Machine Learning-based Energy and Thermal Efficient Resource Management Algorithms for Cloud Data Centres

Shashikant Ilager

Submitted in total fulfilment of the requirements of the degree of

Doctor of Philosophy

Faculty of Engineering and Information Technology
THE UNIVERSITY OF MELBOURNE

February 2021

ORCID: 0000-0003-1178-6582

Machine Learning-based Energy and Thermal Efficient Resource Management Algorithms for Cloud Data Centres

Shashikant Ilager
Principal Supervisor: Prof. Rajkumar Buyya

Abstract

Cloud data centres are the backbone infrastructures of modern digital society and the economy. Data centres have witnessed tremendous growth, consuming enormous energy to power IT equipment and cooling system. It is estimated that the data centres consume 2% of global electricity generated, and the cooling system alone consumes up to 50% of it. Therefore, to save significant energy and provide reliable services, workloads should be managed in both an energy and thermal efficient manner. However, existing heuristics or static rule-based resource management policies often fail to find an optimal solution due to the massive complexity and non-linear characteristics of the data centre and its workloads. In this thesis, we focus on machine learning-based resource management algorithms for energy and thermal efficiency in Cloud data centres which are proven to be efficient in capturing non-linearity between interdependent parameters. We explore how these techniques can be adapted to resource management problems to increase the energy and thermal efficiency of Cloud data centres while simultaneously satisfying application QoS requirements. In particular, we propose algorithms for workload placement, consolidation, application scheduling, and configuring efficient frequencies of resources in Cloud data centres. This thesis advances the state-of-the-art by making the following key contributions:

1. A comprehensive taxonomy and literature review on learning-based resource management approaches in Cloud computing environments for energy and thermal efficiency.

2. A data-driven energy-efficient frequency scaling in GPUs and a deadline aware application scheduling. It first configures the best suitable frequency for application by predicting execution time and energy consumption of an application based on its workload characteristics and finds an efficient schedule sequence to meet its

deadline.

3. Energy and thermal-aware dynamic consolidation technique for Virtual Machines (VM) to achieve integrated energy efficiency of computing and cooling systems while maintaining the Service Level Agreements (SLAs).

4. A machine learning-based fast and accurate thermal prediction model to aid resource management system's online decision. We also propose an energy-efficient VM scheduling algorithm to minimize peak temperature in the data centre.

5. A Deep Reinforcement Learning (DRL)-based method for management of workloads to optimize the energy and thermal aspects in Cloud data centres.

6. A detailed study outlining challenges and research directions in AI-centric resource management of distributed systems.

Declaration

This is to certify that

1. the thesis comprises only my original work towards the PhD,
2. due acknowledgement has been made in the text to all other material used,
3. the thesis is less than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices.

Shashikant Ilager, February 2021

Preface

Main Contributions

This thesis research has been carried out in the Cloud Computing and Distributed Systems (CLOUDS) Laboratory, Faculty of Engineering and Information Technology, The University of Melbourne. The main contributions of the thesis are discussed in Chapters 2-7 and are based on the following publications:

- **Shashikant Ilager**, Kotagiri Ramamohanarao, and Rajkumar Buyya,"ETAS: Energy and Thermal-Aware Dynamic Virtual Machine Consolidation in Cloud Data Center with Proactive Hotspot Mitigation", *Concurrency and Computation: Practice and Experience (CCPE)*, Volume 31, No. 17, Pages: 1-15, ISSN: 1532-0626, Wiley Press, New York, USA, September 2019.

- **Shashikant Ilager**, Rajkumar Buyya, "Energy and Thermal-aware Resource Management of CloudData Centres: A Taxonomy and Future Directions", *ACM Computing Surveys*, USA, 2021 (in review).

- **Shashikant Ilager**, Rajeev Muralidhar, Rammohanrao Kotagiri and Rajkumar Buyya, "A Data-Driven Frequency Scaling Approach for Deadline-aware Energy Efficient Scheduling on Graphics Processing Units (GPUs)", *In Proceedings of the 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGrid 2020)*, Melbourne, Australia, May 11-14, 2020. **[Best Paper Award]**

- **Shashikant Ilager**, Rajeev Muralidhar, and Rajkumar Buyya, "Artificial Intelligence (AI)-Centric Management of Resources in Modern Distributed Computing

Systems", *In Proceedings of the IEEE Cloud Summit*, Harrisbury, Pennsylvania, USA, October 21-22, 2020.

- **Shashikant Ilager**, Kotagiri Ramamohanarao, and Rajkumar Buyya, "Thermal Prediction for Efficient Energy Management of Clouds using Machine Learning", *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, Volume 32, No. 5, Pages: 1044-1056, ISSN: 1045-9219, IEEE CS Press, USA, May 2021.

- **Shashikant Ilager**, Rajkumar Buyya, "TEDRL: Thermal and Energy-aware Deep Reinforcement Learning approach for Workload Scheduling in Cloud Data Centres", *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, USA, 2021 (in review).

Supplementary Contributions

During the PhD candidature, I have also contributed to the following collaborative works (this thesis does not claim them as its contributions):

- Minxian Xu, Chengxi Gao, **Shashikant Ilager**, Huaming Wu, Chengzhong Xu, and Rajkumar Buyya, "Green-aware Mobile Edge Computing for IoT: Challenges, Solutions and Future Directions", *Mobile Edge Computing (MEC)*, A. Mukherjee, D. De, S. K. Ghosh, and R. Buyya (eds), Springer, USA, 2021 (in press).

- Shreshth Tuli, **Shashikant Ilager**, Kotagiri Ramamohanarao, and Rajkumar Buyya, "Dynamic Scheduling for Stochastic Edge-Cloud Computing Environments using A3C Learning and Residual Recurrent Neural Networks", *IEEE Transactions on Mobile Computing*, ISSN: 1536-1233, IEEE Computer Society Press, USA (in press, accepted on Aug 13, 2020).

Acknowledgements

PhD not only allows one to train and specialise on their topic of interests but also teaches valuable life lessons and skills. I am thankful to many people who helped me to pursue this wonderful journey and cross the finish line.

I want to thank my principal supervisor, Professor Rajkumar Buyya, who has given the opportunity to pursue PhD under his guidance. His supervision, hard work, and encouragement to aim high have been inspiring to my PhD studies. I am grateful for his constant support and guidance throughout this journey. I also thank my co-supervisor, Professor Ramamohanarao Kotagiri, who has helped me with many research problems. His wonderful insights and comments helped formulate research questions more precisely and solve it. I also thank my other co-supervisor Dr. Maria Rodriguez, for her guidance. I am truly grateful for all my supervisor's invaluable support, advice and motivation throughout my candidature. I would also like to express my gratitude to my PhD advisory committee's chair, Prof. Rui Zhang, for his comments and suggestions.

Special thanks to my Master's supervisor, Professor Rajeev Wankar, who has helped me develop a keen research interest and continually encouraged and guided me in joining PhD. I am forever indebted to all the teachers at the University of Hyderabad, where I did my Master's; it was indeed two incredible years of learning that helped progress in my PhD studies. I would also like to thank all the past and current CLOUDS Laboratory members at the University of Melbourne. In particular, I thank Dr. Adel Nadjaran Toosi, Dr. Yaser Mansouri, Dr. Satish Narayana Srirama, Dr. Jungmin Jay Son, Dr. Safiollah Heidari, Dr. Minxian Xu, Dr. Xunyun Liu, Dr. Sara Kardani Moghaddam, Dr. Redowan Mahmud, Dr. Muhammad Hilman, Dr. Muhammed Tawfiqul Islam, TianZhang He, Mohammad Goudarzi, Zhiheng Zhong, Samodha Pallewatta, Amanda Jayanetti, Anupama Mampage, Mohammad Reza Razian and Jie Zhao for their support. I cherish my memories with Prof. Vlado Stankovski, who made possible for me to see a bit of Australian outback with travel plans during his time at CLOUDS Lab.

I thank Rajeev Muralidhar for his guidance and for helping me to get AWS credits through Cloud research grant. I thank Laurent Lefevre and Marcos Assuncao at INRIA, France, who granted access to Grid'5000 testbed, which helped me conduct experiments on the testbed. I also thank Bernard Meade and Justin Mammarella at Research Platform Services, The University of Melbourne, to support and access the infrastructure cloud

and data that immensely helped one of my research work.

I also thank the School of Computing and Information System's admin staff, especially Rhonda Smith, who continuously supported and responded to many queries throughout my PhD candidature.

I acknowledge the University of Melbourne and the Australian Research Council (Discovery Project) for providing me with scholarships to pursue my doctoral studies.

Finally, I am thankful to my parents (Shankar and Adamma) and siblings (Vaishali, Vinod, and Shridhar) for their selfless love, support, and understandings.

Shashikant Ilager
Melbourne, Australia
February 2021

Contents

List of Figures

List of Tables

Chapter 1
Introduction

Cloud computing has seen tremendous growth in recent years. The transition from ownership-based on-premise IT infrastructure to subscription-based Cloud has changed the way computing services are delivered to end-users [3] [4]. Cloud computing's fundamental principle is to provide computing resources as utility services (e.g., water and electricity). It offers on-demand access to elastic resources with a pay as you go model based on actual resource usage. This unique and flexible service delivery model ensures that individuals and businesses can easily access required computing services.

Cloud computing services are broadly categorised into three types. First, the Infrastructure as a Service (IaaS) model offers computing, storage, and networking resources either in the virtual or physical form. Second, the Platform as a Service (PaaS) model offers tools for rapid application development and deployment such as middleware platforms, Application Programming Interfaces (APIs), and Software Development Kits (SDKs). Finally, Software as a Service (SaaS) model offers direct access to application software to the users, and the software is developed and managed by service providers completely.

The rapid growth in digital services, Internet of Things (IoT), Industry 4.0, and 5G-based application scenarios are creating a massive demand for Cloud services [5] [6]. Clouds have become application back-end and storage infrastructures for these modern IT services. Along with remote Clouds, recently, Cloud services are delivered from the edge of the network to satisfy Quality of Service (QoS) requirements for latency-sensitive applications such as autonomous vehicles, emergency healthcare services [7] [8]. To seamlessly deliver services for applications and their users, Cloud computing uses massive network-based infrastructures. In particular, Data Centres (DCs) are the core and

backbone infrastructure of this network system. The DCs hosts thousands of servers, networking equipments, cooling systems, and facility-related subsystems to deliver reliable and uninterrupted services. By default, Cloud workloads require a continuous, always-on, and 24×7 access to its deployed services. For instance, the Google search engine is expected to achieve an almost 100% availability rate [9]. Similarly, Amazon AWS witnesses thousands of Elastic Compute (EC2) instances created [10] in a day through their automated APIs, thus requiring massive geo-distributed DC infrastructures to support such critical demand. According to Gartner, by 2022, 60% of organisations will use external Cloud service provider [11], and by 2024, Cloud computing alone accounts for 14.2% of total global IT spending [12].

To cater for the demand of Cloud services, major Cloud service providers such as Amazon AWS[1], Microsoft Azure[2], and Google Cloud[3] are deploying a large number of hyper-scale data centres in multiple regions worldwide. A snippet of Azure global data centre locations can be found in Figure 1.1 [13]. Data centres have seen huge growth both in number and size. There are over 8 million data centres globally, from private small-scale to hypers-scale DCs, and they are estimated to grow rapidly at 12% annually [14]. As their numbers and size grow, they are consuming an increasing amount of energy, resulting in massive energy challenges. DCs are power-hungry and require a continuous energy supply to power their computing, networking, and cooling systems. It is estimated that the DCs consume 2% of global electricity generated [15]. Furthermore, this massive energy consumption leads DCs to rely on fossil-fuel based or brown energy sources that hugely contribute to greenhouse gas emissions. DCs are responsible for emitting 43 million tons of CO_2 per year and continues to grow at an annual rate of 11% [16] leaving high carbon footprints. Therefore, improving the Cloud data centre's energy efficiency is quintessential for sustainable and cost-effective Cloud computing.

The Cloud users and service provider should address the above-mentioned energy problems of DCs through various abstraction layers of the Cloud computing stack. Energy-efficient resource management policies need to be incorporated from an individual server's silicon chip to workloads running across geo-distributed data centres. It is important to

[1]https://aws.amazon.com/
[2]https://azure.microsoft.com/
[3]https://cloud.google.com/

Figure 1.1: Data Centre Locations of Microsoft Azure Cloud

identify the system's inefficiencies and optimise resource usages for energy efficiency. Although a data centre contains numerous subsystems, computing and cooling are two main subsystems that contribute to the significant energy consumption in a DC. Each rack in DC consumes up to 30-40 kW of power which makes them one of the highest energy density Cyber Physical Infrastructures (CPS). This high-density energy is translated as heat and dissipated into the environment. Thus, cooling systems are employed to keep the data centre environment within the safe temperature threshold. Hence, it is imperative to optimise both these subsystems together to achieve significant energy efficiency.

Resource Management Systems (RMS) in DCs are middleware platforms that perform different tasks such as resource provisioning, monitoring, workload scheduling, and many others. RMS tasks need to be designed with intelligent algorithms and policies, keeping energy consumption as a "first-class" optimisation parameter. Current approaches follow the "time-to-solution" approach, which is optimised for application execution speed. However, faster execution may not always yield better energy efficiency [17]. Hence, the paradigm shift to focus on "kW-to-solution" is essential. New algorithms and techniques towards this direction are required focusing on how much electrical energy has been spent on workload execution. Furthermore, today's Clouds

serves diverse users and their heterogeneous workloads, serving static webpages to highly latency-sensitive and dynamic stream applications where petabytes of data must be processed in realtime. Thus, along with energy efficiency, RMS policies should be aware of the varied QoS requirements of applications.

To address the above challenges of Cloud data centre's energy efficiency, many solutions have been proposed, including resource management principles, policies, and algorithmic techniques for energy-efficient resource provisioning, workload scheduling, and consolidation. Utilising renewable energy is also an important direction to reduce the carbon footprints of Cloud data centres. Techniques like workload shifting are useful where application workloads are migrated across geo-distributed data centres to exploit renewable energy and mitigate its intermittent availability issues. Some solutions have explored free cooling mechanisms to reduce cooling energy cost [18]. However, energy-efficient Cloud computing is still a challenging problem to be solved. A holistic approach to managing Cloud resources is required to achieve significant energy efficiency. One of the challenges in holistic optimisation is a conflicting trade-offs between computing and cooling systems. Optimising computing system alone often increases cooling energy cost due to increased temperature in the data centre. Moreover, many of the current approaches optimise compute and cooling subsystems independently, thus failing to achieve significant efficiency. Hence, innovative holistic solutions addressing energy efficiency can make Cloud data centres environmentally and economically sustainable. To that end, this thesis focuses on the problem of energy and thermal efficient resource management in Cloud data centre by proposing efficient energy management of resources at various abstraction layer, from an individual machine to data centre level workload management. It ensures computing resources are efficiently utilised without increasing cooling energy cost while guaranteeing the required Quality of Service (QoS) for applications.

1.1 Motivations

Cloud Data centres tremendous growth has introduced massive energy challenges. If necessary steps are not taken, data centres may consume up to 8000 terawatts of power

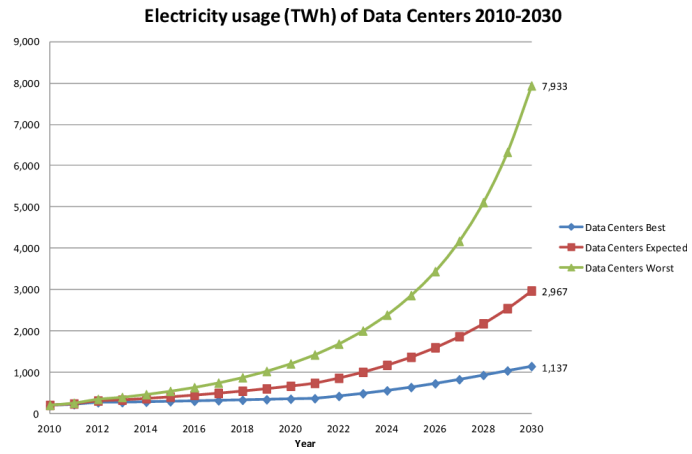


Figure 1.2: Estimation of Data Centre Energy Consumption by 2030 [1]

in the worst case by 2030. However, if best practices are adopted across the Cloud computing stack, this massive energy consumption can be brought down to around 1200 terawatts [1] (see Figure 1.2). To achieve this best-case scenario needs adopting energy-efficient practices into the various level of data centre resource management platforms (such as optimised use of computing and cooling resources). Hence, it is of utmost importance to address this energy problem and achieve sustainability both environmentally and economically.

Resource management in data centres is extremely challenging due to complex subsystems and heterogeneous workload characteristics. It is impossible to fine-tune the controllable parameters by resource management systems manually. For example, *"Just* 10 *pieces of equipment, each with* 10 *settings, would have* 10 *to the* 10^{th} *power, or* 10 *billion, possible configurations a set of possibilities far beyond the ability of anyone to test for real"* [19][20]. Moreover, these large-scale systems have numerous subsystems interacting with each other and often have a non-linear relationship between their parameters. For instance, increasing utilisation of resources is a vital optimisation parameter for a service provider, which reduces the operational cost by reducing the number of active machines and using available resources to offer their services to more users. However, over-utilisation potentially results in degraded QoS for users, as applications now compete for constrained resources in these shared environments. Furthermore, over-utilisation also increases energy consumption which translates as dissipated heat resulting in com-

plex thermal management and cooling energy cost. Hence, optimising a single parameter often leaves a trade-off with other parameters essential for optimising infrastructure as a whole. Existing heuristics or rule-based policies often fail to find an optimal solution under such environments. To that end, innovative learning-based solutions are promising where Machine Learning (ML) techniques are proven to be efficient in capturing complex non-linearity between different interrelated parameters. We can carefully adapt these techniques to various resource management problems. For instance, Google has achieved a 40% efficiency in managing its cooling infrastructure using simple ML techniques and learning from historical data [21]. Many other methods explored problems such as device placement, scheduling, and application scaling using data-driven methods [22], [23]. At the system architecture level, the work in [24] used massive data sets of hardware performance counters and profiles collected from large-scale Google data centre servers and utilised this data to reason, analyse and mitigate front-end stalls in warehouse-scale systems. However, Machine Learning (ML)-based resource management solutions are in their superficial stage. They require meticulous attention to address the challenges they pose and simultaneously identify potential avenues to incorporate these methods.

Hence, in this thesis, we explore how we can leverage the capabilities of machine learning and deep learning methods into various data centre resource management problems to optimise energy and thermal aspects. This requires suitable ML techniques need to be trained to learn and predict desired outputs. Besides, problems need to be carefully designed so that models can learn appropriate resource management policies in realtime. This thesis focuses on multiple resource management problems proposing ML-based policies for Cloud data centre energy efficiency.

An ideal way to reduce the server's energy consumption is applying Dynamic Voltage and Frequency Scaling (DVFS) technique [25]. Modern computing elements such as CPUs and GPUs have a vast number of frequency ranges, and usually, operating system level device drivers scale frequencies based on simple heuristics observing utilisation level. However, regulating optimal frequency is challenging since different workloads exhibit different execution speed and energy consumption concerning different operating frequencies. In this thesis, we explore how to configure energy-efficient frequencies

in GPUs using machine learning models. GPUs are a highly used computing device in modern Cloud data centres, as demand for AI and ML workloads is continuously increasing. Nevertheless, our principle approach can be applied to CPUs as well.

Another way to reduce the power consumption of a data centre servers is increasing resource utilisaiton [26] [27] [28]. Underutilisation and even idle servers consume a significant amount of power. An idle server consumes up to 70% of its peak power [29]. Thus, increasing resource utlisaiton helps to achieve proportional energy computing by effectively utilising the active power consumption for useful computation. In IaaS Cloud, utilisation is usually increased by dynamically consolidating the Virtual Machines (VMs) through live migration and switching off inactive machines. This dynamic VM consolidation is a widely used technique in Cloud data centres for energy efficiency. However, consolidation causes overutilisation of servers which results in increased server temperature, thus potentially creating local hotspots. This not only affects cooling energy but also affects the reliability of the system (due to device failures caused by high temperature). Cooling energy cost exponentially increases since supply air temperature needs to be set to a much lower value, demanding more energy. Hence, an optimal balance needs to be found in dynamic VM consolidation, both energy and thermal aware.

Another issue in the data centre is managing peak temperature [30]. Every degree increase in data centre peak temperature costs millions of dollars in operational cost [31] as the cooling system's thermal load drastically increases. Furthermore, as described earlier, increased temperature affects the cooling cost and further decreases the system's reliability due to failures under high thermal conditions. It is essential to understand that the data centre's peak temperature and the cooling system setpoint temperature are different (also called supply air temperature). If the data centre's peak temperature increases, supply air temperature needs to be set to a lower value, requiring higher cooling energy. Hence, to solve these problems, a workload management system should be aware of such trade-offs, and resources should be managed holistically. The data centre workloads should be managed to reduce energy consumption and keep the peak temperature of the data centre within the recommended thresholds, thus keeping cooling energy cost minimum.

However, estimating the server's accurate temperature for a given workload and data centre conditions is non-trivial. Inaccurate temperature estimation leads to suboptimal resource management decisions such as wrong scheduling, placement, and consolidation decisions. Currently, Computational Fluid Dynamics (CFD) models [32] [33] and analytical models [34] [35] are used to estimate the temperature. Although CFD models are accurate but they are computationally expensive making them infeasible for online decisions. In contrast, analytical models are inaccurate in predicting the temperature as they are highly dependent on static mathematical variables. Therefore, fast and accurate temperature prediction models are essential in energy and thermal efficient data centre resource management. In this regard, ML-based temperature predictions are highly suitable as they are built from actual measurements, and they capture the important variations that are induced by different factors in data centre environments which make their predictions accurate.

Holistic management of energy is a challenging task that requires capturing complex dynamics of data centre workloads and physical environments. Recent advancements in RL have made it possible to learn different policies by interacting with the environments and learning from experience. RL techniques can be more adaptive and automatically understand the policies. Many resource management solutions have explored applying Reinforcement learning (RL) methods for optimising device placement [22], scheduling [36]. Careful design of state management, action, and rewards are important for applying RL techniques to data centres' holistic energy management.

Therefore, this thesis scope is energy and efficient thermal management of resources. It proposes various resource management algorithms by leveraging learning-based techniques to optimise Cloud data centres' energy and thermal aspects. Our solutions' primary focus is workload management through frequency scaling, consolidation, and scheduling policies while providing required QoS or Service Level Agreements (SLAs). The proposed solutions are evaluated using a set of simulation toolkits (CloudSim) and real testbeds. We have used actual workload traces (PlanetLab VMs and Bitbrain's data sets), standard benchmarking applications (Rodinia and PolyBench), and to train ML models, we collected data from our own University's data centre and through profiling

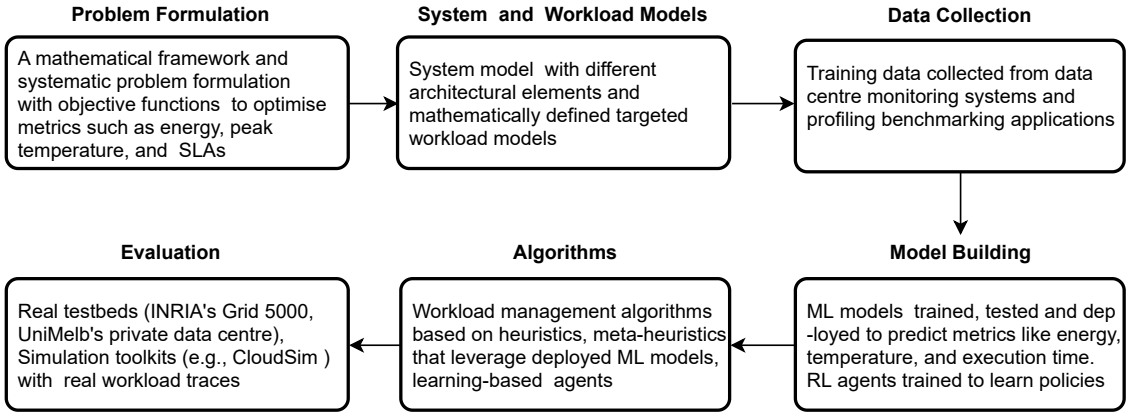


Figure 1.3: The research methodology used in the thesis

on Grid5K testbed's resources [4].

1.2 Methodology

This thesis aims to reduce the energy consumption of data centres by focusing on different abstraction layers of data centre resources. To achieve this goal, we follow the systematic research methodology as shown in Figure 1.3 in our research works.

Problem Formulation: For each of the research problem, we formulate the problem by focusing on specific optimisation objectives of interests, including energy, peak temperature, SLAs, and QoS.

System and Workload Models: We provide the system model showing different architectural elements involved in our system. Also, we formally define targeted workload models in our research problems.

Data Collection: We collect the data from real-world environments such as our University's Cloud data centre monitoring systems. Data is also collected from profiling benchmarking applications on INRIA's Grid5K testbed in Europe. This data is essential to train machine learning models.

Model Building: We train suitable machine learning models using data collected from the previous step. We test, validate and deploy those models, which are then

[4] http://cloudbus.org/ai4clouds/

used in resource management algorithms. These models are trained to predict different metrics such as energy consumption of an application, execution time, and server temperature under diverse workload and data centre conditions.

Algorithms: We propose different resource management algorithms for different tasks such as workload consolidation and scheduling. These algorithms are aided with prediction models built in the previous step. They are designed based on heuristics, meta-heuristics, and the system model showing different architectural elements in our Cloud data centre environments. Along with achieving optimisation objectives, they are designed to satisfy the application QoS and user's SLAs.

Evaluation: We evaluate our proposed approaches using real testbeds (Grid 5K [37]) and simulation toolkits such as CloudSim [38]. We also build our prototype system and evaluate the metrics such as energy, peak temperature, SLAs and QoS (deadline).

Our research methodology has produced innovative algorithms, methods, open-sourced data sets, and software systems.

1.3 Research Problems and Objectives

In data centres, numerous subsystems, including computing (application and storage servers), networking equipment, cooling system, and other facility-related systems, closely work together to provide reliable services to users. Cooling and computing are two major subsystems that consume a significant amount of energy. Hence, it is important to address efficiency of these two subsystems to make cloud data centres energy efficient. This thesis investigates energy efficiency at different abstraction layers of data centre stacks from the individual server to middleware platforms optimising computing and cooling energy while simultaneously providing application's Quality of Service (QoS) and Service Level Agreements (SLAs). To achieve this objective, we solve important resource management problems by addressing the following research questions:

- Q1. *How can we configure energy-efficient GPU frequencies and schedule applications on them to meet their QoS?* : Modern computing paradigms, such as Cloud computing, are increasingly adopting GPUs to boost their computing capabilities primarily

due to the heterogeneous nature of AI/ML/deep learning workloads. In GPUs, Dynamic Voltage Frequency Scaling (DVFS) is a popular technique to reduce active power by varying the GPU frequencies. However, configuring optimal operating frequency for application execution is a challenging problem. It is more challenging for GPUs since they provide hundreds of frequency configurations, and application kernels behave differently concerning energy and performance. Hence, it is required to build frequency scaling techniques and scheduling algorithms that account for application workload characteristics and accordingly configure GPU operating frequencies that are energy efficient and yet satisfy the application QoS requirements.

- Q2. *How to dynamically consolidate the workloads to reduce energy and yet avoid potential hotspots?* : Dynamic Virtual Machine (VM) consolidation is a widely adopted technique to reduce computing systems' energy consumption in Cloud data centres. However, aggressive consolidation leads to creating local hotspots that have adverse effects on cooling energy consumption and the system's reliability. Besides, aggressive consolidation also violates SLAs due to over utilisation of resources. Hence, it is necessary to design consolidation techniques and algorithms that are both energy and thermal-aware and yet satisfy users' SLAs.

- Q3. *How to predict server temperature accurate and fast to guide resource management systems' online decisions?*: Precise prediction of host temperature is crucial for managing the resources effectively. Several resource management tasks, such as scheduling and provisioning, need fast and accurate temperature estimation models for their online decisions. Temperature estimation is a non-trivial and complex problem due to thermal variations in the data centre. Existing solutions such as Computational Fluid Dynamics (CFDs) and analytical models are either computationally expensive or inaccurate. Hence, it is necessary to build accurate and fast prediction models based on the machine learning models built using monitored data from a particular data centre, predicting the server temperature accurately for given workload and data centre conditions with fast inference.

- Q4. *How to automatically learn the scheduling policies to capture the workload and data*

centre characteristics to optimise complex objectives for energy efficiency? The data centre environment exhibits non-linear relationships between different parameters. For instance, a host with a similar state often exhibits non-stationary in its thermal response. Its temperature is effected by heat recirculating, physical position, workload type and many other parameters. Similarly, resource utilisation, power consumption and corresponding temperature response will have a non-linear relationship between them. Existing scheduling algorithms are based on static rules or manually fine-tuned heuristics that fail to capture these intricacies in the data centre. Therefore, it is essential to build an adaptive scheduling algorithm based on Deep Reinforcement Learning (DRL) to deal with such complexity and learning adaptive scheduling policies.

1.4 Thesis Contributions

This thesis systematically addresses the energy efficiency problem of cloud data centres through various resource management techniques. The proposed resource management solutions cover from device architectural level to Cloud data centre level abstractions. It presents a detailed survey and taxonomy covering the existing resource management techniques and identifies the need and motivations for Machine Learning (ML) based solutions. The individual research works have proposed novel resource management algorithms, architectural models, and prototype systems for resource management in Cloud resources. Based on the research problems mentioned in Section 1.3, the key contributions of this thesis are listed below:

1. Proposes a taxonomy on learning-based energy and thermal-aware resource management and reviews the existing energy, thermal and integrated energy and thermal aware resource management approaches.

2. Identifies the need for learning-based resource management techniques, identify the challenges in applying Artificial Intelligence (AI) techniques in the data centre, and explore different avenues to use them on other resource management problems.

3. Builds machine-learning-based models and techniques to configure energy-efficient GPU frequencies and designs deadline aware application scheduling algorithm. (addresses the Q1).

    - A framework to efficiently profile the benchmark application so to observe key architectural, power, and performance counters and metrics.
    - A data-driven prediction model to accurately predict the energy and execution time of applications
    - An efficient frequency scaling configuration mechanism using the prediction models
    - A deadline-aware energy-efficient application scheduling algorithm which leverages the prediction models.
    - A prototype system and evaluation of the proposed solution on a real platform using standard benchmarking applications
    - Evaluations on INRIA Grid5K testbed with NVIDIA Tesla P100 GPUs

4. Investigates the energy and thermal aware dynamic VM consolidation technique to reduce the energy consumption while mitigating potential hotspots and SLA violation (addresses the Q2).

    - A mathematical model to optimise integrated computing and cooling energy for VM consolidation
    - A policy for efficient distribution of VM's workload to optimise the computing and cooling energy and proactively prevent the hotspots.
    - An online scheduling algorithm based on Greedy Random Adaptive Search Procedure (GRASP) meta-heuristic used for dynamic VM consolidation.
    - An event-based simulator used to simulate, test, and compare baseline scheduling policies.

5. Builds ML-based prediction models for accurate temperature prediction models for a server and innovative workload scheduling algorithm to minimise peak temperature in the data centre for energy efficiency (addresses the Q3).

- A complete data collection framework that collects physical-host level measurements from real-world data.
- A detailed study showing the thermal and energy consumption variations between hosts under similar resource consumption and cooling settings.
- Several machine learning-based temperature prediction models using fine-grained measurements from the collected data.
- A feasibility study of proposed prediction models with extensive empirical evaluation.
- A dynamic workload scheduling algorithm guided by the prediction methods, which reduces the peak temperature of the data centre that minimises the total energy consumption under rigid thermal constraints.

6. Proposes RL-based scheduling framework to learn and optimise complex scheduling objectives for energy and thermal efficiency (addresses the Q4).

   - An RL model of the workload scheduling problem of energy and thermal efficiency of data centre environments.
   - A action, reward, and state management methods for DRL framework.
   - A RL-based data centre model environment.
   - A DRL agent that works as an adaptive scheduler to optimise energy and thermal efficiency.

