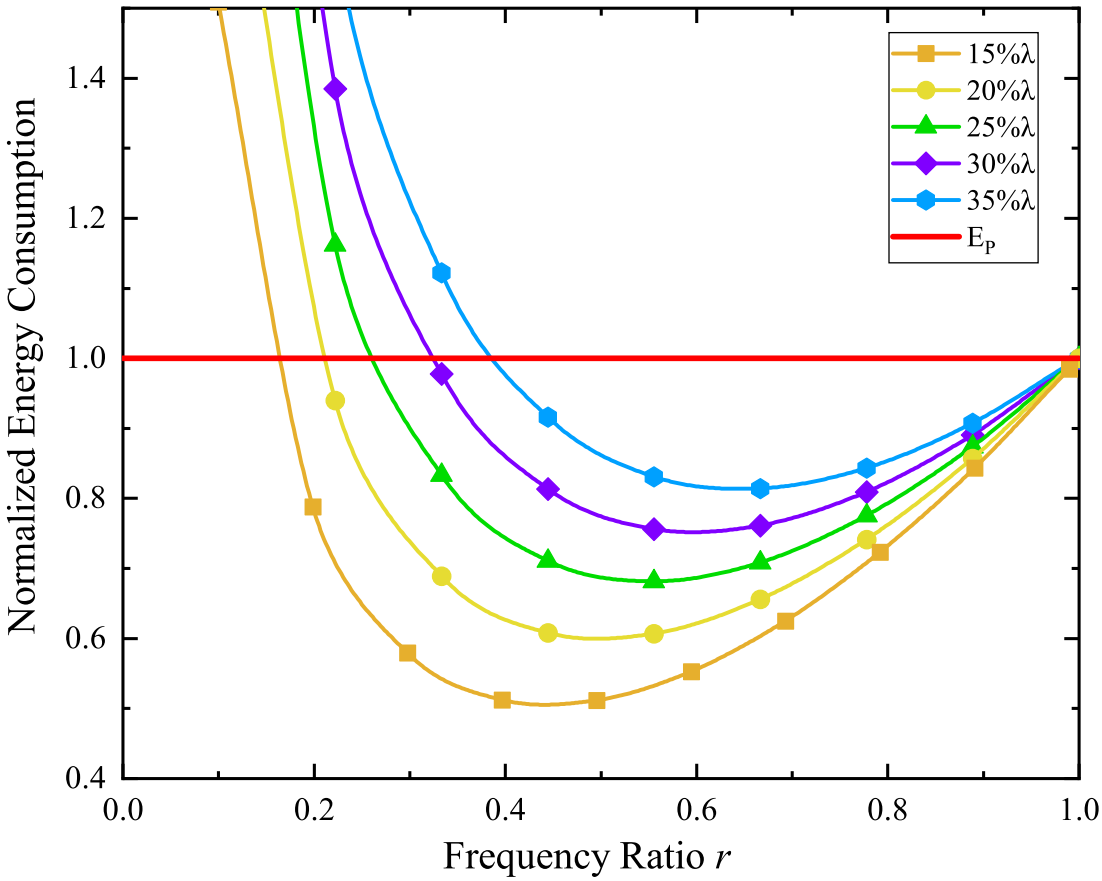


Figure 3.4: Impacts of frequency ratio on normalized energy consumption under various value.

the peak power with the maximal frequency level all the time (i.e., frequency ratio $r = 1$). We refer to the energy consumption plotted by the red line as peak energy consumption or E_p; in this scenario, no DVFS is involved in conserving energy. For comparison purpose, we normalize all the energy consumption derived from the model, within which E_p turns to be 1 - an ideal baseline. The five curves in Fig. 3.4 are the normalized energy consumption when the static power proportion λ is configured to 15%, 20%, 25%, 30%, and 35%, respectively. Fig. 3.4 depicts the energy consumption outcomes of the five λ values that cover a majority of real-world server cases.

The five parabolas in Fig. 3.4 share a similar pattern with respect to energy consumption values under a given frequency ratio. All the parabolas have a raft of points lower than baseline E_p, meaning that the servers embrace ample opportunities to conserve energy. The findings unveil that static power proportion λ is proportional to the energy consumption. The results confirm that a server with a low static power proportion λ has an energy-efficiency edge over its counterparts with high λ values.

3.4.2 Energy-saving Windows

Now we introduce energy-saving windows, within which frequency ratios lead to energy savings using DVFS. In other words, frequency ratios outside an energy-saving window are unable to take leverage DVFS to conserve energy. We refer to a frequency ratio within an energy-saving window as *energy-saving frequency ratio*.

Fig. 3.5 shows an example of energy-saving windows and energy-saving frequency ratios when static power proportion is set to 20%. Recall that when frequency ratio r equals to 1 (i.e., $r = 1$, $f = 100\% f^{max}$), energy consumption spikes at the value of E_p, which is expressed as

$$E_p = \frac{\sum_{i \in T} \Gamma_i}{f^{max}} \cdot (P^{sta} + P^{dmax}). \tag{3.21}$$

Fig. 3.5 suggests that the curve obtained from my model may go beyond the red line (i.e., from 0 to fr_1), meaning that there exist cases where DVFS fails in offering energy savings. This phenomenon is expected, because the DVFS technology reduces frequency and voltage to save dynamic energy at the cost of increasing static power consumption. When frequency ratios are set to a value that is outside the energy-saving window, a large amount of energy is consumed during excessively enlarged execution times. On the flip side, the curve below the red line in Fig. 3.5 represents energy consumption when frequency ratios are in the range of the energy-saving window (i.e., r_1 to 1).

Fig. 3.5 intuitively shows that the upper bound of an energy-saving window is 1 and the lower bound is r_1. Let us represent energy consumption obtained from my model as E_{DVFS}

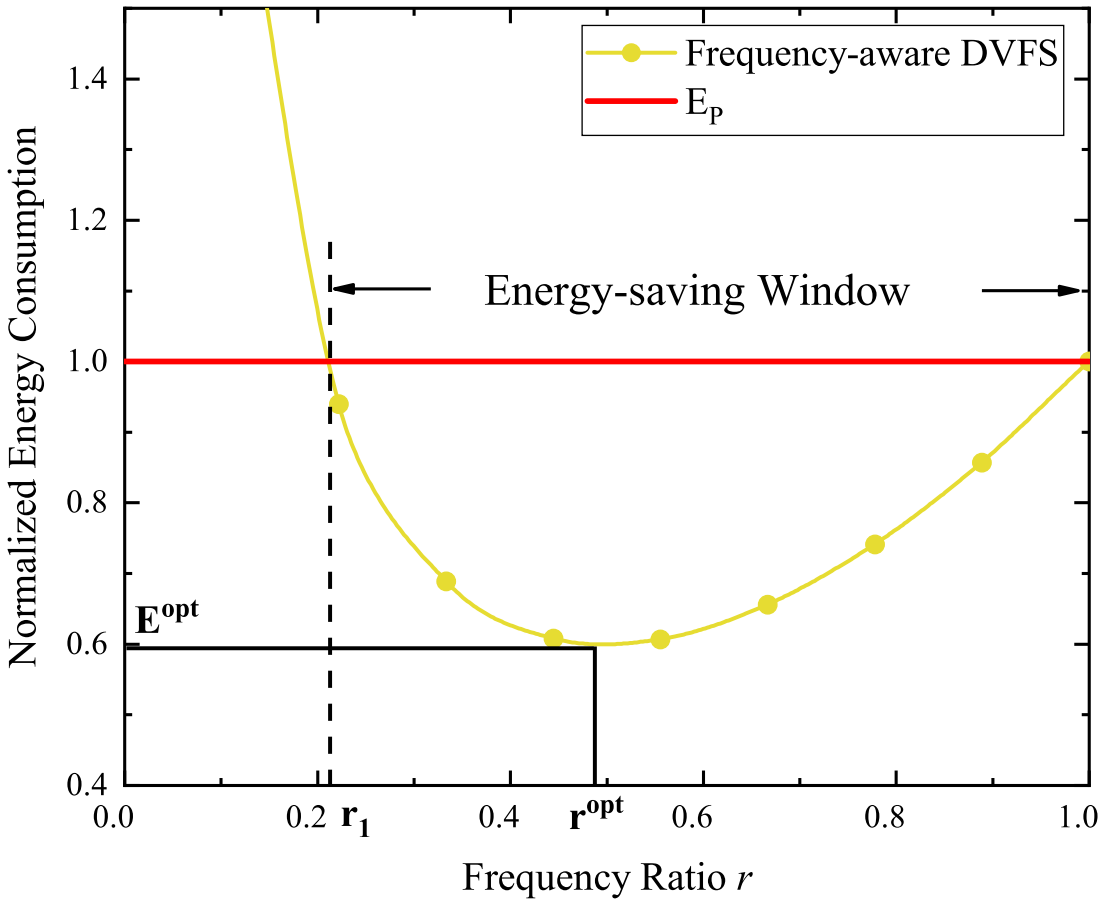


Figure 3.5: An example of energy-saving windows and optimal energy-saving frequent ratio.

rather than E_c for processor c. Because peak energy consumption E_p (see (3.21)) is larger than E_{DVFS} inside the energy-saving window, we express the correlation between E_{DVFS} and E_p in (3.22) by incorporating (4.4). In light of (3.22) we show that frequency ratio r is greater than $\frac{\sqrt{1+4P^{sta}\ /\ P^{dmax}}-1}{2}$, signifying the lower bound of an energy-saving window (see also r_1 in Fig. 3.6 and Fig. 3.7).

$$E_p\ -\ E_{DVFS} = \frac{\sum_{i\in T}\Gamma_i}{f^{max}}(P^{sta}\ +\ P^{dmax}) - \frac{\sum_{i\in T}\Gamma_i}{f^{max}}(\frac{P^{sta}}{r} + P^{dmax}(r)^2) > 0$$

$$r > \frac{\sqrt{1\ +\ 4P^{sta}\ /\ P^{dmax}}\ -\ 1}{2} \tag{3.22}$$

Given the upper and lower bounds of an energy-saving window, we argue that frequent ratio r residing within the energy-saving window offers energy savings. In other words, energy can be conserved by DVFS if we have $r \in \left(\frac{\sqrt{1+\frac{4P^{sta}}{P^{dmax}}}-1}{2}, 1\right)$.

(3.22) suggests that an energy-saving window is dependent of server static power P^{sta} and maximum dynamic power P^{dmax}. To further explore intriguing trends of energy-saving windows, we plot in Fig. 3.6 and Fig. 3.7 energy window size as functions of static power and maximum dynamic power. We shine a bright light on energy-saving window size, because a large window size implies there are ample opportunities to conserve energy using DVFS.

Fig. 3.6 shows the energy-saving window size of servers when the maximum dynamic power settings are 80W, 120W, 160W, respectively. Similarly, Fig. 3.7 plots the energy-saving window size as functions of maximum dynamic power when static power is configured to 30W, 50W, 70W, respectively. We observe from Fig. 3.6 that regardless of the maximum dynamic power, increasing static power continuously curtails the energy-saving window size. On the contrary, Fig. 3.7 reveals that energy-saving window size goes up with an rising maximum dynamic power. Interestingly, energy-saving window size is quite sensitive to maximum dynamic power when the dynamic value is below 120W; when maximum dynamic power is sitting at a high level (e.g., 200W), further pushing up the maximum dynamic power

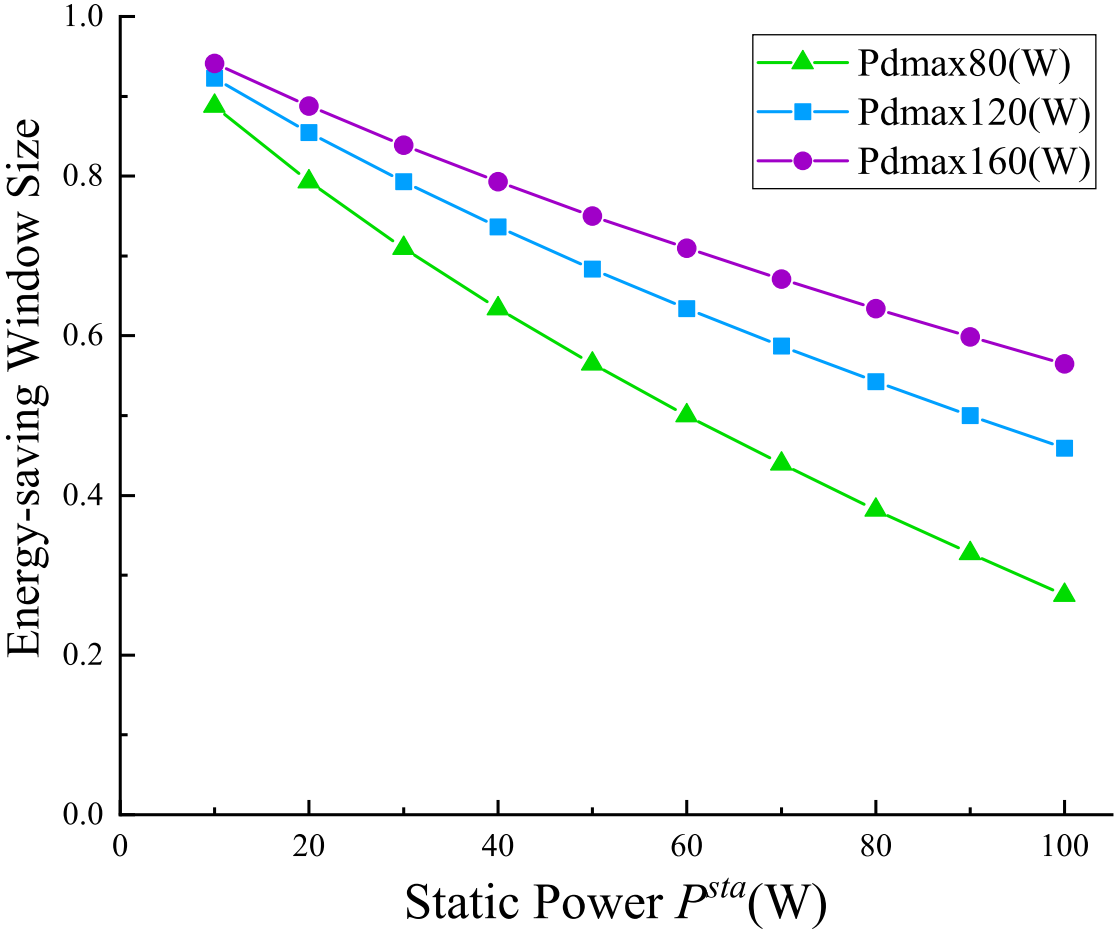


Figure 3.6: Static Power vs. Energy-Saving Window Size

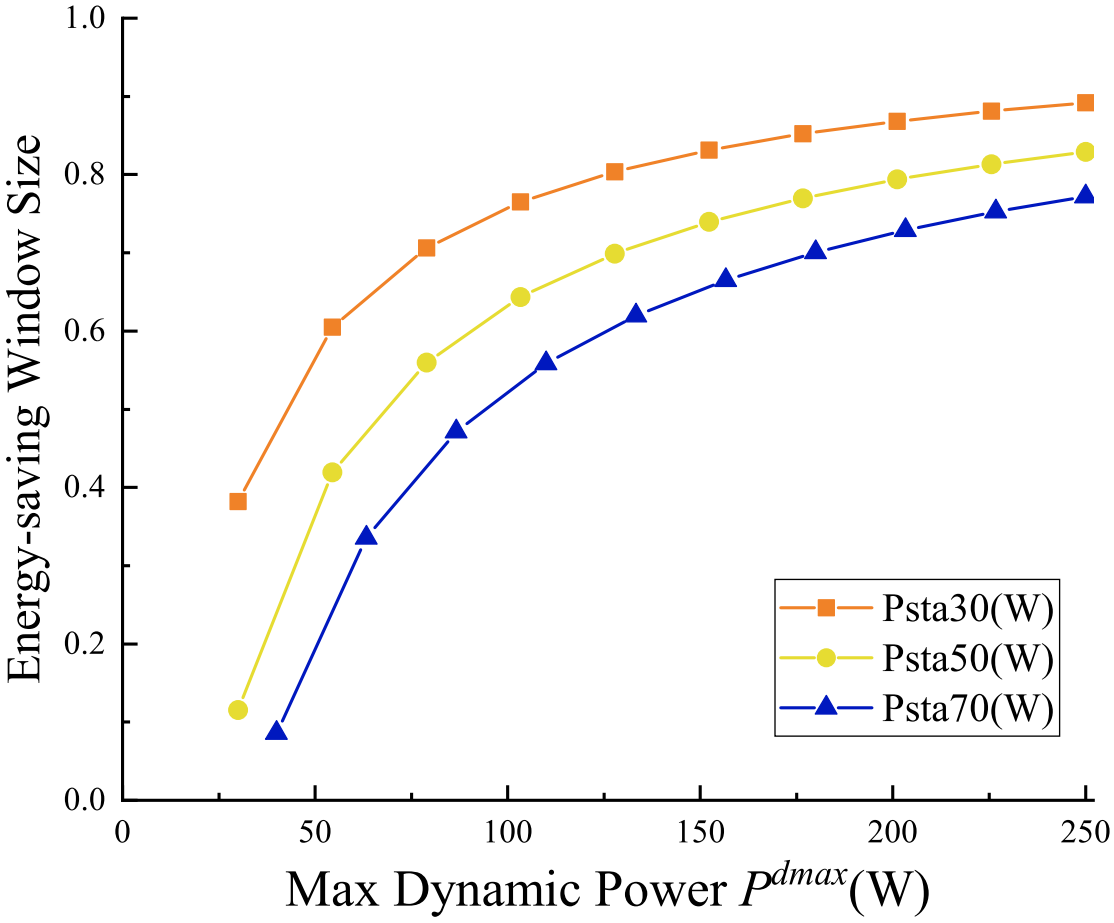


Figure 3.7: Maximum Dynamic Power vs. Energy-Saving Window Size

leads to a marginal growth in energy-saving window size. In summary, the energy-saving window size is proportional to maximum dynamic power and inversely proportional to static power.

3.4.3 Optimal Energy-Saving Frequency Ratio

Fig. 3.5 unravels normalized energy consumption as a function of frequency ratio r. We mark the energy-saving window (i.e., from r_1 to 1) in Fig. 3.5, where an optimal frequency ratio (see r^{opt}) is the lowest point of the energy-consumption value obtained from my frequency-aware DVFS model. It is noteworthy that frequency ratio r^{opt} minimizes energy consumption denoted as E^{opt}. The discovery of the r^{opt} value is the centerpiece of my algorithm, because energy consumption can be minimized by adjusting frequency ratio to reach r^{opt}.

Eq. (4.5) suggests that the optimal frequency ratio is affected by static power and maximum dynamic power. As such, we plot optimal frequency ratio as functions of static power and maximum dynamic power in Fig. 3.8 and Fig. 3.9 .

Similar to Fig. 3.6 and Fig. 3.7, Fig. 3.8 and Fig. 3.9 plots the optimal frequency ratio as the functions of static power and maximum dynamic power, respectively. Comparing the five parabolas in Fig. 3.4, we conclude that E^{opt} climbs with an increasing value of r^{opt}. Moreover, a high E^{opt} value leads to low energy-saving performance. Fig. 3.8 reveals that the optimal energy-saving frequency ratio is proportional to the static power, meaning that a large static power gives rise to a small amount of saved energy. On the flip side, Fig. 3.9 shows that the optimal frequency ratio is inversely proportional to the maximum dynamic power when the static power is a constant.

3.4.4 Static Power Proportion

Recall that energy-saving windows and optimal energy-saving frequency ratios are dependent of static power and maximum dynamic power. To simplify the presentations of

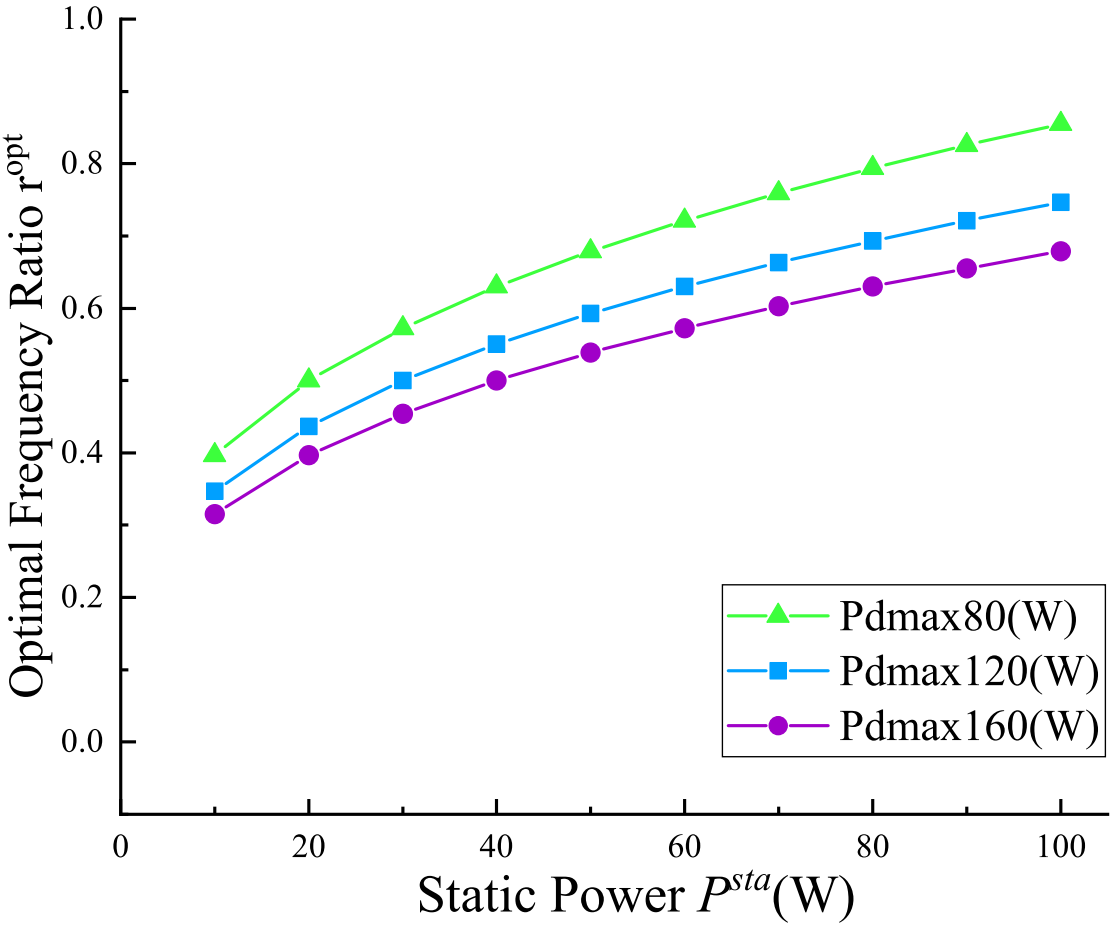


Figure 3.8: Static Power vs. Optimal Frequency Ratio

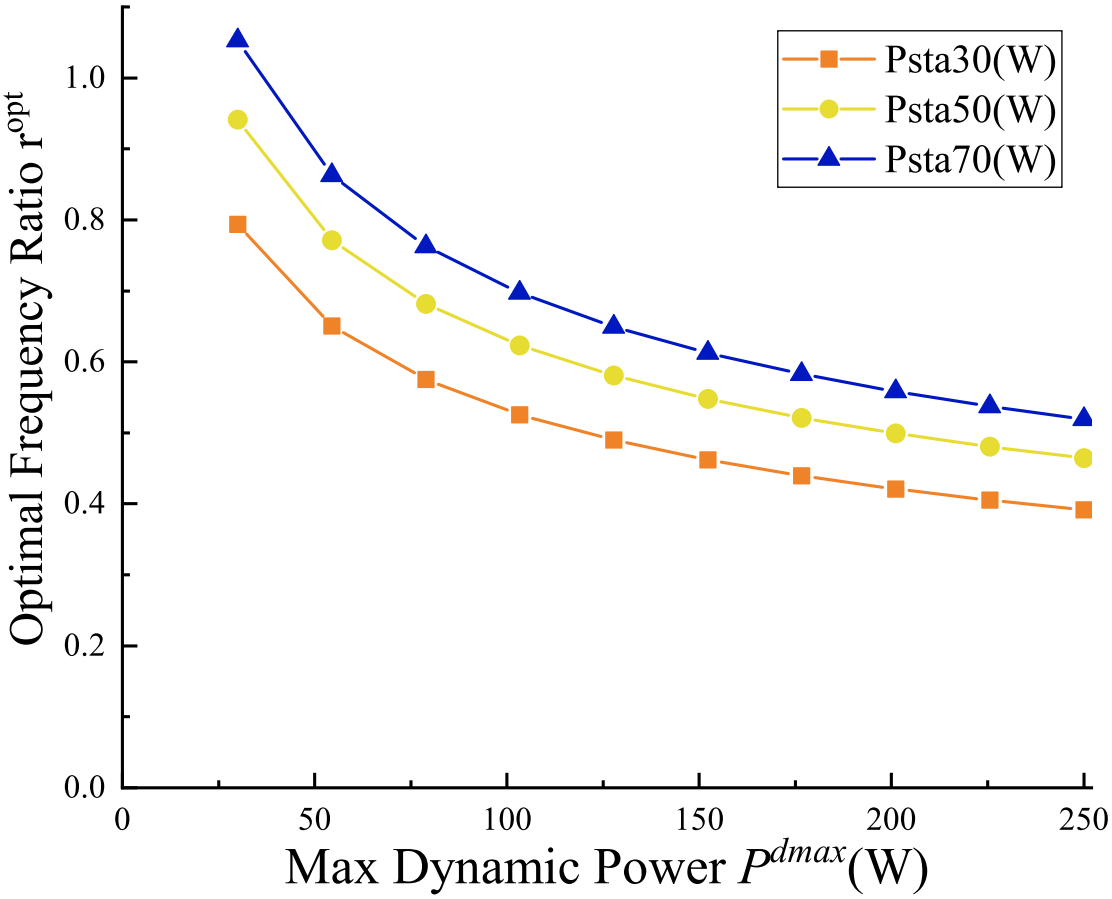


Figure 3.9: Maximum Dynamic Power vs. Optimal Frequency Ratio

the results plotted in Fig. 3.6, Fig. 3.7, Fig. 3.8 and Fig. 3.9 , we adopt a concept of static power proportion (i.e., λ) (see the second paragraph in Section 3.4.1). Fig. 3.10 intuitively illustrates the impacts of static power proportion λ on energy-saving windows and optimal frequency ratios when the static power proportion varies from 10% to 50%, which resembles the static power proportions of most modern servers.