1.5 Thesis Organization

The structure of this thesis is shown in Figure 1.4. The remaining part of this thesis is organised as follows:

- Chapter 2 presents a taxonomy and literature review on energy and thermal efficient resource management algorithms for Cloud data centres. This chapter is derived from:

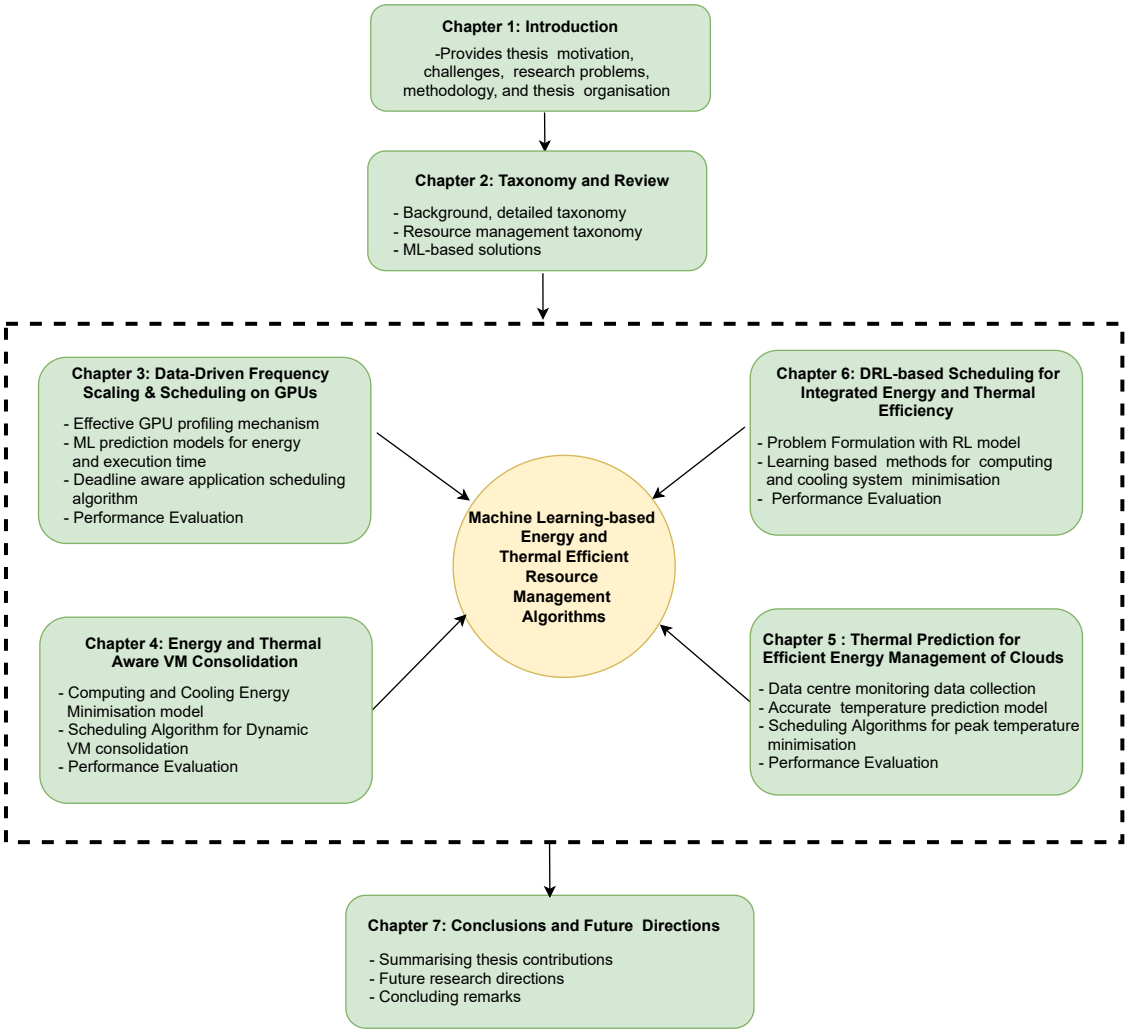


Figure 1.4: The thesis structure

Shashikant Ilager, Rajeev Muralidhar, and Rajkumar Buyya, "Artificial Intelligence (AI)-Centric Management of Resources in Modern Distributed Computing Systems", *In Proceedings of the IEEE Cloud Summit*, Harrisbury, Pennsylvania, USA, October 21-22, 2020.

- **Shashikant Ilager**, Rajkumar Buyya, "Energy and Thermal-aware Resource Management of CloudData Centres: A Taxonomy and Future Directions", *ACM Computing Surveys*, USA, 2021 (in review).

- Chapter 3 presents energy-efficient frequency scaling in GPUs using ML-based

prediction models that facilitate deadline-aware application scheduling algorithm. This chapter is derived from:

- **Shashikant Ilager**, Rajeev Muralidhar, Rammohanrao Kotagiri and Rajkumar Buyya, "A Data-Driven Frequency Scaling Approach for Deadline-aware Energy Efficient Scheduling on Graphics Processing Units (GPUs)", *In Proceedings of the 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGrid 2020)*, Melbourne, Australia, May 11-14, 2020. **[Best Paper Award]**

- Chapter 4 presents energy and thermal aware dynamic virtual machine consolidation algorithm that deals with mitigating potential hotspots that may occur in consolidation and takes care of SLA requirements. This chapter is derived from:

  - **Shashikant Ilager**, Kotagiri Ramamohanarao, and Rajkumar Buyya, "ETAS: Energy and Thermal-Aware Dynamic Virtual Machine Consolidation in Cloud Data Centre with Proactive Hotspot Mitigation", *Concurrency and Computation: Practice and Experience (CCPE)*, Volume 31, No. 17, Pages: 1-15, ISSN: 1532-0626, Wiley Press, New York, USA, September 2019.

- Chapter 5 presents machine-learning-based temperature prediction models, which are built from data collected from our University's private data centre. These prediction models are used in the proposed scheduling algorithm that optimises data centre peak temperature for energy efficiency. This chapter is derived from:

  - **Shashikant Ilager**, Kotagiri Ramamohanarao, and Rajkumar Buyya, "Thermal Prediction for Efficient Energy Management of Clouds using Machine Learning", *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, Volume 32, No. 5, Pages: 1044-1056, ISSN: 1045-9219, IEEE CS Press, USA, May 2021.

- Chapter 6 presents a Reinforcement Learning based scheduling framework where DRL agents learn the complex energy and thermal efficient scheduling policies by interacting with the environments. This chapter is derived from:

  - **Shashikant Ilager**, Rajkumar Buyya, "TEDRL: Thermal and Energy-aware Deep Reinforcement Learning approach for Workload Scheduling in Cloud Data Centres", *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, USA, 2021 (in

review).

- Chapter 7 concludes the thesis, summarises the key findings and identifies future research directions.

Chapter 2

A Taxonomy on ML-driven Energy and Thermal-aware Resource Management

This chapter investigates the existing resource management approaches in Cloud Data Centres for energy and thermal efficiency, focusing on machine learning-based approaches. The chapter identifies the need for integrated computing and cooling systems management and learning-based solutions in resource management systems. A taxonomy on resource management in data centres is proposed based on an in-depth analysis of the literature. A detailed survey of existing approaches is conducted according to the taxonomy. Finally, a conceptual model for managing resources using learning-based techniques in data centre environments is presented.

2.1 Introduction

Internet-based Distributed Computing Systems (DCS) such as Clouds have become an essential backbone of the modern digital economy, society, and industrial operations.

The emergence of the Internet of Things (IoT), diverse mobile applications, smart grids, smart industries, and smart cities has resulted in massive amounts of data generation. Thus, increasing the demand for computing resources [5] to process this data and

This chapter is derived from:

- **Shashikant Ilager**, Rajeev Muralidhar, and Rajkumar Buyya, "Artificial Intelligence (AI)-Centric Management of Resources in Modern Distributed Computing Systems", *In Proceedings of the IEEE Cloud Summit*, Harrisbury, Pennsylvania, USA, October 21-22, 2020.
- **Shashikant Ilager**, Rajkumar Buyya, "Energy and Thermal-aware Resource Management of Cloud-Data Centres: A Taxonomy and Future Directions", *ACM Computing Surveys*,USA, 2021 (in review).

derive valuable insights for users and businesses. According to the report from Norton [39], 21 billion IoT devices will be connected to the internet by 2025, creating substantial economic opportunities.

Computing models such as Cloud have revolutionised the way services are delivered and consumed by providing flexible on-demand access to services with a pay-as-you-go model. Besides, new application and execution models like micro-services and serverless or Function as Service (FaaS) computing [40] are becoming mainstream that significantly reduce the complexities in the design and deployment of software components. On the other hand, this increased connectivity and heterogeneous workloads demand distinct Quality of Service (QoS) levels to satisfy their application requirements [4, 41, 42]. These developments have led to the building of hyper-scale data centres and complex multi-tier computing infrastructures.

The Cloud data centres are the backbone infrastructures of Cloud computing today. A data centre is a complex Cyber-Physical-System (CPS) consisting of numerous elements. It houses thousands of rack-mounted physical servers, networking equipment, sensors (monitoring server and room temperature), a cooling system to maintain acceptable room temperature, and many other facility-related subsystems. It is one of the highest power density CPS consuming up to 30-40 kW per rack, dissipating an enormous amount of heat. This poses a severe challenge to manage resources energy efficiently and provide reliable services to users. Moreover, even a 1% improvement in data centre efficiency leads to savings in millions of dollars over a year and reduces the carbon footprints [31]. However, optimising data centre operation requires tuning the hundreds of parameters belonging to different subsystems where heuristics or static solutions fail to yield a better result. Therefore, optimising these data centres using suitable Artificial Intelligence (AI) techniques is of great importance.

There have been many efforts in this regard using ML for systems focusing on optimising different computing layers [43]. Public Cloud service providers and the data centre industry have also explored energy and thermal efficient resource management solutions using ML techniques. ML-centric Cloud [44] is an ML-based RMS system at an inception stage from the Microsoft Azure Cloud. They built Resource Control (RC)a general ML and prediction serving system that provides insights into the Azure com-

pute fabric resource manager's workload and infrastructure. The input data collected from the virtual machine and physical servers. The models are trained using a gradient boosting tree and trained to predict the different outcomes for a user's VMs, such as average CPU utilisation, deployment size, lifetime, and blackout time. The Azure resource manager interacts with these models in runtime. For instance, the scheduler queries for virtual machine lifetime, and based on the predicted value, an appropriate decision is taken to increase infrastructure efficiency. Applying these models to several other resource management tasks is considered, including power management inside Azure infrastructure.

Similarly, Google has also applied ML techniques to optimise the efficiency of their data centres. Specifically, they have used ML models to change the different knobs of the cooling system, thus saving a significant amount of energy [21]. The ML models are built using simple neural networks and trained to improve the PUEs (Power Usage Effectiveness), which is a standard metric to measure the data centre efficiency. The input features include total IT workload level, network load, parameters affecting the cooling system like outside temperature, wind speed, number of active chillers, and others. The cooling subsystems are configured according to the predictions, and results have shown that the 40% savings are achieved in terms of energy consumption. These applied use cases firmly attest to the feasibility of learning-based solutions in different aspects of resource management of distributed systems. In the next subsection, we describe the need for ML-based resource management solutions explicitly.

2.1.1 Need for learning-based Resource Management Solutions

The existing Resource Management Systems (RMS), from operating systems to large scale data centres, are predominantly designed and built using preset threshold-based rules or heuristics. These solutions are static and often employ reactive solutions [44]; they work well in the general case but cannot adjust to the dynamic contexts [43]. Moreover, once deployed, they considerably fail to adapt and improve themselves in the runtime. In complex dynamic environments (such as Cloud and Edge), they are incapable of capturing the infrastructure and workload complexities and hence fall through.

Consequently, the learning-based approaches built on actual data and measurements collected from respective DCS environments are more promising, perform better, and adapt to dynamic contexts. Unlike heuristics, these are data-driven models built based on historical data. Accordingly, these methods can employ proactive measures by foreseeing the potential outcome based on current conditions. For instance, a static heuristic solution for scaling the resource uses workload and system load parameters to trigger the scaling mechanism. However, this reactive scaling diminishes the users' experience for a certain period (due to the time required for system bootup and application trigger). Consequently, a learning-based RMS enabled by data-driven Machine Learning (ML) model can predict the future workload demand and scale up or scale down the resources beforehand as needed. Such techniques are highly valuable for both users to obtain better QoS and service providers to offer reliable services and retain their business competency in the market. Moreover, methods like Reinforcement Learning (RL) [43]. [45] [46] can improve RMS's decisions and policies by using monitoring and feedback data in runtime, responding to the current demand, workload, and underlying system status.

Machine learning-based RMS is more feasible now than ever for multiple reasons: (1) AI techniques have matured and have proven efficient in many critical domains such as computer vision, natural language processing, healthcare applications, and autonomous vehicles; (2) Most DCS platforms generate enormous amounts of data currently pushed as logs for debugging purposes or failure-cause explorations. For example, Cyber-Physical-Systems (CPS) in data centres already have hundreds of onboard CPU and external sensors monitoring workload, energy, temperature, and weather parameters. Such data is useful to build ML models cost-effectively; (3) the increasing scale in computing infrastructure and its complexities require automated resource management systems that can deliver the decisions based on the data and key-insights from experience, to which AI models are ideal.

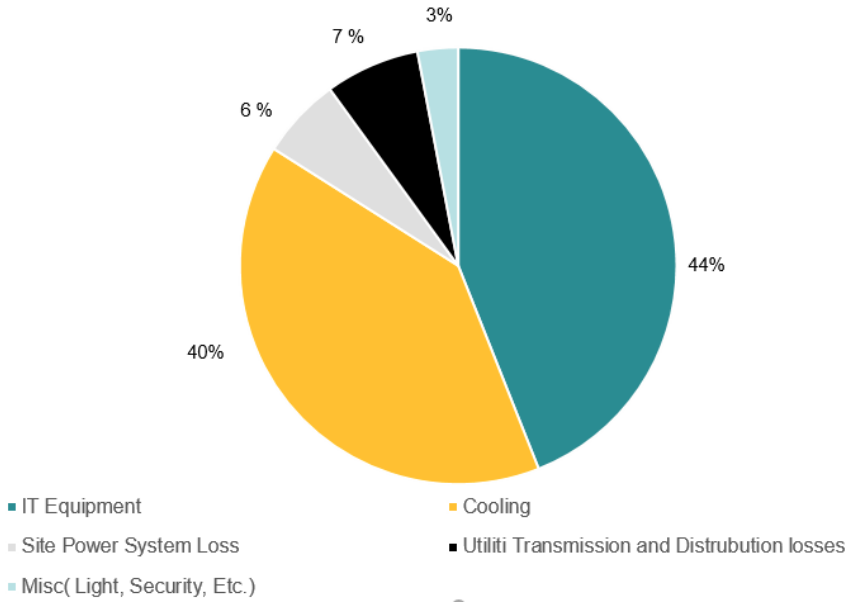


Figure 2.1: Energy Distribution in Data Centres [2]

2.1.2 Need for Integrated Energy and Thermal-aware Resource Management

Data centres host numerous subsystems, including IT/compute (compute servers, network, and storage equipment), cooling system, power distribution, and other facility-related subsystems. However, the majority of power is spent on computing and cooling systems. As shown in Figure 2.1, computing and cooling system together account for 85% of total energy consumption in a data centre, with each of them equally contributing total power consumption [2].

Traditionally, cooling system management is left to the facility management team, and computing system is managed by IT administrator individually. However, optimising a single system has an adverse effect on other systems. For instance, increasing resource utilisation in computing may create hotspots and thus increasing cooling energy cost. Hence, managing these subsystems separately leaves energy inefficiencies in data centre even though individually they are optimised for energy efficiency. The advancement in IoT and smart systems [47] has enabled many mechanical systems associated with cooling to be managed or configured through software systems [48] [49] [50]. Hence, it is imperative to apply resource management techniques holistically to

optimise computing and cooling systems and avoid conflicting tradeoffs between these two subsystems.

The rest of the chapter is organised as follows: The brief background and need for Ml based resource management techniques is given in Section 2.2. Section 2.3 presents high-level taxonomy of energy and thermal aware resource management. Section 2.4 describes existing methods based on taxonomy for energy management in data centre and Section 2.5 covers thermal management solutions. Existing energy and thermal efficient integrated resource management solutions are explained in Section 2.6. Then, Section 2.7 describes different cooling managements systems in a data centre, including air and liquid cooling systems. Section 2.8 presents a conceptual resource management model using learning-based solutions. Finally, Section 2.9 concludes the chapter.

2.2 Background

2.2.1 Challenges of learning-based Resource Management Solutions

Availability of Data

The quality of data used to train the models determines the success of machine learning techniques. Also, this data should be available in large quantities with enough features covering all the aspects of environments [51][52]. Within Cloud data centres, multiple challenges exist concerning the availability of such data. First, currently, different resource abstraction platforms collect the data at different granularity. The physical machine-level data from onboard sensors and counters is gathered and accessed by tools like Intelligent Platform Management Interface (IPMI), while at a higher abstraction level, middleware platforms collect data related to workload level, user information, and surrounding environmental conditions (temperature, cooling energy in the data centre). Also, network elements such as SDN controllers collect data related to network load, traffic, and routing. Unifying these data together and preprocessing it in a meaningful way is a complex and tedious task. The respective tools gather the data in a different format without common standards between them. Hence, building data-

pipelines combining various subsystems data is crucial for the flexible adoption of ML solutions. Secondly, current monitoring systems collect data and push them into logging repositories to be used later for debugging. However, converting this data for ML-ready requires monotonous data-engineering. Hence, future systems should be explicitly designed to gather the information that can be directly fed to the ML models with minimal data engineering and preprocessing effort. Lastly, although several publicly available datasets provide workload traces, there are hardly any public datasets available representing various infrastructure, including physical resource configurations, energy footprints, and several other essential parameters (due to privacy and NDAs). Therefore, getting access to such data is a challenge and needs collaborative efforts and data management standards from the relevant stakeholders. Moreover, it requires standardised data formats and domain-specific frameworks [53].

Managing the Deployment of Models

Training ML models and inference in runtime needs an expensive amount of computational resources. However, one significant challenge is to manage the life cycle of ML models, including deciding how much to train, where to deploy the training modules in multi-tier computing architectures like Edge/Fog. ML models tend to learn with the expense of massive computational resources consuming an enormous amount of energy. Therefore, innovative solutions are needed to decide how much learning is sufficient based on specific constraints (resource budget, time-budget, etc.) and estimate context-aware adaptive accuracy thresholds of ML models [54]. To overcome this, techniques like transfer learning, distributed learning can be applied to reduce computational demands [52]. In addition, dedicated CPUs, GPUs, and domain-specific accelerators like Google TPU, Intel Habana, and FPGAs (Azure) can carry out the inference.

Non-Deterministic Outputs

Unlike statistical models, which are analogous for its deterministic outputs, ML models are intrinsically exploratory and depend on stochasticity for many of their operations, thus producing the non-deterministic results. For example, the cognitive neural nets,

which are basic building blocks for many regressions, classification, and Deep Learning (DL) algorithms primarily rely on the principles of stochasticity for different operations (stochastic gradient descent, exploration phase in RL). When run multiple times with the same inputs, they tend to approximate the results and produce different outputs [55]. This may pose a severe challenge in the Cloud systems, where strict Service Level Agreements (SLAs) govern the delivery of services requiring deterministic results. For example, if a service provider fixes a price based on certain conditions using ML models, consumers expect the price to be similar all the time under similar settings. However, ML models may have a deviation in pricing due to stochasticity creating the transparency issues between users and service providers. Many recent works have focused on this issue, and introduced techniques such as induced constraints in neural nets to produce the deterministic outputs [56]. Yet, stochasticity in the ML model is inherent and requires careful monitoring and control over its output.

Black Box Decision Making

The ML models' decision-making process follows a completely black-box approach and fails to provide satisfactory justification for its decisions. The inherent probabilistic architectures and enormous complexities within ML models make it hard to evade the black-box decisions. It becomes more crucial in an environment such as DCS, where users expect useful feedback and explanation for any action taken by the service provider. This is instrumental in building trust between service providers and consumers. For instance, in case of a high overload condition, it is usual that service provider shall preempt few resources from certain users at the expense of certain SLA violations. However, choosing which users' resources should be preempted is crucial in business-driven environments. This requires simultaneously providing fair decisions and valid reasons. Many works have undertaken to build the explanatory ML models (Explainable AI-XAI) to address this issue [57], [58]. However, solving this continues to remain a challenging task.

Lightweight and Meaningful Semantics

The DCS environment having heterogeneous resources across the multi-tiers accommodates different application services. RMS should interact with different resources, entities, and application services to efficiently manage the resources. However, these requires semantic models that represent all these various entities meaningfully. Existing semantic models are either heavy or inadequate for such complex environments. Therefore, lightweight semantic models are needed to represent the resource, entities, applications, and services without introducing the overhead [59].

Complex Network Architectures, Overlays, Upcoming Features

Network architectures across distributed Clouds and telecom networks are evolving rapidly using software-defined infrastructure, hierarchical overlay networks, Network Function Virtualization (NFV), and Virtual Network Functions (VNF). Commercial Clouds like Amazon, Google, and Microsoft have recently partnered with telecom operators worldwide to deploy ultra-low latency infrastructure (AWS Wavelength and Azure Edge Zone, for example) for emerging 5G networks. The explosion of data from these 5G deployments and resource provisioning for high bandwidth, throughput, and low latency response through dynamic network slicing requires a complex orchestration of network functions [60].

In future Cloud systems, RMS needs to consider these complex network architectures, the overlap between telecom and public/private Clouds, and service function orchestration to meet end-to-end bandwidth, throughput, and latency requirements. These architectures and implementations, in turn, generate enormous amounts of data at different levels of the hierarchical network architecture. As different types of data are generated in different abstraction levels, standardised well-agreed upon data formats and models for each aspect needs to be developed.

Performance, Efficiency, and Domain Expertise

Many ML algorithms and RL algorithms face performance issues like a cold-start problem. Specifically, RL algorithms spend a vast amount of the initial phase in exploration before reaching their optimal policies creating an inefficient period where the decisions are suboptimal, even wholly random or incorrect leading to massive SLA violations [52]. RL-based approaches also face several challenges in the real world including (1) need for learning on the real system from limited samples, (2) safety constraints that should never or at least rarely be violated, (3) need for reward functions that are unspecified, multi-objective, or risk-sensitive, (4) inference that must happen in real-time at the control frequency of the system [61]. In addition, AI models are compute-heavy and designed with a primary focus on accuracy-optimisation resulting in a massive amount of energy consumption [19]. Consequently, new approaches are needed to balance the trade-offs between accuracy, energy, and performance overhead. Furthermore, current ML algorithms, including neural network architectures/libraries are primarily designed to solve computer vision problems. Adapting them to RMS tasks needs some degree of transformation of the way input and outputs are interpreted. Current AI-centric RMS algorithms transform their problem space and further use simple heuristics to interpret the result back and apply to the RMS problems. Such complexities demand expertise from many related domains. Thus, newer approaches, algorithms, standardised frameworks, and domain-specific AI frameworks are required to adopt AI in RMS efficiently.

Despite the challenges associated, machine learning-based solutions provide many opportunities to incorporate these techniques into RMS and benefit from them. This thesis explores different avenues where such techniques can be applied to manage Cloud data centres for energy and thermal efficiency.

2.3 Taxonomy of Energy Thermal-aware Resource Management in Cloud Data centres

As discussed earlier, cooling and computing are two main subsystems that contribute a significant amount of energy. In that regard, many works have focused on optimising these two systems with different methods. However, some works have also focused on

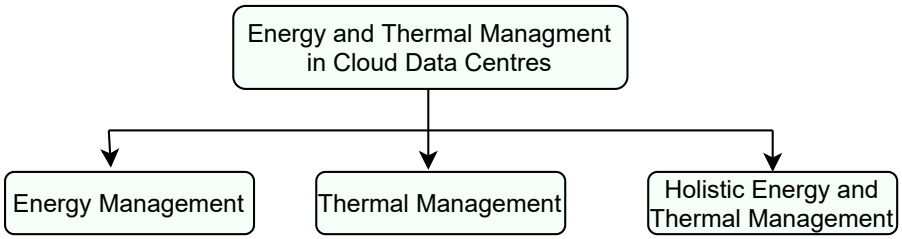


Figure 2.2: A High Level Taxonomy of Energy and Thermal-Aware Resource Management Approaches

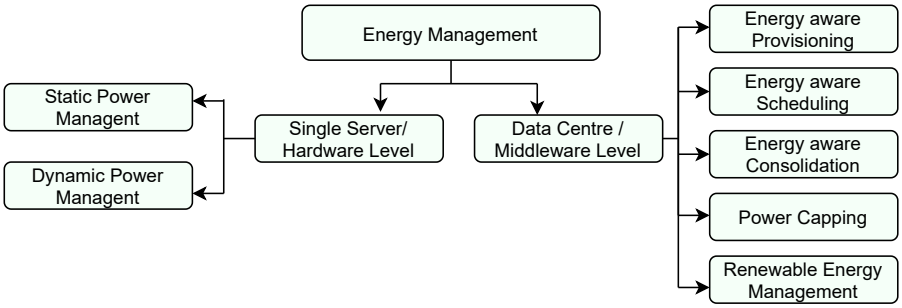


Figure 2.3: Taxonomy of Energy Management in Cloud Data Centres

holistic optimisation of two systems by co-ordinating each other, techniques like power budget shifting and workload scheduling. A high-level taxonomy of these can be found in Figure 2.2. In the following Sections, we review current research works and propose taxonomy covering different optimisation techniques in each of these three categories (energy, thermal and holistic resource management). We focus on server level and data centre level solutions concerning energy, thermal aspects in resource management techniques.

2.4 Energy Management

Many researchers have focused on increasing the energy efficiency of data centres with various resource management techniques. These techniques cover from an individual server to geo-distributed data centres. Taxonomy on the data centre's energy management solutions is presented in Figure 2.3. We categorise these solutions into two broad categories, i.e., single server level and data centre level solutions. Accordingly, we identify the essential techniques used in these two categories and briefly review their meth-

ods.

2.4.1 Server Level

In a computing server, CPU is a predominantly consumes a significant amount of energy. Modern rack-mounted data centre servers consume more than 1000 watts of power. Hence, managing this high power consumption is a challenging task. This server level power management has been mostly left to the operating system and its device drivers that communicate with underlying hardware signals and manage the server power. Server level power management can be broadly categorised into two levels, static and dynamic power management. Static power management deals with minimising leakage power while dynamic power management deals with regulating active runtime power based on utilisation level.

Static Power Management

The silicon chip has static power consumption which is independent of the usage level. The static power mainly accounts for leakage of current inside active circuits. To some extent, static power consumption is unavoidable; however, it can be minimised with better design and processes. There are many solutions from a lower level from circuit level, and architectural techniques [62]. The general approach in managing leakage is with different sleep states of CPUs when the system is idle. For instance, Intel X86 architecture has (C0-C4) sleep states indicating C0 is an active state while C4 is a deep sleep state where most of the CPUs' components are turned off to avoid the static power consumption.

Dynamic Power Management

A large part of silicon chip-based computing elements either in CPU or GPUs spend on dynamic power. Dynamic power represents runtime energy based on workload utilisation level. CPUs operate at a different frequency to regulate the dynamic power. If the operating frequency of a CPU in highest, then it's dynamic power consumption will

also be higher. The frequency is regulated based on utilisation level and workload requirements to increase their speedup. Dynamic Voltage Frequency Scaling (DVFS) is a popular technique to regulate the dynamic power in modern systems [63]. The dynamic power can be defined as below:

$$P_{dynamic} \ \alpha \ V^2F \tag{2.1}$$

In Equation 2.1, F is the frequency, and V is the supply voltage to the processor. Based on the frequency, the voltage is regulated, and some frequency ranges usually have a similar. If a CPU should be at its highest speed or frequency should be set to a higher level, thus consuming the more power. The operating system scales frequency based on its workload and application demands in runtime.

There are many solutions proposed that intend to optimise energy efficiency through DVFS techniques at the data centre level. These solutions include DVFS-aware VM scheduling, and consolidation [64] [65], placement of application based on DVFS capabilities [66], data centre level task scheduling by synchronising the frequency scaling among multiple machines [67]. ML-based techniques have also been explored recently in DVFS optimisations. Authors in [68] proposed ML-based CPU and GPU DVFS regulator for compute-heavy mobile gaming application that coordinates and scales frequencies with performance and energy improvements.

2.4.2 Data Centre Level

A significant amount of energy efficiency can be achieved when data centre level platforms incorporate energy-efficient resource management policies. Distributed data centre applications span hundreds of machine in geo-distributed data centres, hence, providing energy efficiency holistically across data centre resources and applications is more feasible and yields better results. In this section, we discuss important techniques for data centre level energy-efficient solutions.

Energy-aware Provisioning

Cloud data centre offer computing resources in terms of Virtual Machines (VMs) or containers. Allocating the required amount of resources for the application need is vital to satisfy the SLAs. However, overprovisioning of resources may yield higher energy consumption, and monetary cost to the users while underprovisioning will potentially violate the SLAs. Many researchers have proposed energy-aware resource provisioning techniques. Authors in [69] investigated energy-aware resource allocation for scientific applications. The proposed system EnReal leverages the dynamic deployment of VMs for energy efficiency. Similarly, Li et al. [70] proposed an iterative algorithm for energy-efficient VM provisioning for application tasks. Beloglazov et al. [27] propose various heuristic algorithms for resource allocation policies for VMs defining architectural principles.

Some researchers have also proposed data-driven methods for resource provisioning. Mehiar et al. [71] offered clustering and prediction based techniques; they used K-means for workload clustering and stochastic Wiener filter to estimate the workload level of each category accordingly allocate resources for energy efficiency. Recently Microsoft has proposed Resource Control (RC) [44], where they trained ML models to output predictions like VM lifetime, CPU utilisation, maximum deployment of VMs. These predictions use various resource management problems for better decision-making, including resource provisioning with the right container size for applications.

Energy-aware Scheduling

Scheduling is a fundamental and essential task of a resource management system in Cloud data centres. It addresses the following question, given an application or set of VMs (considering application runs inside these isolated VMs), when and where to place these VMs/application among available physical machines. This decision depends on several factors, including application start time, finish time, and required SLAs. In addition, workload models, whether an application is long-running (24 $\times$7) web application, or a scientific workflow model of which it's tasks need to be aware of precedence constraints, or applications based on IoT paradigm that is predominantly event-driven.

Although one can optimise numerous scheduling parameters, many recent studies have focused on energy optimisation as a priority in Cloud data centre scheduling.

Chen et al. [72] propose energy-efficient scheduling in uncertain Cloud environments. They propose an interval number theory to define uncertainty, and a scheduling architecture manages this uncertainty in task scheduling. The proposed PRS1 scheduling algorithm based on proactive and reactive scheduling methods optimises energy in independent tasks scheduling. Similarly, Huang et al. [73] investigate energy-efficient scheduling for parallel workflow application in Cloud. Their EES algorithm tries to slack non-critical jobs to achieve power saving by exploiting the scheduling process's slack room. Energy-efficient scheduling using various heuristics for different application model has been widely studied topic in literature [74] [75] [76].

Machine learning-based solutions are also explored in data centre scheduling focusing on energy efficiency. Some solutions rely on predictive models and then use them in scheduling algorithms, while other techniques model scheduling as a complete learning-based problem using t Reinforcement learning (RL). Berral et al. [77] adopt many ML-based regression techniques to predict CPU load, power, SLAs and then use these in scheduling for better decisions. These solutions still use some level of heuristics with integrated prediction models. However, RL-based scheduling is designed to learn and take actions in a data centre environment without external heuristics. Cheng et al. [36] proposed DRL-based provisioning and scheduling for application tasks in the data centre.

Energy-aware Consolidation

Cloud data centre are designed to handle the peak load to avoid potential SLA violation or overload conditions. Hence, the resources are oversubscribed to manage such an adverse situation. However, this oversubscription leads to resource underutilisation in general. It is estimated that Cloud data centres utilisation level is around 50% on average. Under utilisation of resources is the main factor in the data centre's energy inefficiency as idle or lower utilised servers consume significant energy (up to 70% [29]). Thus, it is necessary to manage workloads under such oversubscribed and under-

utilised environments. To that end, consolidation has been a widely used technique to increase energy efficiency. It aims to bring the workloads (VMs and containers) from underutilised servers and consolidate them on fewer servers, thus allowing remaining servers to be kept in sleep/shut down mode to save energy. Many challenges exist in consolidation, including maintaining VM-affinity, avoiding overutilisation, minimising SLA violation, and reducing application downtime due to workload migrations.