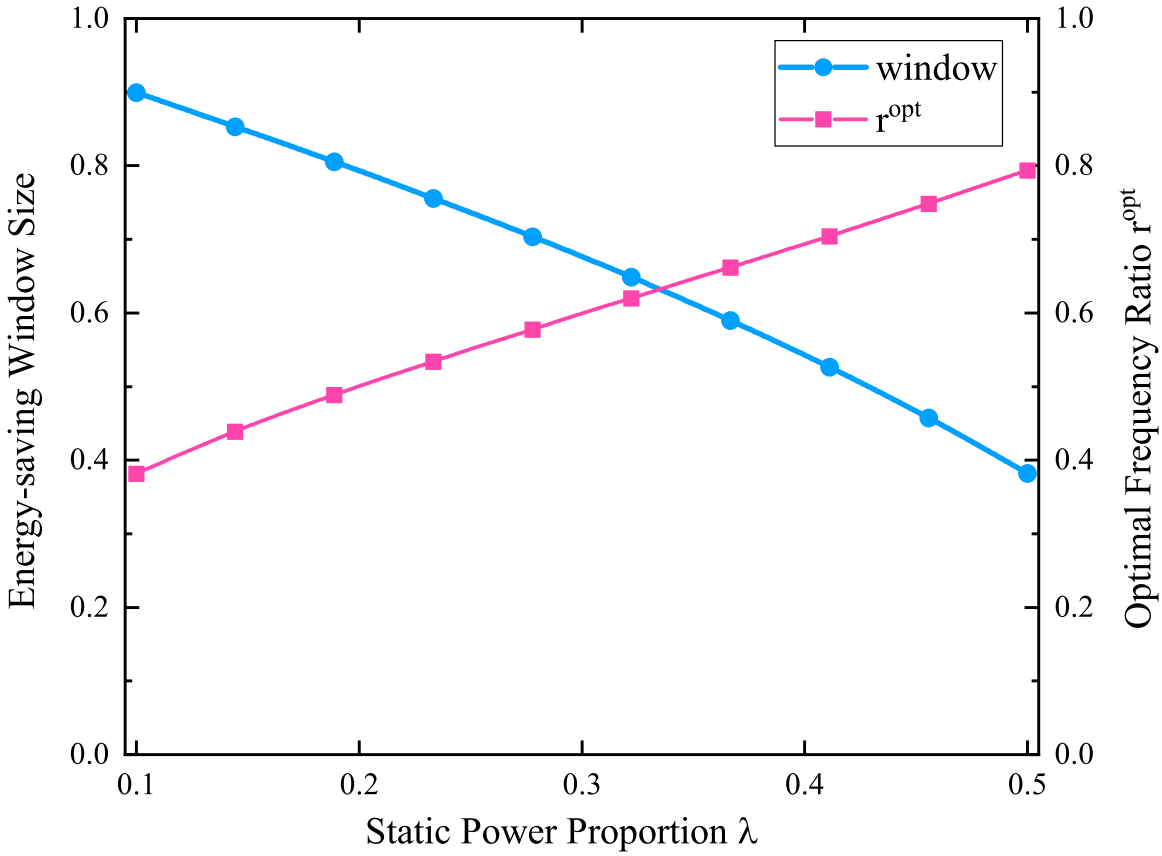


Figure 3.10: Impacts of static power proportion λ on energy-saving windows and optimal frequency ratios.

Fig. 3.10 unravels that a small static power proportion of a server leads to a large energy-saving window size and a low optimal energy-saving frequency ratio. From the findings discussed in Section 3.4.3, we observe that a low r^{opt} value implies good energy-saving performance. On the other hand, a large energy-saving window size means there are ample

opportunities to conserve energy using DVFS. In contrast, when the static power proportion λ grows, the energy-saving window size is narrowed coupled with an increasing r^{opt} value. We conclude that servers with large static power proportions tend to deliver deteriorated energy-saving performance.

In summary, my energy-saving model is evident that servers that have small λ parameters (a.k.a., static power proportion) are capable of taking advantages from the DVFS technique to offer high energy efficiency. This trend is reasonable, because a small static power proportion provide a wide space to tune frequencies to harvest energy savings. On the other hand, DVFS is not an ideal technique to conserve energy for servers where the λ parameters are high.

3.4.5 Power Consumption Compared with Baseline

We compare the energy consumption of servers equipped with and without my frequency-aware DVFS technique. The energy efficiency of the non-DVFS enabled system is represented by peak energy consumption E_p, which is a baseline measure offering no energy savings. In this group of experiments, the CPU static power is 100 W; the workload is configured to 10,000 clock cycles. Fig. 3.11 shows the energy savings obtained by the frequency-aware DVFS scheme under various static power proportion values. The findings confirm that energy savings offered by frequency-aware DVFS becomes pronounced under low static power proportion; the energy-efficiency strength of DVFS is diminishing when the static power proportion grows.

In theory, the shadow area is the the max energy I could save after applied frequency-aware DVFS algorithm, achieving average energy saving 20.7%.

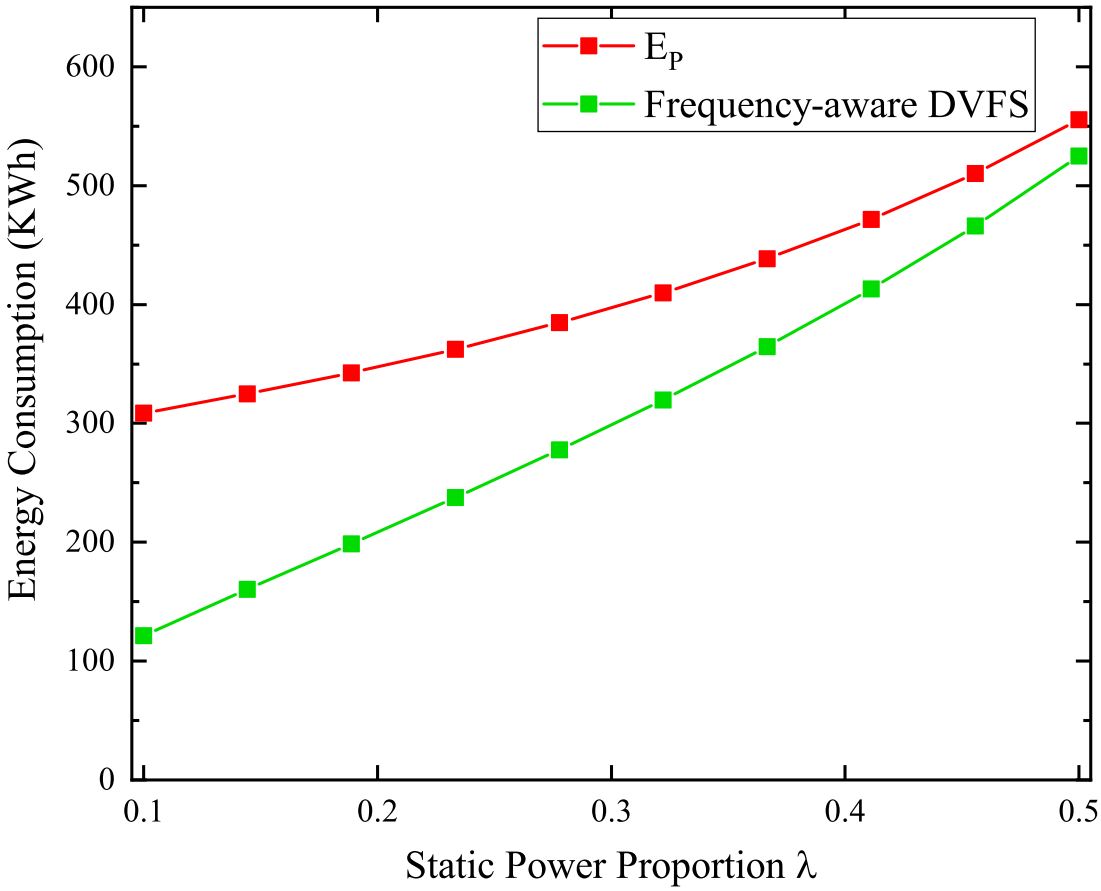


Figure 3.11: An energy consumption comparison between servers equipped with and without the frequency-aware DVFS technique. E_p is the peak energy consumption of the non-DVFS enabled system. Static power proportion λ is configured in a range between 0.1 and 0.5.

3.5 Results and Discussions

I conduct simulations to demonstrate the usage of the frequency-aware DVFS model elaborated in Section 3.3. More specifically, I simulate the energy consumption behaviors of three servers governed by the power management policies.

3.5.1 Experimental Setup

Table 3.2: The CPU configurations of the five tested servers.

Server	**Frequency (GHz)**	P^{sta} **(W)**	P^{dmax} **(W)**
Xeon E5-2670	{1.2,1.3,...,2.5,2.6}	84.1	342.9
Intel i7-4770	{0.8,1.0,...,3.2,3.4}	19	76
AMD Athlon	{0.8,1.5,2.0,2.7,3.0}	53	121.1
Intel Pentium 950	{1.0,1.5,2.2,2.9,3.4}	68	126
Intel Pentium 4630	{1.0,1.2,1.5,2.4,3.0}	30	64

As a numerical study, I implement the model to test five popular CPUs, namely, Xeon E5-2670 [12][117], Intel i7-4770 [111][131], AMD Athlon, Intel Pentium 950, and Intel Pentium 4630 [86]. I make use of the static power p^{sta}, maximum dynamic power p^{dmax}, and the frequency levels of the five real-world processors to demonstrate a way of applying my model to determine the most energy-efficient frequency for real-world settings. Table 3.2 summarizes the three parameters of the five CPUs.

I implement the proposed mathematical model, the input of which includes a static power and the max dynamic power. The output of the model implementation is an optimal frequency ratio accompanied by the lowest energy consumption. To illustrate the usage of my implemented model, I compare my frequency-aware DVFS with the two competitive approaches in the context of the five tested processors. After deriving the optimal frequency ratios coupled with the minimized energy consumption, I put spotlight on the energy-efficiency comparison of my frequency-aware DVFS scheme against the utilization-based DVFS scheme and a baseline scheme. Let us briefly articulate the key ideas of the three schemes below.

- **The baseline scheme.** In this scenario, the simulated servers are operated at the maximum frequency. The energy consumption of the servers managed by the baseline scheme is referred to as peak energy consumption E_p (see also (3.21) in Section 3.4.2).

- **The utilization-based DVFS scheme [42][15].** This popular DVFS scheme adjusts CPU frequency according to processor utilization that is dynamically changing. A server's power consumption is proportional to the server's utilization [42]. The correlation between power consumption and CPU utilization is expressed as

$$P = P_{idle} + (P_{busy} - P_{idle}) \cdot u \tag{3.23}$$

  where P_{idle} and P_{busy} are power consumption of a system when its CPU's power state is idle and busy, respectively; u is the CPU utilization. In prior studies (see, for example, [15]), the optimal utilization in the utilization-based DVFS scheme is set to 80%. Please refer to [15] for the detailed description on the utilization-based DVFS scheme.

- **The frequency-aware DVFS scheme.** My proposed energy-saving method explores frequency ratio to optimize energy efficiency. Obtaining r^{opt} from Eq. (4.5), my scheme minimize energy consumption by electing a frequency-ratio value r that is close to r^{opt}.

3.5.2 Experimental Results

Fig. 3.12 shows the results of normalized energy consumption of the AMD and Intel processors governed by three power management policies (i.e., frequency-aware DVFS, utilization-based DVFS scheme, baseline). The normalized energy consumption of the baseline approach is fixed at the value of 1.0. Therefore, the baseline's energy consumption is marked as a red line rather than multiple bars. Compared with the baseline policy, my frequency-aware DVFS conserves the energy by anywhere between 19% to 41%. The energy

savings measured by the simulator are consistent with estimates offered by the model elaborated in Section 3.4.5. The simulation results validate the correctness of the model developed in this chapter.

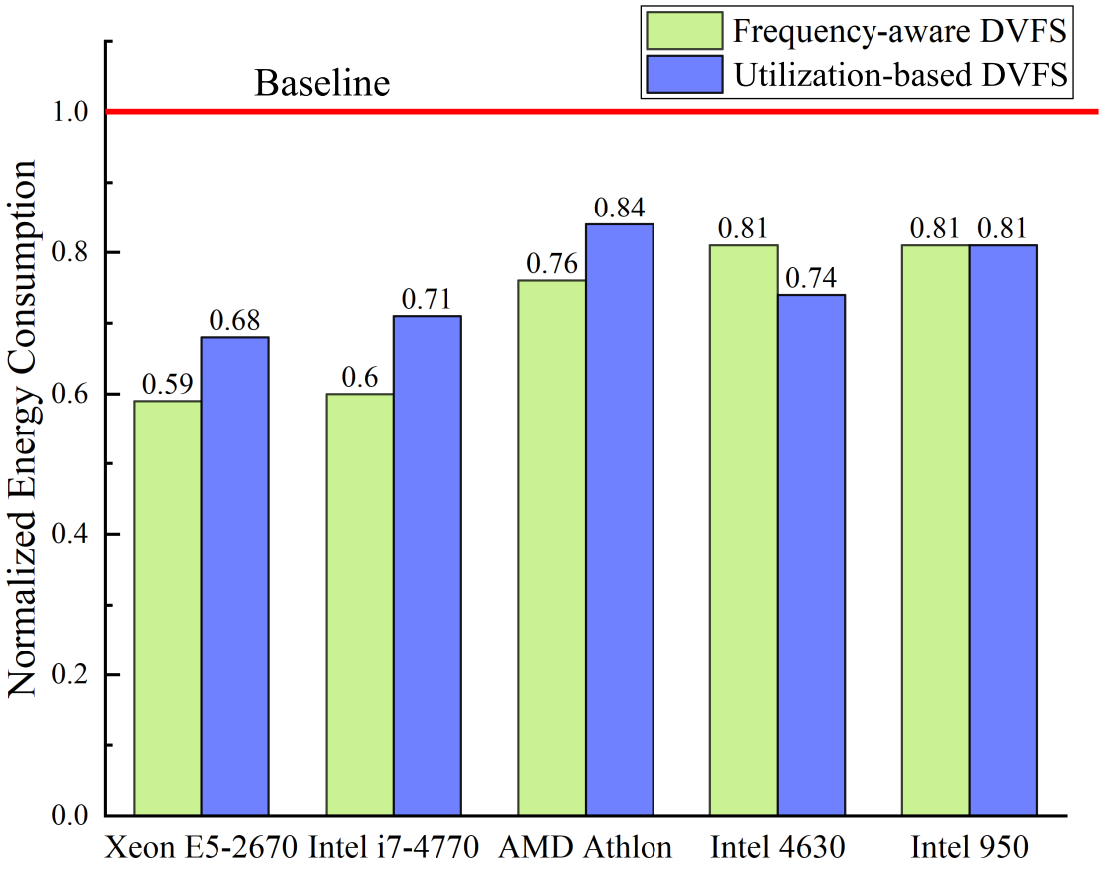


Figure 3.12: Normalized energy consumption of the AMD and Intel processors managed by the frequency-aware DVFS scheme, utilization-based DVFS scheme, and the baseline scheme.

Comparing the energy efficiency between my frequency-aware DVFS method and the utilization-based DVFS technique, I observe three intriguing and surprising trends. When it comes to the Xeon E5-2670, Intel i7-4770, and the AMD Athlon, the frequency-aware DVFS technique conserves more energy than the utilization-based DVFS counterpart. For instance, compared with utilization-based DVFS, frequency-aware DVFS slashes the energy

consumption of Intel i7-4770 by 15.5%. From the perspective of the Intel Pentium 4630 processor, the utilization-based DVFS method is superior to the frequency-aware DVFS policy. In the context of the Intel Pentium 950 processor, the frequency-aware and utilization-based DVFS schemes share identical energy efficiency. Such versatile trends motivate us to shed some light on the applicability of the frequency-aware DVFS in the next subsection.

3.5.3 Applicability Discussions

The results plotted in Fig. 3.12 suggest that frequency-aware DVFS outperforms the utilization-based DVFS scheme on three cases (i.e., Xeon, Intel i7, and AMD Athlon), whereas the opposite is true for an outlier - Intel 4630. I delve into the features of these five processors to justify the three winning cases of Xeon, Intel i7, and AMD Athlon, in which the frequency-aware DVFS scheme more energy efficient than the utilization-based alternative.

The frequency-aware scheme is not on par with the utilization-based technique in the case of Intel Pentium 4630, because this Intel processor has no appropriate frequency levels that take full advantage from for my scheme. For example, optimal frequency ratio r^{opt} of Intel Pentium 4630 is 0.617; unfortunately, the nearest ratio close to r^{opt} is just 0.5, which is far away from the optimal energy-saving frequency ratio of 0.617. When it comes to Intel Pentium 950, the frequency-based and utilization based schemes are equally energy efficient. This trend is reasonable, because, Pentium 950's static power proportion λ is over 35% - the highest one among those of the three processors. Recall that (see Section 3.4.4) a high λ value suppresses the capability of DVFS to conserve energy by adjusting frequency, adversely affecting the energy-saving performance.

The aforementioned comparisons of the policies managing the five processors demonstrate that my model is adroit at analyzing the energy efficiency of DVFS-enabled servers. The results suggest that servers powered by processors with more frequency levels have a higher likelihood to achieve the lowest energy consumption, because the frequency ratios of these processors are more likely to be closer to optimal ratio r_{opt}. Furthermore, servers with

low static power proportion can enjoy energy savings offered by my frequency-aware DVFS model.

My frequency-aware DVFS method delivers comparable energy-saving performance as the utilization-based DVFS policy. A distinctive feature of my model is that frequency ratio is introduced to measure the QoS and energy consumption. The existing utilization-based DVFS model [15] assumes that virtual machine execution times are given in a priori; more often than not, the estimates of the execution times are inaccurate. In contrast, my frequency-aware model applies DVFS to conserve energy without estimating virtual machine execution times. My model makes energy-saving decisions in accordance with server configuration data. My model is more practical than the existing utilization-based model.

3.6 Summary

In this chapter, I have developed a frequency-aware DVFS model for virtual machines running on cloud computing platforms. It is conventional wisdom to apply worst-case execution time or WCET as QoS requirements of virtual machines running on DVFS-enabled systems, which conserve energy by scaling down CPU frequency. My model advocates for specifying QoS requirements in form of frequency rather than WCET. The prominent benefit of my proposed frequency requirements is to avert inaccurate WCET estimates. I demonstrated the feasibility of transforming the traditional time-aware QoS model into a novel frequency-aware QoS model. I showed that my model can be applied to govern the power management of DVFS-enabled systems without violating the SLAs of virtual machines. In the model, I introduced a new parameter referred to as *frequency ratio*, which impacts the energy consumption of processors.

Given a processor's hardware information, my model is adept at deriving the optimal frequency ratio that leads to the minimum energy consumption of the processor. A power manager can maximize energy efficiency by adjusting the processor's frequency to meet the optimal frequency ratio. After analyzing the correlations between frequency ratio and energy

consumption, I drew an intriguing conclusion - a small static power proportion leads to high energy-saving performance. To shed bright light on the applicability of my frequency-aware DVFS model, I conducted a simulation study using the parameters from three real-world processors. I confirmed that the energy-efficiency performance of my model is on par with the well-known utilization-based DVFS scheme. The simulation results are evident that my model is conducive to projecting the energy-saving performance of DVFS-enabled computing systems running virtual machines.

Chapter 4

Security-Aware Energy Management in Clouds

Cloud computing over the internet reveals a remarkable potential to provide on-demand services to consumers with great flexibility in a cost-effective manner. Security issues coupled with resource allocations in cloud computing remain a challenging problem to be tackled by the industry and academia. While moving towards the concept of on-demand services and resource pooling in a distributed computing environment, security is a major obstacle for this new dreamed vision of computing capability. At the same time, the research on energy-efficient networking infrastructures is of great importance for service providers, network administrators, and equipment manufacturers. In this chapter, novel energy-aware scheduling policies are developed catering for virtual machines running on clouds, in which service-level agreements (SLAs) are fulfilled. After addressing security concerns in cloud computing, I advocate for a research roadmap towards future security-aware energy management in clouds. I propose a high-level design for a security- and frequency-ware DVFS model or *SF-DVFS*, which orchestrates security services, security overhead analysis, and DVFS control green cloud computing systems. I delve into the main technical challenges associated with the proposed SF-DVFS model. After the description on the SF-DVFS high-level structure, the NSGA-II-SER algorithm (NSGA-II with security and energy-aware requirements) is developed to optimize both energy efficiency and security protections in cloud data centers.

In Section 4.1, I begin this chapter with a roadmap description by presenting the concepts of security services and strengths. Next, Section 4.2 discusses the development of security overhead models for various security services. I propose in Section 4.3.1 an idea to incorporate security and frequency awareness into the context of qualify of service (QoS), and Section 4.3.2 presents a security- and frequency-aware DVFS model (SF-DVFS) in clouds.

Our newly developed NSGA-II with security and energy-aware requirements algorithm is proposed in Section 4.4. Finally, I conclude this chapter with our achievements in Section 4.5.

4.1 Security Services and Strengths

The security of a cloud computing system entails a capability of keeping various attacks at bay. A security system built for clouds consists of a diversity of security services like data integrity, confidentiality, and authentication. Because security services are implemented by different algorithms, the security services experience various strength associated with computational overhead. For instance, data confidentiality may be furnished by the *RC4* (*Rivest Cipher 4*) or *AES* (*The Advanced Encryption Standard*) cryptographic algorithms. RC4 is a fast algorithm with low memory space overhead [130]. Importantly, Fluhrer *et al.* discovered a few vulnerabilities in the RC4 algorithm, meaning that RC4 is unsafe for any key size [44]. In contrast, AES encryption was rigorously reviewed for potential security loopholes before being standardized by *NIST* in 2001. Compared with RC4, AES is more secure at the cost of high overhead.

The security strength of a cryptographic algorithm largely depends on key size and the number of operation rounds. The key size directly resembles the strength of the algorithm against key search attacks. In the AES case, the key size can be configured at 128, 192, and 256 bits. Theoretically speaking, the number of guesses to crack AES protected data is 3.410^{38} for the 128-bit key, 6.210^{55} for the 192-bit key, and 1.110^{77} for the 256-bit key. On the other hand, expanding the number of operation rounds makes ciphers more secure, because a large number of rounds leaves no trails of original data. Therefore, one may make use of the number of operation rounds to gauge the quality of ciphers against potential cryptanalysis attacks [129].

To optimize the security strength of applications running on computing clouds, I advocate for future efforts to quantitatively measure the strength and computational overhead of different security services implemented by cutting-edge algorithms. It is arguably true

that the strength of a security service is proportional to the service's computing and communication overhead, because low-quality security services that bear high overhead should be replaced by either fast-service counterparts or high-quality services with high overhead.

4.2 Security Overhead Models

Among a variety of security services, confidentiality, integrity, and availability are three common services to safeguard sensitive data. Among these three types of services, I first focus on the security overhead models developed to capture the correlation between strength and overhead in the confidentiality and integrity services. Then, I shed some light on the idea of constructing a security overhead model for data availability services.

4.2.1 Confidentiality and Integrity

A security service may be implemented by multiple implementation instances, each of which have distinctive security strength and computing overhead. In this chapter, I refer to the implementation instances as security service instances or security instances for short. Given a security service, I assign 1 as the strength value of the strongest security instance. The strength values of the other security instances in this service type are normalized based on the strongest instance. The overhead of each security instance should be derived from a program profiling study. Let us take the confidential service security as an example. Table 4.1 summarizes the strengths and speed of the encryption algorithms implemented in the five confidentiality instances. Similarly, Table 4.2 lists the hash functions supporting the five integrity instances. The details on these security overhead models can be found in the literature [116][94].

The overhead of the cryptographic instances is measured on virtual machines running on a physical machine powered by a 3.3 GHz duo-core CPU, 2.0 GB main memory, and 400 GB disk [29]. The overhead of each security instance is heavily reliant on the size of data to be protected and the security instance's speed. More specifically, the overhead of securing

data equals to data size divided by the speed of the given security instance. Such a security overhead plays a key role in utilizing slack time to adjust security and frequency levels in a resource management system articulated in Section 4.3.1.

Table 4.1: The Encryption Algorithms for Confidential Service.

Encryption algorithms	Strength	Speed (Mb/s)
IDEA	1.00	17.34
DES	0.90	18.21
Rijndael	0.64	39.88
Blowfish	0.36	39.96
RC4	0.30	87.07

Table 4.2: The Hash Functions for Integrity Service.

Hash functions	Strength	Speed (Mb/s)
TIGER	1.0	48.03
RIFDMD-160	0.77	71.27
SHA-1	0.63	80.67
RIFDMD-128	0.36	86.97
MD5	0.26	138.12

4.2.2 Data Availability

Now I propose an approach to building overhead models for data availability services in cloud storage. High data availability becomes possible with the full support of replication services or erasure code services, which are summarized as follows.

Data replication is a simple yet effective approach to tolerating failures in cloud storage. In case of a lost data block, one replica block is sufficient to fix the problem with the minimum data movements over networks. A high replica factor like triplication boosts storage system performance via parallel I/Os [170]. An overhead model dedicated to data replication is comprised of replication service instances representing different replica factors. In this model, a high replica factor offers high data availability at the cost of creating replications. On the flip side, the overhead can be reduced by lowering the replica factor. Intuitively, in this model security levels of data availability are measured by replica factors. The overheads of read operations are in stark contrast from those of write operations. A high replica factor

leads to fast reads and expensive writes; the opposite is true for a low replica factor. Thus, the overhead model must be separately developed for reads and writes.

Erasure codes are widely adopted in cloud storage housed in data centers [22][45][62]. The Reed-Solomon (RS) code is a popular erasure-code solution, thanks to its optimal storage efficiency and high level of data availability tolerance [151]. These (k+r,k) RS codes encode source data with a k×(k+r) *Generator Matrix*, which involves a k×k *Identity Matrix* and a k×r *Redundancy Matrix* (see the details in [98]). In RS encoding, parity strips are originated by multiplying k data strips with the k×r redundancy matrix. In the security overhead model for data availability services fueled by RS code, security levels and overhead are obtained from parameters k and r. In general, the large values of r offer a high level of availability (high security level) at an expensive cost of constructing parity strips. Reducing r value curtails the overhead by sacrificing data availability.

4.3 Security- and Frequency-aware Modeling

This section consists of two related components: I first investigate the security and frequency awareness issues in QoS (see Section 4.3.1), followed by a DVFS modeling method that incorporates security and frequency awareness (see Section 4.3.2).

4.3.1 Security and Frequency Awareness in QoS

The security overhead models articulated in Section 4.2 can be incorporated into a QoS model to estimate the time spent in performing assigned security services. Specifically, security overhead prolongs task execution times, which in turn triggers performance degradation. Nevertheless, tasks that are slowed down by such security overhead are acceptable as long as QoS requirements can be fulfilled.

In conventional real-time task models, the worst case execution-time (WCET) and deadlines are two key parameters capturing QoS requirements of real-time applications. Besides

WCET and deadlines, CPU frequency is a practical parameter to prescribe QoS requirements. Given memory and I/O resources, a task's execution time largely depends on an assigned processor and its CPU frequency level. It is feasible to convert time-aware requirements into frequency-aware requirements.

Fig. 4.1 outlines a model of converting frequency requirements from deadlines and WCET specified as timing constraints. In this modeling procedure, task requirements are modeled in the format of minimum frequency requirements. By the same token, security overhead incurred in security-sensitive applications should be integrated into the WCET measures, which are converted into frequency requirements. As a future research direction, tremendous efforts will be dedicated to ways of constructing frequency requirements from WCET values that are reliant on time spent in performing security services. Such security service times will be derived from security overhead model (see also Section 4.2).

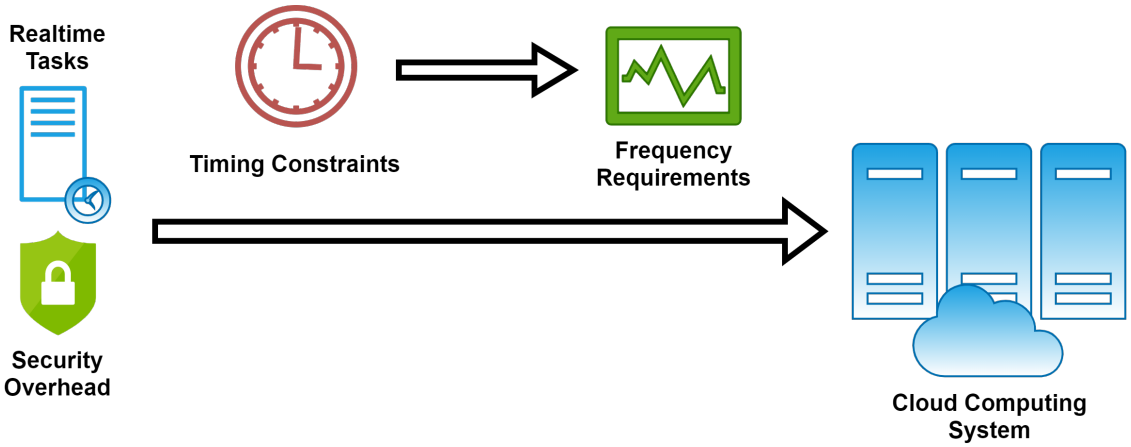


Figure 4.1: A procedure of converting frequency requirements from deadlines and WCET specified as timing constraints. Task requirements are modeled in the format of minimum frequency requirements in clouds.

I investigate multiple virtual machines running on a group of physical machines modeled as $C = \{c_1, c_2, ..., c_m\}$. Let us define a set of n virtual machines as $V = \{vm_1, vm_2, ..., vm_n\}$ running on machine c, where I have $c \in C$. Each virtual machine is denoted as a pair $vm_i = (a_i, f_i^{req})$, where a_i is the creation time of virtual machine vm_i, f_i^{req} is the minimum frequency

requirement of virtual machine vm_i. The correlation between an overall task's frequency requirement and each virtual machine's frequency requirement is formally expressed as:

$$f_{V,c}^{req} = \sum_{i \in V} f_{i,c}^{req}. \tag{4.1}$$

I define a *security-related frequency requirement* co^{fre} as the frequency requirement that is derived from the corresponding security overhead. I layout in Eq. (4.2) the relation between an overall security-related frequency requirement and each virtual machine's security-related frequency requirement. The security-related frequency requirement of virtual-machine set V running on physical machine c is an accumulated measure of the security-related requirements of all the virtual machines in set V. Thus, I have

$$co_{V,c}^{fre} = \sum_{i \in V} co_{i,c}^{fre}. \tag{4.2}$$

Considering the minimum security-related frequency requirement $co_{V,c}^{fre}$, I show that physical machine c has a capability to support all the virtual machines in V without violating SLAs as long as the following requirement (4.15) holds.

$$f_{V,c}^{conf} \geq f_{V,c}^{req} + co_{V,c}^{fre}. \tag{4.3}$$

where $f_{V,c}^{conf}$ is a frequency configured for virtual-machine set V on physical machine c. To meet specified SLA requirements, one has to regulate the frequency $f_{V,c}^{conf}$ in a way to exceed a threshold of $f_{V,c}^{req} + co_{V,c}^{fre}$.

4.3.2 Security- and Frequency-aware DVFS Modeling

Now, I propose a DVFS model embracing security and frequency awareness. Fig. 5.2 unravels a high-level architecture of the security- and frequency-aware DVFS model or *SF-DVFS*, in which the frequency-aware DVFS, a security overhead model, and security services are seamlessly integrated.

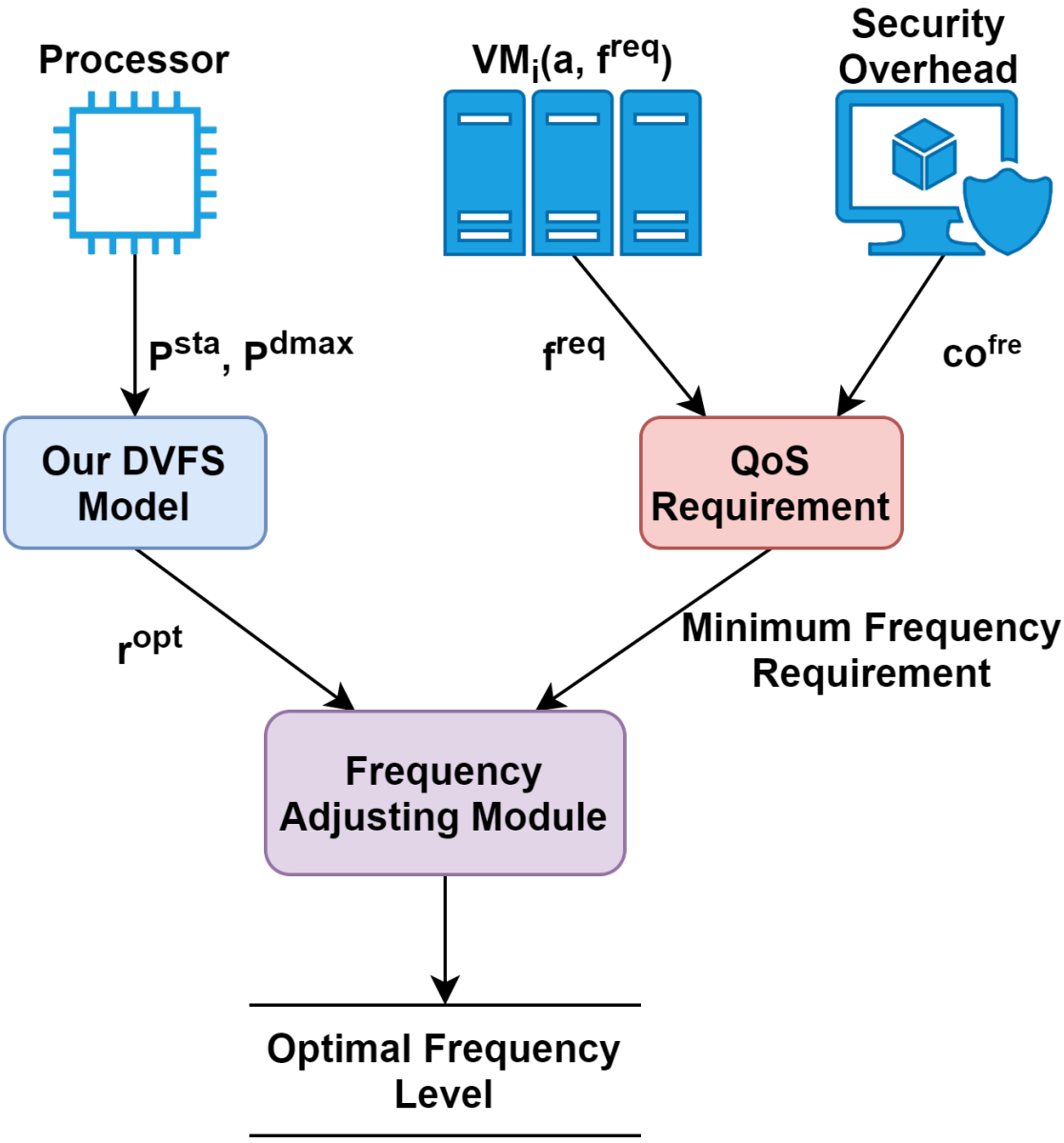


Figure 4.2: The security- and frequency-aware DVFS model *SF-DVFS* integrates the frequency-aware DVFS, a security overhead model, and security services in the context of quality of services (QoS).

In one of my recent studies [99], I proposed a frequency-aware DVFS model aiming to conserve energy consumption of tasks with QoS requirements. In my DVFS model, the energy consumption of processor c is calculated as:

$$E_c = \frac{\sum_{i \in V} \Gamma_i}{f_c^{max}} \left(\frac{P_c^{sta}}{r_c} \; + \; P_c^{dmax} (r_c)^2 \right). \tag{4.4}$$

where Γ_i is the total number of clock cycles of vm_i, f_c^{max} is the max frequency level of processor c, frequency ratio r is the ratio between the current processor frequency f and the maximum frequency f^{max} held by a processor. P_c^{sta} is the static power of processor c; P_c^{dmax} is the maximum dynamic power of processor c.

Let r^{opt} be an optimal frequency ratio that curbs the energy consumption in the system. I obtain optimal ratio r^{opt} as:

$$r_c^{opt} = \sqrt[3]{\frac{P_c^{sta}}{2 P_c^{dmax}}}. \tag{4.5}$$

Given the optimal frequency ratio r^{opt} in Eq. (4.5), I derive the minimized energy consumption E^{opt} from this ratio r^{opt}, static power P_c^{sta}, and maximum dynamic power P_c^{dmax}. Thus, I have

$$E_c^{opt} = \frac{\sum_{i \in V} \Gamma_i}{f_c^{max}} \left(\frac{P_c^{sta}}{r_c^{opt}} + P_c^{dmax} \; \cdot \; (r_c^{opt})^2 \right). \tag{4.6}$$

In the system architecture depicted in Fig. 5.2, the *QoS requirement module* outputs a minimum frequency requirement from two input parameters, namely, (1) the minimum frequency requirement f^{req} and (2) the security-related frequency requirement co^{fre} prescribed in virtual machines. On the left-hand side of the architecture, my *frequency-aware DVFS model* incorporates the static and maximum dynamic power constants to obtain an optimal frequency ratio r^{opt}. Finally, the *frequency adjusting module* compares the optimal frequency ratio and the overall minimum frequency requirement to configure the most appropriate frequency level to reduce the energy consumption of the virtual machines running on a physical machine.

To enhance the system architecture outlined in Fig. 5.2, I advocate for the following future research directions. First, practical VM consolidation and management policies should be blended with DVFS to build energy-efficient clouds running tasks with QoS requirements. Second, machine-learning-based prediction techniques are expected to boost the performance of the VM consolidation and management policy. Third, the security overhead (see also Section 4.2) largely depends on security levels. Hence, it is desirable to dynamically configure security levels to fulfill QoS requirements in my proposed security-aware energy management system. For example, if QoS requirements are permitted, security service instances with strong strengths should be elected to maximize security in clouds. Otherwise, security levels must be lowered to avert performance degradation.

4.4 Secure and Economical DVFS-enabled Scheduling Policy with NSGA-II-SER for Clouds

After proposing the SF-DVFS high-level structure, I am positioned to design the NSGA-II-SER algorithm (NSGA-II with security and energy-aware requirements) to optimize both energy efficiency and security protections in cloud data centers.

This section consists of four related components. I first introduce the procedure and benefits of genetic algorithms in Section 4.4.1. Then, I formulate this multi-objective optimization problem in Section 4.4.2). Next, I introduce the widely adopted NSGA-II algorithm in Section 4.4.3, followed by my newly developed *NSGA-II with security and energy-aware requirements* algorithm in Section 4.4.4 and 4.4.5). Finally, I introduce the Pareto frequency ratio in Section 4.4.6 and obtain the experimental results from three real-world processors under two groups of security methods in Section 4.4.7.

4.4.1 Genetic Algorithms

Recall that (see Section 2.4.4) genetic algorithms or GAs, a random search method that simulates biological evolution process, are widely adopted in resource scheduling for cloud

computing systems [52]. Genetic algorithms are rooted in an early development spearheaded by Holland and his associates back in 1975 [107]. A genetic algorithm - a search heuristic - is inspired by Charles Darwin's theory of natural evolution: the algorithm reflects a process of natural selection where the fittest individuals are chosen for reproduction with a hope to produce offspring of the next generation.

There are six phases are considered in a genetic algorithm:

- **Initialization of Population.** A population consists of a certain number of individuals, each with its unique code, treated as gene chromosomes. More specifically, chromosomes are entities with characteristics, and the external performance of individuals is determined by chromosomes.

- **Fitness Function.** The fitness of individuals is calculated, and the calculation of fitness must be based on the optimization goal. Generally speaking, individuals with high fitness measures are close to an optimization goal.

- **Selection.** According to fitness levels, the algorithm elects individuals who perform crossover mutation operations with different probabilities. Higher fitness corresponds to more possibility of being selected, and the unselected individuals are unable to participate in subsequent evolution processes - thus, the dominant individuals are retained. Overly strong or insufficient fitness selection bias must be protected against, and the awfully strong fitness selection bias can lead to sub-optimal solutions. In contrast, week fitness bias selection may results in an unfocused search.

- **Crossover.** The crossover operator is similar to reproduction and biological crossover. According to the preset crossover probability, two individuals are randomly selected, and some of their coding sequences are exchanged to generate new individuals. The individual obtained by crossover operation may be a combination of the dominant genes of its mother parent - a more optimized individual. There are three commonly

used crossing operations: single point crossing, multiple point crossing, and uniform crossing.

- **Mutation.** Few individuals are randomly chosen to mutate based on a specific probability. In other words, some bits of chromosomes can be flipped. It is used to maintain and introduce diversity in genetic population; normally, the probability is minuscule.

- **Termination Condition.** Determine the termination time of the evolutionary iterative process. The algorithm manages various termination conditions, such as the fitness of optimal individuals that have met the requirements (fitness within the specified range), the number of iterations reaching a specific number, and the fitness of the optimal individual or population tends to be stable, and to name just a few.

Genetic algorithm starts with initialization population to mutation operation, followed by jumping back to individual evaluation. The algorithm continues its computation until the termination condition is met. Considering that each new generation is a set of objective function solutions, I argue that increasing the number of algebras tend to boost the accuracy of an objective function.

When it comes to a single-objective optimization problem, general genetic algorithms simply fulfill the requirements. The original genetic algorithm design, however, is no longer applicable when the problem is extended to the multiple-objective optimization field. This is because there is usually a deep relationship among multiple goals - optimizing one goal usually affects the other goals, and it is hard to get the optimal solution for all the goals. At this time, finding a suitable fitness function becomes the key toward solving the multi-objective genetic algorithm in my dissertation research.

4.4.2 Multiple-objective Optimization Problem Formulation

The Objectives

According to the security- and frequency-aware QoS and *SF-DVFS* structure proposed in Section 4.3, I demonstrate the feasibility of combing my frequency DVFS model (see chapter 3) with security requirements. Now, I formulate the SF-DVFS model as a multiple-objective optimization problem to furnish the development of the generic algorithm - NSGA-II-SER - in Section 4.4.4. Table 4.3 lists the notation used throughout this section.

Table 4.3: Symbol and Annotation 2

Symbol	Annotation
f_c^{opt}	optimal energy-saving frequency for processor c applied my model.
$f_{i,c}^{req}$	frequency requirement of virtual machine vm_i on processor c
$f_{V,c}^{req}$	frequency requirement of virtual machine set V on processor c. V is the set of virtual machine running on processor c.
$f_{V,c}^{conf}$	frequency configured for virtual machine set V on processor c
Γ_i	total execution requirements of virtual machine vm_i.
$\gamma_{i,j}$	execution requirements of the j th segment of virtual machine vm_i.
P_c^{sta}	static power of processor c.
P_c^{dmax}	max dynamic power power of processor c.
r_c	frequency ratio of processor c.
f_c^{max}	max frequency level of processor c.
E_c^{opt}	the optimal energy consumption of processor c.
t_c	the execution time of processor c.
t_i'	the overall executing time t_i' of virtual machine i.
t_i	the original task executing time of virtual machine i.
ts_i	the security overhead encryption time of virtual machine i.
sl_i	the security level of virtual machine i.
te_i	the time overhead of encrypting each clock cycle on virtual machine i.

Per Eq. 3.9 articulated in Section 3.2, the total CPU power is obtained from frequency ratio r as:

$$P = P^{sta} + P^{dmax} \cdot r^3,$$

and Eq. 3.10 in Section 3.3 captures the relation between energy, power and time consumption as:

$$E = P \cdot t = P_i \cdot \frac{\Gamma_i}{f_i} \tag{4.7}$$

The energy consumption of processor c which T VMs running on is:

$$E_c = (P_c^{sta} \ + \ P_c^{dmax}(r_c)^3) \cdot t_c, \tag{4.8}$$

where processor c is executed in time t_c, r_c is the frequency ratio of processor c. P_c^{sta} and P_c^{dmax} is the static and maximum dynamic power of processor c, respectively.

Assessing the security overhead in this model, I partition the overall executing time t_i' on virtual machine into two camps: the original task executing time t_i and security overhead encryption time ts_i. Thus, I have

$$t_i' = t_i + ts_i. \tag{4.9}$$

Each security service is implemented through an algorithm such as encryption/decryption algorithms. For instance, the time cost of using the encryption algorithm to encrypt data is associated with the corresponding quantitative security level achieved by the encryption algorithm. The encryption time of each data unit is generally linear with the security level achieved. I denote the security level of virtual machine vm_i is sl_i, and let time te_i be the time overhead of encrypting each clock cycles using the encryption algorithm to fulfill the security-level requirements. Given T virtual machines running on processor c, I derive overall executing time t_c' of processor c by applying Eqs. (4.7) and (4.9). Hence, I express overall executing time t_c' as

$$t_c' = \frac{\sum_{i \in T} \Gamma_i}{f_c} + \sum_{i \in T} te_i \cdot \Gamma_i \tag{4.10}$$

where Γ_i is the total number of clock cycles of vm_i.

After modeling the new overall time t_c' of processor c, I am able to gauge the overall energy consumption E_c, a portion of which is contributed by security overhead. More formally,

I write the overall energy consumption E_c as follows.

$$E_c = (\frac{\sum_{i \in T} \Gamma_i}{f_c} + \sum_{i \in T} te_i \cdot \Gamma_i) \cdot (P_c^{sta} \ + \ P_c^{dmax}(r_c)^3).$$

$$E_c = (\frac{\sum_{i \in T} \Gamma_i}{r_c \cdot f_c^{max}} + \sum_{i \in T} te_i \cdot \Gamma_i) \cdot (P_c^{sta} \ + \ P_c^{dmax}(r_c)^3). \tag{4.11}$$

Intuitively, the above Eq. 4.11 is not a quadratic function with respect to frequency ratio rc, and security service time te_i for each data unit is also a variable. Consequently, it becomes futile to take the derivative of E_c - a solution proposed in Section 3.3 - to resolve the optimization problem in this section. To tackle this issue, I manage to formulate this challenging problem into a multiple-objective optimization problem. More prosaically, I start off modeling the optimization problem with two distinct objectives: minimizing energy consumption in clouds and maximizing the quality of security offered to applications.

I formally express the first objective - minimizing energy consumption in cloud data centers below.

$$Min\ E_c = Min\ (\frac{\sum_{i \in T} \Gamma_i}{r_c \cdot f_c^{max}} + \sum_{i \in T} te_i \cdot \Gamma_i) \cdot (P_c^{sta} \ + \ P_c^{dmax}(r_c)^3). \tag{4.12}$$

The second objective - maximizing the quality of security - is written as the following expression. It is noteworthy that the quality of security is enhanced by lifting the corresponding security levels.

$$Max \sum_{i \in T} sl_i. \tag{4.13}$$

Aiming to solve the multiple-objective optimization problem, I ought to seamlessly integrate these two objective functions. Thus, I translate the security function from the

maximization problem into the minimization problem as the following format

$$Min(-\sum_{i \in T} sl_i). \tag{4.14}$$

The Constraint Conditions

Here I present the constraint conditions that play a critical role in my multi-objective optimization model.

Recall that (see also Section 4.3.1) to fulfill the minimum security-related frequency requirement $co^{fre}_{V,c}$, I demonstrate that physical machine c has a capability to support all the virtual machines in V without violating SLAs as long as the following requirement (4.15) holds.

$$f^{conf}_{V,c} \geq f^{req}_{V,c} + co^{fre}_{V,c}. \tag{4.15}$$

where $f^{conf}_{V,c}$ is a frequency configured for virtual-machine set V on physical machine c. To meet specified SLA requirements, one has to regulate the frequency $f^{conf}_{V,c}$ in a way to exceed a threshold of $f^{req}_{V,c} + co^{fre}_{V,c}$.

The frequency ratio of each possessor has its own rage rather than directly from 0 to 1. As I studied, *frequency ratio* r is a ratio between the current processor frequency f and the maximum frequency f^{max} held by a processor. Hence, the lowest range of frequency ratio largely depends on the lowest frequency level of the processor. Hence, I have

$$r > f_{minlevel}/f_{max}. \tag{4.16}$$

4.4.3 The NSGA-II Algorithm

Multi-objective genetic algorithms are a family of evolutionary algorithms analyzing and solving multi-objective optimization problems. A core component of genetic algorithms is to

coordinate the relationships among objective functions and to discover an optimal solution set that makes each objective function reach a relatively large or relatively small function value as much as possible.

The Non-Dominated Sorting Genetic Algorithm or *NSGA-II* is a powerful multi-objective genetic algorithm for solving multi-objective optimization problems. Deb *et al.* improved the NSGA algorithm in 2002 by proposing a non-dominated sorting genetic algorithm with an elite strategy, and the new algorithm is referred to as Elitist Non-Dominated Sorting Genetic Algorithm or *NSGA-II* [33]. The following three aspects have been addressed in the updated version of the algorithm.

- A fast non-dominated sorting algorithm was devised to suppress the complexity of non-dominated sorting.
- Elite strategy was designed to expand the sampling space.
- The crowding and crowding comparison operators were applied to ensure the diversity of a given population.

Now I am positioned to design and implement a multi-objective genetic algorithm, the security- and energy-aware scheduling algorithm, to analyze and solve my multi-objective optimization problems in scheduling. My algorithm, which is referred to as *NSGA-II-SER*, is an improved NSGA-II algorithm in nature. The key and distinct feature of NSGA-II-SER lies in a coordinator that orchestrates the relationship between the energy-saving function and the security function, aiming to locate an optimal solution.