Beloglazov et al. [27] proposed various heuristics to consolidate the workload and answer the question, including which VMs to migrate, where to migrate and when to migrate to reduce potential SLA Violation. Many other solutions have broadly focused on energy efficiency along with optimising different parameters (cost reduction, failure management, etc) while consolidating workloads in data centre [78] [79] [80].

Data-driven solutions are predominantly used in consolidation [26] [81]. Hsieh et al. [81] studied VM consolidation to reduce power cost and increase QoS. They predict the utilisation of resources using Gray-Markov-based model and use the information for consolidation. Similarly, authors in [26] also use prediction for consolidation. They predict memory and network usage and perform consolidation of VMs in a data centre along with CPU. Few researchers have also used RL in energy-aware consolidation [82] [83]. Basu et al. [83] proposed Megh— a system that learns to migrate VMs in the data centre using RL. It proposes the dimensionality reduction technique using dimensional polynomial space with a sparse basis to minimise the state-space in their problem. Their system has shown that it achieves better energy efficiency and cost reduction compared to existing heuristics.

Power Capping

Data centres are designed to handle the peak power consumption based on the workload and cooling system requirements. Hence, in general, data centres are under-provisioned with power. This power capping on data centre servers restricts the amount of energy available to individual servers even though they can consume their maximum limit, thus providing required speed for workloads [84]. Managing resources and workload effectively in these power-constrained environments is necessary. It is essential to avoid

power inefficiencies in limited power allocated across servers to achieve power proportional computing [85].

In this regard, different power capping mechanisms at Cloud data centre level are studied. Authors in [86] proposed a fast decentralised power capping (DPC) technique to reduce latency and to manage power at the individual server. Dynamo [87] is the power management system used by Facebook data centres, which has hierarchical power distribution. The lowest level leaf controller regulates power in a group of servers. This leaf controller based on high-bucket-first heuristic determines the amount of energy to be reduced in each server to meet the power cap limits to which it is constrained. It also considers workload priorities and avoids potential performance degradation due to its power capping. Some researchers have investigated controlling peak power consumption [88] by designing feedback controller, which periodically reads system-level power and configures highest power state of servers keeping server within its power budget. Authors in [89] studied optimal power allocation in servers, which accounts for several factors including power-to-frequency, the arrival rate of jobs, maximum and minimum server frequency configuration. They have shown that allocating full power may not always result in the highest speed as expected. Some techniques have also explored enabling data centre service providers to dynamically manage the power caps by participating in an open electricity market and achieve cost and energy efficiency [90]. However, due to close interconneciton between power capping effect on CPU speed, thermal dissipiation and also presence of heterogenety in servers and workloads, data centre level power capping workload management is a difficult task to achieve [91] as compared to other energy efficiency methods that are discussed in this chpater.

Renewable Energy Management

Data centres consume colossal energy and contribute significantly to greenhouse gas emissions (CO_2). Data centre service providers continuously increase renewable or green energy (solar, wind) usage with minimal carbon footprints to decarbonise the data centres. However, green energy usage in the data centre is extremely challenging due to

its intermittent nature availability. In contrast, the Cloud data centre needs uninterrupted power supply since Cloud workloads tend to run 24×7. Therefore, managing workloads under the uncertain availability of renewable energy is a challenging research problem.

Several resource management techniques explored maximising renewable energy in data centres. They include workload shifting and placement across geo-distributed data centres [92] [93] [94] based on their carbon efficiency. Besides, delaying job execution if an application can tolerate the QoS [95] and job dispatching or load balancing workloads to match the available renewable energy at different data centres [96] are some popular techniques in this regard.

Machine learning-based algorithms are promising in renewable energy management, as predicting the available green energy based on an environmental condition is crucial in workload management [97]. Along with prediction models, RL methods are also used to solve optimisation problems in increasing green energy usages in data centres [98].

2.4.3 Summary of Energy Management in Data Centres

To achieve significant energy efficiency in data centres, we need algorithms and software systems that manage resources and workloads across different computing layer. In addition to the energy management techniques we discussed in this section, researchers propose various other solutions. The proposed solutions cover designing more energy-efficient processor architecture, building middleware platforms that manage resources efficiently, and finally including energy efficiency in the software development process, itself [99] [100]. As data centre systems' complexities increase, machine-learning-based solutions are becoming predominant that either aid externally for different algorithms or directly taken action if modelled accordingly.

2.5 Thermal Management

Thermal efficient resource management in the data centre is vital to increase energy efficiency. It also helps manage resources and workloads reliably by avoiding device

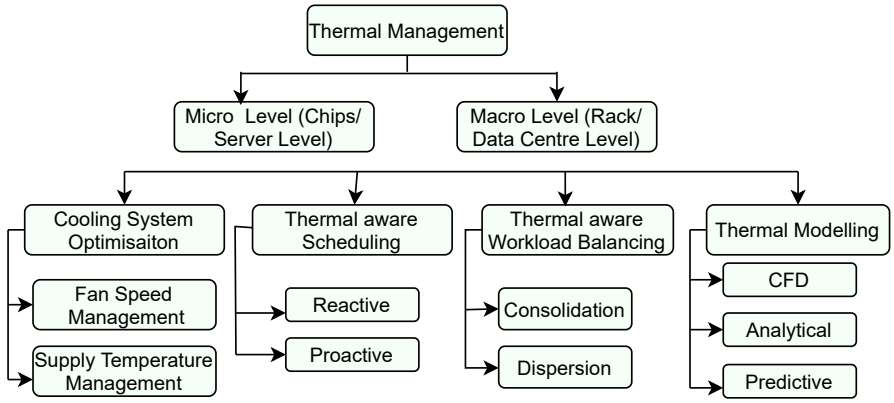


Figure 2.4: Taxonomy of Thermal Management in Cloud Data Centres

failures induced by peak temperatures and performance degradation due to thermal throttling. Similar to energy management, thermal management techniques span from an individual server to data centres. A taxonomy on thermal management solutions is presented in Figure 2.4. This section categorises these techniques into two broad categories, i.e., micro-level or single server level and macro-level or data centre level thermal management techniques. We describe and review essential approaches used in these two categories.

2.5.1 Server Level

Computing servers consume an enormous amount of energy and dissipate this energy as heat. It is crucial to keep processor or CPU temperature within the threshold limit to avoid damage to the processor's silicon components, thus permanently producing catastrophic device failures. Modern rack servers reach peak temperature up to 90-100°C. In reality, the processor speed of servers is limited by their thermal management capacity. Generally, onboard fans are responsible for taking out heat from the server cabinet to the outside ambient environments in data centres.

Like DVFS in energy management, its corresponding thermal dissipation is regulated in servers by controlling the amount of power consumed. Dynamic Thermal Management (DTM) [101] is a popular thermal management technique at the individual server level which regulates Multiprocessors Systems-on-chip (MPSoCs) performance,

power consumption, and reliability. This is controlled at the operating system level by closely communicating with underlying hardware interfaces. If a server's temperature is potentially exceeding the predefined threshold, the operating system takes major by employing thermal throttling mechanisms that reduce the energy consumption, thus reducing the CPU speed. Moreover, techniques like application scheduling [102] [103], optimal onboard fan speed configuration [104] techniques are employed for energy and thermal efficiency at server level.

Machine-learning based solutions are recently used to optimise temperature management at individual server level [105]. For instance, Iranfar et al. [106] investigated how to proactively estimate the required number of active cores, operating frequency, and fan speed. Accordingly, the system is configured to achieve reduced power consumption.

The server-level thermal management involves solutions including processor architecture design, manufacturing technology and resource management solutions within the operating system, including server fan control and others. As our focus is entirely on data centre solutions, we do not delve into server-level thermal management.

2.5.2 Data Centre Level

A typical large scale data centre hosts thousands of servers. Data centre servers are arranged in rack-layout, where each rack (e.g., standard 42U rack) can accommodate 10-40 rack blade servers based on vendor-specific dimensions. This high density of equipment makes data centre one of the highest-energy density physical infrastructures. Disipiated heat from these rack server can result in data centre ambient temperature to reach extremely high. Thus, cooling systems in data centres make sure that data centre temperature is within the threshold. Many approaches exist optimising different parameters to reduce cooling energy. In this section, we review and describe data centre level thermal management techniques.

Cooling System Optimisation

Traditional rack layout data centres have Computer Room Air Conditioning (CRAC) cooling system that blows cold air to the racks across data centre (more details of cooling technologies can be found in Section 2.7). The entire cooling system efficiency requires multiple parameters to be configured in the design and operational phase. In the design phase, efficiency can be increased by better physical layout and vent designs to reduce heat recirculations. While runtime cooling energy efficiency can be increased by fine-tuning the fan speeds of CRAC systems and cold air supply temperature which mainly determines the cooling system energy consumption [107] [108] [109]. In this section, we focus on runtime cooling system optimisation.

Fan Speed Management: Within the CRAC system, fans are used to regulate the airflow rate within the data centre. It is important to note that these fan speeds are separate from the onboard server's fan equipped to eject heat from CPU to outside of the server cabinet. Increasing airflow require higher fan speeds, thus consuming more energy. Hence, regulating fan speed optimally can save a significant amount of cooling power. However, this depends on the status of the data centre, and it's temperature level. Many researchers have proposed solutions to optimally configure the CRAC's fan speed based on cooling load [110] [107] by monitoring thermal load in the data centre and accordingly varying fan speeds dynamically to reduce energy consumption.

Supply Temperature Management: CRAC system blows cold air to racks through vented floor tiles in the data centre to take out dissipated heat. Passing colder air requires higher energy consumption as chiller's in CRAC consumes energy to supply cold air. Hence, the inaccurate configuration of supply air temperature significantly affects cooling energy cost in the data centre. For a safer operation, the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) [111], recommends supply air temperature in the data centre to be in the range of 17-27 °C. Thus, it is beneficial to set the supply temperature closer to 27 °C. However, most data centres are overcooled as supply temperature in the data centre is set to much lower temperature conservatively, leaving energy inefficiencies in the cooling system. Setting a higher supply air temperature requires careful handling of peak temperature in data centres.

Many solutions have been proposed to raise the supply air temperature. Zhou et al.

[112] have shown that significant power saving can be achieved when the workload is managed efficiently and allowing supply air temperature to be increased. In essence, to raise supply air temperature, data centre peak temperature should be minimised. It can be done through various means, including thermal aware workload scheduling, and avoiding thermal imbalance in the data centre.

Thermal-aware Scheduling

Workload scheduling in the data centre has a significant effect on cooling system efficiency [113]. If workload scheduling strategy results in peak temperature in the data centre, it generates a higher thermal load, thus increasing cooling cost. To address this, many researchers have proposed thermal-aware scheduling methods in Cloud data centres. Some solutions are proactive, which intend to avoid adverse temperature effects beforehand. In contrast, some scheduling policies follow reactive approaches. If a temperature violation is found, workloads are rescheduled to other nodes; however, the reactive scheduling method may result in higher QoS violation for application due to rescheduling and migration. Mhedheb et al. [114] investigated load and thermal aware scheduling in Cloud that optimises temperature and load while scheduling tasks in data centres. Sun et al. [35] proposed thermal-aware scheduling of HPC jobs. They have used analytical models to estimate server temperature and model heat recirculation in the data centre. Proposed thermal aware job assignment heuristics have shown that increased performance with thermal balancing. Furthermore, authors in [115] have further extended thermal aware batch job scheduling across geo-distributed data centres.

Many of the existing works have employed machine-learning-based techniques in thermal-aware scheduling. Wang et al. [116] proposed Artificial Neural Networks (ANN)-based temperature prediction model and used it for task prediction in data centres. The results have shown that machine learning models can capture the thermal phenomenon in a data centre.

Thermal-aware Workload Balancing

In Cloud data centres, thermal agnostic placement of workload triggers adverse temperature effect. Hence, balancing the workloads thermal efficiently yields better efficiency, consolidation and workload dispersion are two popular techniques in workload balancing. ***Workload Consolidation:*** Consolidation is a widely used technique to optimise computing system's energy consumption. However, aggressive consolidation leads to the creation of hotspots that further increases cooling cost. Hence, thermal-aware consolidation is necessary to balance the computing and cooling system energy consumption. Many researchers have proposed many solutions for this [117] to balance the temperature response due to workload placement. ***Workload Dispersion:*** Opposite to consolidation, workload dispersion technique aims to spread out workloads evenly across data centre's servers [118]. It has shown to be thermal efficient workload management as it minimises temperature in a data centre, avoiding servers to reach peak utilisation. Although it minimises peak temperature, it significantly increases the computing system energy due to resource underutilisation. Hence, there should be a balance between consolidation and workload dispersion techniques to achieve cooling system efficiency.

Thermal Modelling

Thermal modelling in data centre plays a vital role in resource management. Thermal modelling includes capturing thermal behaviour in a data centre and accurately estimating server temperature. Thermal models that predict accurately and fastly are useful aids in scheduling, configuring cooling system and other resource management techniques. However, temperature prediction is a difficult problem. Server ambient temperature in a data centre depends on multiple factors including CPU heat dissipation, inlet temperature and complex heat recirculation effects. There are mainly three types of thermal modelling techniques in data centres: (1) Computational Fluid Dynamic (CFD)-based models; (2) Analytical models; and (3) Predictive models.

CFD: The CFD models accurately captures the room layouts, heat recirculation effects and and accurately estimates temperature in data centre [32] [119] [33]. However, they are computationally expensive, and even a single calibration requires models to be run

for multiple days. Hence, they are incapable of using resource management systems that require for its fast online decisions.

Analytical: These models depend on modelling data centre and workloads based on mathematical frameworks [34] [35]. They represent cooling, computing and workload elements with formal mathematical models and build a framework to establish relationships between all elements [35]. Although they are fast in temperature estimation, the accuracy is compromised due to their rigid static models.

Predictive: ML-based models use actual measurement data from the data centre to predict the accurate temperature of the server. These data-driven models, once trained, are accurate, and quickly deliver the results in runtime. Moreover, they can automatically model physical layout, air conditioning and the heat generated by Cloud data centres. Unlike CFD's where each of these needs to be modelled explicitly, this is a huge benefit. To that end, Wang et al. [116] proposed server temperature prediction model using Artificial Neural Network (ANN) based ML technique, results have shown that it can accurately predict the temperature in data centres. In addition, some studies have explored using machine learning models to identify temperature distribution [120], and to predict server inlet temperature [121].

The drawback of the data-driven model is that the model is only applicable to the data centre where the data is collected from. This means data need to be collected for each data centre extensively. However, this is not a massive disadvantage as such data need to be collected to monitor the data centres' health.

Sumary of Thermal Management in Data Centre

Efficient thermal management in a data centre is essential for achieving energy efficiency and guaranteeing system reliability. In this section, we reviewed various thermal management solutions spanning individual server to data centre level methods. Compared to energy management, machine-learning-based approaches in thermal management is limited or less explored. However, there exist vast opportunities to incorporate learning-based solutions across thermal management stack in Cloud data centres.

2.6 Integrated Energy and Thermal Management

Traditionally cooling system and computing systems are optimised individually. However, these two subsystems in the data centre are closely interdependent and optimising one system often have a counter effect on others. Hence, the joint optimisation of two subsystems is beneficial. Many solutions have been proposed; the well-known technique is to make workload scheduling and cooling system optimisation a multiobjective optimisation problem. Accordingly configure different parameters aim to minimise energy consumption [122] holistically. Other techniques include CRAC fan speed management by interplaying with IT load and its heat dissipation, configuring supply air temperature, and distributing the workload to minimise peak temperature, among many others.

Wan et al. [123] studied holistic energy minimisation in data centres through a cross-layer optimisation framework for cooling and computing systems. This energy minimisation problem is formulated as a mixed-integer nonlinear programming problem. To solve this problem, authors proposed a heuristic algorithm called JOINT, that dynamically configures parameters (such as server frequency, fan speed, and CRAC supply air temperature) based on workload demand and minimises computing and cooling system energy holistically.

Li et al. proposed [124] joint optimisation of computing and cooling systems for energy minimisation in data centres by modelling IT systems interactions (load distributions) and it's corresponding thermal behaviour, i.e., heat transfer. The proposed analytical models for load distribution across rack servers to minimise computing and cooling system energy, thereby configuring different knobs of two systems while ensuring required throughput and resource constraints of workloads.

Power budget shifting is another important resource management techniques in Join optimisation of these two systems. Using available power to trade between two systems in runtime can increase energy efficiency and resource utlisaiton. PowerTrade [125] is a technique that trades-off data centre computing system's idle power and cooling power with each other to reduce total power. Overprovisioning is necessary for such condition to accommodate extra workload and use excessive power obtained.

Machine learning-based techniques have also been explored in joint optimisation of computing and cooling systems. Ran et al. [126] used DRL and designed hybrid action space that optimises the IT system and the airflow rate of the cooling system. Furthermore, the proposed control mechanism coordinates both the IT system's workload and cooling systems for energy efficiency.

2.7 Cooling Management Technologies in Data Centre

When servers/IT equipment uses electricity for their operations, the electrical energy is transferred as heat. This heat will be drawn across the server cabinet by the rear-mounted server fans within allowing heat to transfer from the server's components to the outside ambient environment. Many technologies are employed to take out this heat from the data centre environment and keep the data centre's operational temperature within its threshold. These cooling technologies can be broadly categorised into two categories, including air and liquid cooling technologies.

2.7.1 Air Cooling

Air cooling is widely used data centre cooling technologies due to their inexpensive and flexible design and operational conveniences. In rack-layout based data centres, the dissipated heat from servers is extracted from the cooling system's environment. The **Computer Room Air Conditioning** (CRAC) is a cooling system responsible for monitoring and managing the temperature in data centre [127]. The CRAC blows cold air through the perforated tiles under the racks of a data centre. The cold air passes from bottom to the top of rack taking out the dissipated heat from rack equipment sand this hot exhaust air is pushed to intake of the CRAC units to the ceiling of the room where it is taken out of the room. This allows separating hot exhaust air from the cold inlet air. The CRAC unit then transfers the hot exhaust air via a coil, to a fluid using refrigerant.

Many data centre also equip **Computer Room Air Handler (CRAH)**, where chilled water is used as fluid [127]. These fluids remove the heat from the data centre environment. The CRAC/CRAH continuously blow cold air using constant-speed fans, and

this return cold-air temperature also called inlet temperature. It is configured to manage the dynamic thermal threshold in the data centre. It directly controls the cost of cooling in general. Lower the inlet temperature higher will be the cooling energy cost due to increased energy required to transfer the lower temperature air from CRAC/CRAH. The American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) [111], a leading technical Committee in cooling system technology recommends that the device inlet be between 18-27C for the safe operation of the environment. The design goal of any data centre operators will be to provide the inlet temperature close to 27 C to reduce the cooling cost. However, the safer operation threshold should be maintained while configuring this parameter. Many works have looked into optimising this parameter using different techniques by minimising the peak temperature [34] by balancing the workloads [128] and optimally configuring other parameters [129] of the cooling system.

Some modern system also use **evaporative** [130] and **air side economisers/ free cooling** techniques [18]. In evaporative technique, instead of fluid refrigerant, the hot air carried from the data centre is directly exposed to water. Water evaporates, taking out the heat from the hot air. Cooling towers are employed to dissipates the excess heat to the outside atmosphere. However, it doesn't require expensive CRAC or CRAH units but needs a large amount of water, limiting factor in many data centre locations. On the other hand, air side economisers or free cooling methods use outside free air for direct cooling instead of depending on the fluids to cool down the hot air extracted from CRAC/CRAH. This saves a huge amount of cooling cost. Nonetheless, these techniques vastly depend on the weather and geographical condition where the data centre are located, and thus they are used in limited computing infrastructures in practice.

2.7.2 Liquid Cooling

The recent advancement in the data centre cooling technology has seen the adoption of liquid cooling as it is more efficient than air cooling in general [131]. The liquid cooling system also effectively avoids heat mix up and heat re-circulation issue, which is a common problem in air cooling techniques.

In **direct liquid cooling** system, liquid pipes are used to deliver liquid coolant directly to the heat sink present in the server's motherboards. The dissipated heat from the server is extracted to heat the chiller plant from these pipes, where the chilled water loop takes out the heat extracted from servers.

Immersion cooling. The computing system (servers and networking equipment are directly immersed in a non-conductive liquid. The liquid absorbs the heat and transfers it away from the components [132]. In some cases, equipment is arranged in isolated cabinets and immersed in tanks or cabinets are directly immersed in natural water habitats such as lakes/ocean. For instance, Microsoft has tested underwater data centre with their project Natick [133] which allows them to operate the data centre in an energy-efficient manner by leveraging heat-exchange techniques with outside water. This technique is commonly used in submarines. This experimental project shows that immersion cooling is viable in large scale computing systems with a group of servers sealed into large submarine cabinets.

Some other techniques have also explored but rarely used in large scale settings, such as Dielectric fluid, where server components are coated with a non-conductive liquid. The heat is removed from the system by circulating liquid into direct contact with hot components, then through cool heat exchangers. Such methods are not widely adopted yet in practice. The common issue with rack-level liquid cooling is a lack of standardisation and specifications among multi-vendors. However, due to its energy-efficiency compared to air cooling, it is expected that liquid cooling would become mainstream in future data centre cooling systems.

2.8 A Conceptual Machine Learning-based Resource Management System Framework

In the AI-centric or Machine Learning (ML)-based Resource Management Systems (RMS) system, models need to be trained and deployed for the RMS inference for different tasks. However, integrating data-driven models into Distributed Computing Systems (DCS) platforms in a scalable and generic manner is challenging and is still at a conception stage. In this regard, as shown in Figure 2.5, we provide a high-level architectural

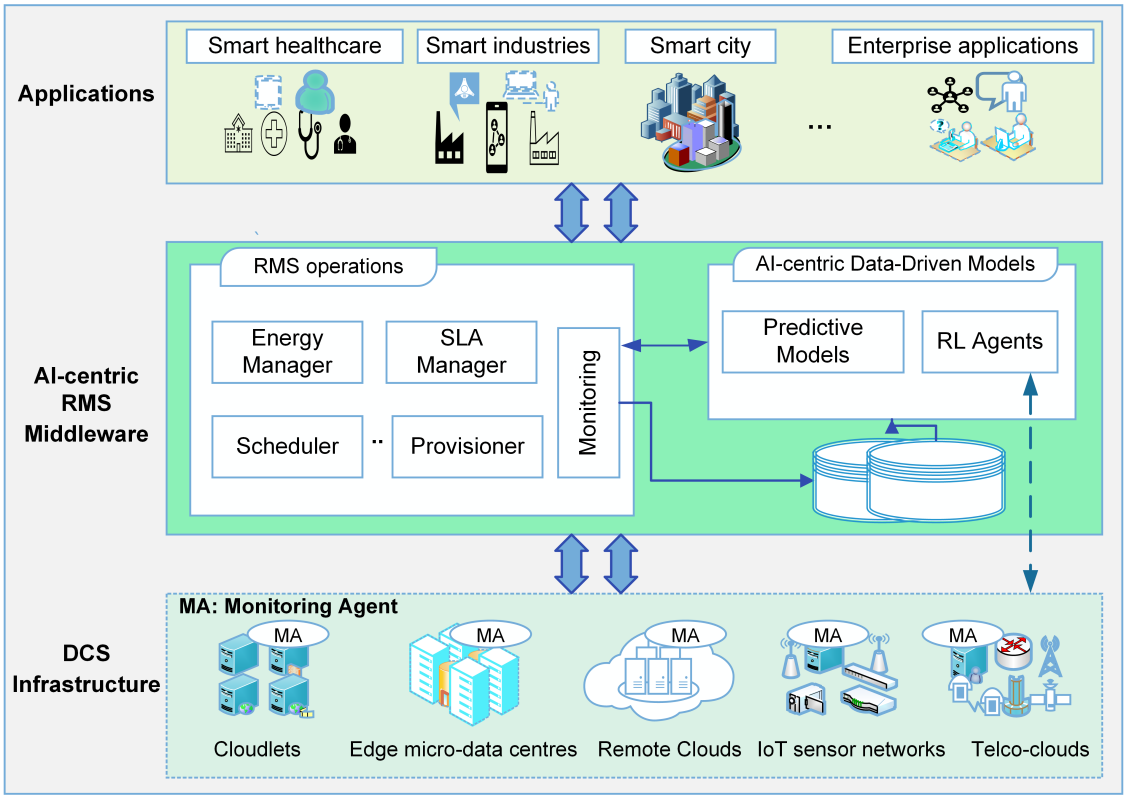


Figure 2.5: Conceptual Machine Learning based RMS Model

model for such data-driven RMS. Here, anticipating future Cloud systems, and workloads, we include different Cloud system infrastructures, including Remote Cloud, Edge Cloud, IoT infrastructures, etc. The essential elements of this system are explained below. It consists of three entities:

Users/ Applications: Users requiring computing resources or services interact with the middleware using APIs or interfaces.

AI-centric RMS Middleware: This is responsible for performing different tasks related to managing user requests and underlying infrastructure. The AI-centric RMS tasks continuously interact with the data-driven models for accurate and efficient decisions. The RMS needs to perform various tasks, including provisioning the resources, scheduling them on appropriate nodes, monitoring in runtime, dynamic optimisations like migrations, and consolidations [44] to avoid the potential SLA violations. The data-driven AI models are broadly categorised into two types, (1) predictive models and (2) adap-

tive RL models. In the former, models are trained offline using supervised or unsupervised ML algorithms utilising historical data collected from the DCS environment that includes features from resources, entities, and application services. This data is stored in databases, and data-engineering is done, such as preprocessing, cleaning, normalising, to suit AI models' requirements. Thus, this offline training can be done on remote Cloud nodes to benefit from the specialised, powerful computing resources. The trained models can be deployed on specialised inference devices like Google Edge TPU and Intel Habana. Choosing the optimal place and deciding where to deploy these ML models depends on where the RMS engine is deployed in the environment, and this is itself a challenging research topic that should be addressed as described in Section 2.2.1. In the latter case, runtime adaptive models such as Reinforcement Learning (RL) that continue to improve their policies based on agents' interactions and system feedback. It requires both initial learning and runtime policy improvement methods that need to be updated after every episode (certain time reaching to terminal state). The RMS operations can interact with both the predictive and RL-based data-driven models using the RESTful APIs in runtime [44].

DCS Infrastructure: The computing infrastructure comprises heterogeneous resources, including sensors, gateway servers, edge data centres, and remote Clouds. Therefore, adopting the learning-based RMS models needs a significant change in the way current RMS systems are designed and implemented, as well as monitoring agents, interfaces, and deployment policies that can be easily integrated into existing environments.

2.9 Summary

Cloud computing platforms are massively complex, large scale, and heterogeneous, enabling the development of highly connected resource-intensive business, scientific, and personal applications. Data centres have become a backbone infrastructure of Cloud. Holistic energy and thermal management in such complex infrastructure has become a challenging task. The state-of-the-art rule-based or heuristics resource management solutions have become inadequate in modern Cloud data centres. The RMS policies need to deal with massive scale, heterogeneity, and varying workload requirements. Hence,

we require data-driven AI approaches that derive key insights from the data, learn from the environments, and take resource management decisions accordingly. In this chapter, we have discussed the challenges associated with adopting AI-centric solutions from the perspectives of energy and thermal management. We provided taxonomy for energy and thermal management in Cloud data centres. Finally, we presented the conceptual AI-centric RMS model.

Chapter 3

Data-Driven Frequency Scaling and Scheduling on Graphics Processing Units (GPUs)

Modern computing paradigms, such as Cloud computing, are increasingly adopting GPUs to boost their computing capabilities primarily due to the heterogeneous nature of AI/ML/deep learning workloads. However, the energy consumption of GPUs is a critical problem. Dynamic Voltage Frequency Scaling (DVFS) is a widely used technique to reduce the dynamic power of GPUs. Yet, configuring the optimal clock frequency for essential performance requirements is a non-trivial task due to the complex nonlinear relationship between the applications runtime performance characteristics, energy, and execution time. It becomes more challenging when different applications behave distinctively with similar clock settings. Simple analytical solutions and standard GPU frequency scaling heuristics fail to capture these intricacies and scale the frequencies appropriately. In this regard, we propose a data-driven frequency scaling technique by predicting the power and execution time of a given application over different clock settings. We collect the data from application profiling and train the models to predict the outcome accurately. The proposed solution is generic and can be easily extended to different kinds of workloads and GPU architectures. Furthermore, using this frequency scaling by prediction models, we present a deadline-aware application scheduling algorithm to reduce energy consumption while simultaneously meeting their deadlines. We conduct real extensive experiments on NVIDIA GPUs using several benchmark applications. The experiment results have shown that our prediction models have high accuracy with the average RMSE values of 0.38 and 0.05 for energy and time prediction, respectively. Also, the scheduling algorithm consumes 15.07% *less energy as compared to the baseline policies.*

3.1 Introduction

Graphics Processing Units (GPUs) have become ubiquitous in modern computing paradigms and platforms, such as Cloud computing and supercomputing environments, due to their massive computational capabilities. Furthermore, the Single Instruction Multiple Data (SIMD) architecture of GPUs is ideally suitable for many parallel and compute-intensive scientific and business workloads [134, 135].. These advantages manifested into the deployment of a large number of GPU clusters in many data centres, including Top500 supercomputers and also in the public Clouds [136–139]. In spite of this increased usage, the power consumption of GPUs has become a significant bottleneck for designing hyper-scale GPU systems [140, 141]. On the other hand, GPU workloads are more sensitive to their Quality of Service (QoS) constraints requiring faster execution and thus spending more energy. Therefore, energy-efficient workload management with QoS satisfaction is exceedingly essential.

Dynamic Voltage Frequency Scaling (DVFS) is a popular technique to reduce active power by varying the GPU frequencies [142] [143] [144]. The modern GPUs have two frequency domains, core and memory, each with many numbers of frequency ranges. While former regulates the Streaming Multiprocessors (SM) (including register, texture cache, shared memory, and l2 cache), and the latter governs bandwidth of DRAM [144]. For instance, NVIDIA Tesla P100 GPU supports one memory frequency (715 MHz) and 62 core frequencies ([544-1328] MHz), and NVIDIA GTX 980 supports four memory frequencies ([3505-324] MHz) and 87 different core frequencies([135-1428] MHz) with the total number of 267 possible frequency combinations. A particular combination of memory and core frequency can be set using the NVIDIA Management Library (NVML). However, the principle DVFS notion- higher frequency range increases the performance requiring more power, while lower frequency consumes less power by decreasing the performance do not hold in all the scenarios [145]. In addition, different GPU applica-

This chapter is derived from:

- **Shashikant Ilager**, Rajeev Muralidhar, Rammohanrao Kotagiri and Rajkumar Buyya, "A Data-Driven Frequency Scaling Approach for Deadline-aware Energy Efficient Scheduling on Graphics Processing Units (GPUs)", *In Proceedings of the 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGrid 2020)*, Melbourne, Australia, May 11-14, 2020. **[Best Paper Award]**

tion kernels behave differently concerning energy and performance with the frequency settings due to their different resource footprints and the intricate instruction execution patterns. Thus, due to such non-linear dependencies, estimating and optimally scaling the frequencies for a given application is non-trivial.

Furthermore, frequency scaling becomes more challenging when a scheduler needs to schedule multiple applications with their deadline requirements. In such a case, the scheduler should not only identify the energy-efficient frequency combinations, but it also needs to take care of the applications execution time. Such scenarios are highly prevalent in real-time HPC and Cloud environments [146, 147].

However, existing analytical and heuristic-based GPU frequency scaling [148, 149] methods are inefficient as they fail to capture the complex non-linearity between the frequency settings, performance, and power. To that end, data-driven DVFS scaling is a promising technique that is built using actual measurements. Models built using such methods can accurately scale the frequencies based on application demands[63]. Moreover, once the model is trained, the new applications can be scheduled on-the-fly with minimum profiling data.

In this chapter, we present a data-driven approach for frequency scaling by observing key architectural, power, and performance counters and predicting the estimated application power and execution time. In addition, guided by these prediction models, we propose a deadline-aware energy-efficient scheduling algorithm that accurately scales the GPU frequency according to the application requirements. We use twelve applications for evaluation from two standard GPU benchmarking suites, Rodinia [150] and Polybench [151]. The training data is generated from profiling the applications using *nvprof*, a standard profiling tool from NVIDIA. Furthermore, several machine learning models are explored to accurately predict the energy and execution time of applications for the given frequency domains. Based on the experimental results, CatBoost, an ensemble-based gradient boosting learning model, is chosen for prediction modeling. We implement the prototype scheduling system and evaluate the proposed techniques on real platforms. The experimental results conducted on the NVIDIA GPU device, Tesla P100 (Pascal micro-architecture), have shown that our prediction models have high accuracy and proposed scheduling algorithm consumes less energy as compared to the

baseline algorithms.