NSGA-II, one of the most efficient multi-objective evolutionary algorithms, only needs to be run once to obtain a Pareto optimal solution (see also 4.3). The NSGA-II procedure starts off with a randomly selected population, ranked by non-dominance. The first generation is initiated using binary selection, crossover, and mutation. After the first generation, elitism is introduced by comparing the population with the best non-dominated solution obtained in the previous phase. To create a new population, binary selections based on non-dominant

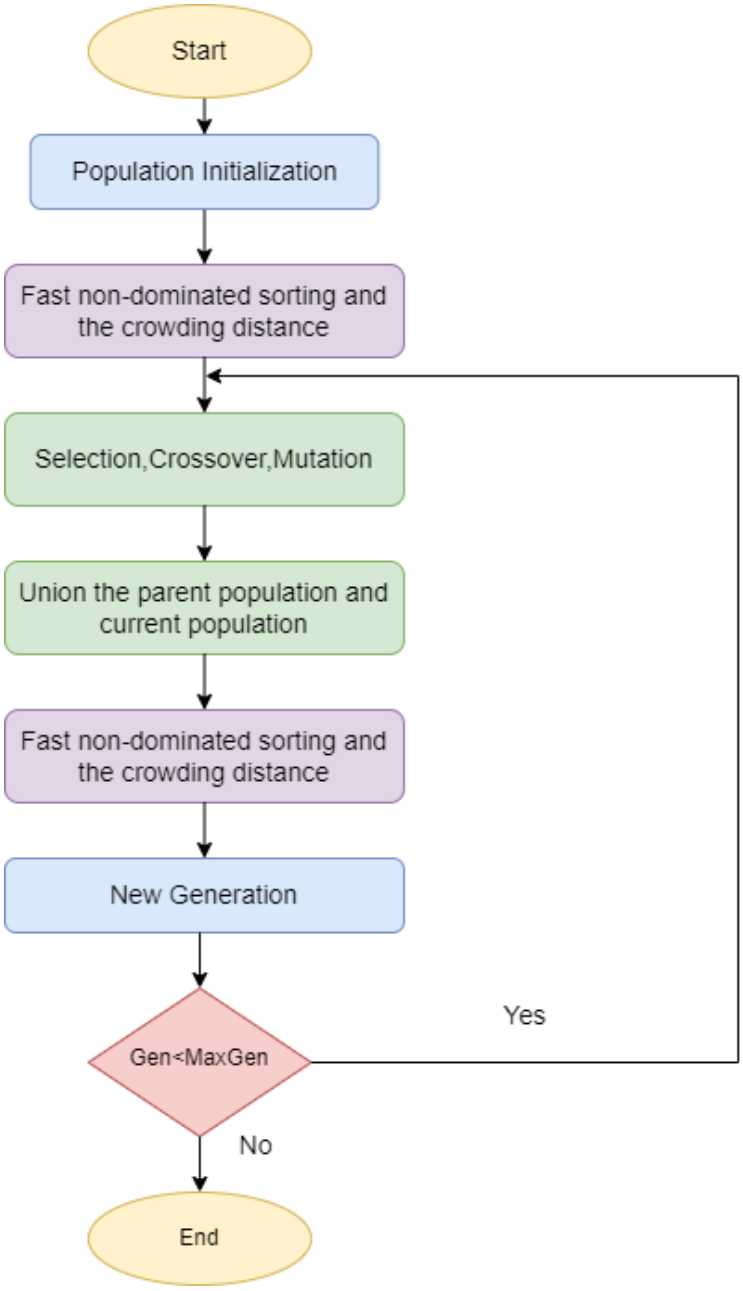


Figure 4.3: The workflow of the NSGA-II genetic algorithm.

and crowding distances are employed first, followed by the crossover and mutation operators. The algorithm deploys two functions, namely, the non-dominated sorting function and the crowding distance function.

It is noteworthy that the non-dominant sorting function is a cyclic classification process. This function consists of the following two steps. First, the non-dominated set in a group is identified as the first non-dominated layer, which is is removed from the group; the search for the non-dominated set in the remaining group is continued. Second, all solutions in the solution set are sorted according to the dominance relationship. Crowding distance is gauged for each individual, thereby making calculation results evenly distributed in a target space while maintaining the population's diversity. The crowding distance of each individual is calculated as the sum of distance differences between two adjacent individuals in each sub-objective function.

4.4.4 NSGA-II-SER: Non-dominated Sorting Genetic Algorithm II Processing Security and Energy Requirements

Now I am primed to design NSGA-II-SER - my NSGA-II algorithm that handles security and energy requirements. In what follows, I present the NSGA-II-SER algorithm from the perspectives of encoding, energy genes, and security genes. All the three pieces are seamlessly integrated in the newly designed NSGA-II-SER.

The Encoding Scheme. I implement an encoding scheme to furnish a mapping method from candidate solutions to chromosomes amid the process. This scheme, one of the critical components in a genetic algorithm, directly influences the operators carried out in the following crossover and mutation steps. Hence, the selection of a reasonable coding scheme immensely impacts the quality and efficiency of the NSGA-II-SER algorithm. In my implementation, I advocate for applying the binary code to divide the chromosomes into two parts: the left part is dedicated to energy genes, whereas the right portion is reserved for security genes.

Energy Genes. Energy genes represent a critical parameter of the energy objective function – frequency ratio. Recall that in Chapter 3, I measure frequency ratio as a ratio between current and max frequency. Obviously, parameter frequency ratio is a non-integer number sitting in a window between 0 and 1. I round this decimal number to two decimal places for the computing purpose. I opt for this configuration because choosing three decimal places does not necessarily improve optimization results. Therefore, I make use of seven genes to resemble the energy genes because the seventh power of 2 is 128, which is sufficiently large to cover all the possibilities of the frequency ratio.

Security Genes. Security genes represent the security strengths of a given tasks to be scheduled by the NSGA-II-SER algorithm. I ought to guarantee that the number of bits for genes is large enough to denote all the possibilities of security strengths. The range of security strengths is determined by the highest strength. For instance, in Table 4.4 illustrates that the security strengths are represented in a range between 29 and 118. This table is only for demonstration purpose, meaning that the table along with the security strengths should be dynamically updated according to deployed security services and measures. The implementation of NSGA-II-SER maintains an interface to take any security-strength table as input configuration data.

Process Description. The process is comprised of the following five steps. First, the algorithm I run the encoding scheme outlined above. Second, after carrying out the encoding scheme, I randomly originate initial population, referred to as the initial energy-security population, using a uniform distribution ranging from a lower bound to a upper bound. For example, the lower and upper bounds for frequency ratios are 0 and 1; the lower and upper bounds of security strengths are 29 and 118 as stipulated in Table 4.4. Third, I spark the fast non-dominated sorting and the crowding distance methods to implement the elitism policy. Fourth, I kick off the binary tournament, intermediate crossover, and Gaussian mutation operations to process the aforementioned energy genes and security genes. Fifth, I union the parent energy-security population and the current population generate a new

energy-security population. Hence, elite energy-security individuals in the parent population are retained, and the diversity is guaranteed during the process. The above five steps are repeatedly performed until the energy-security generation is reach to the generation number specified by us.

4.4.5 How to use the NSGA-II-SER Algorithm?

As an instance case, Table 4.4 summarizes the strengths and speed of the encryption algorithms implemented for the seven selected variants of RC6, running at the maximum frequency. The detailed measures are documented in the literature by Jiang *et al.* [68].

Table 4.4: The strength and encryption/decryption time of different RC6 rounds.

RC6 rounds	**4**	**6**	**8**	**10**	**12**	**14**	**16**
Strength	29	45	61	78	94	110	118
Time/Block (μs)	17	26	35	44	52	61	70

In my example environment, I apply implemented algorithm to solve the multi-objective optimization problem where the CPU is Intel i7-4770: the frequency level are 0.8, 1.0, ...,3.2, 3.4 GHZ with the 19W static power, and 76W max dynamic power. The frequency ratio is from $0.8/3.4 = 0.23$ to 1 ($r \in [0.23, 1]$)

I observe from Table 4.4 that the correlation between the strength and the speed is an approximately linear relationship. Consequently, I deploy the linear regression method to bridge the relation between security strength *stren* and encryption time for per block T_B using the following expression.

$$T_B = 0.568 * stren + 0.104 \tag{4.17}$$

The linear regression relation represented in Eq. 4.17 is also outlined in Fig. 4.4.

To normalize the input of the devised NSGA-II-SER algorithm, I have to figure out the relationship between energy objective function Eq. 4.12 and security objective function

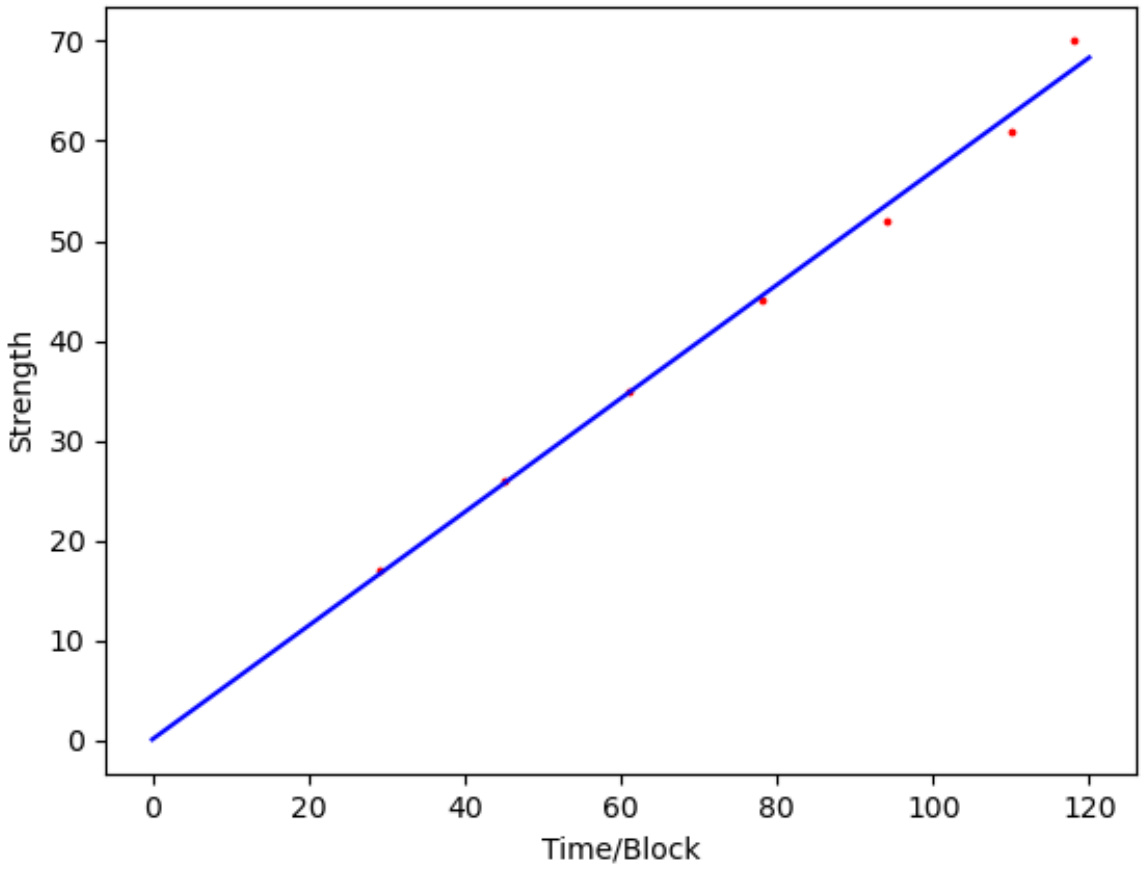


Figure 4.4: Linear regression for security strength and encryption speed. See also Eq. 4.17.

Eq. 4.14. After obtaining the linear relation between security strength *stren* (the quality of security sl_i in this example) and encryption time per block T_B (see eq. 4.17), I link the encryption time per block T_B with sl_i, which is derived from Eq. 4.14. Furthermore, encryption time te_i in Eq. 4.12 can be expressed as:

$$te_i = \frac{T_B * f_{max}}{f} = \frac{T_B}{r}. \tag{4.18}$$

After Eq. 4.18 is combined with the energy consumption objective functions written in 4.12, the energy objective function E_c is updated as

$$\begin{aligned} E_c &= (\frac{\sum_{i \in T} \Gamma_i}{r_c \cdot f_c^{max}} + \sum_{i \in T} te_i \cdot \Gamma_i) \cdot (P_c^{sta} \ + \ P_c^{dmax}(r_c)^3). \\ &= (\frac{\sum_{i \in T} \Gamma_i}{r_c \cdot f_c^{max}} + \sum_{i \in T} \frac{T_B}{r} \cdot \Gamma_i) \cdot (P_c^{sta} \ + \ P_c^{dmax}(r_c)^3). \end{aligned} \tag{4.19}$$

With the two objective functions Eq. 4.14 and Eq. 4.19 in place, the above question model is plugged into the NSGA-II-SER algorithm to facilitate solving the optimization problem. Assuming the clock cycle of virtual machine Γ_i is 10^{12}, I obtain the Pareto front of the NSGA-II-SER Model as depicted in Fig. 4.5, where the population size is 50 and maximum generation is 100.

Table 4.5 list the result of the implemented algorithm to demonstrate the format of a solution set, which consists of the optimal frequency ratio and picked security strength sl_i along with corresponding energy consumption. The security function is the opposite number of the normalized security strength - a ratio between current quality of security sl_i and the maximum measure sl_i (see, for example, 118 in Table 4.4). The normalized security strength makes the algorithm analysis more universal than using the directly opposite number of sl_i as the security function. Noticing that frequency f and strength sl_i are discrete, I choose the

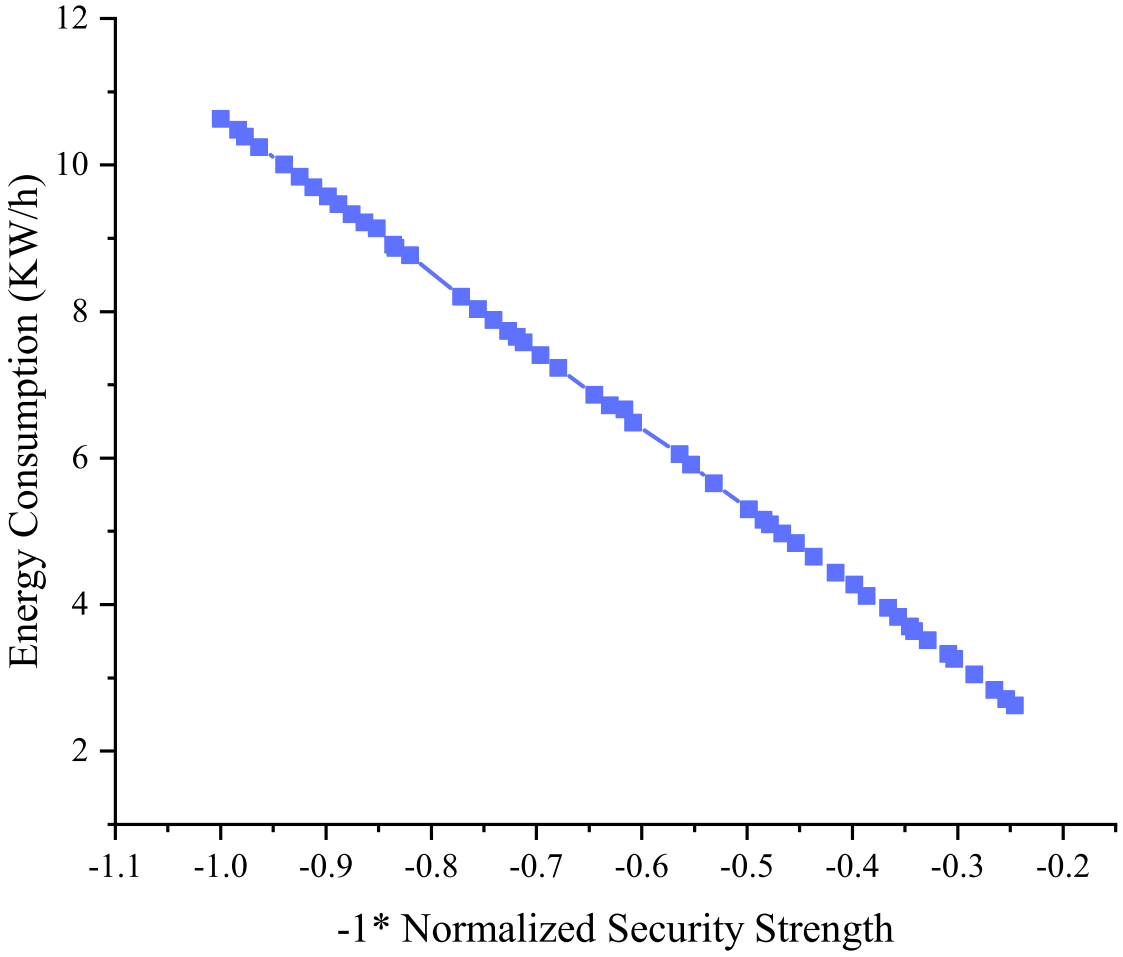


Figure 4.5: Pareto Front of the NSGA-II-SER algorithm. The population size is 50 and maximum generation is 100.

nearest r and sl_i values to optimally balance energy-saving and security measures. When the optimal security method is elected according to the QoS requirements, the frequency ratio is rounded up with rounded- down security requirements (e.g., satisfying Eq. 4.15).

The optimal solution, of course, can be customized by users. For example, let us consider a data center that is more sensitive to security. In that case, a Pareto solution is chosen with a high quality of security: frequency ratio 0.23 and 110.572, which use the 0.8 frequency level and RC6 14 rounds (strength 110). In contrast, if an overarching goal of a data center is high energy efficiency, the Pareto solution (0.23,29) will be selected to use RC6 4 rounds with a 0.8 frequency level to aggressively conserve energy while maintaining the encryption requirement. Therefore, the NSGA-II-SER algorithm is deployed to balance the energy-saving and security requirements in cloud data centers, where hardware information - including processors' static power and maximum dynamic power - is specified. Recall that, as detailed in Chapter 3, optimal energy conversation can be derived from static power and maximum dynamic power.

4.4.6 Pareto Frequency Ratio

Now I introduce the concept of Pareto frequency ratio, which is a frequency ratio when the current situation is included in the Pareto front. According to the data displayed in Fig. 4.5, each security strength has its own Pareto frequency ratio, leading to an optimal balance between energy savings and security protection.

Fig. 4.6 unravels the trend of the Pareto frequency ratio under different security strengths. It is noteworthy that the Pareto frequency ratio corresponds to different security levels hovering between 0.45 and 0.55, which agrees the modeling trend plotted in Fig. 3.4 in Chapter 3. It is evident that NSGA-II-SER keeps adjusting frequency ratio and tries to reach the best energy-saving performance in various security-level scenarios.

Table 4.5: Pareto Solutions of NSGA-II-SER after 100 generation

Frequency ratio r	**Quality of Security** sl_i	**Energy consumption (KWh)**	**Security function**
0.500582	118	10.6287	-1
0.4995419	29	2.624579	-0.162136
0.4791961	70.53771	6.371529	-0.4760989
0.5087977	88.58534	7.985717	-0.6509584
0.4776638	93.77896	8.467762	-0.7064368
0.4193662	49.94709	4.640713	-0.306681
0.5493056	79.5809	7.239084	-0.5603759
0.4848251	110.998	10.0083	-0.9088532
0.5311597	73.05245	6.610943	-0.4988907
0.5057545	82.17824	7.408037	-0.5857984
0.5141162	47.1203	4.25753	-0.2856448
0.5023456	86.52297	7.797979	-0.6295966
0.4762359	91.41571	8.257052	-0.68089
0.5180988	44.71431	4.042986	-0.2681331
0.5109865	48.5413	4.38408	-0.2961557
0.4956241	67.82411	6.116628	-0.4520447
0.5174023	41.93097	3.791992	-0.2483151
0.5589574	90.2774	8.240618	-0.6687657
0.4847718	52.9495	4.782961	-0.3295811
0.4631311	34.70877	3.155959	-0.199021
0.4947543	32.68228	2.95607	-0.185723
0.4963171	55.94063	5.047714	-0.3529829
0.4712487	57.32992	5.190178	-0.3640559
0.4841618	83.2804	7.51389	-0.5967568
0.5228693	115.2429	10.4018	-0.9634639
0.5016829	77.64313	6.999314	-0.5417697
0.5096007	112.8821	10.1721	-0.9328494
0.4607338	36.82	3.349542	-0.21312
0.4661541	75.26748	6.818173	-0.5193714
0.4771739	31.3188	2.83921	-0.176903
0.5219624	39.2922	3.556832	-0.2299529
0.455111	96.11403	8.734763	-0.7321888
0.5398858	43.3142	3.935568	-0.2581047
0.5295522	97.88661	8.849442	-0.7520782
0.4976362	113.9629	10.2658	-0.9467942
0.539322	78.04541	7.076789	-0.5456065
0.4764708	100.02	9.032287	-0.77642
0.5255105	39.7729	3.602479	-0.233269
0.4657222	58.92027	5.341618	-0.3768928
0.4678242	84.80415	7.676331	-0.6120768
0.4668708	95.3643	8.632454	-0.7238635
0.4920991	59.79678	5.3956	-0.3840412
0.5239896	61.22749	5.535233	-0.3958261
0.441664	29.44781	2.704321	-0.164966
0.5418952	116.4211	10.5565	-0.9789781
0.4901881	102.86	9.270642	-0.8095025
0.4984331	63.92497	5.765552	-0.418435
0.44888	35.93019	3.284359	-0.207148
0.4627028	109.6119	9.932212	-0.8914537
0.4825869	74.09629	6.688537	-0.508494

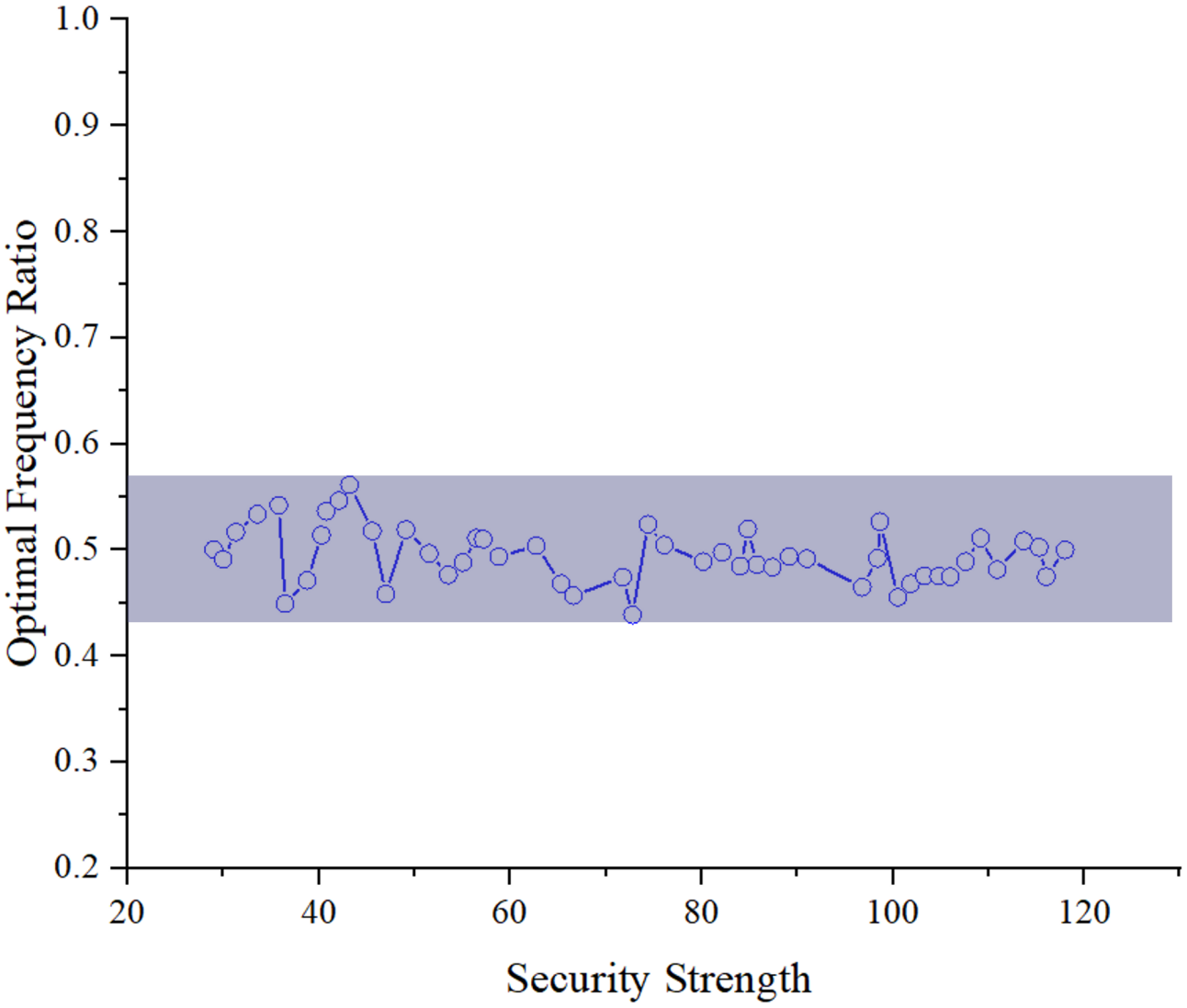


Figure 4.6: The frequency ratio of the point on Pareto front

4.4.7 Preliminary Result the NSGA-II-SER Algorithm

Table 4.6: The CPU configurations of the three tested servers.

Server	Frequency (GHz)	P^{sta} (W)	P^{dmax} (W)
Xeon E5-2670	{1.2,1.3,...,2.5,2.6}	84.1	342.9
Intel i7-4770	{0.8,1.0,...,3.2,3.4}	19	76
AMD Athlon	{0.8,1.5,2.0,2.7,3.0}	53	121.1

Recall that Table 3.2 summarizes the parameters of the five popular CPUs, from which I choose Xeon E5-2670, Intel i7-4770 , AMD Athlon (Table 4.6) to evaluate the performance of the NSGA-II-SER algorithm.

Given the security method chosen from Table 4.4, the Pareto front of Intel i7-4770 was discussed in Section . Fig. 4.7 and Fig. 4.8 reveal the Pareto fronts of the AMD Athlon and Xeon E5-2670 processors.

Running different security methods, I compare in Figs. 4.9, 4.10 and 4.11 the energy consumption of servers equipped with and without the NSGA-II-SER algorithm. Comparing the NSGA-II-SER algorithm with the baseline solutions, I assert that NSGA-II-SER is able to offer energy savings for servers running security services tabulated in Table 4.4 in Section 4.4.5.

Table 4.7: The strength and encryption/decryption time of other popular encrypt algorithm.

Encrypt Algorithm	Algorithm performance (KB/ms)	Security Level
SEAL	168.75	0.08
RC4	96.43	0.14
Blowfish	37.5	0.36
Knufu/Khafre	33.75	0.40
RC5	29.35	0.46
Rijndael	21.09	0.64
DES	15	0.90
IDEA	13.5	1.00

Aiming to validate the applicability of the NSGA-II-SER algorithm, I apply my NSGA-II-SER algorithm to these three processors running the other security methods listed in Table 4.7. Fig. 4.12, Fig. 4.13 and Fig. 4.14 depict the Pareto fronts of the Intel i7-4770, AMD Athlon and Xeon E5-2670 processors. The Pareto fronts of these processors using

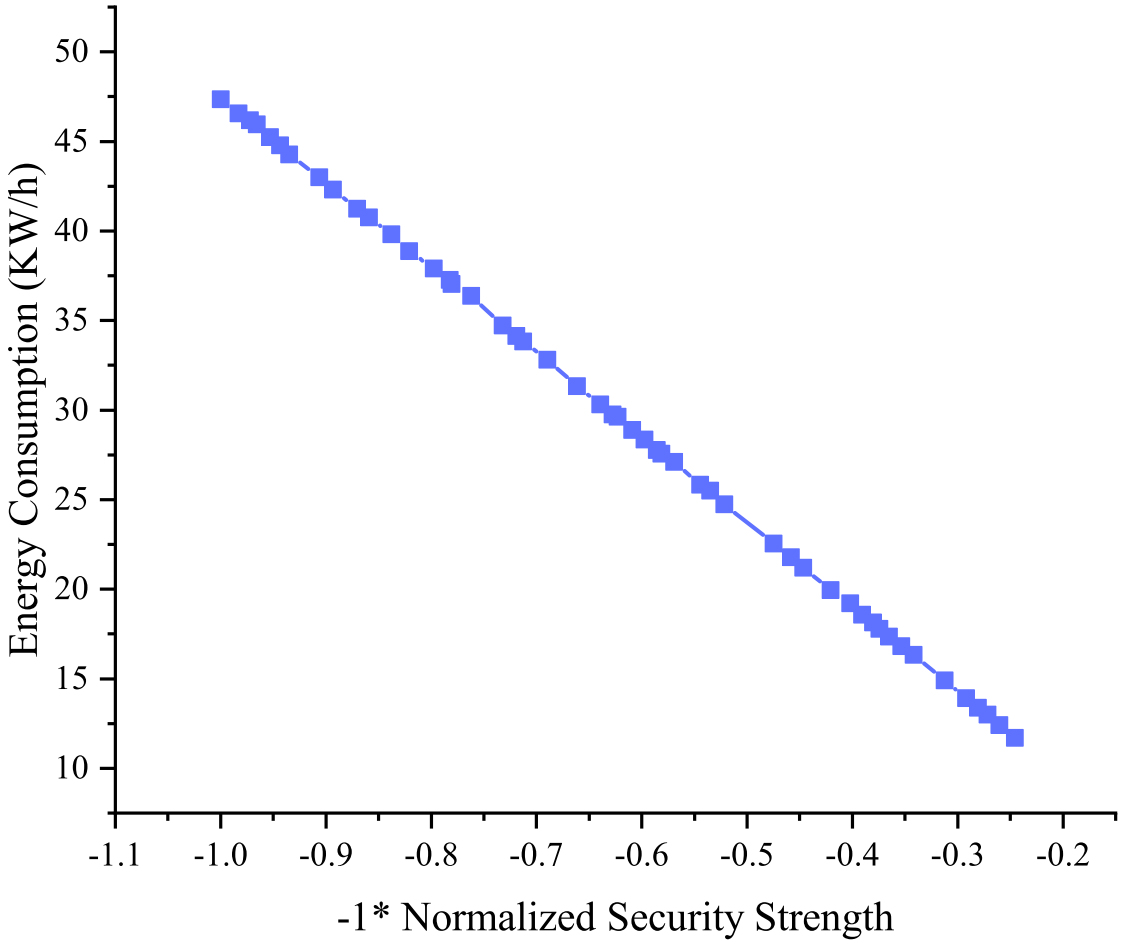


Figure 4.7: Pareto Front of the NSGA-II-SER algorithm of AMD Athlon using method from Table 4.4

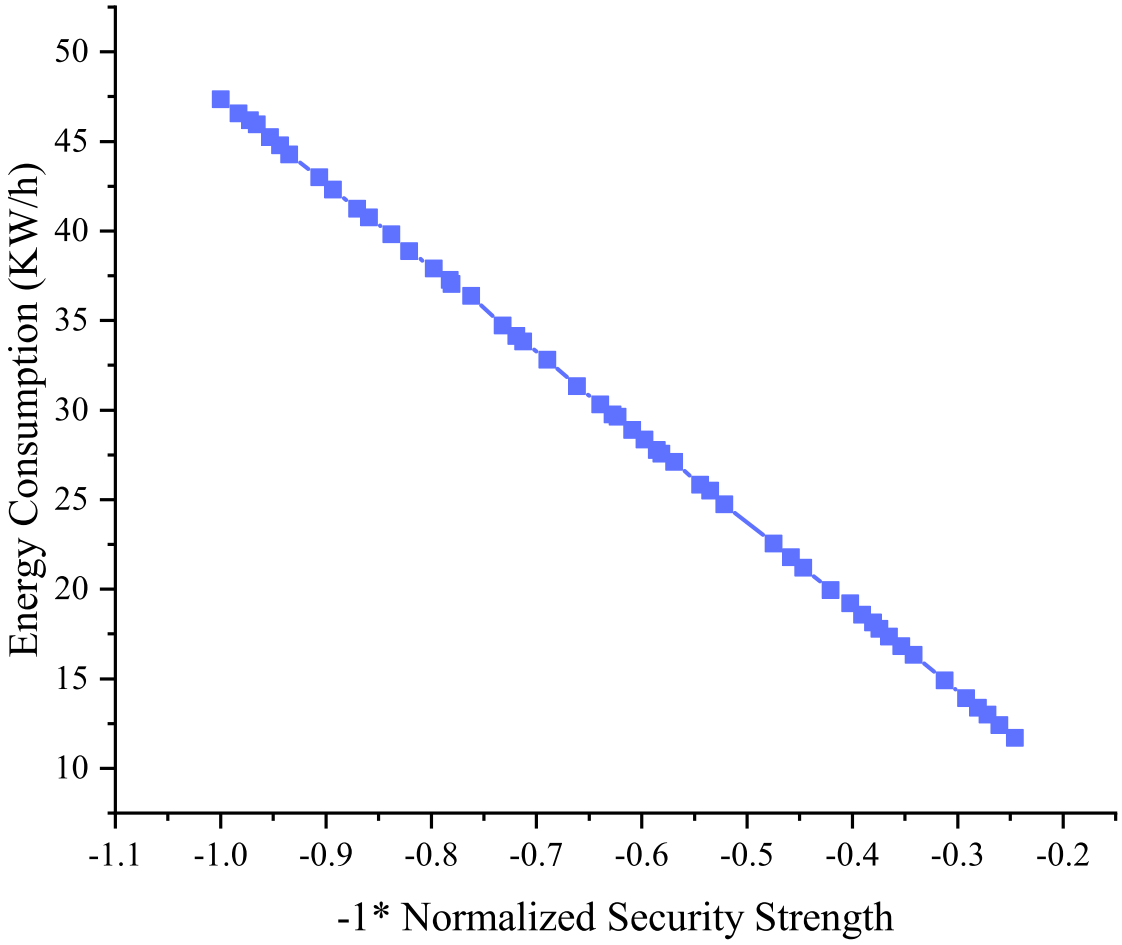


Figure 4.8: Pareto Front of the NSGA-II-SER algorithm of Xeon E5-2670 using method from Table 4.4

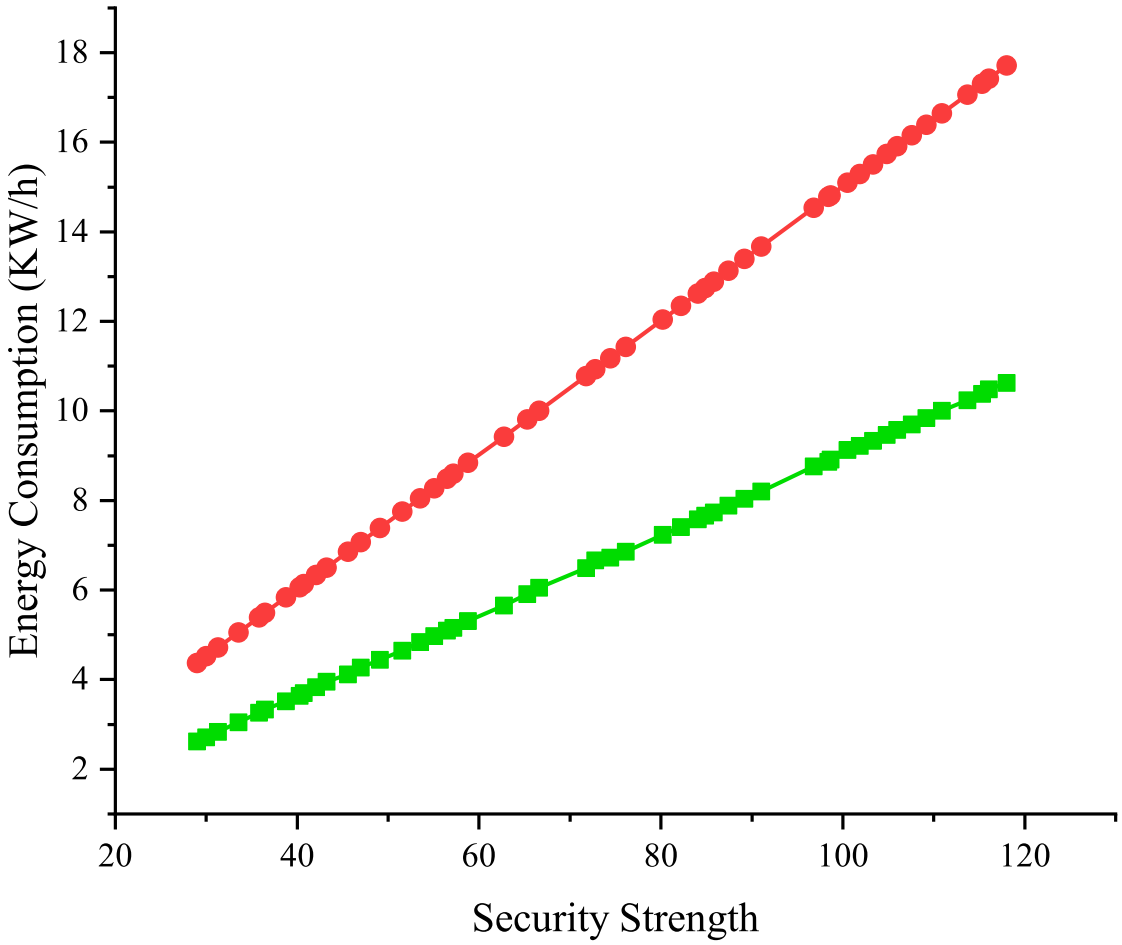


Figure 4.9: Energy consumption comparison for Intel i7-4770 equipped with and without the NSGA-II-SER algorithm using method from Table 4.4

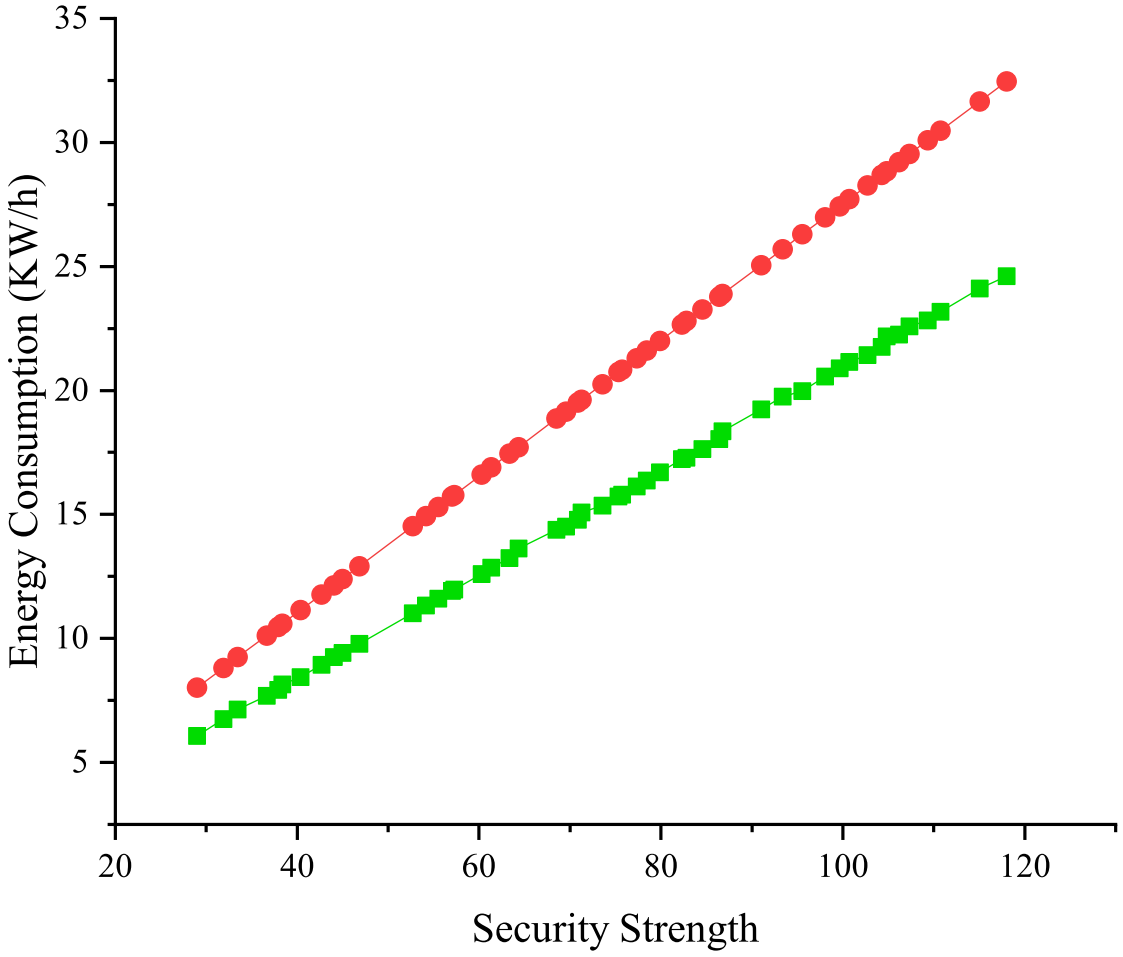


Figure 4.10: Energy consumption comparison for AMD Athlon equipped with and without the NSGA-II-SER algorithm using method from Table 4.4

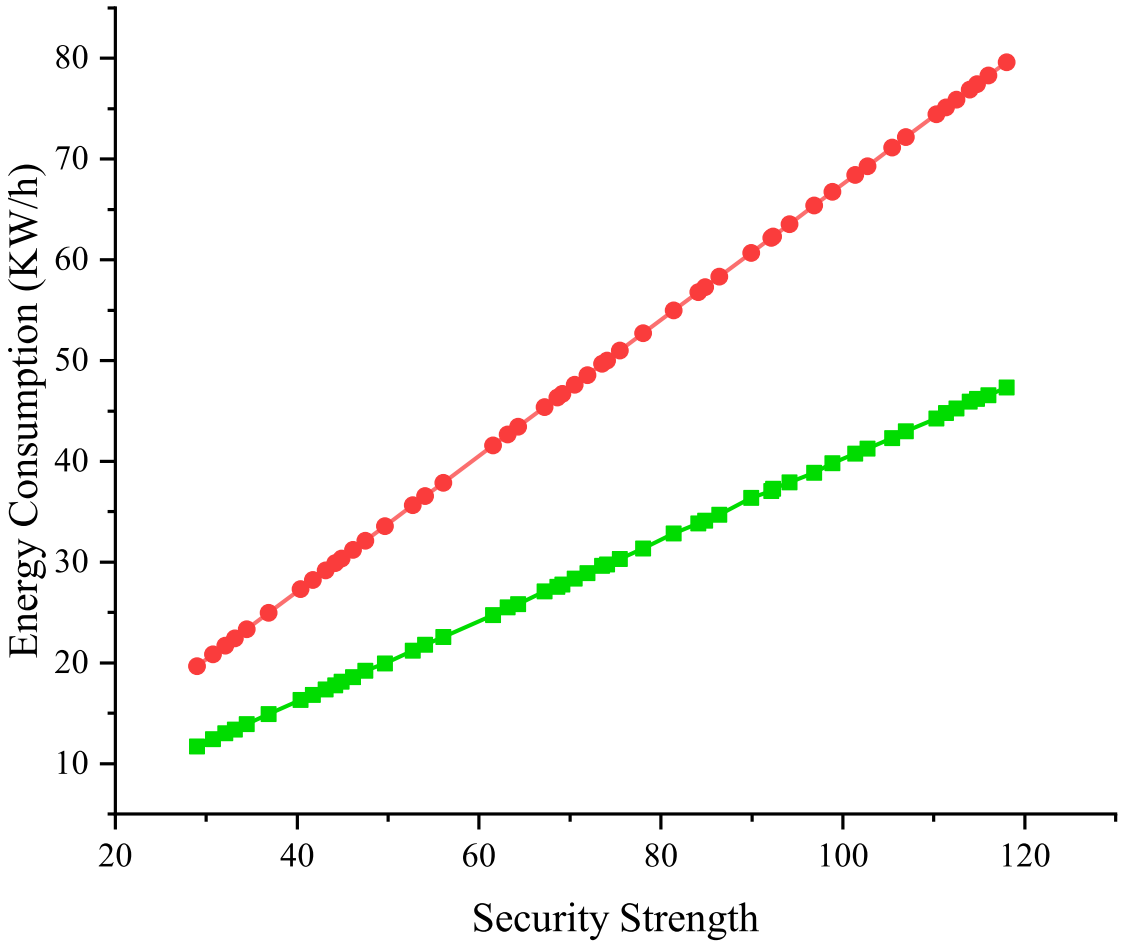


Figure 4.11: Eergy consumption comparison for Xeon E5-2670 equipped with and without the NSGA-II-SER algorithm using method from Table 4.4

methods in Table 4.7) share the same trend pattern as that of the processors executing services summarized in Table 4.4. The comparison result entails that my NSGA-II-SER algorithm is capable of discovering optimal solutions for the processors running new security services.

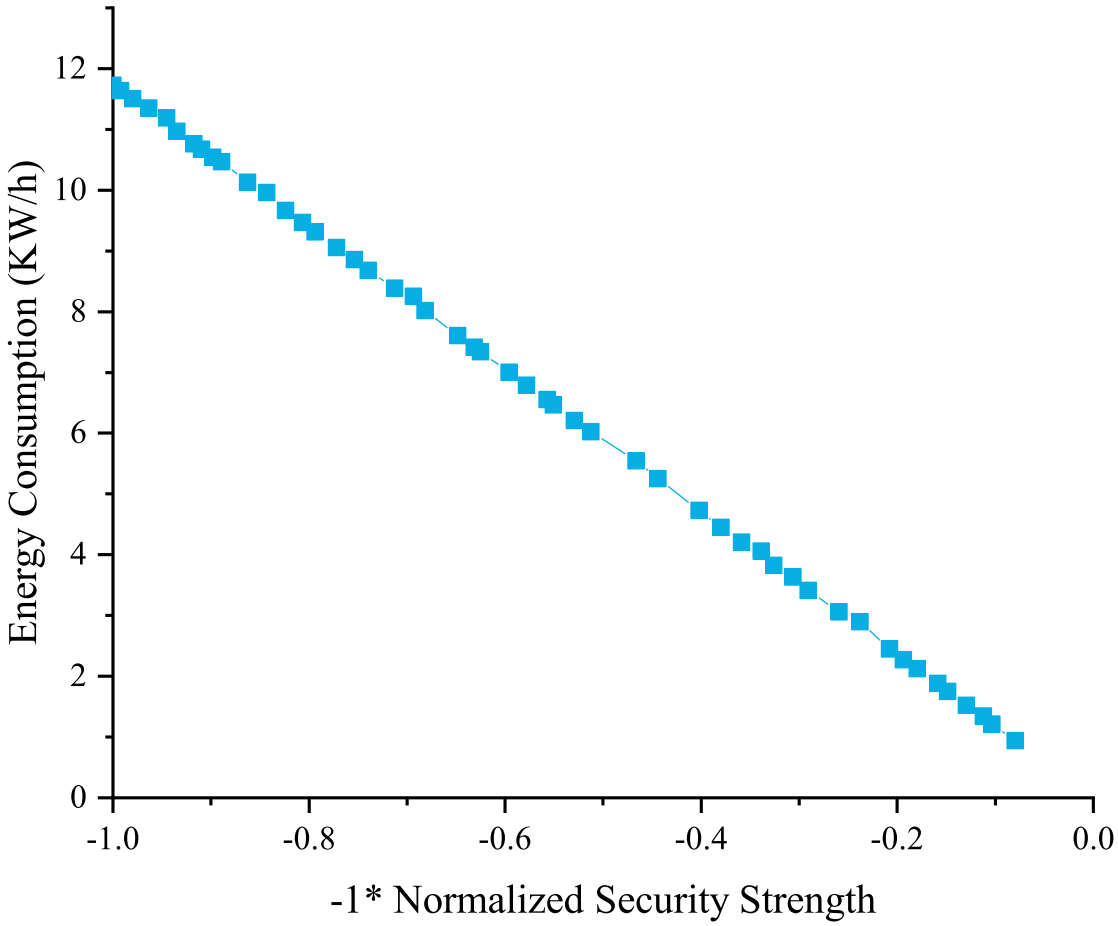


Figure 4.12: Pareto Front of the NSGA-II-SER algorithm of Intel i7-4770 using method from Table 4.7

Fig. 4.15, Fig. 4.16 and Fig. 4.17 unveil the energy consumption of the NSGA-II-SER -enabled and NSGA-II-SER-disabled servers executing the security method outlined in Table 4.4.