In summary, we make the following key **contributions**:

- We propose a data-driven prediction model to accurately predict the energy and execution time of applications to assist the efficient frequency scaling configuration by observing key architectural, power, and performance counters and metrics.
- We design and present a deadline-aware energy-efficient application scheduling algorithm using the prediction models.
- We implement a prototype system and evaluate the proposed solution on a real platform using standard benchmarking applications
- We show the efficiency and efficacy of our proposed solution with extensive experiments, and results are compared and analyzed with the existing state-of-the-art solutions.

The remainder of the chapter is organized as follows. Section 3.2 related work, Section 3.3 describes the DVFS background, motivation of the work, and system model. Section 3.4 presents the data-driven frequency scaling techniques. Section 3.5 shows our proposed deadline aware energy-efficient scheduling algorithm. Section 3.6 describes the performance evaluation with the analysis of the results. Finally, Section 3.7 summarises the chapter.

3.2 Related Work

Several researchers have studied a different aspect of GPU DVFS optimization. The existing GPU frequency and performance estimation models can be classified into three types. First, the analytical models [152] [153], [154], which uses the mathematical relationships between different system components and workload characteristics. Second, static models [148, 149], usually constructed using source code level metrics or static hardware specifications. Finally, machine learning models [145, 155], where different predictive models are employed to estimate the required parameters accurately.

Losch et al. [156] present an accurate analytical energy model for a task execution on heterogeneous nodes by characterizing the application execution and energy model. Some works have also explored techniques like power capping and scheduling [153, 157] for energy optimization using DVFS and task mapping. The authors in [157] have used empirical, analytical models to configure the CPU-GPU frequency to execute applications within a power budget. Chau et al. [153] have studied energy-efficient job scheduling in CPU-GPU systems by regulating the DVFS. The authors proposed analytical approximation algorithms with linear programming (ILP) model and introduced a heuristic algorithm to solve this problem. Although analytical models are fast, they fail to accurately estimate the intended metrics due to their sensitivity to different parameters involved in the modeling.

The static models rely mostly on the source-code or compiler level metrics to build the models. Wang et al. [148] proposed a hybrid framework for fast and accurate GPU performance estimation through source-level analysis. They used a total of 23 parameters collected from the hardware specifications, simulation traces, and the source code.

Fan et al. [149] also studied predicting the energy and performance using the static source code features from several real and synthetic open-CL kernels. Although their prediction model relies on ML techniques (Support vector regression- SVR), the training data is collected from the static source code features. They use Pareto-set of frequency configurations to find the optimal scaling values between speedup and energy further. However, models built using static features perform poorly when applied to different GPU architectures as each device has a different response to the energy and execution time. Therefore, it is beneficial to build models with actual data from the real-platforms.

Machine Learning (ML) models have been used by researchers recently in GPUs DVFS management. Wu et al. [155] proposed a neural network model to estimate the scaling curve of application with different hardware configurations. While their objective is tuning different hardware parameters, we instead focus more on configuring the frequency domains to facilitate the efficient DVFS for application execution. Similarly, Guerreiro et al. [145], investigated the DVFS-aware application classification to improve GPU efficiency. They characterize the applications using the nine different application profiling features and classify the workloads based on the hierarchical clustering and

neural network classifier. Our approach is different where we predict the energy and time with varying settings of the clock while this method classifies application into different domains and optimize accordingly. Furthermore, Tang et al. [158] carried out an empirical study of GPU DVFS on energy and performance of deep learning workloads. They analyze the effect of DVFS with different core frequencies while training the deep neural networks on NVIDIA $GTX2080T_i$. The empirical results have shown that optimal frequency settings can significantly save energy consumption. Most of these works focus on a single objective. However, in this work, we propose a data-driven frequency scaling approach for the deadline-aware scheduling algorithm.

3.3 Background Motivation and System Model

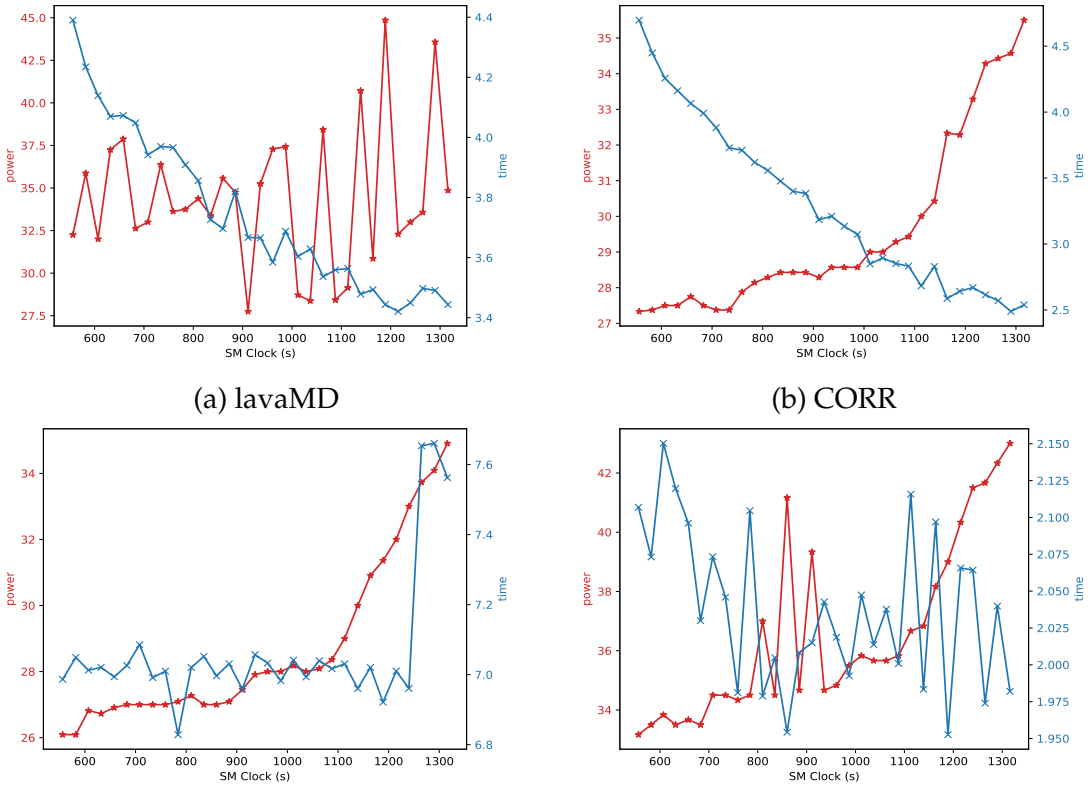
3.3.1 GPU DVFS

The power consumption of a GPU (or any semiconductor logic block in general) is a combination of dynamic ($P_{dynamic}$) and static power (P_{static}).

The static power (P_{static}) is related to the leakage and energy consumed when the system is idle, and usually it is managed by hardware and/or software using different sleep states [159]. However, large amounts of energy is spent on the dynamic power ($P_{dynamic}$) which is proportional to the run time of the workload. Performance management of GPU typically rely on the DVFS-based heuristics to regulate the dynamic power to save the energy. The frequency is normally regulated based on the application's utilisation parameters or system's temperature threshold throttling mechanisms [63]. Thus, the dynamic power can defined as below:

$$P_{dynamic} \ \alpha \ V^2F \tag{3.1}$$

In Equation 3.1, F denotes the operating frequency, while V denotes the supply voltage. Based on the current operating frequency, a combination of hardware and software changes the frequency (and thereby the underlying voltage); certain frequency ranges can share the same voltage level. Furthermore, GPUs have multiple frequency domains



(a) lavaMD (b) CORR

(c) myocyte (d) SYR2K

Figure 3.1: Power, time and clock relationship of different applications

$F = \{f_{smclock}, f_{memclock}\}$, regulating hardware components related to streaming multiprocessor or graphics processor and the memory components, respectively [145]. Thus, considering that, usually, hardware logic manages the voltage based on operating frequency, we focus on benefiting from regulating frequency and scaling it based on application behaviors.

3.3.2 Motivation

Estimating the optimal frequency is a non-trivial problem due to the complex behaviors of applications regarding their energy consumption and execution time. To analyze this complexity, we plot the behavior of different applications towards energy and execution time by changing the core frequency of the GPU, as shown in Figure 3.1. These execu-

Application	Domain/Description	Suite	Input
particlefilter_naive	Medical Imaging	Rodinia	-x 128 -y 128 -z 10 -np 1000
particlefilter_float	Medical Imaging	Rodinia	-x 128 -y 128 -z 10 -np 1000
myocyte	Biological Simulation	Rodinia	10000, 1000, 1
lavaMD	Molecular Dynamics	Rodinia	-boxes1d 50
Backprop	Pattern Recognition	Rodinia	983040
SYRK	Symmetric rank k operations	Polybench	M 1024, N 1024
SYR2K	Symmetric rank 2k operations	Polybench	M 2048, N 2048
GEMM	Matrix Multiply C = α A x B + β C	Polybench	NI 2048, NJ 2048, NK 2048
COVAR	Covariance Computation	Polybench	M 2048, N 2048
CORR	Correlation Computation	Polybench	M 2048, N 2048
ATAX	Matrix Transpose and Vector Multiplication	Polybench	NX 16384, NY 16384
2MM	2 Matrix Multiplications (D=A.B; E=C.D)	Polybench	NI 4096, NJ 4096, NK 4096, NL 4096

Table 3.1: Description of applications

tions are from NVIDIA Tesla P100 GPU that has one memory frequency and 62 core frequencies. So we only vary the core frequencies. In Figure 3.1, we can observe that when the core frequency increases, energy consumption is not always linear. And also, the lavaMD (Figure 3.1.a) application has a completely inconsistent response to frequency variations. Furthermore, some application produces different functionalities between certain frequency range, for application CORR (Figure 3.1.b), energy consumption has a non-convex curve between [730-920] MHz. Similar nonlinear behavior is present in Figure 3.1.c and 3.1.d, where execution time and energy consumption have some unexpected spikes and dips in their energy response. Moreover, optimising such non-linear non-convex functions is an NP-hard problem [160]. Therefore, it is extremely challenging to find energy-efficient frequency combinations under application's QoS constraints. Simple analytical models that linearly regulate the core frequencies are inaccurate and inefficient to reduce the energy or increase the performance [63]. Motivated by these factors, we model the frequency scaling problem based on the data-driven methods.

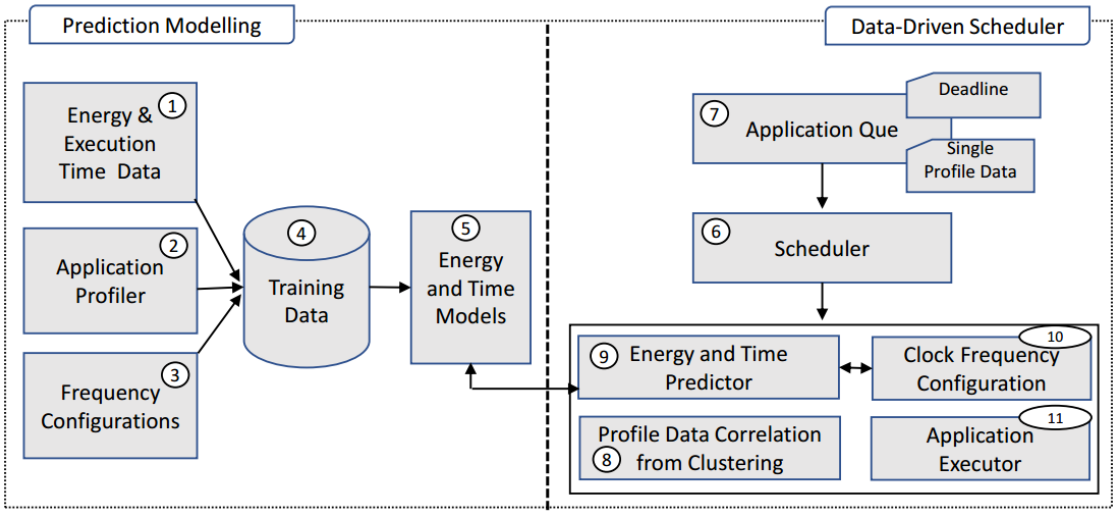


Figure 3.2: System Model for Data Driven Frequency Scaling

3.3.3 System Model

A high-level system model of the proposed framework is shown in Figure 6.1. It is broadly classified into two parts, *predictive modeling* and *data-driven scheduler*. In the first part, we collect the training data that consists of three parts, profiling information, energy-time measurements, and respective frequency configurations. We then predict two entities for a given application and frequency configuration, i.e., energy consumption and execution time. Afterward, in the second part, the new applications arrive with the deadline requirements and minimal profiling data from a default clock frequency execution. The scheduler finds correlated application data using the clustering technique, and this data is used for predicting the energy and execution time over all frequencies. Finally, based on the deadline requirements and energy efficiency, the scheduler scales the frequencies and executes the applications. The important components of this framework are explained in the following sections.

3.4 Data-Driven Frequency Scaling for GPUs

In this section, we discuss the prediction models in detail, which include data collection, training, and model evaluation.

Power	Time
sm	sm
sm_clock	l2_tex_read_hit_rate
l2_tex_read_hit_rate	l2_tex_read_transactions
tex_cache_throughput	tex_cache_transactions
ipc	dram_write_transactions
flop_dp_efficiency	ipc
shared_load_throughput	inst_executed_shared_loads
stall_exec_dependency	gst_efficiency
stall_inst_fetch	inst_replay_overhead
eligible_warps_per_cycle	inst_executed_shared_stores
stall_constant_memory_dependency	l2_read_throughput
pcie_total_data_transmitted	gst_throughput
dram_read_transactions	warp_execution_efficiency
dram_read_bytes	dram_read_bytes
issue_slots	local_store_throughput
l2_tex_write_throughput	gld_efficiency
inst_bit_convert	global_store_requests
l2_global_load_bytes	stall_memory_throttle
gld_requested_throughput	dram_utilisation
pcie_total_data_received	inst_fp_32

Table 3.2: Features used in energy and time prediction (top 20)

3.4.1 Feature Collections

ML-based models are trained using real-time measurement data from the environment, and these models are used in the run time to predict the outcomes. In case of GPU energy and performance prediction, several existing studies rely either on static source code metrics [149, 153], or on run time traces and profiling data [155]. The profiling based method is most suitable due to its ability to gather the real resource footprints and hardware counters of applications on particular GPU architectures, which is crucial to estimate energy and time accurately.

The input to our training model consists of (1) applications profiled features $F = \{f1, f2, f3...fn\}$, (2) respective GPU frequency pair $\{f_{smclock}, f_{memclock}\}$, and (3) energy

and execution time measurements. The profiling features contain the information of an application's run time metrics related to its different hardware components utilization values, instruction counts, communication, and cache metrics, etc.

Applications: We have considered twelve different applications from two heterogeneous computing benchmark suites, Polybench [151] and Rodinia [150]. These two benchmark suites cover a wide range of applications. The Polybench suit covers many linear algebraic applications while the Rodinia suit covers different scientific applications. The details of these applications, including their domain and input specifications, are shown in Table 3.1.

Profiling: For profiling the applications, we use *nvprof*, a standard profiling tool by NVIDIA for CUDA applications. Although NVIDIA has recently released new *nsight-systems* tool kits for monitoring and profiling, they do not support many existing GPU architectures and CUDA versions, so we use *nvprof*. We have also used the *nvidia-smi* toolkit, which is built on top of *nvml* library, a C-based API for monitoring and managing various states of the NVIDIA GPU devices. This tool allows application users to set the supported GPU application clocks and also to measure the energy consumption metrics.

We have developed Python scripts to collect the profiling metrics that runs all the applications iteratively on different frequency domains supported by GPU. Initially, we collect all the available metrics provided by *nvprof* using *–metrics all* argument and export the output in *csv* format. The energy and execution time are gathered by running applications separately to avoid the effect of profiling on these metrics.

We collect metrics from every alternative clock pair of the Tesla P100 GPU from its supporting 62 combinations of core and memory frequencies to reduce the data collection time. It is important to note that, some applications take up to ten minutes for each profiling session as *nvprof* replays the application kernels over several passes to collect the metrics. The *nvprof* provides more than 120 metrics of GPU counters for each execution. For the sake of brevity, we list the top twenty features that dominate in energy and execution time prediction in Table 3.2. The details about selecting these features are explained in feature analysis Section 3.4.3. Here, the features *sm* (SM's utilisation level) is collected from *nvidia-smi dmon* API and remaining all are from *nvprof*.

These collected application profiling metrics, along with the frequency configura-

tion, energy, and time metrics, constitute the total training data, which are then used to build predictive models.

3.4.2 Prediction Models

When building any predictive models, it is often required to test the suitable candidate algorithms and adopt the model that works best for the given input training data and the problem domain. In this regard, the prediction of energy and execution time requires regression-based machine learning (ML) models. We have investigated several suitable candidate ML algorithms, including Linear Regression (LR), lasso-linear regression (Lasso), and Support Vector Regression (SVR). Also, we explored ensemble-based gradient boosting techniques, eXtreme Gradient Boosting (XGBoost), and CatBoost. The goal is to build energy and execution time prediction models for each GPU device to assist the frequency configuration. To that end, prediction models are trained for two outputs, i.e., energy (E) and execution time(T). The profiling data from all the applications are partitioned into training and testing datasets with 70% and 30%, respectively.

The input feature set now includes a set of tuples with each tuple having profiled features and frequency combination i.e, $F = \{f_1, f_2, f_3, ...f_n\} \cup \{f_{smclock}, f_{memclock}\}$ while the two models output predicted energy consumption P and execution time T.

We use the sci-kit learn package to implement the LR, Lasso, and SVR. For XGBoost and CatBoost, the standard python packages are used that are publicly available. A few of the profiling parameters from $nvprof$ are categorical, representing different components utilization as low, mid, and $high$. Among a total of 120 features collected, 15 features were categorical including $dram\_utilisation$, $double\_precision\_fu\_utilisation$. Only numerical features are used in the models except for CatBoost. However, CatBoost is specifically designed to work with the categorical or mixed data, and it has an efficient way of representing the categorical variables. Here, the categorical features are transformed into numerical features by the technique of order target statistics.

The parameters for each of the algorithms are the default and self-explanatory in

https://scikit-learn.org/stable/
https://catboost.ai/
https://github.com/dmlc/xgboost

our implementation. To avoid over-fitting of the models, we adopt the leave-one-out cross-validation, where we exclude the data from a particular application in training and evaluate this model with the excluded application's test data. This helps to assess the robustness of models, and proven efficiency will help to use these offline trained models for new applications without retraining the models.

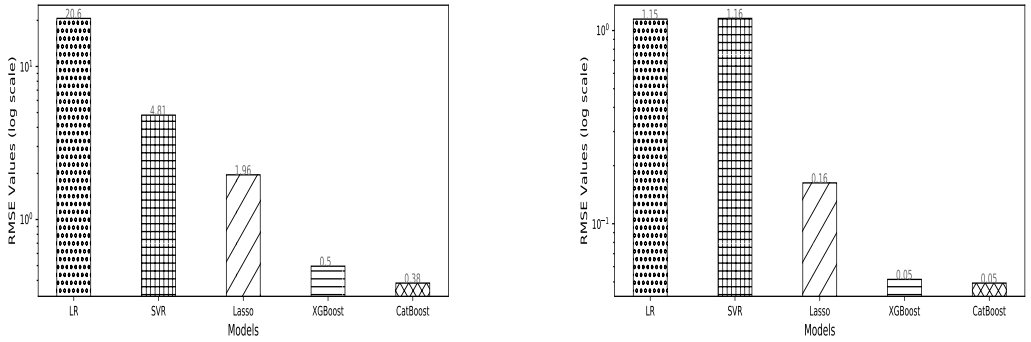
The goodness of fit is measured using the Root Mean Square Error (RMSE) metric, which is a standard evaluation metric in regression-based problems [23]. The RMSE is defined as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(y_i - y^i)^2} \tag{3.2}$$

In Equation 3.2, y_i is the observed value, y^i is the predicted output variable by prediction model, and n is the total number of predictions. The value of RMSE represents the standard deviation of the residuals or prediction errors. It also indicates how far are the data points from the model fitted line. Thus, lower RMSE values are preferred.

The performance of different algorithms is shown in Figure 3.3. Here, we can observe that the CatBoost has the lowest RMSE value of 0.38 in energy prediction, indicating residuals or prediction errors are less, and its predictions are more accurate. We observed that linear models like LR, SVR, and Lasso perform worst in estimating energy and slightly better in predicting the execution time. It is because energy consumption has more non-linearity with the input features than the execution time, and simple linear models do not perform well at it. While in execution time prediction (Figure 3.3.b), both XGBoost and CatBoost has equal performance (RMSE value of 0.05). As the performance of the CatBoost is promising in both models, we choose it as our prediction algorithm.

We perform hyperparameter tuning to further optimize the CatBoost model; we use the grid search technique over the parameter space. The results of the grid search are shown in Table 3.3. Here, the parameters *iterations* and *depth* decides the number and size of the decision trees while *learning_rate* is used for reducing the gradient step. The *l2_leaf_reg* represents the coefficient at the L2 regularisation term of the cost function.



(a) Energy prediction (b) Time prediction

Figure 3.3: Performance of different models for energy and execution time prediction (lower RMSE value is preferred)

3.4.3 Feature Analysis

We carry out feature analysis to understand the importance of individual features towards the performance of the prediction model. It also represents the features that are highly influential on the prediction output.

Figure 3.4 and 3.5 indicates the Feature Importance ($F.I$) score of different features. We plot the twenty most significant features sorted in descending order of their score. Here, $F.I$ value represents the difference between the loss value of the model with and without that feature. The model without this feature is similar to the one that would have been trained if this feature was excluded from the data set. Since, *RMSE* is our loss function, the F.I score on y axis shows change in *RMSE* value.

We can observe from Figure 3.4 and Figure 3.5, different types of features contribute to different magnitude while predicting energy and time of application, respectively. In both cases, feature *sm*, which represents the streaming multiprocessor's utilization, has the highest $F.I$ score showing its high impact on the energy and time. Furthermore, *sm_clock* is the second most important feature in predicting energy, reflecting a direct correlation between energy and frequency clock settings. Please note, since our testbed GPUs (Tesla-P100) have only one memory frequency ($f_{memclock}$), it does not feature in the top twenty features as there is no variation introduced by it in the data set. For the system with multiple $f_{memclock}$, it is expected that it would have a significant effect on the

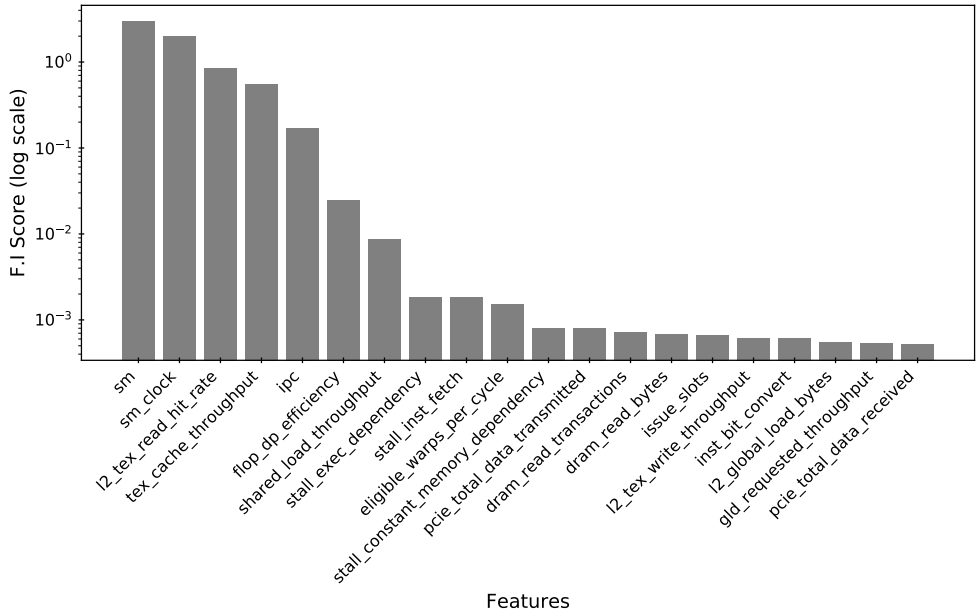


Figure 3.4: Energy prediction model. Top 20 features sorted based on Feature Importance (F.I) score (difference in loss value with and without the feature

model's performance. We can also observe from time model in Figure 3.5, different features present when compared to the energy model (Figure 3.4). A total of 5 features are in common between two models. The features related to l2 cache and stall dependencies have more impact in the energy model, while in time prediction, separate features like *inst_executed_shared_stores*, $inst\_fp\_32$ have occurred in top 20 features showing the higher co-relation between execution time and the metrics related to instruction count.

To further analyze the impact of the number of features on the prediction model's performance, we carry out a threshold analysis. Here, features are sorted based on their $F.I$ score, and recursively added to the training data set, and resulting loss value ($RMSE$) is calculated accordingly. As shown in the Figure 3.6, for both the power and time model, the top 20 features are sufficient enough to achieve the reasonable performance with excellent RMSE value and further inclusion of features do not yield much improvement in the result without increasing the training cost.

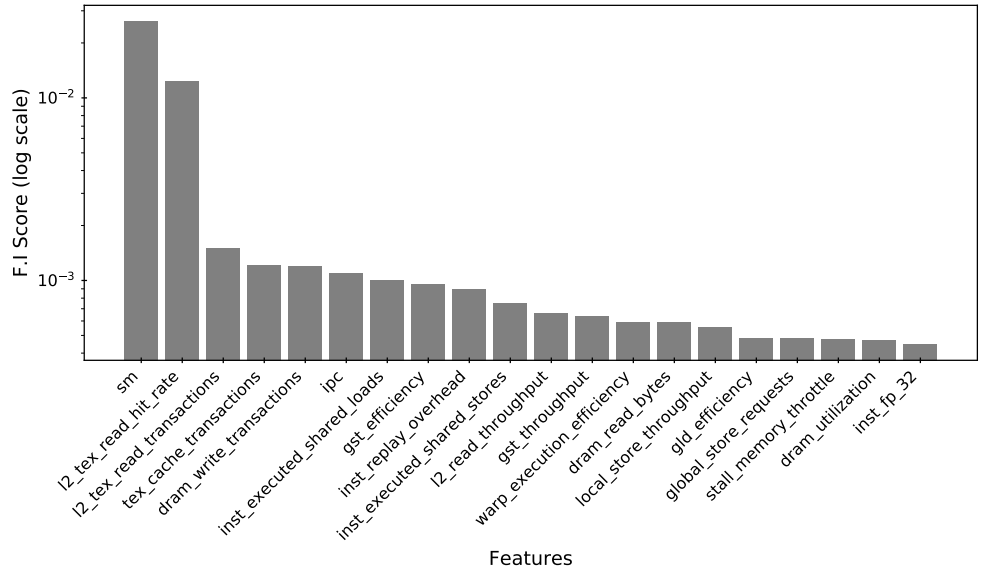


Figure 3.5: Time prediction model. Top 20 features sorted based on Feature Importance (F.I) score (difference in loss value with and without the feature

Catboost model	*depth*	*l2_leaf_reg*	*iterations*	*learning_rate*
Power	4	5	1200	0.1
Time	4	3	1200	0.03

Table 3.3: Optimal parameters obtained for CatBoost from grid search technique

3.4.4 Feature Correlation with Clustering

The prediction models need exhaustive application profiling data from multiple frequency combinations. However, profiling every new application is tedious and infeasible in real-time. Thus, using the existing data and correlating with the new application is a common practice in profiling-based predictive modeling scenarios [161]. In such a case, a new application should have at least one set of profiling data of one frequency combination; we take the default clock as a reference for this. We generate the clusters based on already collected exhaustive data and predict the cluster label for a new application. Furthermore, a highly correlated application within the cluster is chosen from simple heuristic, i.e., application with the lowest absolute difference in execution time is

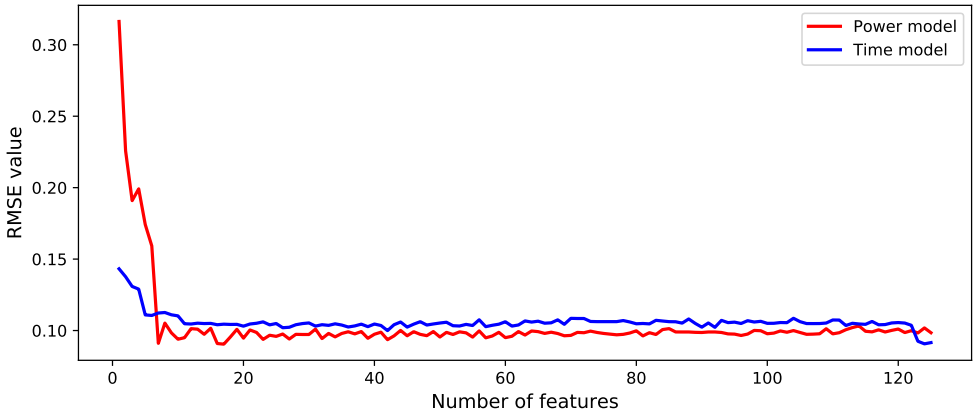


Figure 3.6: Threshold analysis of features

selected to further match the applications performance-similarity.

We use the same set of twelve applications to perform this analysis. To generate the clusters, we use K-means. We found that an optimal number of clusters is five based on the weighted sum of the squared error metric. Applications belonging to a different group and their correlated application can be seen in Table 3.4. The cluster-3 has only one application, i.e., 2MM, suggesting the essence of having a still more number of applications in the sample space to have at least two or more applications in each cluster. The robustness of this method is evaluated by predicting the execution time and energy for all the applications using the profile data of corresponding correlated applications. The average RMSE value of 3.19 and 1.11 is achieved for energy and time prediction, respectively, proving the feasibility of this method.

3.5 Deadline Aware Application Scheduling by Data-Driven DVFS

The advantages of power and performance estimations of GPU workloads are manifold. It is used in resource management techniques like scheduling [147], power capping[157], and also in the analysis of performance bottlenecks of workloads [143]. In this work, we propose deadline aware energy-efficient application scheduling guided by the data-driven DVFS.

The workload model of our scheduling is shown in Figure 3.7.a. It consists of a set of

Applications	Cluster label	Correlated application
particlefilter_naive	0	particlefilter_float
particlefilter_float	0	particlefilter_naive
myocyte	1	lavaMD
lavaMD	1	myocyte
Backprop	2	ATAX
SYRK	0	particlefilter_float
SYR2K	0	particlefilter_naive
GEMM	4	CORR
COVAR	4	CORR
CORR	4	COVAR
ATAX	2	Backprop
2MM	3	2MM

Table 3.4: Cluster labels and correlated app

applications represented as a vector $W = \{A_1, A_2, A_3..A_n\}$, with their own arrival and deadline times, represented as vectors $A = \{a_1, a_2, a_3, ..., a_n\}$ and $D = \{d_1, d_2, d_3, ..., d_n\}$ where $\forall a_i \in A$ and $\forall d_i \in D$, respectively.