After testing these three processors in two experiment groups of security methods, I obtain similar Pareto front and energy consumption comparison trends. The empirical study

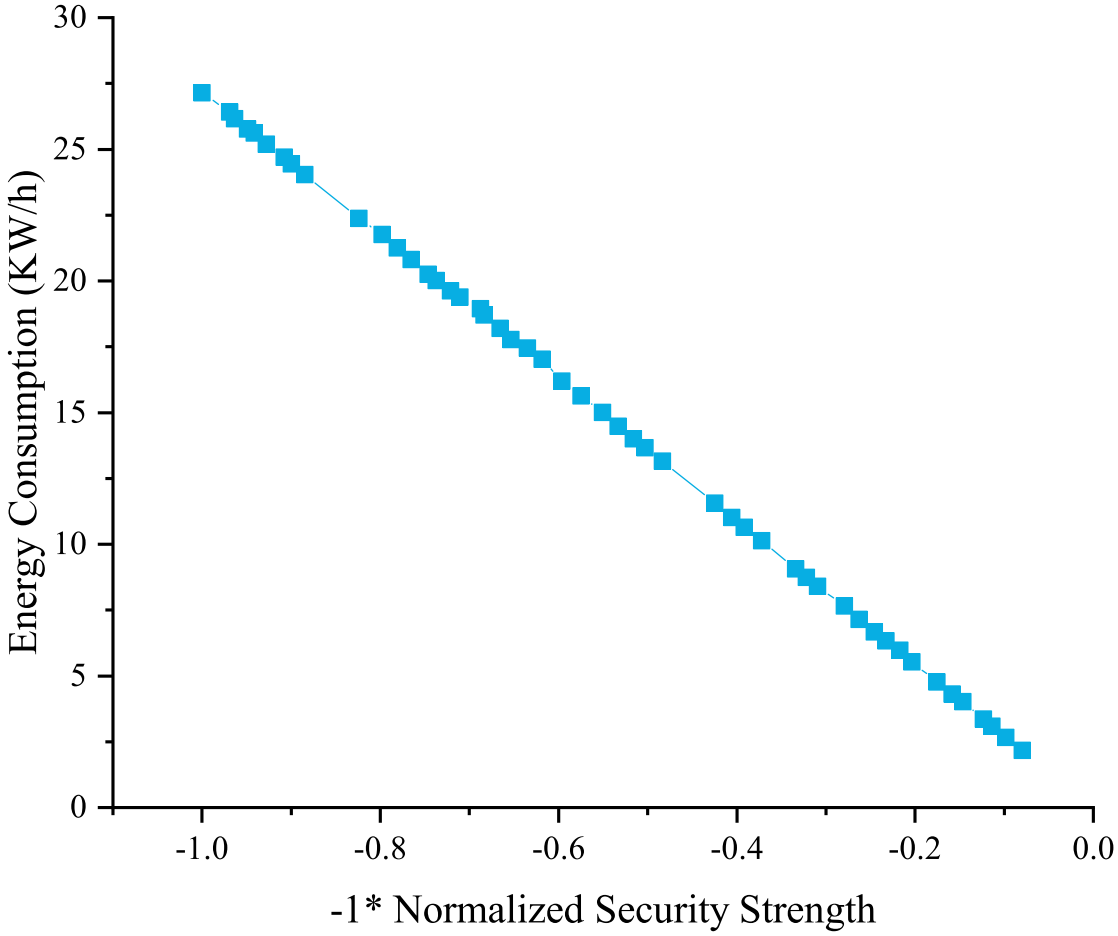


Figure 4.13: Pareto Front of the NSGA-II-SER algorithm of AMD Athlon using method from Table 4.7

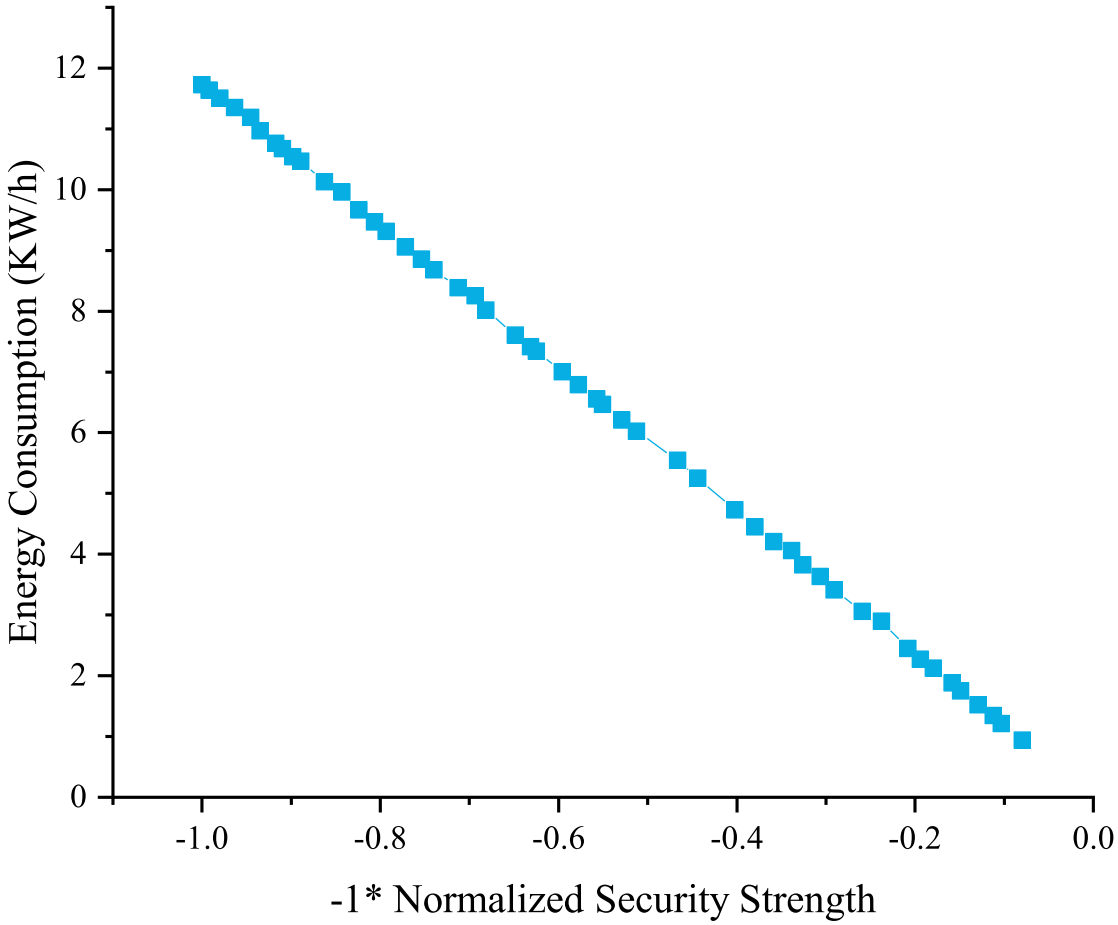


Figure 4.14: Pareto Front of the NSGA-II-SER algorithm of Xeon E5-2670 using method from Table 4.7

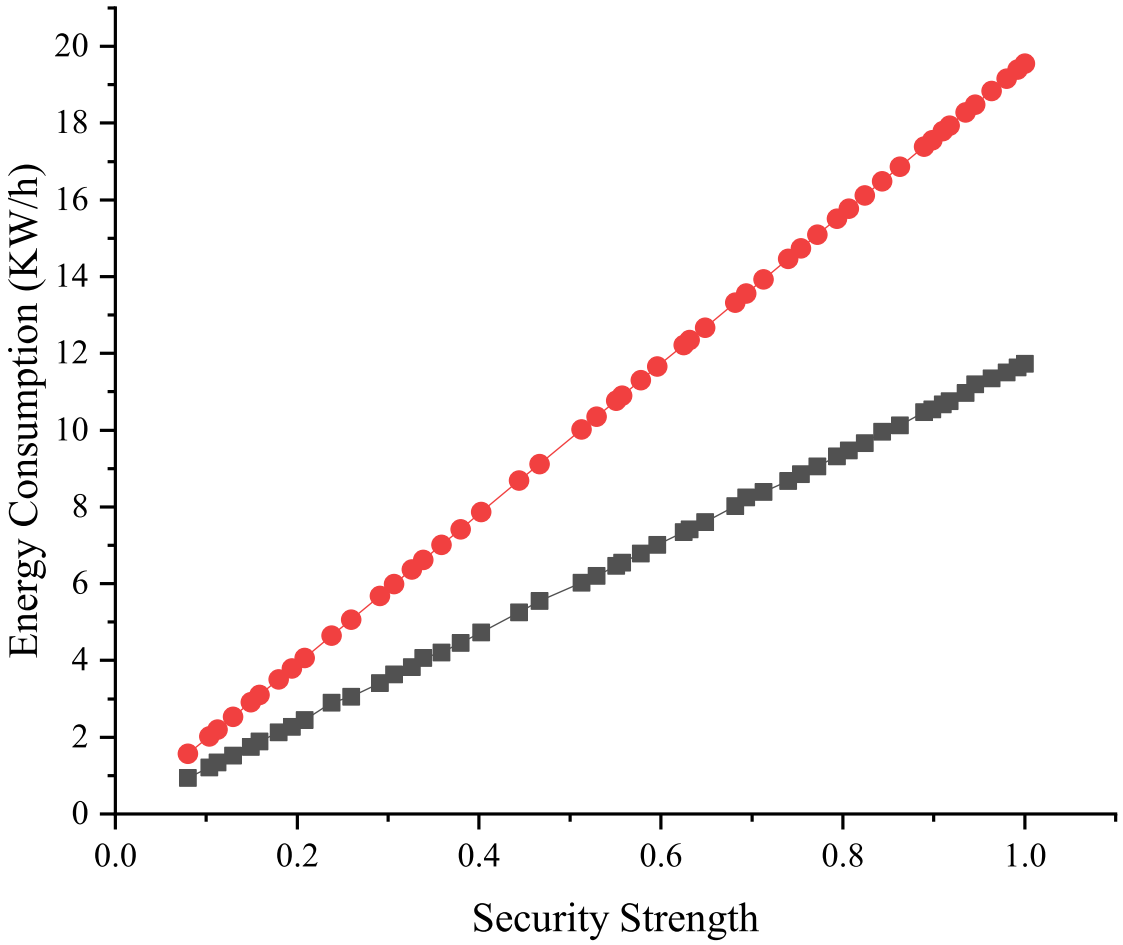


Figure 4.15: Energy consumption comparison for Intel i7-4770 equipped with and without the NSGA-II-SER algorithm using method from Table 4.4

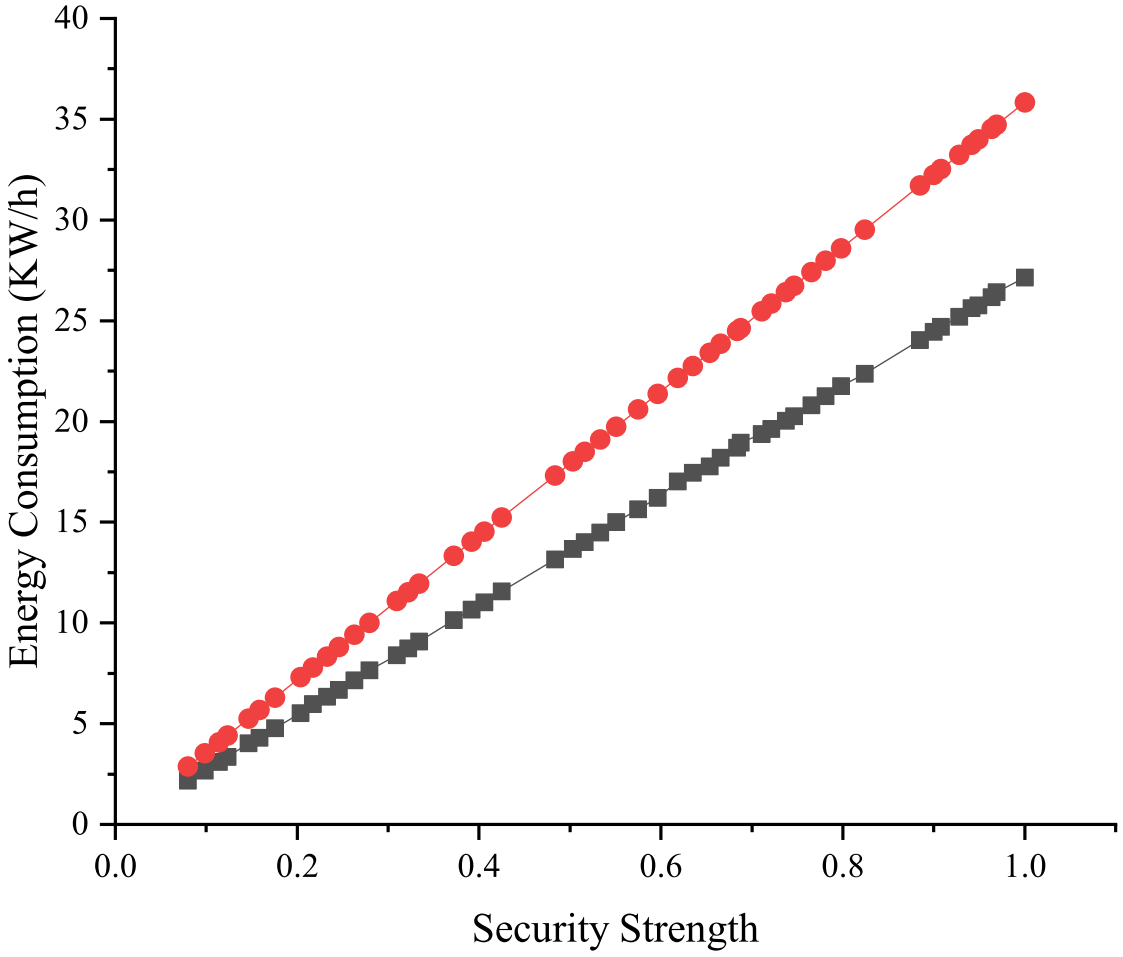


Figure 4.16: Energy consumption comparison for AMD Athlon equipped with and without the NSGA-II-SER algorithm using method from Table 4.4

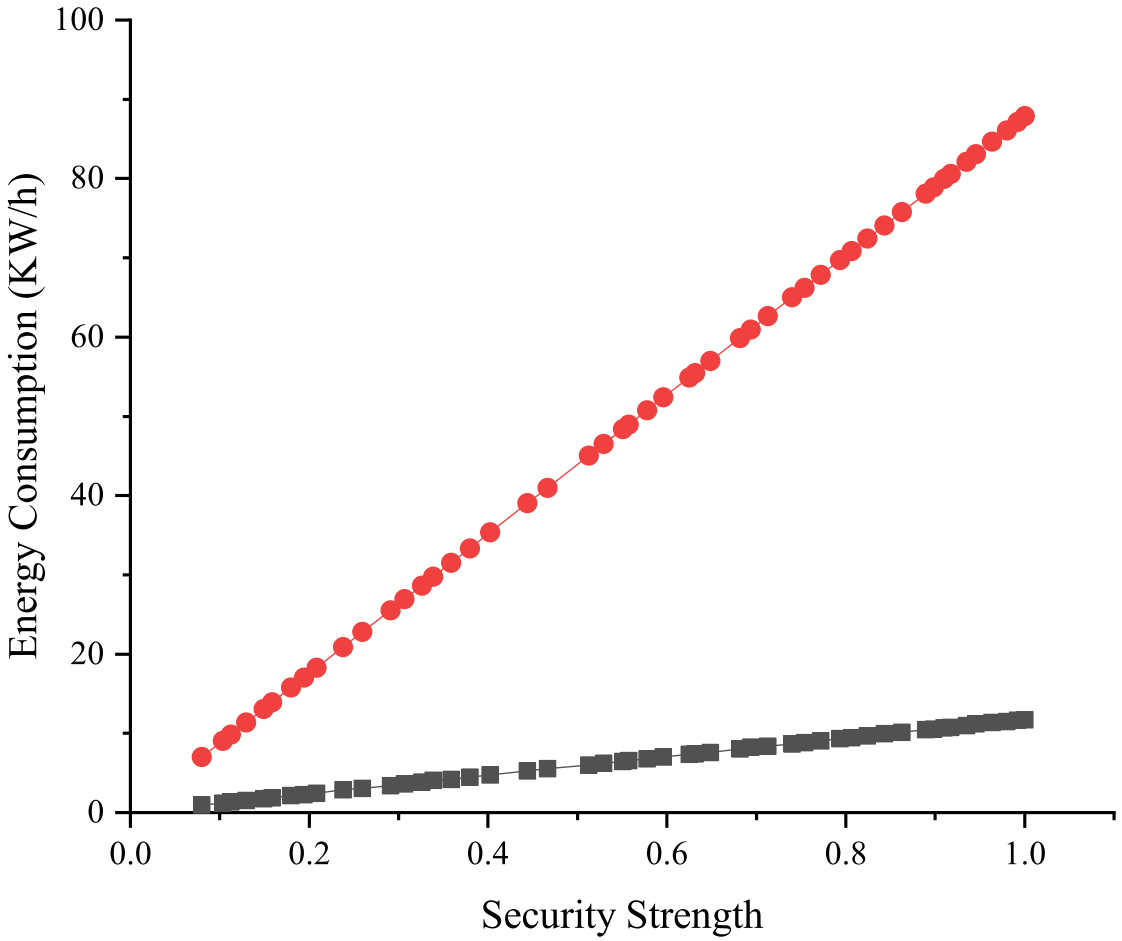


Figure 4.17: Energy consumption comparison for Xeon E5-2670 equipped with and without the NSGA-II-SER algorithm using method from Table 4.4

confirms the efficacy of the NSGA-II-SER approach in terms of energy saving and security protection for cloud datacenters.

4.5 Summary

In this chapter, I proposed the research roadmap towards the security-aware energy management in clouds. The roadmap possesses four connected components: (1) security services, (2) security overhead models, (3) security- and frequency-aware QoS, and (4) security-enabled DVFS. I formulated the SF-DVFS model as a multiple-objective optimization problem to furnish the development of the generic algorithm referred to as *NSGA-II-SER*. My novel energy management system achieves high security and energy efficiency in clouds by seamlessly integrating the security services, a security overhead model, and the security- and frequency-aware DVFS model.

Chapter 5

The blockchain-based privacy protection for the VM consolidation mechanism coupled with the frequency-aware DVFS model

Cloud computing has radically changed the landscape of computing, storage, and communication infrastructures and services. Cloud computing's benefits encompass on-demand capacity, low cost of ownership, and flexible pricing. While moving towards the concept of on-demand services and resource pooling in distributed computing environments, privacy protection becomes a major concern due to the sharing and consolidation features of clouds. At the same time, blockchain is an ideal privacy protection technology characterized by decentralization, transparency, data security, and system autonomy. In this chapter, I investigate privacy controls in blockchain systems. Inspired by modern blockchain and cloud computing techniques, I articulate a research roadmap towards future energy-aware privacy protection mechanisms in clouds. As a case study, I propose a blockchain-based VM consolidation framework accompanied by the DVFS (Dynamic Voltage and Frequency Scaling) technique to offer energy savings and privacy controls in clouds. I expect that the roadmap will open up the potential to develop energy-efficient blockchain-based cloud computing platforms.

The remainder of this chapter is organized as follows. In Section 5.1, I introduce representative blockchain-based privacy protection mechanisms. I present a research roadmap, where new approaches and directions are discussed in Section 5.2, Section 5.3 and Section 5.4. Finally, Section 5.5 presents concluding remarks.

5.1 Privacy of Blockchain Systems

The blockchain technology has emerged as a creative way of maintaining distributed systems thanks to its high efficiency, high data security, and high credibility at a low cost [168]. Blockchain techniques employ a linked block structure to store and verify data, the changes of which are synchronized by a trusted consensus mechanism. A growing number of novel consensus mechanisms catering to cryptocurrencies have been proposed in the past few years. For example, an innovative consensus method was incorporated in the Kraft system to avert multiple hash-rate scenarios [78], thereby offering stable average block times. Sompolinsky and Zohar designed the *GHOST chain selection* rule, which weights branches to speed up selection tasks for miners [136]. The PeerCensus system is capable of maintaining a strong consistency in Bitcoin-like systems [34]. *Discoin*, built atop PeerCensus, enhances consensus efficiency by decoupling block creations from transaction confirmation operations.

The blockchain techniques make it feasible to forge a tamper-proof storage system catering to data storage. It is noteworthy that data privacy challenges may hinder the wide applications of the blockchain technology. For instance, Kosba *et al.* unveiled that blockchain may not guarantee the privacy of transactions because the values of all transactions and balances for a public key are publicly visible [77]. Biryukov *et al.* developed a technique to link user pseudonyms to IP addresses even in case that users are behind firewalls [20]. with this technique in place, clients have potentials to be identified by a set of nodes, which are learned and used to discover a transaction's origin.

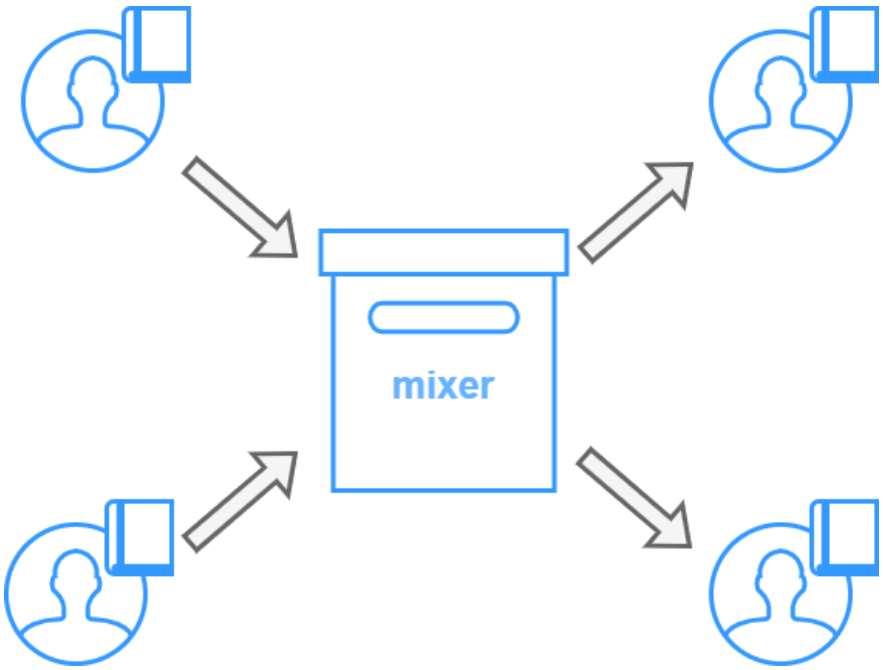
Similar to the privacy-preserving methods for clouds, a raft of privacy protection mechanisms were developed in the arena of the blockchain techniques. Representative mechanisms include, but not limited to, mixing service [27], anonymous signatures [26], and encryption techniques[143].

5.1.1 Mixing Service

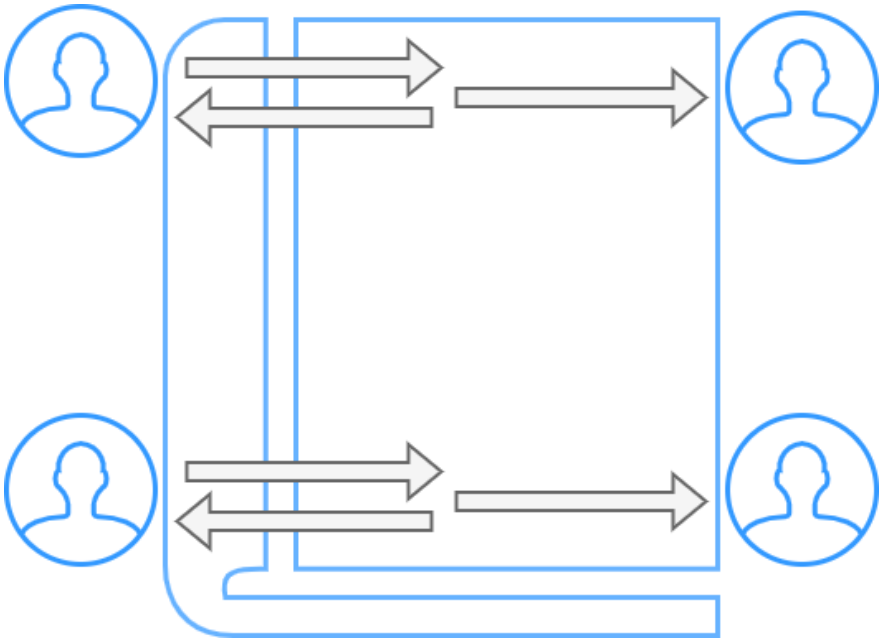
Chaum *et al.* proposed a mechanism of coin mixing [27]. Fig. 5.1 (a) depicts that this mechanism allows privacy-seeking coin users to deliver transactions to a mixer service, which blends a pool of coins to delink a transaction trail. Bonneau *et al.* [21] developed *Mixcoin*, where a central server is in charge of mixing transaction addresses to offer external anonymity. Such a centralized mixing mechanism relies on third-party servers, where dishonest mixers may stealthily archive transaction records or provide poor mixing services. To tackle the thread imposed by untrusted servers, decentralized mixing pattern was proposed (see Fig.5.1 (b)). Because decentralized mixing mechanisms are independent of the credibility of third-party servers, decentralized designs effectively avert third party thefts and leakages of coin mixing information.The decentralized coin mixing mechanism cancels the participation of the third-party coin mixing providers and merges multiple one-to-one transaction records into a many to many transaction record. The attacker cannot directly find the relationship between them. The decentralized approaches also eliminate mixing fees. *CoinJoin* is the earliest decentralized mixing scheme first proposed by Maxwell *et al.* [104].

5.1.2 Anonymous Signatures

Unlike the mixing methods suffer a delay in which participants discover their partners for transactions to be mixed, anonymous signature schemes have strengths in furnishing anonymity for signers. Among anonymous signature schemes, the group signature and ring signature solution are two representative schemes [26][121]. The group signature method, designed by Chaum and Heyst [26], enables group members to set up anonymous signatures on behalf of a group, the managers of which may open signatures when the signature is disputed. A ring signature is constructed by a group member, whose identify is protected from the other members [121]. *Monero* is a successful implementation of the ring signature approach, where ring signatures coupled with hidden addresses to camouflage the linkage between input and output addresses [2].



(a) Mixing transaction



(b) CoinJoin transaction

Figure 5.1: Mixing service mechanisms in blockchain.

5.1.3 Encryption Methods

The homomorphic encryption and attribute-based encryption methods are widely applied in blockchains. Recall that (see Section 2.6.3) homomorphic encryption allows computations to be accomplished on encrypted data without accessing a decryption key. Distributed electronic voting and bidding systems deploy the homomorphic encryption technology to protect data privacy, to enhance the anonymity of participants, and to boost data reliability and verifiability [143]. It is evident that attribute-based encryption is powerful. For example, Lewko *et al.* implemented a decentralized attribute-based encryption scheme on a blockchain [83]. Nevertheless, applications of attribute-based encryption schemes ought to be further explored.

Another cryptographic technology that embraces privacy-preserving properties is *zero knowledge proofs (ZKP)* [56]. A prover in a zero knowledge proof system makes a verifier believe that a message is correct without sending valid information to the verifier. Non-interactive zero-knowledge proof or NIZK is an extension of ZKP, where only a single message is transferred from a prover to a verifier. NIZK was deployed in *Zcash* - a privacy-protecting digital currency system that shields transaction information [128].

All the aforementioned privacy-preserving techniques (mixing, anonymous, and encryption) are tabulated in Table 5.1, which summarizes the strengths and weaknesses of the leading-edge privacy protection techniques.

Table 5.1: Summary of Privacy Techniques on Blockchian.

Techniques	Applications	Advantages	Disadvantages
Mixing	Mixcoin [21]	It can obfuscate users' addresses from being	Cause a Delay waiting to be mixed. High risky on
	CoinJoin [104]	linked.	unprotected transaction content.
Group signature		It is efficient anonymity and revocability.	Need a trusted manager.
Ring signature	Monero [2]	It can hide tradition origin and no need for trusted participant.	The identity of the signer cannot be revealed even in a dispute. The storage overhead is heavy.
Homomorphic encryption		It enable to perform computations on ciphertexts without decrypting data in advance.	Low efficiency for complex functions and no support for auditing.
NIZK	Zcash [128]	It can simultaneously achieve anonymity and transaction privacy.	Heavy computation overhead.

5.2 Energy-aware Privacy Protections in Clouds

5.2.1 Energy Efficiency of Privacy Protections

High energy efficiency and privacy protections are two vital design objectives for cloud computing platforms. In the first phase of my research roadmap, I focus on bringing forth energy-efficient privacy protection techniques in clouds.

Splitting takes constant work to split an original data set into fragments. Extra energy consumption is expected because operations on the fragments lead to additional input/output overhead. Jaikar *et al.* devised a secure data distribution scheme anchored on secret splitting to preserve data privacy over clouds [66]. Although this technique protects sensitive data, the secret splitting technique inevitably increases energy usage due to extra bandwidth and storage utilization.

Most anonymization methods cost energy to generate anonymized data sets. Once anonymized data sets are generated and uploaded to the cloud, no further intervention is required. Any query like search and retrieval on anonymized data incurs no overhead on clouds. When it comes to homomorphic encryption, the complexity of encryption and decryption are normally higher than that of plain encryption. Searchable encryption's energy efficiency lies between those of plain encryption and homomorphic encryption.