As illustrated in Figure 3.7.b, the power curve for individual applications are non-linear with their execution time. The objective is to configure the frequency that meets application A_i's deadline d_i and also has the lowest energy consumption according to it's power curve. Therefore, considering the energy consumption of application i is P_i, then reducing the overall GPU energy consumption is formulated as a minimisation problem as below:

$$\begin{aligned} &\text{minimize} \quad P_{total} = \sum_{i}^{n} P_i \\ &\text{subject to} \quad \forall T_i \leq d_i \end{aligned} \tag{3.3}$$

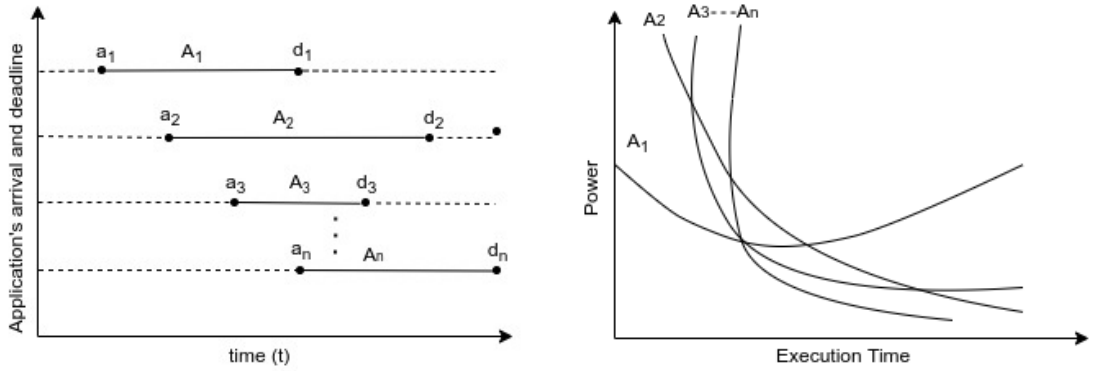
In the above Equation 3.3, T_i is application's execution time and d_i is deadline or execution time requirement of an application. The objective function minimises the total GPU energy consumption, and the constraint makes sure that the application is executed within the deadline. With all applications having different arrival times to system in a stochastic manner, without knowing applications arrival and deadline time apriori,

Algorithm 1 Deadline-aware Scheduling by Data-Driven DVFS

Inputs:
W : list of applications to be executed (workload)
D : list of application's deadline (d)
Output: GPU application clock and schedule
while true **do**
  jobQue ← GETCURRENTARRIVALJOBS(W)
  jobQue ← SORT(jobQue, key= D, order= asc)
  clockList ← GETGPUSUPPORTEDCLOKS(deviceID)
  for each *job* in jobQue **do**
    predictionInput ← GETCORRELATEDDATA(*job*)
    minPower ← MAX
    maxTime ← *job.d*
    $(f_{smclock}, f_{memclock})$ ← NULL
    for each *clockSet* in clockList **do**
      $\hat{P_{job}}$ ← PREDICTPWR(predictionInput.*clockSet*)
      $\hat{T_{job}}$ ← PREDICTTIME(predictionInput.*clockSet*)
      if $\hat{P_{job}} <$ minPower **and** $\hat{T_{job}} <$ maxTime **then**
        minPower ← $\hat{P_{job}}$
        maxTime ← $\hat{T_{job}}$
        $(f_{smclock}, f_{memclock})$ ← *clockSet*
      end if
    end for
    SETGPUAPPLICATIONCLOCK($f_{smclock}$, $f_{memclock}$)
    if not $(f_{smclock}, f_{memclock})$ NULL **then**
      EXECUTE(job)
    end if
  end for
end while
End

solving Equation 3.3 becomes an NP-hard problem. This is equivalent to constrained global optimization [162]. As it is impractical to find an optimal solution in real-time, we present a heuristic algorithm in order to reduce the problem complexity suitable for on-line scheduling and find a near-optimal solution within reasonable amount of time.

Algorithm 1 shows the proposed scheduling algorithm for deadline-aware application scheduling with the data-driven DVFS. Its objective is to reduce the energy while meeting the application's deadline and it is achieved by generating an efficient schedule sequence and also suitably scaling the GPU frequencies. The algorithm takes the appli-



(a) (b)

Figure 3.7: Workload and Power-Execution time models (a) Workload Model (b) Power and Execution time

cation list, and their corresponding deadlines as an input and outputs suitable predicted clock and scheduling sequence for applications.

First, according to applications arrival time, the available jobs are sorted based on the deadlines in ascending order (line 4,5) to make sure the jobs with the earliest deadline are executed first. Considering the new arrived application has only default clock input profile data, we find its correlated application and use it's exhaustive profile data for prediction (as explained in Section 3.4.4). Furthermore, the power and execution time is predicted for all the supported GPU frequency clock sets (lines 12-14) using the prediction models proposed in Section 3.4.2. For a given job, the clocks which have the lowest power consumption and also the predicted execution time less than its deadline is selected (15-18). Finally, the selected application clock is configured, and the application is executed.

The time complexity of Algorithm 1 is polynomial. Assuming we have n jobs to be scheduled on a GPU with c number of clocks. The sorting of jobs requires worst-case time complexity of $n \log n$ (line 5). Furthermore, each job has to be evaluated on all clock-sets and executed, that has a time complexity of nc (line 7-20). Hence, the total complexity will be $(n \log n + nc)$. Considering c is a constant for any given GPU, the Algorithm 1 has polynomial time complexity of $O(nlogn)$.

3.6 Performance Evaluation

We discuss the implementation of our proposed algorithm integrated with the prediction models. We also analyze and discuss the results compared to baseline algorithms.

3.6.1 Implementation

We implemented the proposed scheduling framework using Python language. We developed a multithreaded application where the main thread executes the algorithm 1's logic and invokes the application execution files. Additionally, it also launches two other background threads, one to collect the GPU data related to energy metrics (by running bash scripts with $nvidia - smi\ dmon$) and other kills the background thread once the application execution is done. Furthermore, the application clocks are predicted based on the proposed model in Section 3.4.2, and these predicted clocks are set before executing the scheduled application using the NVML's $nvidia - smi$'s API.

3.6.2 Experimental Setup

We use Grid'5000 testbed [37] for our experiments. It is a large-scale flexible testbed for experiment-driven research, specifically, designed for experimental evaluations of the energy-efficient techniques [37]. We have used Tesla-P100 GPUs for our experiments. This machine has a dual CPU Intel Xeon Gold 6126 processor with 12 cores per CPU and 192 GB of primary memory. The GPU itself has 3584 cores with 16 GB primary memory. The machines are deployed with Debian 10 as the operating system installed with CUDA 10.1 drivers.

3.6.3 Benchmarking Applications

We have used twelve applications from two bench-marking suits (PolyBench and Rodinia), which are also part of our predictive modeling (Table 3.1). To formally produce the workload model described in Figure 3.7, initially, we use GPU default application clocks run time ([715, 1189] MHz for Tesla-P100) as a reference to our application's ex-

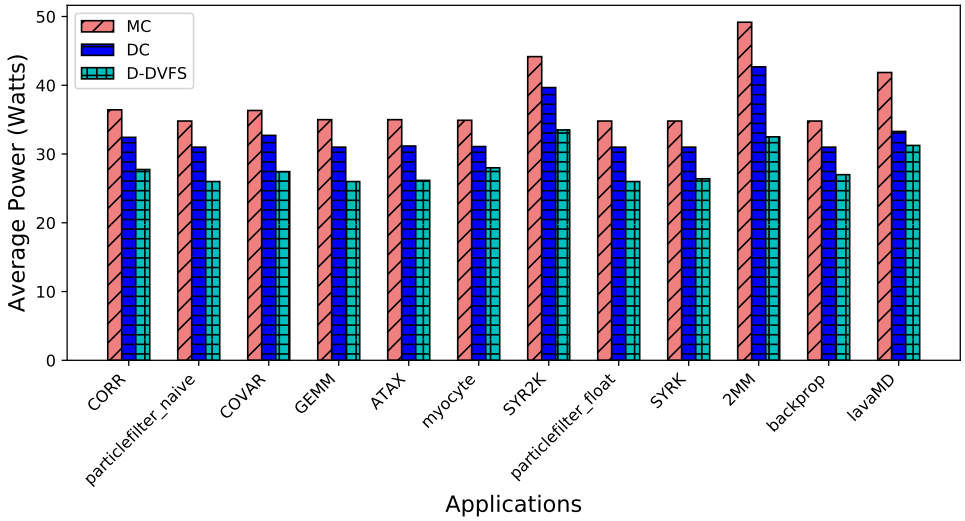


Figure 3.8: Average energy consumption of applications

ecution time. Also, the application's arrival time and deadline are calculated based on the normal distribution. For the arrival time, the minimum and maximum value range of distribution are set to (1, 50), and for the deadline, it is set to (1, 2× execution time). This is to make sure that the application's deadline can have a maximum value of twice their execution time. All these applications are CUDA-based implementation, and configurations are shown in Table 3.1.

3.6.4 Analysis of Results

We evaluate the proposed algorithm 1 against two baselines. *(1): Default Clock (DC)*: GPU frequencies are set to default application clocks. The applications usually run on default clocks under normal conditions *(2): Max Clock (MC)*: GPU frequencies are set to maximum possible frequency domains. This is a widely used technique in the form of near-threshold computation or computational sprinting to finish the execution as fast as possible under strict performance requirements [163]. We represent our proposed policy as D-DVFS, data-driven DVFS.

Figure 3.8 depicts energy consumption of various applications by different policies. Both the MC and DC policies consume a much higher amount of energy as compared to our proposed data-driven frequency scaling (D-DVFS). Particularly, the MC consumes

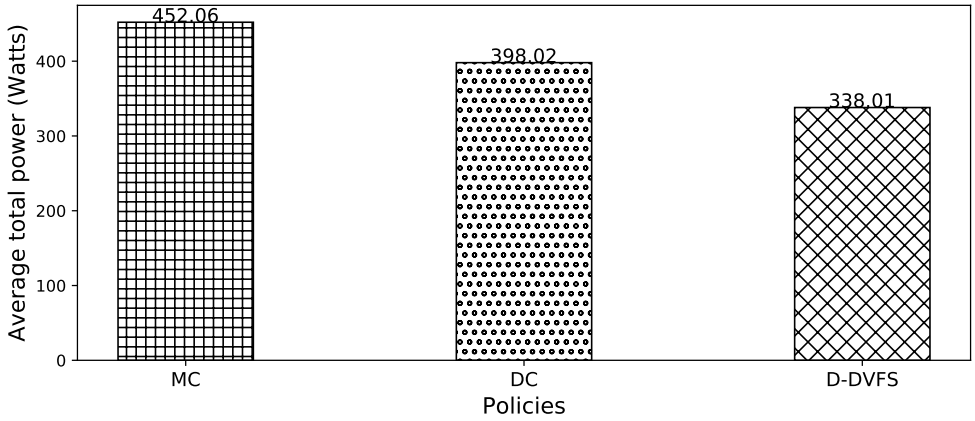


Figure 3.9: Average total energy consumption of GPU

more energy than the other two policies. Since D-DVFS sensibly configures the clocks to the lowest possible energy consumption, it leads to significant energy savings for the application.

The total average GPU energy consumption can be seen in Figure 3.9. MC, DC, and D-DVFS have an average of 452.06 (W.S) (with confidence interval of 95 % (446.73, 457.38)), 392.02 (W.S), and 338.01 (W.S), respectively. In other words, D-DVFS consumes 15.07% and 25.3% less than MC and DC policies, respectively. The results confirms that D-DVFS selects energy-efficient frequencies for application execution.

The application's arrival and deadline time generated using distribution are shown in Figure 3.10. Accordingly, the normalized application completion time achieved using our scheduling and baseline policies is shown in Figure 3.11. The D-DVFS policy meets all the deadlines. It tends to execute near to the deadline requirement of applications as it scales to the frequency that has high energy efficiency and predicted execution time that meets the deadline. Although DC and MC policies execute faster, their deadline-agnostic nature tends to run the applications with high frequency and thus spending more energy. Furthermore, D-DVFS with much lower frequency executes faster in a few scenarios (for application backprop and particle_float, refer Figure 3.11), which represents, faster execution of applications do not have high frequency all the time. Such condition usually happens when the application has significant I/O wait or dependency stalls and setting the higher frequency in such scenarios wastes more power. Neverthe-

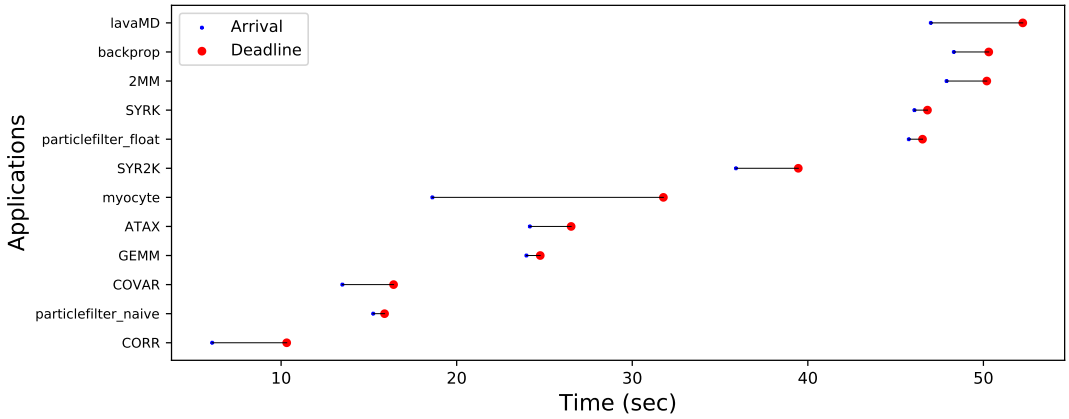


Figure 3.10: Application arrival and deadline times

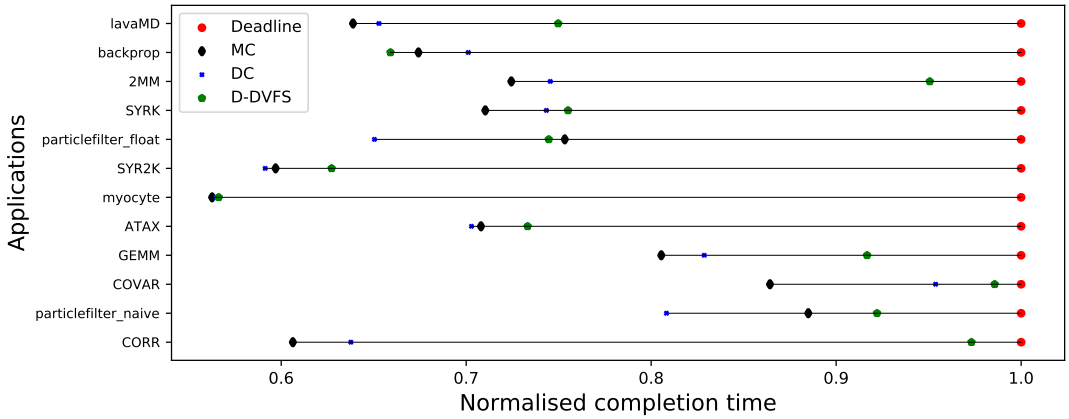


Figure 3.11: Normalised application completion time compared to deadline

less, our approach inherently learns such behavior and adapts accordingly.

The GPU frequency scaling behavior for different applications is depicted in Figure 3.12. Here, MC and DC always operate statically with the highest possible and default frequencies of the GPU. However, D-DVFS selects much lower frequencies for most of the applications, which are sufficient enough to meet the deadlines. Moreover, for applications that demand faster execution to meet their deadlines, it appropriately scales the frequency and chooses the efficient higher frequency range, this can be evidenced in Figure 3.12 for the applications *lavaMD* and *myocyte*.

The accuracy of prediction models in the scheduler is vital for achieving the stated objective. The performance of energy and time prediction models is shown in Figure

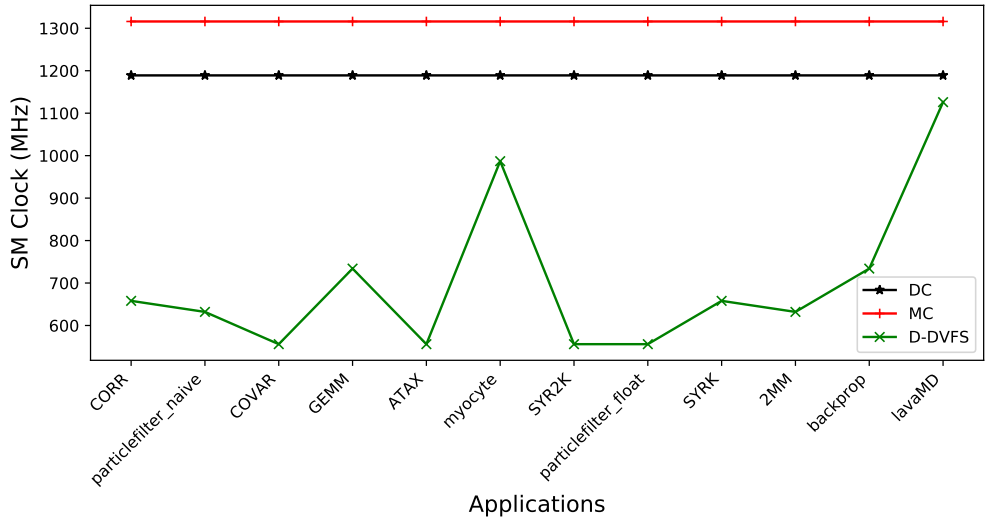


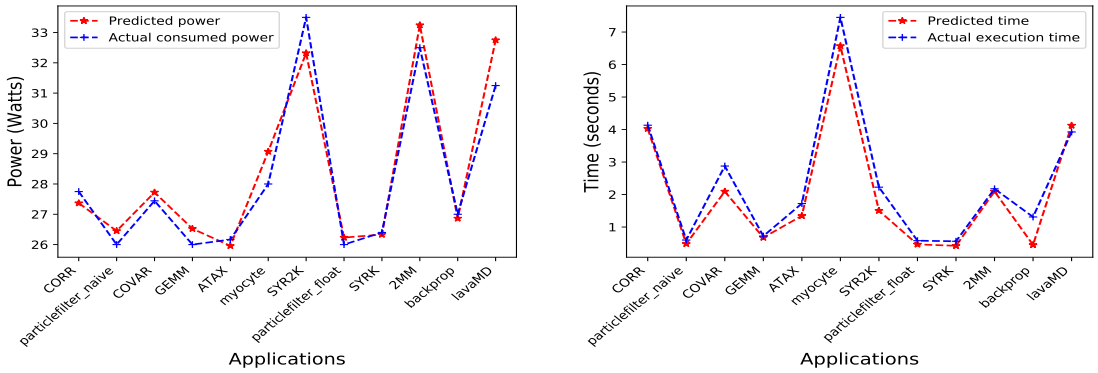
Figure 3.12: Frequency Scaling by different policies

3.13. The predicted values closely follow the actual measurements from the executions showing the accuracy of predictions and thus assisting the scheduler efficiently for frequency scaling.

Therefore, the optimal configuration of frequencies is vital to reduce GPU energy consumption. It is more necessary when different applications have different deadlines. This is the most real case where multiple users submit parallel GPU jobs with their expected deadlines (in the form of wall-time in HPC environments). Employing such techniques, provided they have single execution profiled data, will benefit primarily to save the system energy and also provide better service for application users.

3.7 Summary

Optimal configurations of GPU frequencies can significantly reduce energy consumption. However, identifying the suitable frequencies that result in lower energy consumption with the strict application's deadline requirement is extremely challenging. This is mainly due to the complexity induced by the application's response to energy, execution time, and clock settings. In this chapter, we present a framework that selects suitable GPU frequencies for a given application using the data-driven techniques and



(a) (b)

Figure 3.13: Actual and prediction values in scheduling (a) Power (b) Time

accordingly schedule the applications while reducing energy consumption and meeting deadline. Our model achieves high accuracy with average RMSE values of 0.38 and 0.05 for energy and time, respectively, indicating that predicting the energy is quite difficult as compared to the execution time. Additionally, our proposed scheduling algorithm consumes 15.07% less energy as compared to the baselines while satisfying the deadline requirements.

This chapter presented an energy efficiency solution at an individual computing element level in a data centre (i.e., a GPU). In the next chapter, we study a data centre level resource management, in particular, consolidation technique for energy and thermal efficiency.

Chapter 4

Energy and Thermal-Aware Dynamic Virtual Machine Consolidation

Data centres consume an enormous amount of energy to meet the ever-increasing demand for Cloud resources. Computing and Cooling are the two main subsystems that largely contribute to energy consumption in a data centre. Dynamic Virtual Machine (VM) consolidation is a widely adopted technique to reduce the energy consumption of computing systems. However, aggressive consolidation leads to the creation of local hotspots that has adverse effects on energy consumption and reliability of the system. These issues can be addressed through efficient and thermal-aware consolidation methods. We propose an Energy and Thermal-Aware Scheduling (ETAS) algorithm that dynamically consolidates VMs to minimize the overall energy consumption while proactively preventing hotspots. ETAS is designed to address the trade-off between time and the cost savings and it can be tuned based on the requirement. We perform extensive experiments by using the real world traces with precise power and thermal models. The experimental results and empirical studies demonstrate that ETAS outperforms other state-of-the-art algorithms by reducing overall energy without any hotspot creation.

4.1 Introduction

A significant part of Cloud data centres' energy consumption emanates from computing and cooling systems.

In particular, the contribution of cooling system power is almost equal to the com-

This chapter is derived from:

- **Shashikant Ilager**, Kotagiri Ramamohanarao, and Rajkumar Buyya,"ETAS: Energy and Thermal-Aware Dynamic Virtual Machine Consolidation in Cloud Data Center with Proactive Hotspot Mitigation", *Concurrency and Computation: Practice and Experience (CCPE)*, Volume 31, No. 17, Pages: 1-15, ISSN: 1532-0626, Wiley Press, New York, USA, September 2019.

puting system [164]. In this context, a data centre resource management system should holistically contemplate computing and cooling power together to achieve overall energy efficiency.

In pursuance of reducing the computing energy, workloads are consolidated on the fewest hosts as possible, and remaining hosts are shut down or turned to low power mode [165–168]. However, such aggressive consolidation leads to localized hotspots. The effect of hotspots is manifold. It has a catastrophic effect on the entire system by affecting the reliability of the data centre [169]. In addition, the temperature beyond the threshold at host damages the silicon components of CPU leading to the failure of the host itself. Moreover, to prevent further complications, the cooling system is enforced to pass more cold air which further increases the cost of cooling. This entails for thermal management through optimal workload distribution to avoid the hotspots and simultaneously reduce the overall data centre energy.

The temperature variations in a data centre are caused by many factors. Firstly, the power consumed by a host is dissipated as heat to the ambient environment [170], this power consumption is directly proportional to the utilization level of resources. Secondly, the supplied cold air from Computer Room Air Condition (CRAC) itself carries certain temperature along with it which is known as cold air supply temperature. Finally, existing studies have shown that the inlet temperature of hosts exhibits the spatio-temporal phenomenon [34]. The dissipating heat from one host affects the temperature of other hosts, this heat recirculation within a data centre happens due to the thermodynamic feature of hot air. The air that has passed through or over hosts does not completely reach the return-air plenum but instead remains in the space to pass over the hosts again. In this aspect, it is important to address this spatio-temporal aspect to optimize energy usage.

On the other hand, estimating the temperature in a data centre is a non-trivial problem. There are three approaches to predict the thermal status of the data centre. First, CFD (Computational Fluid Dynamics) models [32], which are accurate in predictions; however, the inherent complexity in rendering makes it computationally expensive and thus infeasible for real-time online scheduling. The second approach is to use predictive models with techniques like machine learning [170], it largely depends on prediction

models and the quality and quantity of the data. The last approach is analytical modeling [34], which is based on the thermodynamic features of heat and physical properties of a data centre It is computationally inexpensive and efficient compared to other two approaches. Therefore, it is requisite to use an analytical model to design online schedulers which are computationally inexpensive than others.

Dynamic VM consolidation has proven to be a prominent approach for data centre energy savings [166, 171]. These consolidation algorithms are not aware of the physical layout and the location of the physical machine. Furthermore, due to the skewed temperature distribution in the data centre, consolidating the workload on the fewest hosts may not always save the energy as it may increase the cooling cost and create hotspots [170]. However, there exists a restricted amount of work to address this aspect. Power and thermal-aware workload allocation for the heterogeneous data centre are proposed in [172]. Similarly, dynamic voltage frequency scaling (DVFS) coupled spatio-temporal aware job scheduling is discussed in [35]. These solutions either cannot be applied directly to the virtualized Cloud data centres or their solutions are application specific.

In this chapter, we propose a new online scheduling algorithm Energy and Thermal-Aware Scheduling (ETAS) for dynamic consolidation of VMs that uses analytical models for thermal status estimation. The randomized online algorithms commonly perform better than deterministic algorithms designed for the same problems in real-time decision making systems [173]. Therefore, this algorithm for dynamic consolidation is based on Greedy Random Adaptive Search Procedure (GRASP) [174] meta-heuristic which is fast, adaptive and suitable for online decision systems. We analyze the proposed algorithm with the extensive simulation-based experiments using CloudSim [38] with real-time workload traces from PlanetLab systems. The proposed algorithm reduces the significant amount of overall energy consumption by preventing hotspot creation with the small amount of performance overhead in terms of Service Level Agreement (SLA) violations. The key **contributions** of this chapter are summarised as follows:

- Proposes policies for efficient distribution of workloads (VMs) to optimize the computing and cooling energy holistically and proactively prevent the hotspots.
- Designs an online scheduling algorithm based on GRASP meta-heuristic which is

used for dynamic VM consolidation.

- Implements the proposed algorithm and validate its efficiency with extensive experiments using real workload traces through simulation and demonstrate its superiority by comparing to the several baseline algorithms.

The rest of the chapter is organized as follows: Section 4.2 describes related work. Section 4.3 introduces system model. Section 4.3 provides problem formulation and an overview of our algorithm. The experiments and results are discussed in Section 4.5. Finally, Section 4.6 concludes the chapter.

4.2 Related work

Energy management in a Cloud data centre has been the topic of interest for many researchers in recent years. The high energy consumption in data centre incurs a huge operational cost and diminishes the profit margin of Cloud service providers. The literature on energy management in the Cloud data centre itself is vast and we identify some of the relevant works in this section. Techniques like hardware optimization and Dynamic Voltage Frequency Scaling (DVFS) have been practically examined in the literature [175] to manage the energy of a host by adaptively varying the frequency of processor based on its utilization. However, such solutions are restricted to a single node level. For the data centre level, techniques like VM Consolidation and load balancing [166] [167] are extensively used to increase the resource utilization and reduce the energy consumption of computing nodes.

Utilization-based consolidation has been studied by Beloglazov et al. in [166], where they consolidate the VMs on the fewest server as possible based on current CPU utilization. The Modified Best Fit Decreasing (BFD) algorithm is used to place the VM into a target host. They also proposed different heuristics to detect overloaded and underloaded hosts and to select a set of VMs from those hosts and migrate to new hosts. Kansal et al. [176] have proposed Energy-aware VM consolidation algorithm using firefly meta-heuristic optimization technique. The target physical machine for VM to be migrated is selected based on a certain distance metric. The results have shown that

44% energy can be saved compared to other baseline algorithms. Li et al. [177] studied about VM migration and consolidation algorithms and proposed multi-resource energy-efficient models for the same. To avoid local optima and to reach global optima, particle swarm optimization based policies have been proposed. Verma et al. [167] proposed server consolidation using the workload analysis. They have stressed identifying the correlation between applications to consolidate on a server. These consolidation techniques are better at saving the computer system energy, however, they completely ignore the effect of consolidation on thermal status of a data centre.

Thermal management is an important task for a data centre resource management system. At first, Moore et al. identified the effect of workload on CPU temperature and heat recirculation effect in the data centre. [170]. The authors have proposed workload placement strategies to reduce the heat recirculation effect. Similarly, Tang et al. investigated [34] thermal-aware task scheduling for homogeneous HPC data centre. The scheduling policy is derived to minimize peak inlet temperature through task assignment (MPIT-TA). Moreover, they quantified the heat recirculation effect into a heat distribution matrix that was initially identified in [170]. In a similar way, DVFS coupled, thermal-aware HPC job scheduling has been investigated by Sun et al. [35] where the primary focus of the work is to reduce the makespan. Lee et al. [178] have proposed proactive thermal-aware resource management policies for virtualized HPC Clouds. The authors have formulated heat imbalance model based on heat generation and heat extraction metrics. In addition, virtual machine allocation policies VMAP and VMAP+ are proposed to consolidate the workload.

All these solutions addresses either thermal-aware static job placement or mostly confined to HPC workloads. Such workload specific solutions cannot be directly applied to a Cloud data centres where applications are deployed within virtualized resources and service providers usually don't have knowledge of the application characteristics running inside. The IaaS Cloud services require application agnostic resource management techniques with a high abstraction of input data.

Thermal management specific to Cloud data centre is studied in many of the works in literature. Al-Qawasmeh et al. [172] presented power and thermal-aware workload allocation in the heterogeneous Cloud. They have developed optimization techniques

Table 4.1: Related Work

Research Works	Thermal-aware	Heat Recirculation -aware	Consolidation	Online	Dynamic
Beloglazov et al.[166]	N	N	Y	Y	Y
Verma et al. [167]	N	N	Y	Y	Y
Ferreto et al.[180]	N	N	Y	Y	Y
Kansal et al. [176]	N	N	Y	Y	Y
Li et al. [177]	N	N	Y	Y	Y
Moore et al. [170]	Y	Y	N	Y	N
Qinghui et al. [34]	Y	Y	N	Y	Y
Sun et al. [35]	Y	Y	N	Y	N
Al-Qawasmeh et al. [172]	Y	Y	N	Y	N
Li et al. [179]	Y	Y	N	Y	N
Lee et al. [178]	Y	N	Y	Y	Y
Li et al.[181]	Y	Y	Y	Y	Y
Mhedheb et al.[114]	Y	N	Y	Y	Y
Our Work (ETAS)	Y	Y	Y	Y	Y

to assign the performance state of the CPU core at data centre level. Li et al. [179] have investigated the failure and energy-aware scheduling. In this paper, they have extracted failure model from the workload and developed failure and energy-aware static task assignment problem. However these approaches are static in their nature and do not consider runtime variation in utilization and consolidate accordingly. Moreover, Cloud workloads typically run into few days to many years and they need to be dynamically consolidated at regular intervals which is the focus of this chapter.

Mhedheb et al. [114] proposed heuristic algorithms with the goal of reducing energy by balancing load and temperature inside the Cloud data centre. The evaluation results through CloudSim has resulted that thermal-aware scheduling outperforms compared to power-aware only algorithms. The thermal models considered in this work are incomplete and excludes heat recirculation effect.

In a similar context, Ferreto et al. [180] designed consolidation algorithms with migration control based on linear programming and heuristic techniques. The results have evaluated based on a number of migrations and active physical machines as primary

factors. The proposed consolidation tries to optimize only computing system of energy while completely ignoring thermal and cooling aspects.

In a similar way, Lee et al. [181] have proposed scheduling algorithm for holistic energy minimization of computing and cooling system in Cloud data centres. The authors have proposed a greedy heuristic scheduling algorithm GRANITE that balances workload after fixed scheduling interval. However, the algorithm balances the workload only from the overloaded hosts that are decided based on a temperature threshold. The threshold is set by ranking hosts based on their temperature and selecting lowest temperature among top 10% of those hosts. The policy does not discuss managing underloaded hosts and setting an optimal percentage of servers that are to be considered as overloaded

In a consolidation enabled Cloud data centres, managing overloaded hosts for unknown nonstationary workloads poses challenging work for resource management systems. In this regard, Beloglazov et al. [182] proposed a solution to predict overloaded hosts and manage resources efficiently with explicitly set QoS. They have used a multi-size sliding window estimation model and Markov chain model to solve this problem. However, this work solely focuses on overload detection with regard to CPU resources. In a dynamic environment of Cloud data centre, the overload detection algorithm should integrate both thermal and computing resource aspects together.

The comparison of the related work can be found in Table 4.1. Here, *Dynamic* means the ability to consolidate VMs in a regular interval after the initial placement based on optimization criteria.

4.3 System Model

A model of a Cloud computing system is shown in Figure 4.1. It consists of three elements- infrastructure, resource management system (RMS) and users. An RMS receives the request for resources from users and it allocates requested resources from the data centre infrastructure. We assume all the requests are submitted as virtual machines which are hosted on some physical machines. Table 4.2 shows the definition of all the symbols that are used in this section and the rest of the chapter. A discussion on key

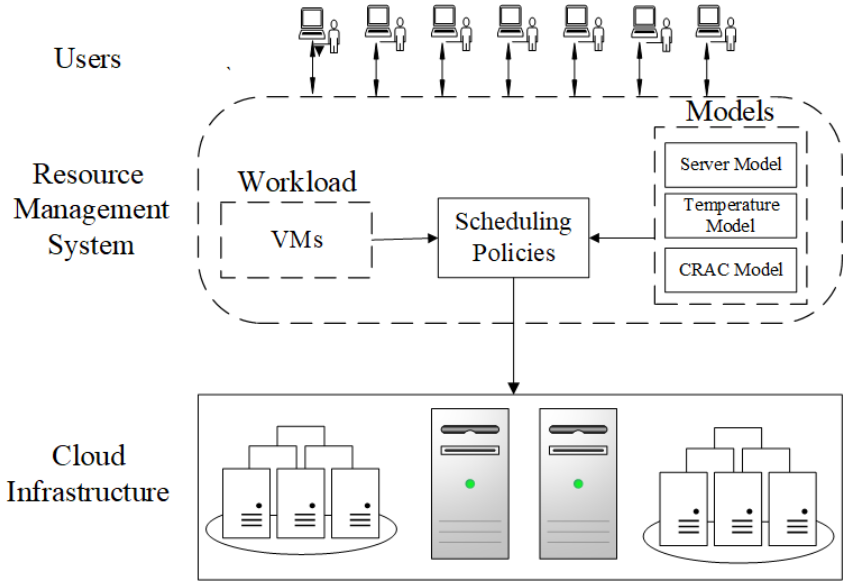


Figure 4.1: System Model for ETAS

elements of RMS, i.e., server model, temperature model, CRAC model, and workload model is presented below.