5.2.2 Design Issues in Energy-aware Privacy Protection Systems

There are four design issues in developing energy-aware privacy protection systems in clouds. First, I design an energy-efficient data splitting mechanism by storing fragments on energy-aware data storage systems such as *Eco-storage* [11]. Second, energy-efficient anonymization and encryption strategies are expected to become technological underpinnings of energy-aware privacy preserving mechanisms in clouds. Third, it is critical to navigator an approach to making a good trade-off between privacy protection and energy efficiency

in clouds. Last, a key research component is the development of blockchain-based privacy-preserving techniques, which offer high energy efficiency in clouds. Please refer to Section 5.4 for the details.

- *Design Issue 1.* To build energy-efficient data splitting mechanisms by storing fragments on energy-aware data storage systems such as *Eco-storage* [11].
- *Design Issue 2.* To design energy-efficient anonymization and encryption strategies, which expect to become technological underpinnings of energy-aware privacy preserving mechanisms in clouds.
- *Design Issue 3.* To navigator an approach to making a good trade-off between privacy protection and energy efficiency in clouds.
- *Design Issue 4.* To develop blockchain-based privacy-preserving techniques, which offer high energy efficiency in clouds. Please refer to 5.4 for the details.

5.3 Energy-Efficient Blockchains for Privacy Controls

Unlike traditional centralized solutions, the *blockchain* technology securely manages chain data across a distributed and interlinked network of nodes. Blockchains, serving as a tamper-resistant distributed ledger, naturally offer data privacy protections in clouds.

Blockchain-based data provenance provides tamper-proof records, enables the transparency of data accountability in clouds, and enhances data privacy. Very recently, Ali *et al.* demonstrated that the blockchain techniques embrace immutable, deterministic, and public natures that play a vital role in data provenance [13]. The concept of *smart contract* balances data provenance, functionality, and trusted environment, regardless of on-chain or off-chain data storage. Liang *et al.* devised a decentralized and trusted cloud data provenance architecture - *ProvChain* - powered by the blockchain technology [90]. To glean provenance data on storage clouds, ProvChain is slated to detect user operations on cloud files. User's privacy

is protected by ProvChain because users' identities are constructed in a hashed form, where only service providers are authorized to map hashed values to the identities.

Because wasting resources of mining networks becomes a key drawback of blockchain technology, I will explore the following research directions to construct energy-efficient blockchain techniques.

- *Research Direction 2-1.* I will pilot low-energy architecture designs to furnish the development of energy-efficient blockchains to preserve user privacy.

- *Research Direction 2-2.* I intend to explore new ways of enhancing the energy efficiency of consensus mechanisms in blockchains. I will kick off this direction by profiling energy consumption of popular consensus mechanisms on edge computing platforms.

- *Research Direction 2-3.* I plan to extend novel ideas [55] of applying renewable energy to the blockchain techniques. I will evaluate the energy efficiency of blockchain algorithms powered by solar and wind farms.

5.4 Blockchain-based Energy Management for Clouds

In one of my recent studies [99], I proposed a frequency-aware DVFS (Dynamic Voltage/Frequency Scaling) model aiming to conserve energy consumption of tasks imposing QoS requirements. My model advocates for specifying QoS requirements with respect to frequency rather than execution time.

A proposed framework depicted in Fig. 5.2 combines virtual machine (VM) migrations with the DVFS technique to further improve energy efficiency. To protect data during VM migrations and data movement, I propose to make use of blockchain-enabled resource allocation to offer a transparent and trustworthy service on clouds (see also Section 5.3). I promote the blockchain technique as an advanced decentralized structure to avoid privacy leakage during VM migrations while guarding data against malicious tampering.

To facilitate an energy-efficient cloud platform, my *DVFS model* intends to derive an optimal frequency ratio that leads to the minimum energy consumption of each active server while shutting down idle servers. The model makes power management decisions by incorporating the servers' hardware information such as static power P_c^{sta} and maximum dynamic power P_c^{dmax}. The *frequency adjusting module* compares the optimal frequency ratio and an overall minimum frequency requirement to appropriately configure frequency levels to cut back the energy consumption of VMs running on clouds.

I will spearhead this research effort along with the two directions below.

- *Research Direction 3-1.* I will develop a blockcahin-based VM consolidation mechanism accompanied by the DVFS technique to offer energy savings and privacy protection in clouds.

- *Research Direction 3-2.* I will incorporate reinforcement-learning algorithms into my privacy-aware energy management system to optimize the performance of resource allocation in the realm of cloud computing.

5.5 Summary

I introduced in this chapter the blockchain techniques make it feasible to forge a tamper-proof storage system catering to data services on clouds. I surveyed an array of representative mechanisms such as mixing service, anonymous signatures, and encryption techniques. Among all the energy-saving and privacy protection schemes for cloud computing, I shed bright a light on blockchain-based VM consolidation combining DVFS to offer energy savings and privacy protection in clouds.

As the research roadmap towards the privacy-aware energy management in clouds, I proposed three connected research activities: (1) building energy-aware privacy protection services, (2) developing energy-efficient blockchains, and (3) devising blockchain-enabled

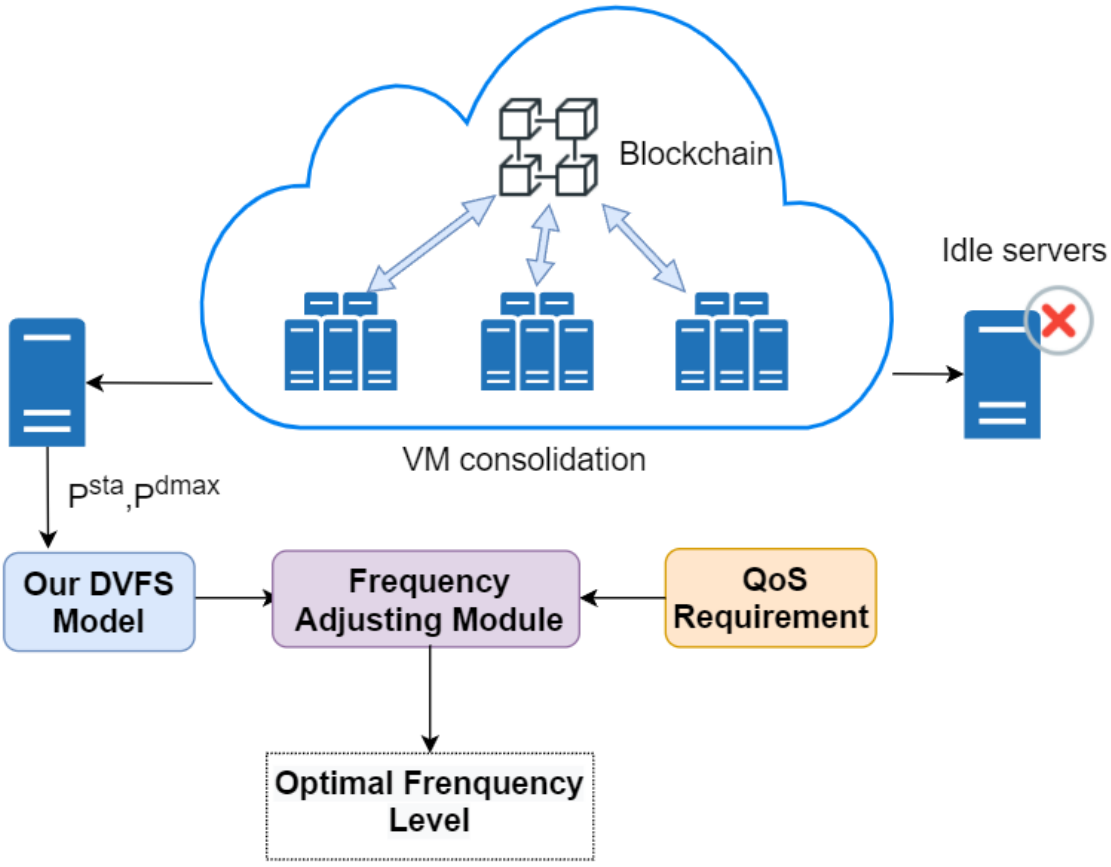


Figure 5.2: The blockchain-based privacy protection for the VM consolidation mechanism coupled with the frequency-aware DVFS model.

energy management modules in clouds. Currently, I am in the process of designing a privacy-aware energy management system for cloud computing environments. my novel energy management system is expected to achieve high privacy and energy efficiency in clouds by seamlessly integrating the blockchain, VM consolidation and frequency-aware DVFS model.

Chapter 6

Conclusions and Future Research Directions

In this dissertation, I proposed a frequency-aware management strategy - a new way of controlling dynamic power and static power of processors running virtual machines in data centers. The frequency-aware model is adept at deriving an optimal frequency ratio that minimizes processors' energy consumption. With my model in place, the energy efficiency of a data center can be maximized by adjusting the processor's frequency to meet the optimal frequency ratio. A management approach was devised to judiciously adjust frequency ratio to conserve energy without violating the frequency requirements imposed by virtual machines. The results demonstrate that my model lays out a solid theoretical foundation catering to the development of power management software in DVFS-enabled clouds. Furthermore, I constructed a high-level design for a security- and frequency-ware DVFS model or *SF-DVFS*, orchestrating security services, security overhead analysis, and DVFS control green cloud computing systems. After proposing the SF-DVFS high-level structure, I developed the NSGA-II-SER algorithm (NSGA-II with security and energy-aware requirements) to optimize both energy efficiency and security protections in cloud data centers. Last but not least, I proposed a blockchain-based VM consolidation framework accompanied by the DVFS technique to offer energy savings and privacy controls in clouds.

In what follows, I first present in Section 6.1 the main contributions made in each chapter. Then, I weigh in on the future research directions in Section 6.2.

6.1 Main Contributions

The scale of data centers has increased dramatically in recent years, and the energy consumption of these large-scale datacenters is truly tremendous. At the same time, cloud-based datacenters, not surprisingly, are becoming a new trend of enterprise data repositories

replacing traditional datacenters. Evidence clearly show that virtual-machine-based cloud computing platforms are technical underpinnings for modern datacenters in the future. Thus, this dissertation research is focused on building energy-efficient cloud data centers.

While moving towards the concept of on-demand services and resource pooling in distributed computing environments, security is a major obstacle for deploying this new leading-edge computing capability. A centralized structure is prone to data leakage of VMs running in cloud data centers when access privileges of some nodes are comprised. Therefore, the projects presented in this dissertation, expected to solve energy efficiency and security problems, facilitate the development of modern cloud-based data centers.

6.1.1 Frequency-aware Management for DVFS-based Clouds

In the first part of this dissertation study, I made the following three contributions.

Contribution 1: Frequency-Aware Quality of Service Unlike the conventional wisdom applying worst-case execution time or WCET as QoS requirements, the frequency model proposed in Chapter 3 advocates for specifying QoS requirements in the form of frequency rather than WCET. The proposed frequency requirements' primary benefit is to avert inaccurate WCET estimates. I demonstrated the feasibility of transforming the traditional time-aware QoS model into a novel frequency-aware QoS model. I showed that the developed model can be applied to govern the power management of DVFS-enabled systems without violating the SLAs of virtual machines. My model derives optimal energy savings for servers without estimating execution times or specifying deadlines.

Contribution 2: A Frequency-aware DVFS Model Given the hardware information of the processor, the frequency-aware DVFS model is able to derive the optimal frequency ratio, which leads to the minimum energy consumption of the processor. A power manager can maximize energy efficiency by adjusting the frequency of the processor to meet the optimal frequency ratio. After analyzing the correlations between frequency ratio and energy consumption, Several intriguing conclusions were drawn. First, the energy-saving

window size is proportional to maximum dynamic power and inversely proportional to static power. Second, the optimal energy-saving frequency ratio is proportional to the static power, meaning that a sizeable static power gives rise to a small amount of saved energy. On the other hand, the optimal frequency ratio is inversely proportional to the maximum dynamic power when the static power is a constant. Third, a small static power proportion leads to high energy-saving performance.

Contribution 3: Applicability and Energy-efficiency Performance My frequency-aware DVFS model, being practical, is ready to be adopted by power management systems. Given a server's static power and maximum dynamic power, the model can control a power manager to achieve high energy efficiency. To shed a bright light on the applicability of my frequency-aware DVFS model, I conducted a simulation study using the parameters from three real-world processors. I also confirmed that the energy-efficiency performance of my model is on par with the well-known utilization-based DVFS scheme. The simulation results show that my model is conducive to projecting the energy-saving performance of DVFS-enabled computing systems running virtual machines.

6.1.2 Security and Frequency Awareness in QoS

By the same token of the frequency-Aware QoS management designed in Chapter 4, security overhead incurred in security-sensitive applications is integrated into WCET measures, which are converted into frequency requirements. I investigated the relation between an overall security-related frequency requirement and each virtual machine's security-related frequency requirement. The security-related frequency requirement of virtual machines running on a physical machine is an accumulated measure of the security-related requirements of all the virtual machines. To meet specified SLA requirements, I argued that one has to regulate the frequency in a way that exceeds a threshold of execution and security requirements.

6.1.3 Security- and Frequency-aware DVFS Modeling and NSGA-II-SER

In the security- and frequency-aware system architecture, the QoS requirement module outputs a minimum frequency requirement from two input parameters, namely, (1) the minimum frequency requirement and (2) the security-related frequency requirement prescribed in virtual machines. My frequency-aware DVFS model incorporates the static and maximum dynamic power constants in this architecture to obtain an optimal frequency ratio. Last but not least, the SF-DVFS model was formulated as a multiple-objective optimization problem to furnish the development of the generic algorithm - NSGA-II-SER.

6.2 Future Projects

As future work directions, I plan to extend my model by combining different energy efficiency and security protection methods. I believe that advanced machine learning algorithm including reinforcement learning could become a game changer for the cloud data center management. The detailed future research projects are articulated below.

6.2.1 Power Management Software

Recall that the first part of this dissertation (see also Chapter 3) lays out a solid theoretical foundation for power management software tailored for DVFS-enabled clouds. There are three future research directions that will further expand this project of the dissertation research. First, I intend to dabble a little bit in the development of a power manager where the frequency-aware model is incorporated to adjust CPU frequencies in a heterogeneous computing environment on clouds. The frequency-aware model will offer the power manager insightful hints on optimal frequency ratios, which boost the energy efficiency of heterogeneous processors. Second, I have a solid plan to dive into the development of a hybrid mechanism that seamlessly blends the DVFS and VM consolidation techniques. Third, thanks to the prominent energy-saving features offered by GPU DVFS, I plan to explore energy conservation techniques in GPU computing systems. I will forge my model to guide

the configuration of frequency ratios minimizing GPU's energy consumption. I also have a desire to push my future directions further by upgrading the model to handle CPU-GPU heterogeneous systems in the face of cloud computing.

6.2.2 Energy-aware Distributed File Systems

From the perspective of green file systems, I plan to study the energy-efficient data management in distributed file system in general and in Hadoop distributed file system or HDFS in particular. In the HDFS system, there are usually three replicas for each data block: when a file is created in HDFS, three copies of the same file are created. I intend to design an energy-efficient HDFS, which will coordinate with my proposed frequency-aware management strategy articulated in Chapter 3. One of a few key component in my new design include a green data transmission mechanism, which will energy efficiently transfer massive amounts of data between clients and HDFS.

6.2.3 Security-aware Energy Management in Clouds

To enhance the security- and frequency-aware system architecture, I advocate for the following future research directions. First, practical VM consolidation and management policies should be blended with DVFS to build energy-efficient clouds running tasks with QoS requirements. Second, machine-learning-based prediction techniques are expected to boost the performance of the VM consolidation and management policy. Third, the security overhead largely depends on security levels. Hence, it is desirable to dynamically configure security levels to fulfill QoS requirements in my proposed security-aware energy management system. For example, if QoS requirements are permitted, security service instances with strong strengths should be elected to maximize security in clouds. Otherwise, security levels must be lowered to avert performance degradation. Developing a security-aware energy management system should incorporate underpinning techniques from multiple areas like

machine learning solutions, DVFS techniques, real-time scheduling, security services, security strength evaluation, and security overhead analysis.

6.2.4 Energy-aware Privacy Protections in Clouds

Now I present future research directions in the arena of developing energy-aware privacy protection systems in clouds. I propose to build energy-efficient data splitting mechanisms by storing fragments on energy-aware data storage systems such as *Eco-storage*. I plan to design energy-efficient anonymization and encryption strategies, which expect to become technological underpinnings of energy-aware privacy preserving mechanisms in clouds; I will navigator an approach to making a good trade-off between privacy protection and energy efficiency in clouds. I will delve in the development of blockchain-based privacy-preserving techniques, which offer high energy efficiency in clouds.

6.2.5 Blockchain-based Energy Management for Clouds

Among all the energy-saving and privacy protection schemes for cloud computing, I shed bright a light on blockchain-based VM consolidation combining DVFS to offer energy savings and privacy protection in clouds. As a future project, a framework will be implemented to combine virtual machine (VM) migrations with the DVFS technique to further improve energy efficiency.

Blockchain-enabled resource allocation will be utilized to offer a transparent and trustworthy service on the cloud. I plan to promote the blockchain technique as an advanced decentralized structure to avoid privacy leakage during VM migrations while guarding data against malicious tampering. Along with this direction, three connected research activities will be undertaken in the future: (1) building energy-aware privacy protection services, (2) developing energy-efficient blockchains, and (3) devising blockchain-enabled energy management modules in clouds.

Furthermore, I will incorporate reinforcement-learning algorithms into privacy-aware energy management systems to optimize the performance of resource allocation in the realm of cloud computing. Thanks to the explosion of the machine learning technology, reinforcement learning is widely used in the field of resource allocation. It is expected that the development of a privacy-aware energy management system should incorporate underpinning techniques from reinforcement learning.

Bibliography

[1] Data center power market size, share trends analysis report by product (pdu, ups, busway), by end use (it telecom, bfsi, energy, healthcare, retail), by region, and segment forecasts, 2019 – 2025.

[2] The Monero Project.

[3] Data Center Market – Global Outlook and Forecast 2018 – 2023. Technical report, Arizton 4577457, 6 2018.

[4] 2019 Outlook for Energy: A perspective to 2040. Technical report, ExxonMobil, 2019.

[5] BP Statistical Review of World Energy 68th edition. Technical report, BP p.l.c., 2019.

[6] Energy Outlook: 2020 edition . Technical report, BP p.l.c., 2020.

[7] S. Abrishami, M. Naghibzadeh, and D. H. Epema. Deadline-constrained workflow scheduling algorithms for infrastructure as a service clouds. *Future Generation Computer Systems*, 29(1):158–169, 2013.

[8] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, and H. Liu. Energy proportional datacenter networks. In *Proceedings of the 37th annual international symposium on Computer architecture*, pages 338–347, 2010.

[9] G. Aggarwal, M. Bawa, P. Ganesan, H. Garcia-Molina, K. Kenthapadi, R. Motwani, U. Srivastava, D. Thomas, and Y. Xu. Two can keep a secret: A distributed architecture for secure database services. *CIDR 2005*, 2005.

[10] N. Akhter and M. Othman. Energy aware resource allocation of cloud data center: review and open issues. *Cluster computing*, 19(3):1163–1182, 2016.

[11] M. M. Al Assaf, X. Jiang, M. R. Abid, and X. Qin. Eco-storage: A hybrid storage system with energy-efficient informed prefetching. *Journal of Signal Processing Systems*, 72(3):165–180, 2013.

[12] P. Alcorn. Intel xeon e5-2600 v4 broadwell-ep review, Mar 2016.

[13] S. Ali, G. Wang, M. Z. A. Bhuiyan, and H. Jiang. Secure data provenance in cloud-centric internet of things via blockchain smart contracts. In *2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pages 991–998. IEEE, 2018.

[14] F. Armknecht, C. Boyd, C. Carr, K. Gjøsteen, A. Jäschke, C. A. Reuter, and M. Strand. A guide to fully homomorphic encryption. *IACR Cryptol. ePrint Arch.*, 2015:1192, 2015.

[15] P. Arroba, J. M. Moya, J. L. Ayala, and R. Buyya. Dynamic voltage and frequency scaling-aware dynamic consolidation of virtual machines for energy efficient cloud data centers. *Concurrency and Computation: Practice and Experience*, 29(10):e4067, 2017.

[16] H. Aydin and Q. Yang. Energy-aware partitioning for multiprocessor real-time systems. In *Proceedings International Parallel and Distributed Processing Symposium*, pages 9–pp. IEEE, 2003.

[17] V. Balasaraswathi and S. Manikandan. Enhanced security for multi-cloud storage using cryptographic data splitting with dynamic approach. In *2014 IEEE International Conference on Advanced Communications, Control and Computing Technologies*, pages 1190–1194. IEEE, 2014.

[18] L. A. Barroso, J. Clidaras, and U. Hölzle. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Second Edition*. 2013.

[19] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. 2007.

[20] A. Biryukov, D. Khovratovich, and I. Pustogarov. Deanonymisation of clients in bitcoin p2p network. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 15–29, 2014.

[21] J. Bonneau, A. Narayanan, A. Miller, J. Clark, J. A. Kroll, and E. W. Felten. Mixcoin: Anonymity for bitcoin with accountable mixes. In *International Conference on Financial Cryptography and Data Security*, pages 486–504. Springer, 2014.

[22] D. Borthakur, R. Schmidt, R. Vadali, S. Chen, and P. Kling. HDFS RAID, 2010. Technical Talk. Yahoo! Developer Network.

[23] V. Brilliantova and T. W. Thurner. Blockchain and the future of energy. *Technology in Society*, 57:38–45, 2019.

[24] T. D. Burd and R. W. Brodersen. Energy efficient cmos microprocessor design. In *Proceedings of the Twenty-Eighth Annual Hawaii International Conference on System Sciences*, volume 1, pages 288–297. IEEE, 1995.

[25] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen. Low-power cmos digital design. *IEICE Transactions on Electronics*, 75(4):371–382, 1992.

[26] D. Chaum and E. Van Heyst. Group signatures. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 257–265. Springer, 1991.

[27] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981.

[28] H. Chen, X. Zhu, H. Guo, J. Zhu, X. Qin, and J. Wu. Towards energy-efficient scheduling for real-time tasks under uncertain cloud computing environment. *Journal of Systems and Software*, 99:20–35, 2015.

[29] H. Chen, X. Zhu, D. Qiu, L. Liu, and Z. Du. Scheduling for workflows with security-sensitive intermediate data by selective tasks duplication in clouds. *IEEE Transactions on Parallel and distributed systems*, 28(9):2674–2688, 2017.

[30] R. Chen, B. C. Fung, N. Mohammed, B. C. Desai, and K. Wang. Privacy-preserving trajectory data publishing by local suppression. *Information Sciences*, 231:83–97, 2013.

[31] Y. Chen, J.-F. Martínez, P. Castillejo, and L. López. A privacy-preserving noise addition data aggregation scheme for smart grid. *Energies*, 11(11):2972, 2018.

[32] V. Ciriani, S. D. C. Di Vimercati, S. Foresti, and P. Samarati. Microdata protection. In *Secure data management in decentralized systems*, pages 291–321. Springer, 2007.

[33] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[34] C. Decker, J. Seidel, and R. Wattenhofer. Bitcoin meets strong consistency. In *Proceedings of the 17th International Conference on Distributed Computing and Networking*, pages 1–10, 2016.

[35] Q. Deng, D. Meisner, A. Bhattacharjee, T. F. Wenisch, and R. Bianchini. Coscale: Coordinating cpu and memory system dvfs in server systems. In *2012 45th annual IEEE/ACM international symposium on microarchitecture*, pages 143–154. IEEE, 2012.

[36] H. Dev, T. Sen, M. Basak, and M. E. Ali. An approach to protect the privacy of cloud data from data mining based attacks. In *2012 SC Companion: High Performance Computing, Networking Storage and Analysis*, pages 1106–1115. IEEE, 2012.

[37] J. Domingo-Ferrer, O. Farras, J. Ribes-González, and D. Sánchez. Privacy-preserving cloud computing on sensitive data: A survey of methods, products and challenges. *Computer Communications*, 140:38–60, 2019.

[38] Z. Dong, N. Liu, and R. Rojas-Cessa. Greedy scheduling of tasks with time constraints for energy-efficient cloud-computing data centers. *Journal of Cloud Computing*, 4(1):1–14, 2015.

[39] B. Dorronsoro, S. Nesmachnow, J. Taheri, A. Y. Zomaya, E.-G. Talbi, and P. Bouvry. A hierarchical approach for energy-efficient scheduling of large workloads in multicore distributed systems. *Sustainable Computing: Informatics and Systems*, 4(4):252–261, 2014.

[40] K. Duan, S. Fong, W. Song, A. V. Vasilakos, and R. Wong. Energy-aware cluster reconfiguration algorithm for the big data analytics platform spark. *Sustainability*, 9(12):2357, 2017.

[41] E. M. Elnozahy, M. Kistler, and R. Rajamony. Energy-efficient server clusters. In *International Workshop on Power-Aware Computer Systems*, pages 179–197. Springer, 2002.

[42] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *ACM SIGARCH computer architecture news*, volume 35, pages 13–23. ACM, 2007.

[43] Q. Fettes, M. Clark, R. Bunescu, A. Karanth, and A. Louri. Dynamic voltage and frequency scaling in nocs with supervised and reinforcement learning techniques. *IEEE Transactions on Computers*, 68(3):375–389, 2018.

[44] S. Fluhrer, I. Mantin, and A. Shamir. Weaknesses in the key scheduling algorithm of rc4. In *International Workshop on Selected Areas in Cryptography*, pages 1–24. Springer, 2001.

[45] D. Ford, F. Labelle, F. Popovici, M. Stokely, V. Truong, L. Barroso, C. Grimes, and S. Quinlan. Availability in Globally Distributed Storage Systems. In *Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation (OSDI'10)*, pages 61–74. USENIX, 2010.