The data centre consists of heterogeneous hosts with different processing capability. The power consumed by a host is predominantly determined by its utilization level, we adopt this power model [183] which has a linear relationship with the utilization of the CPU.

$$P_i(t) = \begin{cases} P_i^{idle} + \sum_{j=1}^{M_i} u(VM_{i,j}(t)) \times P_i^{dynamic} & M_i > 0 \\ 0 & M_i = 0 \end{cases} \tag{4.1}$$

The power ($P_i(t)$) consumption of a host is the summation of idle and dynamic power. In Eq. 4.1, P_i^{idle} is power consumption of a host in its idle state which is constant, and $P_i^{dynamic}$ is dynamic power consumption of host which has linear realtionship with CPU utilization. The $u(VM_{i,j})$ is utilization of j^{th} VM on $host_i$, and M_i is the number of VMs running on $host_i$. We consider the host is active if its utilization is more than 0 and inactive if the utilization is 0.

Table 4.2: Definition of Symbols

Symbol	Definition
N	Number of hosts
P_i	Power of $Host_i$
P_i^{idle}	Idle power of $Host_i$
$P_i^{dynamic}$	Dynamic power of $Host_i$
t	Time interval t
T	Total scheduling interval
U_{max}	Maximum CPU utilizayion threshhold of $host_i$
$u(VM_{i,j})$	utilization of VM_j on $host_i$
P_C	Computing system power
P_{CRAC}	Cooling system power
P_{total}	Total data centre power
T_{sup}	Cold air supply temperature
$T_i^{in}(t)$	Inlet temperature at $host_i$ on time t
$T_i(t)$	Temperature at $host_i$ at time t
T_{red}	Maximum threshold of CPU temperature
R	Thermal resistance of Host
C	Heat capacity of host
$d_{i,k}$	Effect of heat recirculation to $host_k$ from i
α	Parameter to decide size of RCL in GRASP
ϵ	Iterations controller parameter in algorithm

4.3.1 Temperature Model

The temperature at the host is dynamic and it depends on several factors such as its power consumption, CRAC settings and physical location of the host itself due to the heat recirculation effect [34].

The focus of this chapter is not to devise new metrics for these, instead, we use the existing approaches to model and incorporate it into our temperature model. The inlet temperature ($T_i^{in}(t)$) of a host is defined as a linear combination of supplied cold air temperature (T_{sup}) from CRAC and temperature increase due to heat circulation.

$$T_i^{in}(t) = T_{sup} + \sum_{k=1}^{N} d_{i,k} \times P_k(t). \tag{4.2}$$

Considering the heat recirculation effect exist within particular zones of data centre based on its current physical layout, this recirculation effect can be quantified as a heat distribution matrix D where each entry $d_{i,k}$ in the matrix D indicates the factor by which inlet temperature of $host_i$ is affected by the $host_k$ and this factor is magnitude of power consumption ($P_k(t)$) of $host_k$. In the Eq. 4.2, $k \in 1, N$ is the number of hosts in recirculation zone. In abstract, it can be noted based on Eq. 4.2, though the CRAC passes similar cold air supply temperature (T_{sup}) across all the hosts in a data centre, the inlet temperature varies at each host based on its physical location and heat recirculation effect.

The CPU temperature at the $host_i$ is dominated by dissipating heat by its CPU, hence, the temperature at time t can be defined by adopting a widely used RC model [184] as follows:

$$T_i(t) = PR + T_i^{in} + (T_{initial} - PR - T_i^{in}) \times e^{-\frac{t}{RC}} \tag{4.3}$$

where P is the dynamic power of host, R and C are thermal resistance (k/w) and heat capacity (j/k) of the host respectively and $T_{initial}$ is the initial temperature of the CPU. Here, $T_i(t)$ refers to the dissipated CPU temperature (CPU temperature dissipated by $host_i$ at time t). Based on Eq. 5.4, it can be noted that, CPU temperature of host is not only governed by amount of power it is consuming (though it has a major effect and it is proportional to the CPU speed or workload level), it is also governed by hardware specific constants like R and C along with the inlet temperature (T_i^{in}). Though we adopted the RC model to estimate the CPU temperature, our proposed work is independent of the temperature model and it can be applied to other models [34]. Eq. 5.4 captures the dynamic behaviour of host temperature including heat recirculation effec

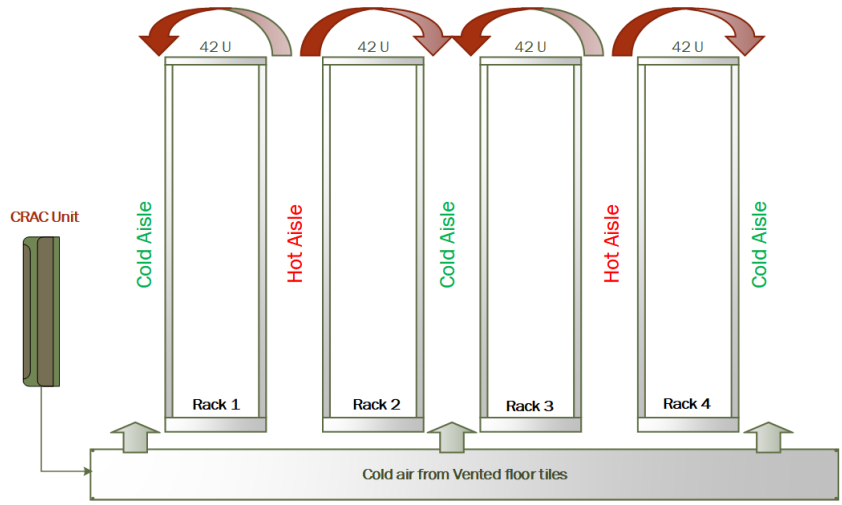


Figure 4.2: Data Center Rack Layout

4.3.2 CRAC Model

The data centre thermal management is done by Computer Room Air Condition (CRAC) system. In a modern data centre, racks are arranged as cold aisle and hot aisle as shown in Figure 4.2. The cold air flows through vented tiles from the bottom of the rack to the top of the rack in the cold aisle. The exhausted hot air is passed through hot aisle i.e, from the rear of the racks and it is collected through the ceiling and supplied back to CRAC [185]. Each data centre consists of multiple CRAC units, $CRAC = \{CRAC_1, CRAC_2, ..., CRAC_n\}$. We consider CRACs are the only cooling facility available in the data centre. The efficiency of such a cooling system is measured by the metric Coefficient of Performance (CoP). The CoP is a function of cold air supply temperature [170] (T_{sup}) and it is defined as the ratio of total power consumed by the computing system to the total power consumed by the cooling system to extract the dissipated heat.

$$CoP(T_{sup}) = \frac{P_C}{P_{CRAC}} \tag{4.4}$$

In the Eq. 4.4, P_C and P_{CRAC} represent computing and cooling system power. The CoP of data centre varies for various settings in a different data centre. It depends on the

physical layout and thermodynamic feature of the data centre. It can be modeled using the regression techniques with multiple experiments using the different workloads and supply air temperature. In this chapter, we use the following model from HP lab [170] data centre to estimate the CoP.

$$CoP(T_{sup}) = 0.0068T_{sup}^2 + 0.0008T_{sup} + 0.458 \tag{4.5}$$

Eq. 4.5 indicates that by increasing the value of T_{sup} we can increase the efficiency of the cooling system and reduce the cooling power.

4.3.3 Workload Model

The requests from users are considered as tasks or cloudlets. Suppose n is the total number of cloudlets submitted, we consider the same number of VMs are required to execute these tasks which are represented as $VM = \{VM_1, VM_2, VM_3....., VM_n\}$. We model the workload in terms of the virtual machine, hence, our solution is independent of the workload type. We consider each VM executes a single cloudlet and the VM is terminated after the cloudlet is executed. Each cloudlet has CPU requirement R_{cpu}, memory requirement R_{mem} and task length l, hence each cloudlet has triplet attributes $\{R_{cpu}, R_{mem}, l\}$. Since we are addressing the dynamic consolidation, we submit all cloudlets at the beginning of the experiment.The VMs are consolidated dynamically based on certain scheduling policies at every interval.

4.4 Energy and Thermal Aware Scheduling

4.4.1 Problem Formulation

The larger part of the data centre energy consumption is contributed by the computing and cooling systems. The computing system consists of hosts and its energy consump-

tion can be defined as follows:

$$P_C = \sum_{t=0}^{T} \sum_{i=1}^{N} x_j^t P_i \tag{4.6}$$

According to Eq. 4.6, computing system energy (P_C) is a summation of energy consumed by all hosts. The binary variable x_j^t holds value 1 if the host i is active at timestep t and 0 otherwise. It is imperative that computing systems energy is governed by the number of active hosts. Therefore, keeping an optimal number of active hosts at each scheduling interval is important.

The cooling system (CRAC) power consumption is defined as the ratio between the thermal load and CoP of the data centre [170]. Considering the fact that energy consumed by a computing system is almost dissipated as heat to an ambient environment of the data centre [34], thermal load can be represented as P_C. Accordingly, the cooling system energy consumption can be defined as follows:

$$P_{CRAC} = \frac{ThermalLoad}{CoP(T_{sup})} = \frac{P_C}{CoP(T_{sup})} \tag{4.7}$$

Based on Eq. 4.7, it can be inferred that cooling system energy (P_{CRAC}) can be reduced either by increasing the CRAC cold air supply temperature (T_{sup}) or by decreasing the thermal load. Therefore, by using consolidation technique, we aim to decrease the thermal load of the data centre and simultaneously avoid hotspots with a proactive approach for a given static cold air supply temperature (T_{sup}). Thus, the total energy consumption of the data centre can be given as:

$$P_{total} = P_C + P_{CRAC} = (1 + \frac{1}{CoP(T_{sup})})P_C \tag{4.8}$$

The VM placement and consolidation algorithm must be aware of the orthogonal tradeoff between computing and cooling systems, where higher concentrated consolidation leads to hotspots and a highly sparsed distribution increases the energy consumption.

$$
\begin{aligned}
\underset{X}{\text{minimize}} \quad & P_{total} = \sum_{t=0}^{T} \sum_{i=1}^{N} x_j^t (1 + \frac{1}{CoP(T_{sup})}) P_i \\
\text{subject to} \quad & u(h_i) \leq U_{max} \\
& T_i(t) < T_{red} \\
& \sum_{j=0}^{m} VM_{j,i}(R_{cpu}, R_{mem}) \leq h_i(R_{cpu}, R_{mem}) \\
& x_j^t \in \{0, 1\}
\end{aligned}
\tag{4.9}
$$

The objective function in Eq. 4.9 takes care of the holistic minimization of energy. The constraints ensures the potential thermal violation and CPU threshold violation does not occur due to the added workload on the host. The constraints also satisfy the capacity constraints, if the host has enough resource (R_{cpu}, R_{mem}) for an accommodating VM, then the host is considered suitable for the VM placement. x_j^t is a binary variable whose value is 1 at timestep t, if the VM is allocated to $host_i$, and 0 otherwise. The above optimization function in Eq. 4.9 should be executed at each scheduling interval to decide the target host for the VMs. Considering that scheduling is an NP-hard problem and scale of Cloud data centre where a single data centre hosts thousands of physical hosts, solving this optimization function in Eq. 4.9 is time-consuming and infeasible for real-time systems. Consequently, in the next section, we propose an online scheduling algorithm with reduced time complexity based on GRASP metaheuristic which finds the near-optimal solution in a reasonable amount of time.

4.4.2 The Scheduling Algorithm

Overview

In this subsection, we propose a scheduling algorithm based on GRASP metaheuristic. GRASP is simple to implement and easy to adapt based on the problem specific domain [174]. It is an iterative randomized optimization technique. Each iteration has two phases: 1) Greedy construction phase- where the solution list is constructed based on the greedy function by random sampling from the solution space. 2) Local search phase- a

neighborhood search to find the current best solution from the previously constructed solution list. The iteration continues until certain stopping criteria is reached which can be chosen based on problem-specific constraints.

Its adaptive nature which provides an opportunity to dynamically update the greedy value of the objects and the simple probabilistic nature which selects the solution by random sampling is viable to achieve the near optimal solution. Moreover, its inherent capabilities like the flexibility to select the size of solution space, and to tune the stopping criteria is useful to adjust the amount of greediness and computational complexity.

Algorithm 2 ETAS: Energy and Thermal Aware Scheduling

Input: VMList, hostList
Output: Energy consumed, Number of hotspots, SLA violation percentage

```
1: Initialize T_red, α, ϵ
2: for t ← 0 to T do
3:    VMList ← getVMsFromOverAndUnderUtilizedHosts()
4:    for all vm in VMList do
5:       allocatedHost = ∅
6:       isSolutionNotDone ← true
7:       while isSolutionNotDone do
8:          SolutionList ← ConstructGreedySolution(VM, hostList)
9:          newHost ← LocalSearch(SolutionList)
10:         δ = allocatedHost.τ − newHost.τ
11:         if δ ¿ ϵ then
12:            allocatedHost ← newHost
13:         else
14:            isSolutionNotDone ← false
15:         end if
16:      end while
17:      if allocatedHost == ∅ then
18:         allocatedHost = getNewHostFromInactiveHostList()
19:      end if
20:   end for
21: end for
```

In this problem, our primary focus will be on step 3, i,e, placement of VMs to new hosts based on thermal and energy status. These 3 optimization steps are applied to each scheduling interval to reduce the number of active hosts and keep remaining hosts in a low power mode or complete power off state to reduce the energy consumption.

Algorithm 3 Construct Greedy Solution

Input: VM, hostList
Output: SolutionList

1: SolutionList $\leftarrow \emptyset$
2: RCL $\leftarrow$ makeRCLFromActiveHostList
3: **for all** s in RCL **do**
4: **if** s is suitable for VM **then**
5: s.$\tau \leftarrow (1 + \frac{1}{CoP(T_{sup})})P_i$
6: **end if**
7: SolutionList $\leftarrow \cup$ s
8: **end for**
9: **return** SolutionList

Algorithm 4 Local Search

Input: SolutionList
Output: Host with local optima

1: LocalOptimalHost $\leftarrow \emptyset$
2: **for all** s in SolutionList **do**
3: **if** s.$\tau <$ LocalOptimalHost.τ **then**
4: LocalOptimalHost = s
5: **end if**
6: **end for**
7: **return** LocalOptimalHost

Algorithm

For the first two steps of dynamic consolidation, we use the following procedure. To detect overloaded hosts, we use the static CPU threshold (U_{max}) and the maximum threshold of CPU temperature (T_{red}) together as threshold parameters. To detect the underloaded hosts, we use the same approach used by Beloglazov et al. [166] where all the active hosts that are not overloaded are iterated and if all the VMs from such particular host can migrate to other active hosts then that host is considered as underloaded. For the second step of consolidation, we select VMs from overloaded hosts to migrate in an iterative manner until the host condition is not overloaded. The VM which has minimum migration time (mmt) is selected to reduce the migration bottleneck in the system (which has minimum RAM usage and takes less time to migrate with the available bandwidth).

We assume that, prior to optimization, the data centre has reached to a steady state, i,e., all the requested VMs are placed into hosts and thermal status of the data centre has reached to steady state. Algorithm 2 runs at the beginning of each scheduling interval (five minutes in this case), and identifies the VMs that are needed to be migrated (based on VM selection policies described above) from overloaded and underloaded hosts and migrates them to the destination hosts.

In the first step of the algorithm, all the essential parameters like T_{red} (redline temperature), α which decides the size of Restricted Candidate List (RCL) in GRASP technique and ϵ (expected amount of improvement over the previous iteration) are initialized, definitions of these are given in Table 4.1. At each interval, all the VMs from over and underutilized hosts are identified (line 3) based on previously discussed policies, and for each VM to be migrated, the best possible host is allocated.

For each VM to be migrated from the migrate list, the process starts with initializing the allocated host to null initially (line 5). The line number 7-16 in Algorithm 2 shows the generic schema of the GRASP technique. At this stage, each iteration has two main steps: 1) constructing a feasible solution list from the search space, in this case, it is a list of possible hosts that can accommodate the current VM, 2) performing the local search to find a local best candidate, a sub-optimal solution. To reach the global best solution, at each iteration, the solution i,e., allocated host is updated based on the greedy value

(τ) computed during the construction phase (line no 12).

If the difference (δ) between the current allocated host and newly allocated host's greedy value (τ) is greater than the predefined parameter ϵ, then the iteration process is continued. Otherwise, iteration is stopped and the current allocated host is returned as a result (line no 11-15). Here, ϵ acts as the parameter to decide the expected amount of improvement over the previous solution. If the new solution at current iteration does not give improvement greater than ϵ to the previous iteration, the process is terminated. If this process is failed to find the suitable host for VM then the new host is initiated from the inactive host's list in the data centre resource pool (line no 17-18).

Algorithm 3 refers to the greedy construction solution phase. It takes VM and host list as input and returns the feasible solution list of hosts for a current VM upon each call to this procedure. The first step of this procedure is constructing the RCL which represents the finite solution search space to construct the solution list. The RCL is formed to limit the number of search in the solution space and thus reduce the time complexity. To that end, we completely exclude the inactive hosts and include a α percentage of hosts from the active number of hosts, this ensures the search space is vastly reduced. In addition, selecting the α fraction of active hosts into the RCL is done through random sampling which depicts the probabilistic part of GRASP. The cost for each host in the RCL represented as τ is calculated based on Eq. 4.9 for current time interval (i.e. t=1, i=1). It is important to note that, there can be a repetitive selection of the same sample in different iterations, however, with the sufficiently large number of iterations, it is assumed that the random sampling provides enough distinct distributions from the solution space.

Algorithm 4 shows the local search applied to find the local best for each iteration. Based on the calculated greedy value τ, the best local candidate is returned as a solution.

This algorithm not only reduces the energy consumption, but it also circumvents the potential thermal violation along with satisfying capacity constraints like CPU, memory, and bandwidth, this is evidenced in line 4 of the Algorithm 3 that satisfies all the constraints that are listed in Eq. 4.9. Moreover, the parameters α and ϵ together act as tuning parameters to adjust the amount of greediness and decision time. If accuracy is most crucial in a system, these parameters can be set to a higher value which increases

Table 4.3: Host and VM Configuration

Name	Core	CPU MIPS	RAM	Bandwidth
Intel Xeon_X5670	2	1860	4 GB	1 Gbit/s
Intel Xeon_X5675	2	2660	4 GB	1 Gbit/s
VM1 (Extra Large)	1	2500	870 MB	100 Mbit/s
VM2 (Large)	1	2000	1740 MB	100 Mbit/s
VM3 (Micro)	1	1000	1740 MB	100 Mbit/s
Vm4 (Nano)	1	500	613 MB	100 Mbit/s

Table 4.4: Host Power Consumption at Different Utilization level in Watts

Servers	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100
IBM x3550 M3 (Interl Xeon X5670 CPU)	66	107	120	131	143	156	173	191	211	229	247
IBM x3550 M3 (Intel Xeon X5675 CPU)	58.4	98	109	118	128	140	153	170	189	205	222

the time complexity to find the solution. Consequently, if finding a quick solution is crucial, these values can be set to lower which may compromise the quality of the solution.

4.5 Performance Evaluation

We evaluated the feasibility and performance of our proposed algorithm ETAS and compared it to other baseline algorithms. We created a simulation environment using CloudSim [38] as it allows to model and simulates Cloud computing environments that resemble real-world infrastructure elements. As the default CloudSim toolkit does not include thermal aspects of a data centre, we extend the base classes to incorporate all the thermal parameters into it.

4.5.1 Simulation Setup

In our setup, data centre infrastructure comprises 1000 heterogeneous hosts. The capacity of these hosts are configured based on the IBM x3550 M3 machine with Intel Xeon

X5670 and X5675 processor, configuration of these machines are shown in Table 5.4. The reason for adopting CPUs with less number of cores is to demonstrate the efficiency of dynamic consolidation with a large number of VM migrations. Oppositely, hosts with a large number of cores can accommodate more number of VMs granting less opportunity for VMs migration. Nevertheless, the proposed policies do not affect this factor and also considering the fact that Cloud data centres with massive heterogenous workloads induce enough triggers that generate a large number of VM migrations. The power usage of these systems is adopted from SPECpower benchmark [186], which provides power usage in watts for the respective machines at different CPU utilization level, the power usage of the two hosts that are used in this chapter can be seen in Table 4.4.

VMs are modeled according to the AWS offerings as shown in Table 5.4. The experiments are conducted on a desktop system with 64 bit Ubuntu operating system which is equipped with the Intel(R) Core(TM) i7-6700 processor, 16 GB of primary memory and 1 TB of storgae.

We assume that the hosts/servers are arranged in rack layout, and the racks are arranged in zones. Each zone consists of 10 racks that are laid in 5×2 rows, and each rack has 10 servers, this setup is inspired by the experimentally validated setup in [34]. We assume heat recirculation effects exist within each zone and is negligible across the zones, therefore we do not consider recirculation effect across zones. The heat distribution matrix that represents the recirculation effect within the zone is adopted based on the matrix that was used in [34].

We derive the workload from realistic traces from PlanetLab systems [187], this workload has several months of utilization history record of more than a thousand VMs that are geographically distributed. The data is recorded at an interval of 5 minutes. We use the one-day traces from this to generate the workloads for the VMs.

4.5.2 Baseline Algorithms

In order to compare the performance and efficiency of our proposed algorithm, we consider the following baseline algorithms.

https://aws.amazon.com/ec2/

- **Random:** In this algorithm, all the VMs are placed on randomly selected hosts. This is a most intuitive method which does not consider either thermal or power status of the host.

- **Round Robin (RR):** In this algorithm, all the VMs are placed in a round robin fashion. This method tries to equally distribute the workloads among active hosts.

- **PABFD:** Power-aware Modified Best Fit Decreasing algorithm is proposed by Beloglazov et al. [166]. This energy efficient policy only considers CPU utilization for consolidation while ignoring the thermal aspects.

- **GRANITE:** Greedy VM scheduling algorithm to minimize holistic energy in Cloud data centre proposed in [181]. This policy dynamically migrates VM to balance workload based on a certain temperature threshold.

- **TAS:** Thermal aware scheduling selects the lowest temperature host as the target host. The protective nature towards the thermal status of the host tries to avoid hotspot creation.

For all the aforementioned algorithms, similar policy for the initial two steps (over and underload host detection and VM selection) of dynamic consolidation is used as described in Section 3.2. However, they differ in VM placement strategy.

4.5.3 Parameter Selection

The parameters that occur in different equations are set as follows. We set thermal resistance and the heat capacity in Eq. 5.4 as 0.34 K/w and 340 J/K respectively and the initial CPU temperature is set to 318 K [184]. According to the recommendation from American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) [111], the cold air supply temperature T_{sup} from CRAC is set to 25 °C.

The maximum allowable temperature of the host is between 85 and 100 °C [188], [35], we set it to 95 °C. It is important to note that, the temperature at the host is not only a factor of dissipated CPU temperature, it also includes the temperature that is associated with CRAC supply air (inlet temperature if we exclude recirculation effect).

Excluding this supply air temperature (T_{sup} set to 25 °C) which is regarded as static, the CPU dynamic maximum threshold temperature (T_{red}) is 70 °C. This means, a host CPU is allowed to dissipate the maximum of 70 °C. In other words, we can say that the maximum temperature threshold is 95 °C, and after exclusion of the static part (T_{sup}) dynamic temperature threshold is 70 °C.

The CPU static utilization threshold (U_{max}) is set to 0.9. The hyperparameters α and ϵ are set to 0.4 and 10^{-1} respectively, the choice of selection and effects of these parameters are discussed in Section 4.5.

4.5.4 Results and Analysis

The experiments were run for 5 times and the average results are reported. We ran the simulation for periods of 24 hours and scheduling algorithm was executed after each 5-minute interval to consolidate VMs dynamically. For the PABFD algorithm, there are many combinations based on different VM selection and allocation policies. We use local regression (LR), minimum migration time (MMT) as overload detection and VM selection policy respectively, which has shown to be the most efficient, we vary safety parameter of this algorithm from 1.0 to 1.4 with the increasing step value of 0.1 as described in their work [166].

Metrics

In order to analyze the effectiveness of our proposed solution, we evaluate the results with the following metrics:
Energy: This metric indicates energy consumption of each approach in Kilowatts(kW).
SLA violation: This metric captures performance overhead caused due to dynamic consolidation. This overhead can be captured by the SLA violation metric [166] ($SLA_{violation}$) as shown in Eq. 4.12. Due to the oversubscription policy, hosts may reach its full utilization level (100%), in such case, the VMs on such host experiences the less performance level, this can be described as SLA violation Time per Active Host (SLA_{TAH}), and it is defined as in Eq. 4.10. Furthermore, the consolidation of VMs comes with performance overhead that has caused due to live VM migration [189], this Performance Degradation

due to Migration (PDM) is defined as in Eq. 4.11.

$$SLA_{TAH} = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{max}}{T_{active}} \tag{4.10}$$

$$PDM = \frac{1}{M}\sum_{j=1}^{M}\frac{pdm_j}{C_{demand_j}} \tag{4.11}$$

$$SLA_{violation} = SLA_{TAH} \times PDM \tag{4.12}$$

Here, N is total number of hosts, T_{max} is amount of time $Host_i$ has expereienced 100% of utilization and T_{active} is total active time of $Host_i$. M is total number of VMs, pdm_j is performnace degradation due to live migration of VM_j, in our experiment, it is set to 10%, this value is similar to the one used by Beloglazov et al. [166] . The C_{demand_j} is total amount of CPU resource (MIPS) requested by VM_j in its lifetime. The overall SLA violation of Cloud infrastructure ($SLA_{violation}$) can be captured by combining both of these SLA_{TAH} and PDM as shown in Eq. 4.12.

Hotspots: This metric indicates the number of hosts that have exceeded the redline temperature.

Active hosts: This metric shows the number of active hosts present during the experimented period.

Peak Temperature: This metric indicates maximum temperature attained by any host during a scheduling interval.

Evaluating Energy, Hotspots, and SLA

The energy consumption from each of the policies is shown in Figure 4.3. The random policy has the highest energy usage of 363.19kWh. RR, PABFD, GRANITE, and TAS have 342.82kWh, 235.2kWh, 265. 68kWh and 327.38kWh, respectively, while ETAS has 250.30 kWh of energy consumption with 95% confidence interval (CI): (247.7, 252.5). In

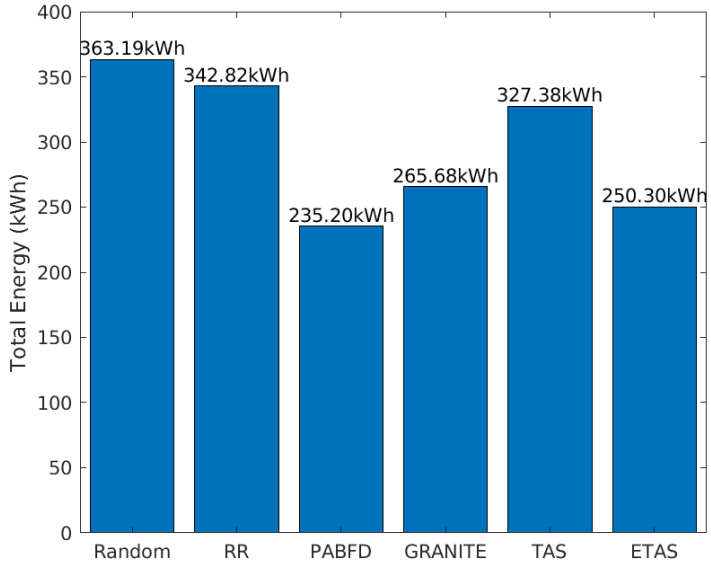


Figure 4.3: Energy Consumption

other words, ETAS consumes 31.1%, 27%, 5%, and 23.5% less than Random, RR, GRANITE, and TAS, respectively. Compared to PABFD, ETAS has a slight increase of 6.4%, PABFD consumes less energy due to the fact that it consolidates VMs on extremely less number of hosts compared to ETAS. This is due to PABFD is aggressive towards the consolidation and accounts for only optimizing computing energy while ignoring the potential thermal constraints which might have unfavorable effects on the system.

GRANITE though integrates both the thermal and energy aspect, it solely considers temperature as threshold parameter to balance the workload. Moreover, it sets the temperature threshold as the lowest temperature of a host among top 10% of high-temperature hosts in the data centre and migrates VMs from those 10% hosts to balance the workload. This particular method is highly correlated with the workload type data centre processes. For example, In the case where not all top 10% servers are exhibiting overload condition, it causes overhead due to excessive VM migrations, oppositely, if more than 10% servers are experiencing overload, GRANITE doesn't account this case too. In addition, identifying % of servers that are experiencing overload is unexplored in this approach (set to 10 % by default). Moreover, it is important to note that, default

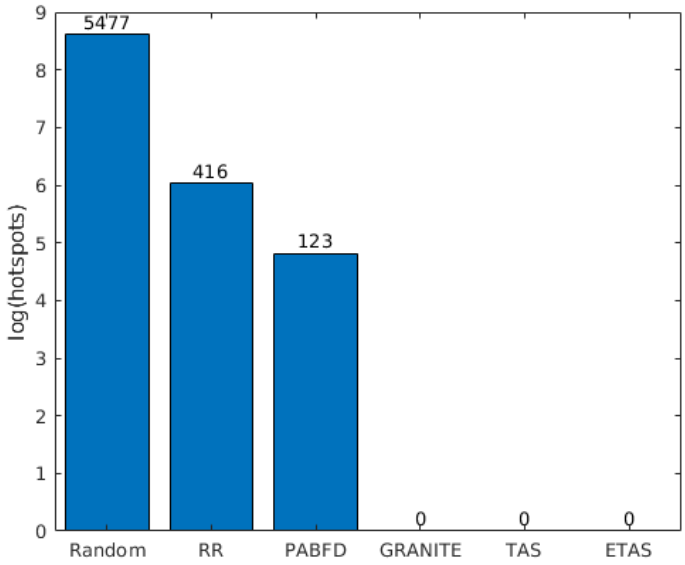


Figure 4.4: Number of Hotspots

GRANITE algorithm balances the workload for only overloaded hosts, here we have applied the underload host management techniques similar to our approach with the GRANITE algorithm to balance the workload. In the process of balancing workload, there will be multiple underloaded hosts over a time, for which GRANITE doesn't have proposed policy. Applying GRANITE without underloaded hosts management resulted in a high amount of energy consumption compared to the other approaches.

Though PABFD consumes slightly less energy compared to our ETAS, it creates a significant number of hotspots. This is evidenced in the bar chart Figure 4.4, where the Y axis in the Figure has a logarithmic scale to respond to the skewness of large values. The consequence of randomness in Random policy has a high impact on both energy consumption and hotspots, this correlation can be observed in the result. Particularly, the Random policy has resulted in 5477 thermal violations in the experimented period. The fair policy distribution of RR performs better than Random policy and it accounts for 416 hotspots, nevertheless, its obliviousness towards thermal and energy parameters cause hotspots and more energy consumption. PABFD has resulted in a total number of 123 hotspots during the experimented period whereas ETAS resulted in 0 hotspots.

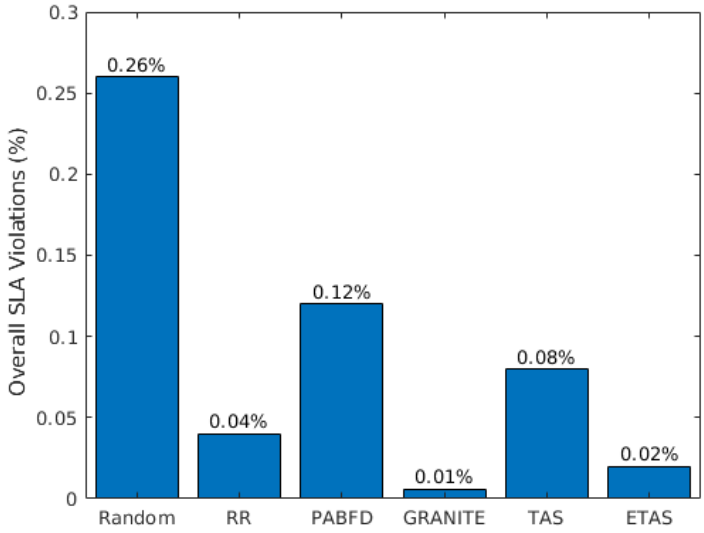


Figure 4.5: SLA Violations

It may seem like ETAS consumes slightly higher amount of energy (250.30 kWh) than PABFD (235.20 kWh), however, occurrence of 123 hotspots in case of PABFD has multiple effects, such as 1) there may be high potential of server failure due to overheating 2) whenever hotspots appear, in a reactive action to this, the data centre administrators enforced to set the cooling temperature to much lower degree °C which further increases the energy consumption, this can be evidenced based on Eq.4.7 where cooling system energy is a function of cold air supply temperature (T_{sup}). Therefore, PABFD energy consumption will surpass when the reactive approach is enforced. This indicates that, in order to evade from hotspot creation, ETAS distributes VMs slightly spread out than PABFD and less than Random, RR, and TAS. In Figure 4.4, it can also be observed that GRANITE and TAS also do not account for any thermal violation due to their thermal-aware scheduling policies. The Conservative approach from TAS towards thermal status alone results in increased power consumption as its power agnostic nature spreads workloads too sparsely on more number of hosts. Consequently, thermal proactiveness and energy-aware placement of VMs from ETAS avoids hotspots and saves the energy.

Figure 4.5 compares the percentage of SLA violation between all the policies. Random, PABFD, and TAS are least efficient while RR, GRANITE, and ETAS do not account

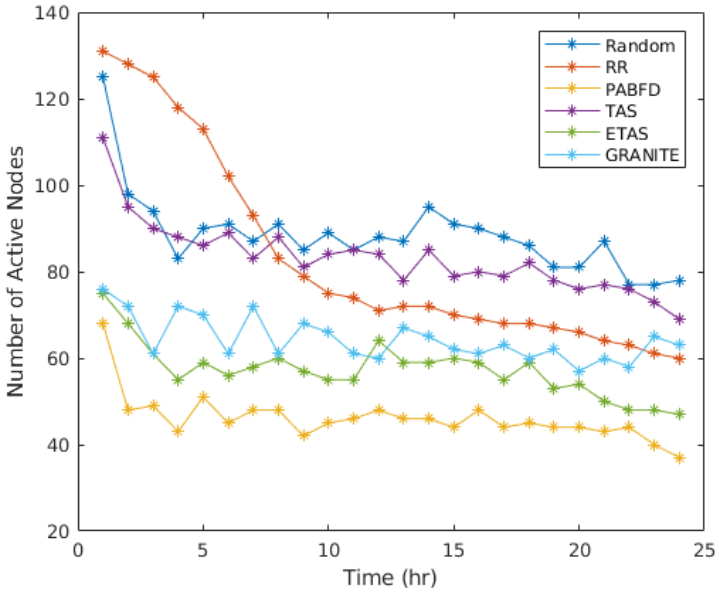


Figure 4.6: Average Number of Active Nodes per Hour

for higher violation of SLA. Even though RR does not consider the SLA requirements while scheduling decision, its inherent characteristics that equally distributes the workload among hosts resulted in reduced SLA violations compared to Random, PABFD, and TAS. However, SLA violation percentage from ETAS (0.02%) may not seem like to completely outperform the RR (0.04%), but this SLA obtained by ETAS is while simultaneously optimizing total energy consumption which has an orthogonal tradeoff with SLA [166], whereas RR does not consider any aspect of energy optimization. Hence, ETAS is capable of minimizing SLA violations with better performance due to its performance and SLA aware scheduling policies.

Runtime Evaluation

To completely understand the performance during runtime, we collect and report runtime data of following metrics: (1) number of active nodes; (2) maximum peak temperature attained by any of the hosts at each scheduling interval (5 min); (3) running time of each of the policies; (4) effect of different CPU threshold on energy and number of active nodes.

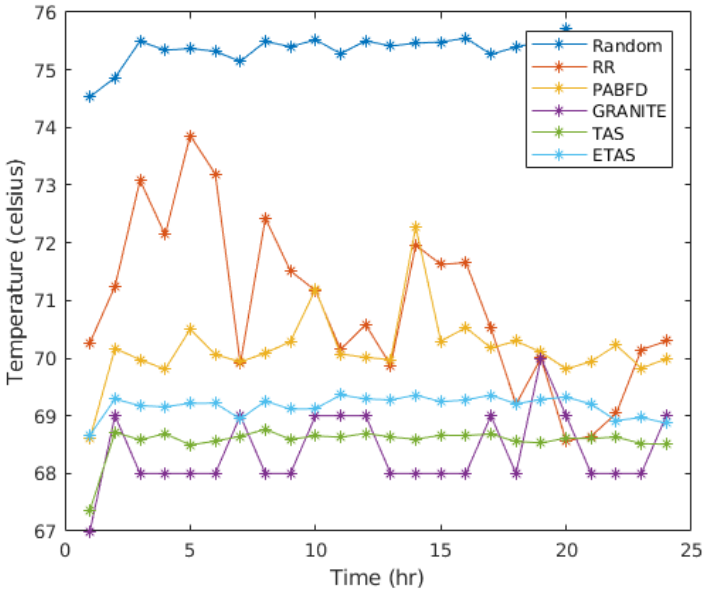


Figure 4.7: Peak Temperature of a Host

The number of active nodes in the experimented duration can be observed in Figure 4.6. For the sake of understanding and clear visibility of plots, we take the average value for each hour (12 intervals average represent a 1-hour data) for the results. PABFD results in less active nodes, while ETAS has a modest increase in a number of active nodes compared to PABFD. GRANITE has a small increase in a number of active nodes than ETAS, the reflection of this is evidenced by energy increase based on Eq. 4.8 and can be observed in Figure 4.3. The Random policy has the highest number of active nodes among all. TAS has a minimal increase in the number of active nodes while less in the case of RR. Moreover, a correlation between the number of active nodes, hotspots, and energy can be derived as the policies with less number of active nodes are prone to the occurrence of hotspots. However, the Random policy is exceptional due to its arbitrary decisions. The difference in a number of the active nodes is not huge among all the policies due to the reason that in our consolidation process, we use same policy to detect over and underloaded hosts along with VM selection policy which contributes largely to this factor. The observed difference exists because of different VM placement decisions by each algorithm. Regardless, it can be inferred that ETAS has a modest

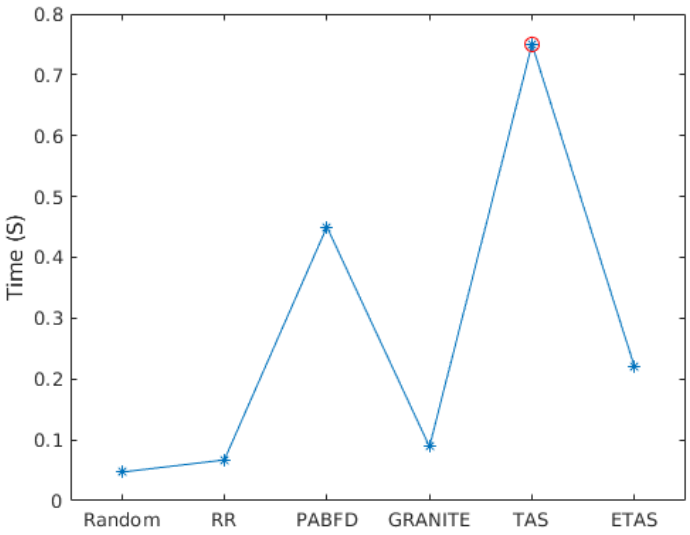


Figure 4.8: Running Time Analysis of Different Policies

increase in a number of active hosts compared to PABFD which is necessary to avoid high concentration of workload and prevent the potential hotspot creation.

Figure 4.7 illustrates the comparison of the peak temperature of a host by each of the policies. The proposed ETAS never exceeds the redline temperature due to its thermal-aware placement of VMs and it operates near to redline which increases the resource utilization and reduces the cooling cost. TAS always operates at a much lower level temperature and PABFD almost operates around redline temperature (70 °C) and exceeds the red line in multiple instances. The peak temperature of a host from GRANITE policy is the lowest among all as it considers temperature as threshold parameter alone and migrates VMs from high-temperature hosts to balance the workload. Though RR equally distributes workload, some of the hosts exceed the redline temperature due to its thermal unawareness. Note that, the represented results are not an average temperature of all the hosts, instead, the values represent the temperature of the hottest machine during each scheduling interval.

To analyze the computation overhead of different policies, we report the empirical values of the running time of each of these policies. This time indicates mean VM allocation time, which includes time taken by an algorithm to migrate a VM to a destination

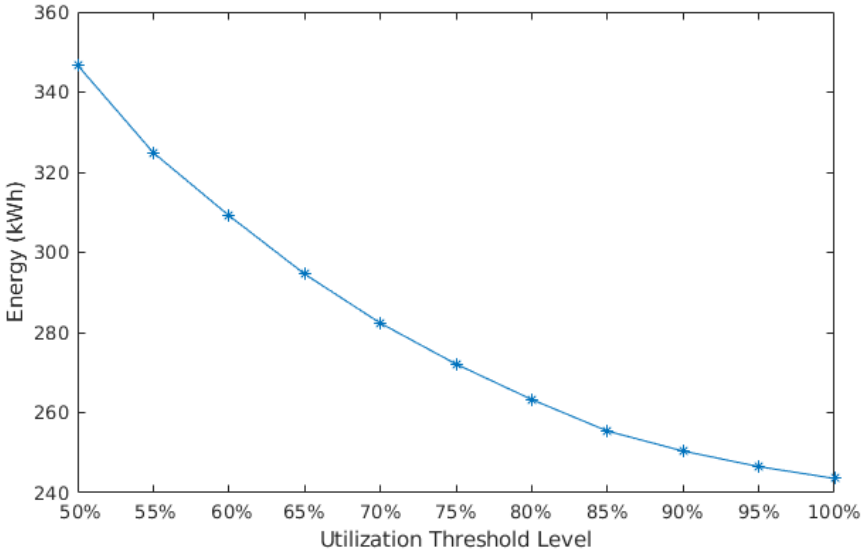


Figure 4.9: Energy Consumption with Different Utilization Thresholds

host, here major variant complexity is from deciding a target host for all VMs in this process. The Figure 4.8 illustrates the running time of each of the policies. It can be observed that, due to the arbitrary selection of hosts by Random policy, it has the lowest running time compared to all. Since RR just need to select the next suitable host in a queue for allocation, it also accounts for less runtime. GRANITE has slightly more runtime compared to Random and RR. TAS and PABFD have high runtime overhead as they have to perform the maximum number iterations to find possible hosts for VM allocation. ETAS has minimal runtime compared to TAS and PABFD as it doesn't search complete solution space, more importantly, we can tune the runtime of ETAS with the energy trade-offs which is discussed in next section.

Performance of ETAS with different CPU utilization threshold values (U_{max}) can be observed in Figure 4.9 and Figure 4.10 which show the energy consumption and number of active nodes with different U_{max} values, respectively. For a lower utilization threshold, the energy consumption is higher as more number of hosts in the data centre will be active to accommodate the given workloads. If we set the threshold to a lower value, the utilization of data centre decreases and energy consumption increases. Hence, to achieve energy efficiency, the threshold should be high enough to utilize the data centre

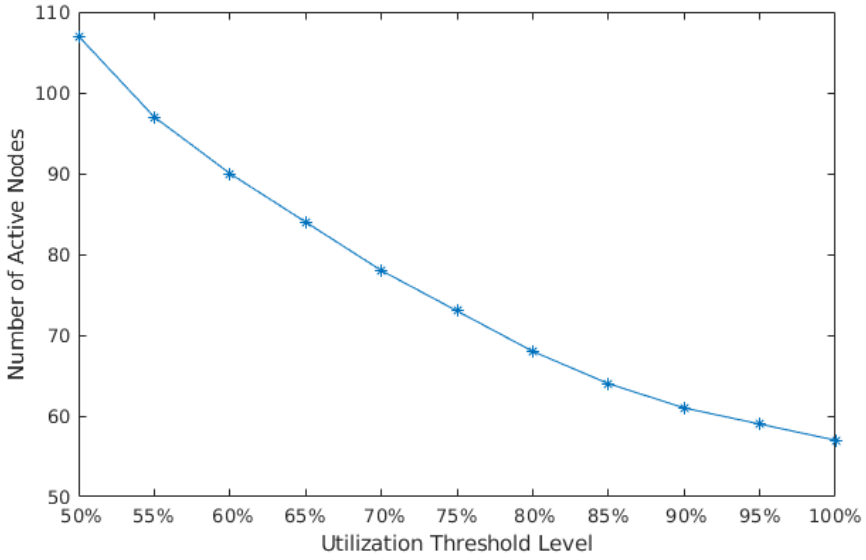


Figure 4.10: Number of Active Nodes with Different Utilization Thresholds

resources efficiently. Furthermore, it can be observed that, after the threshold value of 0.9 (90%), the amount of reduction in energy consumption is less. Consequently, the extremely high value of U_{max} will result in a high number of QoS/ SLA violations.

In conclusion, our proposed energy and thermal-aware algorithm reduces the overall energy consumption of a data centre while circumventing the hotspots by operating within the redline temperature. It also has minimal impact on the SLA violation. ETAS increases the global utilization of resources while ensuring the thermal constraints. In addition, the variant of GRASP heuristic is fast, lightweight and can be used in an online system.

4.5.5 Sensitivity Analysis

The performance of our proposed algorithm is highly influenced by the parameters α and ϵ. To analyze this, we carried a sensitivity analysis and identified the best settings for these parameters. The values for α and ϵ that are considered as listed in Table 4.5. These two parameters form 25 different combinations altogether. We evaluated the effect of these parameters on time and energy. The time represents a mean VM allocation time,

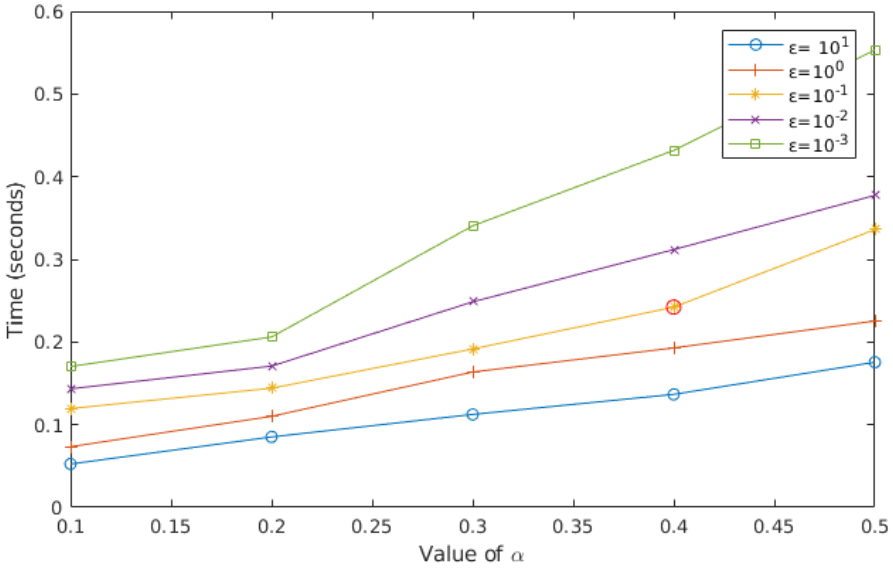


Figure 4.11: Effect on Time

Table 4.5: Parameter and Values

Parameter	Values				
α	0.1	0.2	0.3	0.4	0.5
ϵ	10^1	10^0	10^{-1}	10^{-2}	10^{-3}

i.e., decision time to find the target host for a VM to be migrated.

The effect of hyperparameters, α and ϵ on time and energy with all the 25 combinations can be observed in Figure 4.11 and 4.12, respectively. The higher ϵ and lower α value result in higher energy consumption with the less time. However, after $\alpha = 0.4$, the energy saving is almost linear. Similarly, the lower ϵ yields better energy saving but it increases the time exponentially. The ideal setting for α and ϵ are 0.4 and 10^{-1}, respectively which can be observed from Figure 4.12 and Figure 4.11. Therefore, these parameters can be tuned to manage the trade-off between time and energy.

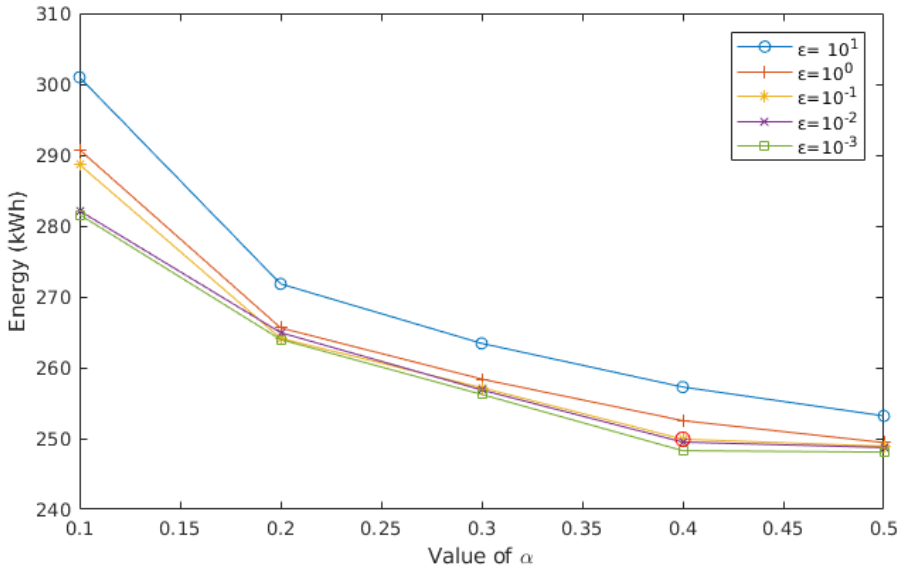


Figure 4.12: Effect on Energy

4.6 Summary

The complexity of thermal behaviour and uncertain workload makes scheduling a complex problem. To address the dynamic consolidation with both thermal and energy awareness, in this chapter, we optimize the computing and cooling energy together with the aim of reducing the overall energy consumption of a data centre while proactively mitigating the effect of hotspots.

Cloud data centres are increasing in both number and size due to the rapid adoption of Cloud computing in many spectrum of IT. Minimizing the energy consumption to increase the profit for Cloud service providers without affecting the performance of user applications is a paramount need.

In this chapter, we proposed a dynamic consolidation framework for holistic management of Cloud resources by optimizing both computing and cooling systems together. Through our proposed ETAS algorithm, we have managed the tradeoff between aggressive consolidation and sparse distribution of VMs which has an effect on energy and hotspots. Moreover, based on the system requirement, ETAS algorithm can be adjusted to manage the computational time and quality of the solution. The experiments

are conducted with traces from real system and results have demonstrated that the proposed ETAS algorithm saves 23.5% and 5% more energy as compared to the thermal-aware algorithms. Compared to the Energy-aware algorithm, ETAS is capable of avoiding hotspots with a modest increase in energy consumption.

This chapter presented a dynamic consolidation technique for integrated computing and cooling system energy optimisation (using analytical thermal models). In the next chapter, we study data-driven ML-based thermal modelling in a data centre. In addition, we propose a scheduling technique for peak temperature minimisation using the proposed prediciton models.

Chapter 5

Thermal Prediction for Efficient Energy Management of Clouds

Thermal management in the hyper-scale Cloud data centres is a critical problem. Increased host temperature creates hotspots which significantly increases cooling cost and affects reliability. Accurate prediction of host temperature is crucial for managing the resources effectively. Temperature estimation is a non-trivial problem due to thermal variations in the data centre. Existing solutions for temperature estimation are inefficient due to their computational complexity and lack of accurate prediction. However, data-driven machine learning methods for temperature prediction is a promising approach. In this regard, we collect and study data from a private Cloud and show the presence of thermal variations. We investigate several machine learning models to accurately predict the host temperature. Specifically, we propose a gradient boosting machine learning model for temperature prediction. The experiment results show that our model accurately predicts the temperature with the average RMSE value of 0.05 or an average prediction error of 2.38 °C, which is 6 °C less as compared to an existing theoretical model. In addition, we propose a dynamic scheduling algorithm to minimize the peak temperature of hosts. The results show that our algorithm reduces the peak temperature by 6.5 °C and consumes 34.5% less energy as compared to the baseline algorithm.

5.1 Introduction

Modern Cloud data centres' rack-mounted servers can consume more than 1000 watts of power each and attain peak temperature as high as 100 °C [111]. The power consumed

This chapter is derived from:

- **Shashikant Ilager**, Kotagiri Ramamohanarao, and Rajkumar Buyya, "Thermal Prediction for Efficient Energy Management of Clouds using Machine Learning", *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, Volume 32, No. 5, Pages: 1044-1056, ISSN: 1045-9219, IEEE CS Press, USA, May 2021.

by a host is dissipated as heat to the ambient environment, and the cooling system is equipped to remove this heat and keep the host's temperature below the threshold. Increased host temperature is a bottleneck for the normal operation of a data centre as it escalates the cooling cost. It also creates hotspots that severely affect the reliability of the system due to cascading failures caused by silicon component damage. The report from Uptime Institute [190] shows that the failure rate of equipment doubles for every 10 °C increase above 21 °C. Hence, thermal management becomes a crucial process inside the data centre Resource Management System (RMS).

Therefore, to minimize the risk of peak temperature repercussions, and reduce a significant amount of energy consumption, ideally, we need accurate predictions of thermal dissipation and power consumption of hosts based on workload level. In addition, a scheduler that efficiently schedules the workloads with these predictions using certain scheduling policies. However, accurate prediction of a host temperature in a steady-state data centre is a non-trivial problem [34, 191]. This is extremely challenging due to complex and discrepant thermal behavior associated with computing and cooling systems. Such variations in a data centre are usually enforced by CPU frequency throttling mechanisms guided by Thermal Design Power (TDP), attributes associated with hosts such as its physical location, distance from the cooling source, and also thermodynamic effects like heat recirculation [34, 191]. Hence, the estimation of the host temperature in the presence of such discrepancies is vital to efficient thermal management. Sensors are deployed on both the CPU and rack level to sense the CPU and ambient temperature, respectively. These sensors are useful to read the current thermal status. However, predicting future temperature based on the change in workload level is equally necessary for critically important RMS tasks such as resource provisioning, scheduling, and setting the cooling system parameters.

Existing approaches to predict the temperature are inaccurate, complex, or computationally expensive. The widely used theoretical analytical models [34, 35, 184, 191, 192] that are built based on mathematical relations between different cyber-physical components lack the scalability and accurate prediction of the actual temperature. In addition, theoretical models fail to consider several variables that contribute towards temperature behavior and they need to be changed for different data centres. Computational Fluid

Dynamics (CFD) models are also predominantly used [32, 33] for accurate predictions, but their high complexity requires a large number of computing cycles. Building these CFD models and executing them can take hours or days, based on individual data centre complexity [193]. The CFD models are useful in initial design and calibration of data centre layout and cooling settings, however, it is infeasible for the realtime tasks (e.g., scheduling in large scale Clouds) that are dynamic and require quick online decisions. Moreover, CFD simulation requires both computational (e.g, the layout of the Data centre, open tiles) and physical parameters, and changes to these parameters need expensive retraining of the models [194]. However, our approach is fast and cost-effective as it solely relies on the physical sensor data that are readily available on any rack-mounted servers and implicitly captures variations. Hence, data-driven methods using machine learning techniques is a promising approach to predict the host temperature quickly and accurately.

Machine learning (ML) techniques have become pervasive in modern digital society mainly in computer vision and natural language processing applications. With the advancement in machine learning algorithms and the availability of sophisticated tools, applying these ML techniques to optimize large scale computing systems is a propitious avenue [21, 36, 195, 196]. Recently, Google has reported a list of their efforts in this direction [43], where they optimize several of their large scale computing systems using ML to reduce cost, energy and increase the performance. Data-driven temperature predictions are highly suitable as they are built from actual measurements and they capture the important variations that are induced by different factors in data centre environments. Furthermore, recent works have explored ML techniques to predict the data centre host temperature [191, 197]. However, these works are applied to HPC data centres or similar infrastructure that relies on both application and physical level features to train the models. In addition, they are application-specific temperature estimations. Nevertheless, the presence of the virtualization layer in Infrastructure Clouds prohibits this application-specific approach due to an isolated execution environment provided to users. Moreover, getting access to the application features is impractical in Clouds because of privacy and security agreements between users and Cloud providers. Consequently, we present a host temperature prediction model that completely relies on

features that can be directly accessed from physical hosts and independent of the application counters.

In this regard, we collect and study data from our University's private research Cloud. We propose a data-driven approach to build temperature prediction models based on this collected data. We use this data to build the ML-based models that can be used to predict the temperature of hosts during runtime. Accordingly, we investigate several ML algorithms including variants of regression models, a neural network model namely Multilayer Perceptron (MLP), and ensemble learning models. Based on the experimental results, the ensemble-based learning, gradient boosting method, specifically, XGBoost [198] is chosen for temperature prediction. The proposed prediction model has high accuracy with an average prediction error of 2.5 °C and Root Mean Square Error (RMSE) of 0.05. Furthermore, guided by these prediction models, we propose a dynamic scheduling algorithm to minimize the peak temperature of hosts in a data centre. The scheduling algorithm is evaluated based on real-world workload traces and it is capable of circumventing potential hotspots and significantly reduces the total energy consumption of a data centre. The results have demonstrated the feasibility of our proposed prediction models and scheduling algorithm in data centre RMS.

In summary, the key contributions of this chapter are:

- We collect physical-host level measurements from a real-world data centre and show the thermal and energy consumption variations between hosts under similar resource consumption and cooling settings.
- We build machine learning-based temperature prediction models using fine-grained measurements from the collected data.
- We show the accuracy and the feasibility of proposed prediction models with extensive empirical evaluation.
- We propose a dynamic workload scheduling algorithm guided by the prediction methods to reduce the peak temperature of the data centre that minimizes the total energy consumption under rigid thermal constraints.

The rest of the chapter is organized as follows. The relevant literature for this chapter is

discussed in Section 5.2. The motivations for this research problem and thermal implications in the Cloud are explained in Section 5.3. Section 5.4 proposes a thermal prediction framework and explores different ML algorithms. Section 5.5 describes the gradient boosting based prediction model. The feasibility of the prediction model is evaluated against a theoretical model in Section 5.6. Section 5.7 presents a dynamic scheduling algorithm. The analysis of scheduling algorithm results is done in Section 5.8 and the feature set analysis is described in Section 5.9. Assumptions and applicability specific to this chapter is explained in Section 5.9.1. Finally, Section 5.10 concludes the chapter.

5.2 Related Work

Thermal management using theoretical analytical models has been studied by many researchers in the recent past [34, 35, 170, 199]. These models based on mathematical relationships to estimate the temperature are not accurate enough when compared to the actual values. Moreover, [34, 199] uses analytical models and targets HPC systems where jobs have specific completion time, while our solution target the virtualized Cloud datacentres with long-running applications that need dynamic scheduling and migration in realtime. Furthermore, some of the studies have also explored using CFD models [32]. Computational Fluid Dynamics (CFD) models provide an accurate thermal measurement, however, their massive computational demand hinders their adoption in realtime online tasks such as scheduling. Researchers are audaciously exploring data-driven ML algorithms to optimize the computing system efficiency [43, 195]. With the help of ML techniques, Google data centres are able to reduce up to 40 % of their cooling costs [21].

Many researchers in recent years study thermal and energy management inside the data centre using machine learning techniques. The vast applications have been used for finding an optimal setting or configurations of systems to achieve energy efficiency [200]. However, ML techniques specific to temperature prediction are studied by Zhang et al. [201] where they proposed the Gaussian process-based host temperature prediction model in HPC data centres. They used a two-node Intel Xeon Phi cluster to run the HPC test applications and collect the training data. In addition, they also proposed a

greedy algorithm for application placement to minimize the thermal variations across the system. In an extended work [191], they enhanced their solution to include more efficient models such as lasso linear and Multilayer Perceptron (MLP). The results have shown that predictive models are accurate and perform well in data centre resource management aspects. Imes et al. [200] explored different ML classifiers to configure the different hardware counters to achieve energy efficiency for a given application. They tested 15 different classifiers including Support Vector Machine (SVM), K-Nearest Neighbours (KNN), and Random Forest (RF), etc. This work only considers energy as an optimization metric ignoring the thermal aspect. Moreover, these works are specific to HPC data centres where temperature estimation is done for application-specific which requires access to application counters. Nevertheless, our proposed solution is for Infrastructure Clouds, where such an approach is not feasible due to limited access to application counters enforced by the isolated virtualized environment. Thus, we rely on features that completely surpass application counters and only consider host-level resource usage and hardware counters and yet achieve a high prediction accuracy.

Furthermore, Ignacio et al. [202] showed the thermal anomaly detection technique using Artificial Neural Networks (ANNs). They specifically use Self Organising Maps (SOM) to detect abnormal behavior in the data centre from a previously trained reliable performance. They evaluated their solution using traces of anomalies from a real data centre. Moore et al. [193] proposed Weatherman, a predictive thermal mapping framework for data centres. They studied the effect of workload distribution on cooling settings and temperature in the data centre. These models are designed to find the thermal anomalies and manage the workload at a data centre level without giving any attention to accurate temperature prediction.

In addition to thermal management, many others applied ML techniques for scheduling in distributed systems to optimize the parameters such as energy, performance, and cost. Among many existing ML approaches, Reinforcement Learning (RL) is widely used for this purpose [36, 203, 204]. Orheab et al. [203] studied the RL approach for scheduling in distributed systems. They used the Q-learning algorithm to train the model that learns optimal scheduling configurations. In addition, they proposed a platform that provides scheduling as a service for better execution time and efficiency.

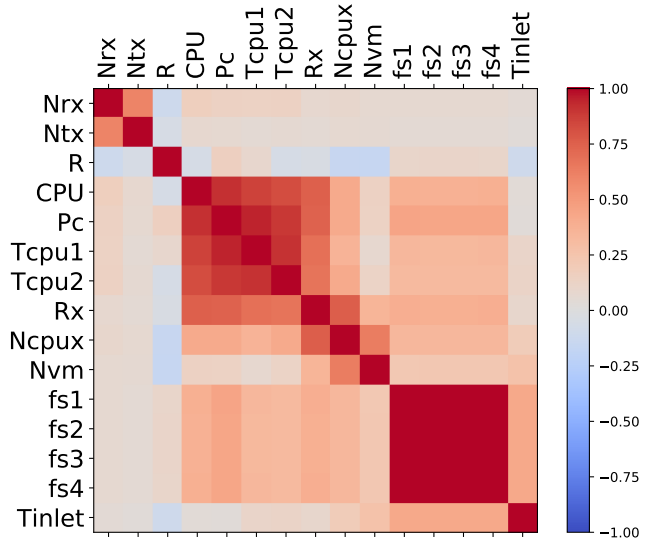


Figure 5.1: Correlation between all features

Cheng et al. [36] proposed the DRL Cloud, which provides an RL framework for provisioning and task scheduling in the Cloud to increase energy efficiency and reduce the task execution time. Similarly, Mao et al. [204] studied deep RL based resource management in distributed systems. Learning to schedule is prominent with RL based methods due to the fact that RL models keep improving in runtime [45] which is convenient for scheduling. However, this chapter is different from these works in a way that, the primary objective of our problem is to estimate the data centre host temperature accurately to facilitate the resource management system tasks. In this regard, solution proposed in this chapter acts as complementary to these solutions where such thermal prediction models can be adopted by these ML-based scheduling frameworks to further enhance their efficiency.