[46] K. Gai, M. Qiu, and H. Zhao. Security-aware efficient mass distributed storage approach for cloud systems in big data. In *2016 IEEE 2Nd international conference on big data security on cloud (bigdatasecurity), IEEE international conference on high performance and smart computing (HPSC), and IEEE international conference on intelligent data and security (IDS)*, pages 140–145. IEEE, 2016.

[47] G.-n. Gan, T.-l. Huang, and S. Gao. Genetic simulated annealing algorithm for task scheduling based on cloud computing environment. In *2010 International Conference on Intelligent Computing and Integrated Systems*, pages 60–63. IEEE, 2010.

[48] V. Ganapathy, D. Thomas, T. Feder, H. Garcia-Molina, and R. Motwani. Distributing data for secure database services. In *Proceedings of the 4th International Workshop on Privacy and Anonymity in the Information Society*, pages 1–10, 2011.

[49] Y. Gao, H. Guan, Z. Qi, B. Wang, and L. Liu. Quality of service aware power management for virtualized data centers. *Journal of Systems Architecture*, 59(4-5):245–259, 2013.

[50] S. K. Garg, C. S. Yeo, A. Anandasivam, and R. Buyya. Environment-conscious scheduling of hpc applications on distributed cloud-oriented data centers. *Journal of Parallel and Distributed Computing*, 71(6):732–749, 2011.

[51] R. Ge, X. Feng, and K. W. Cameron. Performance-constrained distributed dvs scheduling for scientific applications on power-aware clusters. In *SC'05: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing*, pages 34–34. IEEE, 2005.

[52] Y. Ge and G. Wei. Ga-based task scheduler for the cloud computing systems. In *2010 International Conference on Web Information Systems and Mining*, volume 2, pages 181–186. IEEE, 2010.

[53] C. Gentry. *A fully homomorphic encryption scheme*. Stanford university, 2009.

[54] N. K. Gill and S. Singh. A dynamic, cost-aware, optimized data replication strategy for heterogeneous cloud data centers. *Future Generation Computer Systems*, 65:10–32, 2016.

[55] N. Gogerty and J. Zitoli. Deko–currency proposal using a portfolio of electricity linked assets. *Available at SSRN 1802166*, 2011.

[56] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 18(1):186–208, 1989.

[57] I.-T. R. Group et al. Top 10 energy-saving tips for a greener data center. 2007.

[58] K. He, J. Weng, J.-N. Liu, J. K. Liu, W. Liu, and R. H. Deng. Anonymous identity-based broadcast encryption with chosen-ciphertext security. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, pages 247–255, 2016.

[59] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown. Elastictree: Saving energy in data center networks. In *Nsdi*, volume 10, pages 249–264, 2010.

[60] I. Hong, D. Kirovski, G. Qu, M. Potkonjak, and M. B. Srivastava. Power optimization of variable-voltage core-based systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(12):1702–1714, 1999.

[61] C.-H. Hsu, K. D. Slagter, S.-C. Chen, and Y.-C. Chung. Optimizing energy consumption with task consolidation in clouds. *Information Sciences*, 258:452–462, 2014.

[62] J. Huang, P. Zhou, X. Qin, Y. Wang, C. Xie, and J. Jose. Optimizing erasure-coded data archival for replica-based storage clusters. *The Computer Journal*, 62(2):247–262, 2019.

[63] S. Ibrahim, T.-D. Phan, A. Carpen-Amarie, H.-E. Chihoub, D. Moise, and G. Antoniu. Governing energy consumption in hadoop through cpu frequency scaling: An analysis. *Future Generation Computer Systems*, 54:219–232, 2016.

[64] G. Idex. Cisco global cloud index: Forecast and methodology, 2016–2021,". *Cisco, San Jose, CA, USA, White Paper 1513879861264127, Şubat*, 2018.

[65] W. Itani, A. Kayssi, and A. Chehab. Privacy as a service: Privacy-aware data storage and processing in cloud computing architectures. In *2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing*, pages 711–716. IEEE, 2009.

[66] S. P. Jaikar, R. C. Maheshwar, S. P. Mamadapure, and A. A. Bhosle. Secure data distribution using secret splitting over cloud. *Global Journal of Computer Science and Technology*, 2017.

[67] R. Jain, D. Molnar, and Z. Ramzan. Towards understanding algorithmic factors affecting energy consumption: switching complexity, randomness, and preliminary experiments. In *Proceedings of the 2005 joint workshop on Foundations of mobile computing*, pages 70–79. ACM, 2005.

[68] K. Jiang, A. Lifa, P. Eles, Z. Peng, and W. Jiang. Energy-aware design of secure multimode real-time embedded systems with fpga co-processors. In *Proceedings of the 21st International conference on Real-Time Networks and Systems*, pages 109–118, 2013.

[69] W. Jiang, F. Liu, G. Tang, K. Wu, and H. Jin. Virtual machine power accounting with shapley value. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 1683–1693. IEEE, 2017.

[70] S. Kamara and C. Papamanthou. Parallel and dynamic searchable symmetric encryption. In *International conference on financial cryptography and data security*, pages 258–274. Springer, 2013.

[71] K. C. Karki and S. V. Patankar. Airflow distribution through perforated tiles in raised-floor data centers. *Building and environment*, 41(6):734–744, 2006.

[72] A. Kawachi, K. Tanaka, and K. Xagawa. Multi-bit cryptosystems based on lattice problems. In *International Workshop on Public Key Cryptography*, pages 315–329. Springer, 2007.

[73] A. N. Khan, M. M. Kiah, M. Ali, S. A. Madani, S. Shamshirband, et al. Bss: block-based sharing scheme for secure data storage services in mobile cloud environment. *The Journal of Supercomputing*, 70(2):946–976, 2014.

[74] K. H. Kim, R. Buyya, and J. Kim. Power aware scheduling of bag-of-tasks applications with deadline constraints on dvs-enabled clusters. In *Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid'07)*, pages 541–548. IEEE, 2007.

[75] W. Kim, M. S. Gupta, G.-Y. Wei, and D. Brooks. System level analysis of fast, per-core dvfs using on-chip switching regulators. In *2008 IEEE 14th International Symposium on High Performance Computer Architecture*, pages 123–134. IEEE, 2008.

[76] F. Kong, W. Yi, and Q. Deng. Energy-efficient scheduling of real-time tasks on cluster-based multicores. In *2011 Design, Automation & Test in Europe*, pages 1–6. IEEE, 2011.

[77] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *2016 IEEE symposium on security and privacy (SP)*, pages 839–858. IEEE, 2016.

[78] D. Kraft. Difficulty control for blockchain-based consensus systems. *Peer-to-peer Networking and Applications*, 9(2):397–413, 2016.

[79] P. Kumar and A. Verma. Independent task scheduling in cloud computing by improved genetic algorithm. *International Journal of Advanced Research in Computer Science and Software Engineering*, 2(5), 2012.

[80] Y.-K. Kwok and I. Ahmad. On multiprocessor task scheduling using efficient state space search approaches. *Journal of Parallel and Distributed Computing*, 65(12):1515–1532, 2005.

[81] R. Latif, H. Abbas, S. Assar, and Q. Ali. Cloud computing risk assessment: a systematic literature review. In *Future information technology*, pages 285–295. Springer, 2014.

[82] Y. J. Lee, P. K. Singh, and P. S. Lee. Fluid flow and heat transfer investigations on enhanced microchannel heat sink using oblique fins with parametric study. *International Journal of Heat and Mass Transfer*, 81:325–336, 2015.

[83] A. Lewko and B. Waters. Decentralizing attribute-based encryption. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 568–588. Springer, 2011.

[84] D. Li and J. Wu. Energy-aware scheduling for aperiodic tasks on multi-core processors. In *2014 43rd International Conference on Parallel Processing*, pages 361–370. IEEE, 2014.

[85] J. Li, X. Chen, M. Li, J. Li, P. P. Lee, and W. Lou. Secure deduplication with efficient and reliable convergent key management. *IEEE transactions on parallel and distributed systems*, 25(6):1615–1625, 2013.

[86] J. Li, Z. Li, K. Ren, and X. Liu. Towards optimal electric demand management for internet data centers. *IEEE Transactions on Smart Grid*, 3(1):183–192, 2011.

[87] K. Li. Design and analysis of heuristic algorithms for power-aware scheduling of precedence constrained tasks. In *2011 IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum*, pages 804–813. IEEE, 2011.

[88] L. Li, T. Gu, L. Chang, Z. Xu, Y. Liu, and J. Qian. A ciphertext-policy attribute-based encryption based on an ordered binary decision diagram. *IEEE Access*, 5:1137–1145, 2017.

[89] X. Li, P. Garraghan, X. Jiang, Z. Wu, and J. Xu. Holistic virtual machine scheduling in cloud datacenters towards minimizing total energy. *IEEE Transactions on Parallel and Distributed Systems*, 29(6):1317–1331, 2017.

[90] X. Liang, S. Shetty, D. Tosh, C. Kamhoua, K. Kwiat, and L. Njilla. Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. In *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, pages 468–477. IEEE, 2017.

[91] X. Lin, Y. Xue, P. Bogdan, Y. Wang, S. Garg, and M. Pedram. Power-aware virtual machine mapping in the data-center-on-a-chip paradigm. In *2016 IEEE 34th International Conference on Computer Design (ICCD)*, pages 241–248. IEEE, 2016.

[92] D. Liu and N. Han. An energy-efficient task scheduler in virtualized cloud platforms. *International Journal of Grid and Distributed Computing*, 7(3):123–134, 2014.

[93] F. Liu, Z. Zhou, H. Jin, B. Li, B. Li, and H. Jiang. On arbitrating the power-performance tradeoff in saas clouds. *IEEE Transactions on Parallel and Distributed Systems*, 25(10):2648–2658, 2013.

[94] W. Liu, S. Peng, W. Du, W. Wang, and G. S. Zeng. Security-aware intermediate data placement strategy in scientific cloud workflows. *Knowledge and information systems*, 41(2):423–447, 2014.

[95] S. K. Madria and B. Bhargava. A transaction model to improve data availability in mobile computing. *Distributed and Parallel Databases*, 10(2):127–160, 2001.

[96] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski. Algorithms for cost-and deadline-constrained provisioning for scientific workflow ensembles in iaas clouds. *Future Generation Computer Systems*, 48:1–18, 2015.

[97] S. Malik and F. Huet. Adaptive fault tolerance in real time cloud computing. In *2011 IEEE World Congress on services*, pages 280–287. IEEE, 2011.

[98] M. Manasse, C. Thekkath, and A. Silverberg. A Reed-solomon Code for Disk Storage, and Efficient Recovery Computations for Erasure-coded Disk Storage. *Proceeding in Informatics*, pages 1–11, 2009.

[99] J. Mao, T. Bhattacharya, X. Peng, T. Cao, and X. Qin. Modeling energy consumption of virtual machines in dvfs-enabled cloud data centers. In *2020 39th IEEE International Performance Computing and Communications Conference*. IEEE, 2020.

[100] A. Marashi. Power hungry: The growing energy demands of data centers. https://www.vxchnge.com/blog/power-hungry-the-growing-energy-demands-of-data-centers, 2019.

[101] S. Maroulis, N. Zacheilas, and V. Kalogeraki. A framework for efficient energy scheduling of spark workloads. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 2614–2615. IEEE, 2017.

[102] S. Martínez, D. Sánchez, and A. Valls. Towards k-anonymous non-numerical data via semantic resampling. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 519–528. Springer, 2012.

[103] T. Mathew, K. C. Sekaran, and J. Jose. Study and analysis of various task scheduling algorithms in the cloud computing environment. In *2014 International conference on advances in computing, communications and informatics (ICACCI)*, pages 658–664. IEEE, 2014.

[104] G. Maxwell. Coinjoin: Bitcoin privacy for the real world. In *Post on Bitcoin forum*, 2013.

[105] J. Mei, K. Li, A. Ouyang, and K. Li. A profit maximization scheme with guaranteed quality of service in cloud computing. *IEEE Transactions on Computers*, 64(11):3064–3078, 2015.

[106] S. K. Mishra, D. Puthal, B. Sahoo, P. P. Jayaraman, S. Jun, A. Y. Zomaya, and R. Ranjan. Energy-efficient vm-placement in cloud data center. *Sustainable computing: informatics and systems*, 20:48–55, 2018.

[107] M. Mitchell, J. Holland, and S. Forrest. When will a genetic algorithm outperform hill climbing. *Advances in neural information processing systems*, 6, 1993.

[108] M. R. Mohamed and M. H. Awadalla. Hybrid algorithm for multiprocessor task scheduling. *International Journal of Computer Science Issues (IJCSI)*, 8(3):79, 2011.

[109] C. Nadjahi, H. Louahlia, and S. Lemasson. A review of thermal management and innovative cooling strategies for data center. *Sustainable Computing: Informatics and Systems*, 19:14–28, 2018.

[110] C. M. O'Keefe and D. B. Rubin. Individual privacy versus public good: protecting confidentiality in health research. *Statistics in medicine*, 34(23):3081–3103, 2015.

[111] openbenchmarking. cpufreq-info - intel core i7-4770.

[112] A. Pahlevan, Y. M. Qureshi, M. Zapater, A. Bartolini, D. Rossi, L. Benini, and D. Atienza. Energy proportionality in near-threshold computing servers and cloud data centers: Consolidating or not? In *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 147–152. IEEE, 2018.

[113] M. P. Papazoglou. Service-oriented computing: Concepts, characteristics and directions. In *Proceedings of the Fourth International Conference on Web Information Systems Engineering, 2003. WISE 2003.*, pages 3–12. IEEE, 2003.

[114] E. Pinheiro and R. Bianchini. Energy conservation techniques for disk array-based servers. In *Proceedings of the 18th annual international conference on Supercomputing*, pages 68–78, 2004.

[115] X. Qin, M. Alghamdi, M. Nijim, Z. Zong, and K. Bellam. Scheduling of periodic packets in energy-aware wireless networks. In *2007 IEEE International Performance, Computing, and Communications Conference*, pages 210–217. IEEE, 2007.

[116] M. Qiu, L. Zhang, Z. Ming, Z. Chen, X. Qin, and L. T. Yang. Security-aware optimization for ubiquitous computing systems with seat graph approach. *Journal of Computer and System Sciences*, 79(5):518–529, 2013.

[117] A. Quintiliani, M. Chinnici, and D. De Chiara. Understanding "workload-related" metrics for energy efficiency in data center. In *2016 20th International Conference on System Theory, Control and Computing (ICSTCC)*, pages 830–837. IEEE, 2016.

[118] Y. Rahulamathavan, S. Veluru, J. Han, F. Li, M. Rajarajan, and R. Lu. User collusion avoidance scheme for privacy-preserving decentralized key-policy attribute-based encryption. *IEEE Transactions on Computers*, 65(9):2939–2946, 2015.

[119] D. Reinsel, J. Gantz, and J. Rydning. The digitization of the world: from edge to core. *Framingham: International Data Corporation*, 2018.

[120] R. L. Rivest, L. Adleman, M. L. Dertouzos, et al. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

[121] R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 552–565. Springer, 2001.

[122] T. G. Rodrigues, K. Suto, H. Nishiyama, and N. Kato. Hybrid method for minimizing service delay in edge cloud computing through vm migration and transmission power control. *IEEE Transactions on Computers*, 66(5):810–819, 2016.

[123] M. Rodriguez-Garcia, M. Batet, and D. Sanchez. Utility-preserving privacy protection of nominal data sets via semantic rank swapping. *Information Fusion*, 45:282–295, 2019.

[124] S. Roy, A. Rudra, and A. Verma. An energy complexity model for algorithms. In *Proceedings of the 4th conference on Innovations in Theoretical Computer Science*, pages 283–304. ACM, 2013.

[125] X. Ruan, X. Qin, Z. Zong, K. Bellam, and M. Nijim. An energy-efficient scheduling algorithm using dynamic voltage scaling for parallel applications on clusters. In *2007 16th International Conference on Computer Communications and Networks*, pages 735–740. IEEE, 2007.

[126] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 457–473. Springer, 2005.

[127] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. 1998.

[128] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *2014 IEEE Symposium on Security and Privacy*, pages 459–474. IEEE, 2014.

[129] B. Schneier. *Applied cryptography: protocols, algorithms, and source code in C*. john wiley & sons, 2007.

[130] B. Schneier and D. Whiting. Fast software encryption: Designing encryption algorithms for optimal software speed on the intel pentium processor. In *International Workshop on Fast Software Encryption*, pages 242–259. Springer, 1997.

[131] Servethehome. Intel core i7-4770 power consumption.

[132] S. Shin, Y. Kim, and S. Lee. Deadline-guaranteed scheduling algorithm with improved resource utilization for cloud computing. In *2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*, pages 814–819. IEEE, 2015.

[133] E. A. Silk, E. L. Golliher, and R. P. Selvam. Spray cooling heat transfer: technology overview and assessment of future challenges for micro-gravity application. *Energy Conversion and Management*, 49(3):453–468, 2008.

[134] A. Singla, A. Singh, K. Ramachandran, L. Xu, and Y. Zhang. Proteus: a topology malleable data center network. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, pages 1–6, 2010.

[135] G. Sivathanu, C. P. Wright, and E. Zadok. Ensuring data integrity in storage: Techniques and applications. In *Proceedings of the 2005 ACM workshop on Storage security and survivability*, pages 26–36, 2005.

[136] Y. Sompolinsky and A. Zohar. Accelerating bitcoin's transaction processing. fast money grows on trees, not chains (2013). *URL https://eprint. iacr. org/2013/881*, 2018.

[137] M. Song. Minimizing power consumption in video servers by the combined use of solid-state disks and multi-speed disks. *IEEE Access*, 6:25737–25746, 2018.

[138] S. L. Song, K. Barker, and D. Kerbyson. Unified performance and power modeling of scientific workloads. In *Proceedings of the 1st International Workshop on Energy Efficient Supercomputing*, page 4. ACM, 2013.

[139] J. Soria-Comas, J. Domingo-Ferrer, D. Sanchez, and S. Martinez. t-closeness through microaggregation: Strict privacy with enhanced utility preservation. *IEEE Transactions on Knowledge and Data Engineering*, 27(11):3098–3110, 2015.

[140] D. Suleiman, M. Ibrahim, and I. Hamarash. Dynamic voltage frequency scaling (dvfs) for microprocessors power and energy reduction. In *4th International Conference on Electrical and Electronics Engineering*, volume 12, 2005.

[141] C.-J. Tang and M.-R. Dai. Dynamic computing resource adjustment for enhancing energy efficiency of cloud service data centers. In *2011 IEEE/SICE International Symposium on System Integration (SII)*, pages 1159–1164. IEEE, 2011.

[142] Z. Tang, L. Jiang, J. Zhou, K. Li, and K. Li. A self-adaptive scheduling algorithm for reduce start time. *Future Generation Computer Systems*, 43:51–60, 2015.

[143] R. Tso, Z.-Y. Liu, and J.-H. Hsiao. Distributed e-voting and e-bidding systems based on smart contract. *Electronics*, 8(4):422, 2019.

[144] B. M. Tudor and Y. M. Teo. On understanding the energy consumption of arm-based multicore servers. *ACM SIGMETRICS Performance Evaluation Review*, 41(1):267–278, 2013.

[145] A. Vafamehr and M. E. Khodayar. Energy-aware cloud computing. *The Electricity Journal*, 31(2):40–49, 2018.

[146] S. B. Vaghani. Virtual machine file system. *ACM SIGOPS Operating Systems Review*, 44(4):57–70, 2010.

[147] A. Vonderau. Scaling the cloud: Making state and infrastructure in sweden. *Ethnos*, 84(4):698–718, 2019.

[148] L. Wang, J. Tao, M. Kunze, A. C. Castellanos, D. Kramer, and W. Karl. Scientific cloud computing: Early definition and experience. In *2008 10th ieee international conference on high performance computing and communications*, pages 825–830. Ieee, 2008.

[149] L. Wang, G. Von Laszewski, J. Dayal, and F. Wang. Towards energy aware scheduling for precedence constrained parallel tasks in a cluster with dvfs. In *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 368–377. IEEE Computer Society, 2010.

[150] X. Wang and M. Chen. Cluster-level feedback power control for performance optimization. In *2008 IEEE 14th International Symposium on High Performance Computer Architecture*, pages 101–110. IEEE, 2008.

[151] H. Weatherspoon and J. D. Kubiatowicz. Erasure Coding vs. Replication: A Quantitative Comparison. In *Peer-to-Peer Systems*, pages 328–337. Springer, 2002.

[152] B. Wilder. *Cloud architecture patterns: using microsoft azure*. " O'Reilly Media, Inc.", 2012.

[153] R. Wilhelm, J. Engblom, A. Ermedahl, N. Holsti, S. Thesing, D. Whalley, G. Bernat, C. Ferdinand, R. Heckmann, T. Mitra, et al. The worst-case execution-time problem—overview of methods and survey of tools. *ACM Transactions on Embedded Computing Systems (TECS)*, 7(3):36, 2008.

[154] L. Willenborg and T. De Waal. *Elements of statistical disclosure control*, volume 155. Springer Science & Business Media, 2012.

[155] C.-M. Wu, R.-S. Chang, and H.-Y. Chan. A green energy-efficient scheduling algorithm using the dvfs technique for cloud datacenters. *Future Generation Computer Systems*, 37:141–147, 2014.

[156] Q. Wu, Q. Deng, L. Ganesh, C.-H. Hsu, Y. Jin, S. Kumar, B. Li, J. Meza, and Y. J. Song. Dynamo: facebook's data center-wide power management system. In *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, pages 469–480. IEEE, 2016.

[157] T. Xie and X. Qin. Security-aware resource allocation for real-time parallel jobs on homogeneous and heterogeneous clusters. *IEEE transactions on parallel and distributed systems*, 19(5):682–697, 2008.

[158] F. Xu, F. Liu, L. Liu, H. Jin, B. Li, and B. Li. iaware: Making live migration of virtual machines interference-aware in the cloud. *IEEE Transactions on Computers*, 63(12):3012–3025, 2013.

[159] X. Xu, W. Dou, X. Zhang, and J. Chen. Enreal: An energy-aware resource allocation method for scientific workflow executions in cloud environment. *IEEE Transactions on Cloud Computing*, 4(2):166–179, 2015.

[160] C.-Y. Yang, J.-J. Chen, and T.-W. Kuo. An approximation algorithm for energy-efficient scheduling on a chip multiprocessor. In *Design, Automation and Test in Europe*, pages 468–473. IEEE, 2005.

[161] K. Yang, Z. Liu, X. Jia, and X. S. Shen. Time-domain attribute-based access control for cloud-based video content sharing: A cryptographic approach. *IEEE Transactions on Multimedia*, 18(5):940–950, 2016.

[162] 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. IEEE, 1995.

[163] Z.-H. Zhan, X.-F. Liu, Y.-J. Gong, J. Zhang, H. S.-H. Chung, and Y. Li. Cloud computing resource scheduling and a survey of its evolutionary approaches. *ACM Computing Surveys (CSUR)*, 47(4):1–33, 2015.

[164] H. Zhang, S. Shao, H. Xu, H. Zou, and C. Tian. Free cooling of data centers: A review. *Renewable and Sustainable Energy Reviews*, 35:171–182, 2014.

[165] W. Zhang, X. Sun, and T. Xu. Data privacy protection using multiple cloud storages. In *Proceedings 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC)*, pages 1768–1772. IEEE, 2013.

[166] X. Zhang, J.-J. Lu, X. Qin, and X.-N. Zhao. A high-level energy consumption model for heterogeneous data centers. *Simulation Modelling Practice and Theory*, 39:41–55, 2013.

[167] Y. Zhang, Y. Wang, and X. Wang. Greenware: Greening cloud-scale data centers to maximize the use of renewable energy. In *ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing*, pages 143–164. Springer, 2011.

[168] Z. Zheng, S. Xie, H.-N. Dai, X. Chen, and H. Wang. Blockchain challenges and opportunities: A survey. *International Journal of Web and Grid Services*, 14(4):352–375, 2018.

[169] L. Zhou, V. Varadharajan, and M. Hitchens. Achieving secure role-based access control on encrypted data in cloud storage. *IEEE transactions on information forensics and security*, 8(12):1947–1960, 2013.

[170] P. Zhou, J. Huang, X. Qin, and C. Xie. Pars: A popularity-aware redundancy scheme for in-memory stores. *IEEE Transactions on Computers*, 68(4):556–569, 2018.

[171] X. Zhu, X. Qin, and M. Qiu. Qos-aware fault-tolerant scheduling for real-time tasks on heterogeneous clusters. *IEEE transactions on Computers*, 60(6):800–812, 2011.

[172] X. Zhu, L. T. Yang, H. Chen, J. Wang, S. Yin, and X. Liu. Real-time tasks oriented energy-aware scheduling in virtualized clouds. *IEEE Transactions on Cloud Computing*, 2(2):168–180, 2014.

[173] S. Zhuravlev, J. C. Saez, S. Blagodurov, A. Fedorova, and M. Prieto. Survey of energy-cognizant scheduling techniques. *IEEE Transactions on Parallel and Distributed Systems*, 24(7):1447–1464, 2012.

[174] Z. Zong, A. Manzanares, X. Ruan, and X. Qin. Ead and pebd: two energy-aware duplication scheduling algorithms for parallel tasks on homogeneous clusters. *IEEE Transactions on Computers*, 60(3):360–374, 2010.

[175] Z. Zong, X. Qin, X. Ruan, K. Bellam, M. Nijim, and M. Alghamdi. Energy-efficient scheduling for parallel applications running on heterogeneous clusters. In *2007 International Conference on Parallel Processing (ICPP 2007)*, pages 19–19. IEEE, 2007.

[176] L. Zuo, L. Shu, S. Dong, C. Zhu, and T. Hara. A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing. *Ieee Access*, 3:2687–2699, 2015.