5.3 Motivation: Intricacies in Cloud Data Centres' Thermal Management

Thermal management is a critical component in Cloud data centre operations. The presence of multi-tenant users and their heterogeneous workloads exhibit non-coherent behavior with respect to the thermal and power consumption of hosts in a Cloud data

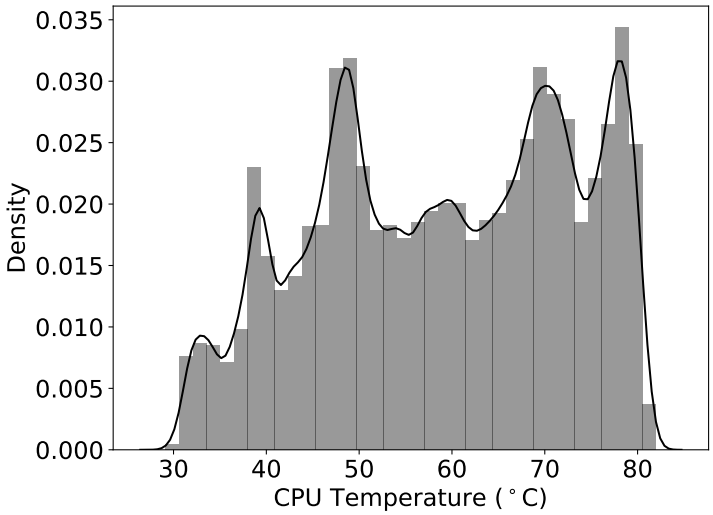


Figure 5.2: CPU temperature distribution

centre. Reducing even one degree of temperature in cooling saves millions of dollars over the year in large scale data centres [21]. In addition, most data centres and servers are already equipped with monitoring infrastructure, that has several sensors to read the workload, power, and thermal parameters. Using this data to predict the temperature is cost-effective and feasible. Thereby, to analyze the complex relationships between different parameters that influence the host temperature, we collected data from a private Cloud and studied it for intrinsic information. This data includes resource usage and sensor data of power, thermal, and fan speed readings of hosts. The detailed information about the data and collection method is described in Section 5.4.2.

The correlation between different parameters (Table 5.1) and temperature distribution in the data centre can be observed in Figure 5.1 and 5.2.These figures are drawn from the data recorded on 75 hosts over a 90 days period. The logging interval was 10 minutes (i.e.,$75 \times 90 \times 24 \times 6$ records). The correlation plot in Figure 5.1 is based on the standard pairwise Pearson correlation coefficient represented as a heat map. Here, the correlation value ranges from -1 to 1, where the value is close to 1 for highly correlated features, 0 for no correlation, and -1 for the negative correlation. For better illustration, the values are represented as color shades as shown in the figure. In addition, the correlation matrix is clustered based on pairwise Euclidean distance to enhance interpretability. It is

evident that the CPU temperature of a host is highly influenced by power consumption and CPU load. However, factors like memory usage and machine fan speeds also have some degree of interdependence with it. Additionally, inlet temperature has a positive correlation with fan speeds and the number of VMs running on a host.

The high number of hosts operating at a peak CPU temperature can be observed from Figure 5.2. The figure represents a histogram of the temperature distribution of all hosts. Thereby each bin on the x axis represents a quantized CPU temperature and the y axis the corresponding probability density value. CPU temperature of hosts can reach more than 80 °C and the occurrence of such conditions are numerous which is evidenced by high-density value on the y axis for the respective bin. In addition, hosts exhibit inconsistent thermal behavior based on several factors. This non-linear behavior of hosts presents a severe challenge in temperature estimation. A single theoretical mathematical model, applied even for homogeneous nodes, fails to accurately predict the temperature. Two homogeneous nodes at a similar CPU load observe different CPU temperatures. For instance, at a CPU load of 50% of the different hosts in our data set, CPU temperature varies up to 14 °C. Furthermore, with similar cooling settings, inlet temperature also varies up to 9 °C between hosts. These temperature variations are caused by factors like physical attributes such as the host's location, thermodynamic effects, heat recirculation, and thermal throttling mechanisms induced by the operating system based on workload behaviors [191]. Therefore, a temperature estimation model should consider the non-linear composite relationship between hosts.

Motivated by these factors, we try to rely on data-driven prediction approaches compared to existing rigid analytical and expensive CFD based methods. We use the collected data to build the prediction models to accurately estimate the host temperature. Furthermore, guided by these prediction models, we propose a simple dynamic scheduling algorithm to minimize the peak temperature in the data centre.

5.4 System Model and Data-Driven Temperature Prediction

In this section, we describe the system model and discuss methods and approaches for Cloud data centre temperature prediction. We use these methods to further optimize

Table 5.1: Definition of features collected

Features	Definition
CPU	CPU Load (%)
R	RAM- Random Access Memory (MB)
R_x	RAM in usage (MB)
N_{CPU}	Number of CPU cores
N_{CPUx}	Number of CPU cores in use
N_{Rx}	Network inbound traffic (Kbps)
N_{Tx}	Network outbound traffic (Kbps)
P_c	Power consumed by host (watts)
T_{cpu1}	CPU 1 temperature (°C)
T_{cpu2}	CPU 2 temperature (°C)
fs_1	fan1 speed (RPM)
fs_2	fan2 speed (RPM)
fs_3	fan3 speed (RPM)
fs_4	fan4 speed (RPM)
T_{in}	Inlet temperature (°C)
N_{vm}	Number of VMs running on host

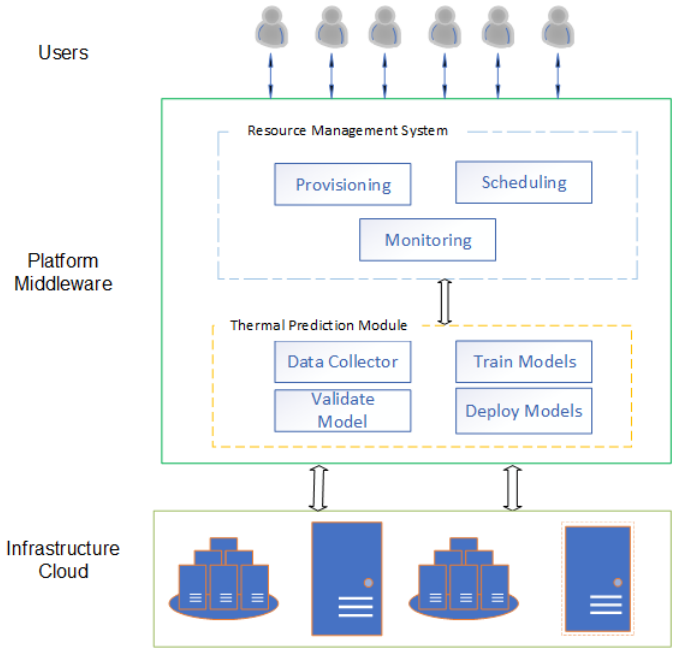


Figure 5.3: System Model for Thermal Prediction for Resource Management of Clouds

our prediction model in Section 5.5.

5.4.1 System Model

A system model for predictive thermal management in the Cloud data centre is shown in Figure 5.3. A Resource Management System (RMS) interacts with both, the users and the thermal prediction module, to efficiently manage the underlying resources of the Cloud infrastructure. The prediction module consists of four main components, i.e., data collecting, training the suitable model, validating the performance of the model, and finally deploying it for runtime usage. RMS in a data centre can use these deployed models to efficiently manage the resources and reduce the cost. The important elements of the framework are discussed in the following subsections.

5.4.2 Data Collection

An ML-based prediction model is as good as the data it has been used to train. In the data centre domain, training data can include application and physical level features

Table 5.2: Description of the feature set variations in the dataset (aggregated from all the hosts)

	$CPU(\%)$	R_x	N_{Rx}	N_{Tx}	N_{vm}	N_{CPUx}	P_c	fs_2	fs_1	fs_3	fs_4	T_{cpu1}	T_{cpu2}	T_{in}
Min	0	3974	0	0	0	0	55.86	5636	5686	5688	5645	29.14	25.46	13.33
Max	64.74	514614	583123.08	463888.76	21	101	380.53	13469	13524	13468	13454	82	75.96	18.05
Mean	18.09	307384.48	2849.00	1354.164	9	54	222.73	9484	9501	9490	9480	59.50	50.78	25.75

Table 5.3: Private Cloud data collected for this chapter

#Hosts	**#VMs**	**Total CPU Cores**	**Total Memory**	**Collection Period**	**Collection Interval**
75	650	9600	38692 GB	90 days	10 Minute

to train the model [191]. The application features include instruction count, number of CPU cycles, cache metrics (read, write and miss), etc. Accordingly, physical features include host-level resource usage (CPU, RAM, I/O, etc.) and several sensor readings (power, CPU temperature, fan speeds). Relying on both of these features is feasible in bare metal HPC data centres where administrators have exclusive access to the application and physical features. However, in the case of Infrastructure as Service (IaaS) Clouds, resources are virtualized and provisioned as VMs or containers, thus, giving users exclusive isolated access to the application execution environment. The presence of a hypervisor or container-based virtualization in IaaS Clouds restricts access to application-specific features. Moreover, a diverse set of users in the Cloud have a different type of workloads exhibiting different application behaviors which impede Cloud RMS to rely on application-specific features. As a consequence, to predict host temperature, the RMS is required to monitor fine-grained resource usage and physical features of the host system that can be directly accessed. In this regard, we show that this data is adequate to predict the host temperature accurately.

The Melbourne Research Cloud (MRC) provides Virtual Machines (VM) to students and researchers. The representative data is collected from a subset of machines from MRC. This computing infrastructure provides computing facilities to students and researchers as a virtual machine (VM). We collect data from a subset of the total machines in this Cloud. A brief summary of this data is presented in Table 5.3. It includes logs

https://docs.Cloud.unimelb.edu.au/

of 75 physical hosts having an average number of 650 VMs. The data is recorded for a period of 3 months and the log interval is set to 10 minutes. The total count of resources includes 9600 CPU cores and 38692 GB of memory. After data filtration and cleaning, the final dataset contains 984712 tuples, each host approximately having around 13000 tuples. Each tuple contains 16 features including resource and usage metrics, power, thermal, and fan speed sensors measurements. The details of these features are given in Table 5.1. As each host is equipped with two distinct CPUs, two temperature measurements are reported per machine. In addition, each system has four separate fans installed to provide cooling. The reason to collect data for an extended period is to capture all the dynamics and variations of parameters to train the model effectively. This is only possible when host resources have experienced different usage levels over time. A model built over such data allows accurate prediction in dynamic workload conditions. An overview of variations of all parameters is depicted in Table 5.2 (N_{CPU} and R are not included as they represent constant resource capacity).

To collect this data, we run a collectd daemon on every host in the data centre, which is a standard open-source application that collects system and application performance counters periodically through system interfaces such as IPMI and sensors. These metrics are accessed through network API's and stored in a centralized server in the CSV format. We used several bash and python scripts to pre-process the data. Specifically, python pandas package to clean and sanitize the data. All invalid measurements (e.g. NaN) were removed.

5.4.3 Prediction Algorithms

The supervised Machine Learning (ML) algorithms broadly falls into two categories, including regression and classification. The choice of regression-based algorithms for our problem is natural since we aim to estimate the numerical output variable, i.e., temperature. In the search for suitable prediction mechanisms, we have explored different ML algorithms including different regression techniques, such as Linear Regression (LR), Bayesian Regression (BR), Lasso Linear Regression (LLR), Stochastic Gradient Descent

https://collectd.org/
https://pandas.pydata.org/

regression (SGD), an Artificial Neural Network (ANN) model called Multilayer Perceptron (MLP), and an ensemble learning technique called gradient boosting, specifically, eXtreme Gradient Boosting (XGBoost).

Since each host in our cluster has two CPUs that are jointly controlled by the same operating system (which may dynamically move workloads between them), we always regard the maximum of the respective two CPU temperature measurements as the systems' effective CPU temperature. We aim to build a model for each host to accurately capture its thermal behavior properties. For that reason, instead of solely predicting CPU temperature, we predict the host ambient temperature (T_{amb}) which is a combination of inlet temperature and CPU temperature [170]. The reason to consider ambient temperature instead of CPU temperature is manifold. First, by combining the inlet and CPU temperature, it is feasible to capture thermal variations that are induced by both the inlet and CPU temperature (cause of these variations are discussed in Section 5.3). Second, at a data centre level, cooling settings knobs are adjusted based on host ambient temperature rather than individual CPU temperature [193]. In addition, resource management systems in the data centre consider host-level ambient temperature as a threshold parameter whereas operating system level resource management techniques rely on CPU temperature.

Therefore, to build the prediction model for individual hosts, we parse the data set and partition it based on host IDs. For each individual host, the feature set consists of a variable number of tuples, with each tuple having these features (CPU, R, R_x, N_{CPU}, N_{CPUx}, N_{Rx}, N_{Tx}, N_{vm}, P_c, $fs_1 - fs_4$, T_{amb}). Note that, we have excluded inlet and CPU temperatures from the list, as we have combined these as ambient temperature (T_{amb}) which is our target prediction variable.

We used sci-kit learn package [205] to implement all the algorithms. For XGBoost, we used a standard python package available on Github. The parameters for each of the algorithms are set to their default settings in our implementation. For MLP, it follows a standard 3 layers architecture, with the number of neurons at a hidden layer set to 5 and a single output neuron, and 'ReLu' as the activation function.

To avoid the overfitting of the models, we adopt k-fold cross-validation where the

https://github.com/dmlc/xgboost

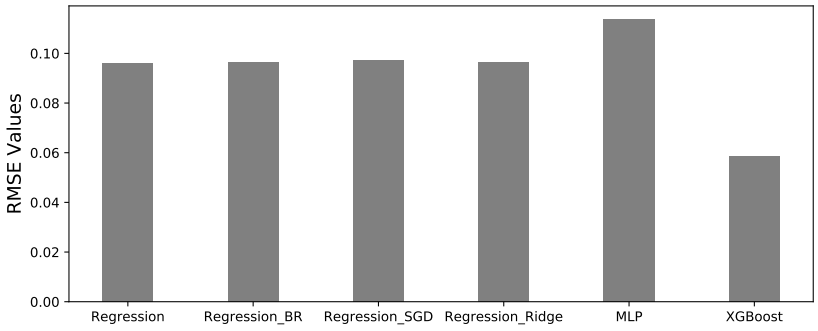


Figure 5.4: Average prediction error between different models

value of k is set to 10. Furthermore, to evaluate the goodness of fit for different models, we use the Root Mean Square Error (RMSE) metric which is a standard evaluation metric in regression-based problems [206]. The RMSE is defined as follows.

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{5.1}$$

In Equation 5.1, y_i is the observed value, $\hat{y}_i$ is the predicted output variable, and n is the total number of predictions. The value of RMSE represents the standard deviation of the residuals or prediction errors. The prediction models attempt to minimize an expectation of loss, thus, lower RMSE values are preferred.

The performance of different algorithms is shown in Figure 5.4. These results are an average of all the hosts' prediction model results. In Figure 5.4, we can observe that XGBoost has a very low RMSE value, indicating that, the residuals or prediction errors are less and its predictions are more accurate. We observed that MLP has a high error value compared to other algorithms.

In addition, different regression variants have performed almost similar to each other. As the gradient boosting method XGBoost results are promising, we focus more on this algorithm to explore it further, optimize, and adapt it for further scheduling as explained in Section 5.7.

5.5 Learning with Extreme Gradient Boosting (XGBoost)

Boosting is an ensemble-based machine learning method that builds strong learners based on weak learners. Gradient boosting is an ensemble of weak learners, usually decision trees. XGBoost (eXtreme Gradient Boosting) is a scalable, fast and efficient gradient boosting variant for tree boosting proposed by Chen et al. [198]. It incorporates many advanced techniques to increase the performance, such as parallelism, cache optimization with better data structure, and out of core computation using block compression and block sharing techniques which is essential to prevent the memory overflow in training large data sets on constrained resource environments. Accordingly, the impact of boosting techniques including XGBoost is evidenced by its dominant adoption in many Kaggle competitions and also in large scale production systems [44, 207, 208].

The XGBoost algorithm is an ensemble of K Classification or Regression Trees (CART) [198]. This can be used for both classification and regression purpose. The model is trained by using an additive strategy. For a dataset with n instances and m features, the ensemble model uses k additive functions to estimate the output. Here, x is a set of input features, $x = \{x_1, x_2, ...x_m\}$ and y is the target prediction variable.

$$\hat{y_i} = \phi\left(x_i\right) = \sum_{k=1}^{K} f_k\left(x_i\right), \quad f_k \in F \tag{5.2}$$

In the Equation 5.2, F is space of all the regression trees, i.e, $F = \{f\left(x\right) = w_{q(x)}\}$, and $\left(q : \mathbb{R}^m \rightarrow T, w \in \mathbb{R}^T\right)$. Here, q is the structure of each tree which maps to corresponding leaf index. T represents the total number of leaves in the tree. Each f_k represents an independent tree with structure q and leaf weights w. To learn the set of functions used in the model, XGBoost minimizes the following regularized objective.

$$\zeta\left(\phi\right) = \sum_{i} l\left(\hat{y_i}, y_i\right) + \sum_{k} \Omega\left(f_k\right),$$

$$where \quad \Omega\left(f\right) = \gamma T = 1\frac{2}{\lambda} \parallel w \parallel^2 \tag{5.3}$$

In Equation 5.3, the first term l is the differentiable convex loss function that cal-

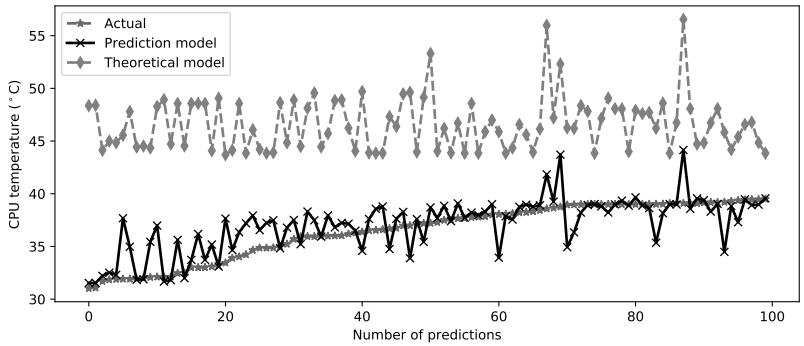


Figure 5.5: Temperature estimation compared to actual values

culates the difference between predicted value $\hat{y}_i$, observed value y_i. Ω penalizes the complexity of the model to control overfitting. Thereby, T is the number of nodes in the tree and w is assigned values for each leaf node of the tree. This regularized objective function attempts to select a model based on simple predictive functions

We use the grid search technique to find the optimal parameters to further enhance the performance of the model. Here, the γ parameter is used to decide the minimum loss reduction required to make a further partition on a leaf node of the tree. Subsample ratio decides the amount of sampling selected from training data to grow the trees. Accordingly, the optimal values for γ are 0.5, the learning rate is 0.1, maximum depth of the tree is 4, minimum child weight is 4, and the subsample ratio is 1, and rest of the parameters are set to default. With these settings, the best RMSE value achieved is 0.05. It is important to note that the prediction based temperature estimation is feasible for any data centre given the historical data collected from the individual data centre.

5.6 Evaluating the Prediction Model with Theoretical Model

To evaluate the feasibility of our temperature prediction models, we compare the prediction results to extensively used theoretical analytical model [34, 35, 184]. Here, the temperature estimation is based on the RC model which is formulated from analytical

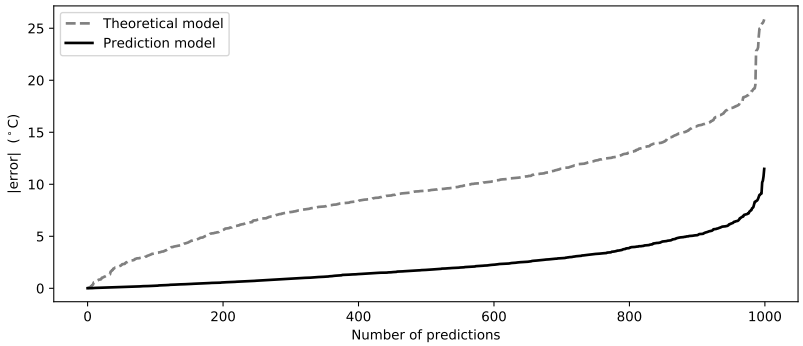


Figure 5.6: Rank order of prediction errors

methods. The temperature of a host (T) is calculated based on the following equation.

$$T = PR + T_{in} + (T_{initial} - PR - T_{in}) \times e^{-\frac{t}{RC}} \tag{5.4}$$

In Equation 5.4, P is the dynamic power of host, R and C are thermal resistance (K/W) and heat capacity (J/K) of the host respectively. $T_{initial}$ is the initial temperature of the CPU. Since analytical models estimate CPU temperature, we also predict CPU temperature to compare the results instead of ambient temperature.

To compare the results, we randomly select 1000 tuples from our whole dataset and analyze the result between prediction and theoretical models. For the theoretical model, the value of P and T_{in} are directly used from our test data set. The value of thermal resistance (R) and heat capacity (C) is set as 0.34 K/W and 340 J/K respectively and $T_{inititial}$ is set to 318 K [184].

The performance of the two models in temperature estimation can be observed in Figure 5.5. For the sake of visibility, Figure 5.5 includes 100 tuples of data. As the figure suggests, our proposed model based on XGBoost's estimation is very close to the actual values, whereas the theoretical model has a large variation from the actual values. Figure 5.6, represents a rank order of the absolute errors (from actual temperature) of two models in °C. The theoretical model deviates as far as 25 °C from the actual values. In this test, the average error of the theoretical model is 9.33 °C and our prediction model is 2.38 °C. These results reflect the feasibility of using prediction models over theoretical models for temperature estimation. It is important to note that, the prediction

models need to be trained for different data centres separately with well-calibrated data that have enough data points to cover all temperature and load conditions in order to predict temperature accurately. Nevertheless, in the absence of such a facility, it is still feasible to use theoretical analytical models that rely on a minimum number of simple parameters.

5.7 Dynamic Scheduling guided by Prediction Models

Applications of temperature prediction are numerous. It can be used to change the cooling settings such as supply air temperature to save the cooling cost [170]. It is also useful in identifying the thermal anomalies which increase the risk of failures and injects performance bottlenecks. Moreover, one foremost usage would be in a data centre resource management system's tasks such as resource provisioning and scheduling.

With the given historical host's data, predictive models are trained and deployed for runtime inference. A scheduling algorithm invokes a deployed prediction model to accurately predict the host temperature. The input to the prediction model is a set of host features. In our model, the features can be easily collected from the host's onboard sensors. These features are accessed from the host's system interface through HTTP APIs. The complexity to retrieve this input feature set information is $O(1)$. The latency of this operation depends on the data centre's local network capabilities. Moreover, the models need to be retrained only when changes are introduced to the data centre environment, like, the addition of new hosts or change in the physical location of hosts. Considering the fact that such changes are not so frequent in a data centre, the cost of building and using such predictive models in resource management tasks like scheduling is highly feasible.

In this regard, we propose dynamic scheduling of VMs in a Cloud data centre based on the temperature prediction model we have proposed. Here, we intend to reduce the peak temperature of the system while consolidating VMs on fewest hosts as possible for each scheduling interval which is a preferred way to reduce the energy in a Cloud data centre [166]. In this problem, n physical hosts in data centre hosting m VMs at timestep t, the objective is to reduce the number of active hosts in a data centre at $t+1$

by consolidating the VMs based on workload level. This consolidation process inside the data centre is critical and carried out regularly to reduce overall data centre energy [167, 209]. This procedure mainly includes three steps. First, identifying under loaded hosts from which we can potentially migrate VMs and shut down the machine. Also finding overloaded hosts and migrate VMs from them to reduce the risk of Service Level Agreements (SLA) violation, here, SLA is providing requested resources to VMs without degrading their performance. Second, selecting VMs for migration from the over-and underloaded hosts identified in previous step, and finally, identifying new target hosts to schedule the selected VMs. The scheduling for consolidation process allows hosts to experience high load and potentially reach the threshold temperature which is useful in evaluating our prediction models effectively. Therefore, the objective of our problem is defined as follows:

$$
\begin{aligned}
\text{minimize} \quad & T^{peak} = \sum_{t=0}^{T} \sum_{j=1}^{m} \sum_{i=1}^{n} \delta_{ji}^{t} T_i^t \\
\text{subject to} \quad & u(h_i) \leq U_{max}, \\
& T_i^t < T_{red}, \\
& \sum_{j=0}^{m} VM_{ji}(R_{cpu}, R_{mem}) \leq h_i(R_{cpu}, R_{mem}), \\
& \delta_{ji}^{t} = \{0, 1\}, \\
& \sum_{i=1}^{n} \delta_{ji}^{t} = 1
\end{aligned}
\tag{5.5}
$$

The objective function in Equation 5.5 minimizes the peak temperature of the hosts while scheduling VMs dynamically in all the time steps $t = \{0, \ldots \quad T\}$. Here, list of VMs that are to be scheduled are represented with the index j where $j = \{1, \ldots \quad m\}$, and list of candidate hosts as i, where $i = \{1, \ldots \quad n\}$. The T_i^t indicates temperature of host i at time t. The constraints ensure that potential thermal and CPU thresholds are not violated due to increased workload allocation. They also assure the capacity constraints, i.e, a host is considered as suitable only if enough resources are available for VM (R_{cpu}, R_{mem}). Here, δ_{ji}^{t} is a binary with the value 1 if the VM_j is allocated to $host_i$ at time interval t, otherwise, 0. The summation of δ_{ji}^{t} is equal to 1, indicating that VM_j is

Algorithm 5 Thermal Aware Dynamic Scheduling to Minimize Peak Temperature

Input: $VMList$- List of VMs to be scheduled
Output: Scheduling Maps

```
1: for t ← 0 to T do
2:   for all vm in VMList do
3:     allocatedHost ← ∅
4:     hostList ← Get list of active hosts
5:     minTemperature ← maxValue
6:     for all host in hostList do
7:       T̂_i ← Predict temperature by invoking prediction model
8:       if (T̂_i < minTemperature) then
9:         minTemperature ← T̂_i
10:        if (T̂_i < T_red and u(h_i) ≤ U_max and vm(R_x) < host(R_x)) then
11:          allocatedHost ← host
12:        end if
13:      end if
14:    end for
15:    if allocatedHost == ∅ then
16:      allocatedHost ← Get a new host from inactive hosts list
17:    end if
18:  end for
19: end for
```

allocated to at most 1 host at time t. The objective function in Equation 5.5 is executed at each scheduling interval to decide the target host for the VMs to be migrated. Finding an optimal solution for the above equation is an NP-hard problem and it is infeasible for on-line dynamic scheduling [162]. To solve the Equation 5.5 optimally, one should know all VMs utilisation level apriori (which affects power and thermal readings) for all the future scheduling time steps. This is impossible in real Cloud workload scenarios where user workload often has stochastic utilisation levels. Accordingly, to achieve the stated objective and provide a near-optimal approximate solution within a reasonable amount of time, we propose a simple Thermal-Aware heuristic Scheduling (TAS) algorithm that minimizes the peak temperature of data centre hosts.

To dynamically consolidate the workloads (VMs) based on current usage level, our proposed greedy heuristic scheduling algorithm is executed for every scheduling interval. The input to the algorithm is a list of VMs that are needed to schedule. These are

identified based on overload and underload condition. To identify overloaded hosts, we use CPU (U_{max}) and temperature threshold (T_{red}) together. In addition, if all the VMs from a host can be migrated to current active hosts, the host is considered as an underloaded host. The VMs that are to be migrated from overloaded hosts are selected based on their minimum migration time, which is the ratio between their memory usage and available bandwidth [166]. The output is scheduling maps representing target hosts for those VMs. For each VM to be migrated (line 2), Algorithm 5 tries to allocate a new target host from the active list. In this process, algorithm initializes necessary objects (lines 3-5) and the prediction model is invoked to predict the accurate temperature of a host (line 7). The VM is allocated to a host that has the lowest temperature among active hosts (lines 8-11). This ensures the reduction of peak temperature in the data centre and also avoids potential hotspots resulting in lower cooling cost. Moreover, this algorithm also assures the constraints listed in Equation 5.5 are met (line 10), so that added workload will not create a potential hotspot by violating threshold temperature (T_{red}). In addition, resource requirements of VM (VM(R_x)) are satisfied, and the CPU utilization threshold is within the limit (U_{max}). If no suitable host is found in the process, a new idle or inactive host is allocated (line 16) from the available resource pool.

Algorithm 5 has a worst-case complexity of $\mathcal{O}(VN)$, which is a polynomial-time complexity. Here, $\mid V \mid$ is the number of VMs to be migrated during a scheduling interval, and $\mid N \mid$ is a number of hosts in a data centre.

5.8 Performance Evaluation

In this section, we evaluate the performance of the proposed algorithm coupled with our prediction model and compare and analyze the results with baseline algorithms.

5.8.1 Experimental Setup

We evaluated the proposed thermal aware dynamic scheduling algorithm through CloudSim toolkit [38]. We extended CloudSim to incorporate the thermal elements and implemented algorithm 5. We used a real-world dataset from Bitbrain [210], which has traces

of resource consumption metrics of business-critical workload hosted on Bitbrain's infrastructure. This data includes logs of over 1000 VMs workloads hosted on two types of machines. We have chosen this data set as it represents real-world Cloud Infrastructure usage patterns and the metrics in this data set are similar to the features we have collected in our data set (Table 5.1). This is useful to construct precise input vectors for prediction models.

The total experiment period is set to 24 hours and the scheduling interval to 10 minutes, which is similar to our data collection interval. Note that, in the algorithm, prediction models are invoked in many places. The prediction is required to identify the host with the lowest temperature, to determine a host overloaded condition, and also to ensure thermal constraints by predicting their future time step temperature.

To depict the experiments in a real-world setting, we model host configurations similar to the hosts in our data centre, i.e., DELL C6320 machines. This machine has an Intel Xeon E5-2600 processor with dual CPUs (32 cores each) and 512 GB RAM. The VMs are configured based on the VM flavours in our research Cloud . We choose four VM types from general flavors, configuration of these VMs are presented in Table 5.4. The number of hosts in the data centre configuration is 75, similar to the number of hosts in our private Cloud collected data, and the number of VMs is set to 750, which is the maximum number possible on these hosts based on their maximum resource requirements. The workload is generated to these VMs according to Bitbrain's dataset.

The CPU threshold (U_{max}) is set to 0.9. According to the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE) [111] guidelines, the safe operable temperature threshold for data centre hosts is in-between 95 to 105 °C. This threshold is a combined value of CPU temperature and inlet temperature together. Accordingly we set temperature threshold (T_{red}) to 105 °C.

The new target machines for VMs to be scheduled are found based on algorithm 5. This requires predicting the temperature of hosts in the data centre. If the $host_i$ temperature is predicted ($\hat{T}_i$) at the beginning of timestep $t+1$ then the input to prediction model is a single vector consisting of a set of features (CPU, P_c, $fs_1 - fs_4$, N_{CPU}, N_{CPUx}, R, R_x, N_{Rx}, N_{Tx}, N_{vm}) representing its resource and usage metrics along with the power and

https://docs.Cloud.unimelb.edu.au/guides/allocations/

Table 5.4: VM Configurations

Name	Core	RAM
VM1 (uom.general.1c4g)	1	4 GB
VM2 (uom.general.2c8g)	2	8 GB
VM3 (uom.general.4c16g)	4	16 GB
VM4 (uom.general.8c32g)	8	32 GB

fan speed measurements. The resource usage metrics are easily gathered from host utilization levels based on its currently hosted VMs' workload level. To estimate the power $\hat{P}_i$, we use SPECpower benchmark [211], which provides accurate power consumption (in watts) for our modeled host (DELL C6320) based on CPU utilization. We estimate fan speeds from simple regression using remaining features to simplify the problem.

We export the trained models as serialized python objects and expose them to our scheduling algorithm by hosting on HTTP Flask application . The CloudSim scheduling entities invoke the prediction model through REST APIs by giving feature vector and host ID as input, the HTTP application returns predicted temperature for the associated host.

5.8.2 Analysis of Results

We compare the results with two baseline algorithms as shown below.

- Round Robin (RR) - This algorithm tries to distribute the workload equally among all hosts by placing VMs on hosts in a circular fashion. The similar constraints are applied as in algorithm 5. We show that the notion of equal distribution of workloads fails to minimize the peak temperature and thermal variations in a data centre.

- GRANITE- This is a thermal-aware VM scheduling algorithm proposed in [181] that minimizes computing and cooling energy holistically. We choose this particular algorithm, because, similar to us, it also addresses the thermal-aware dynamic VM scheduling problem.

http://flask.pocoo.org

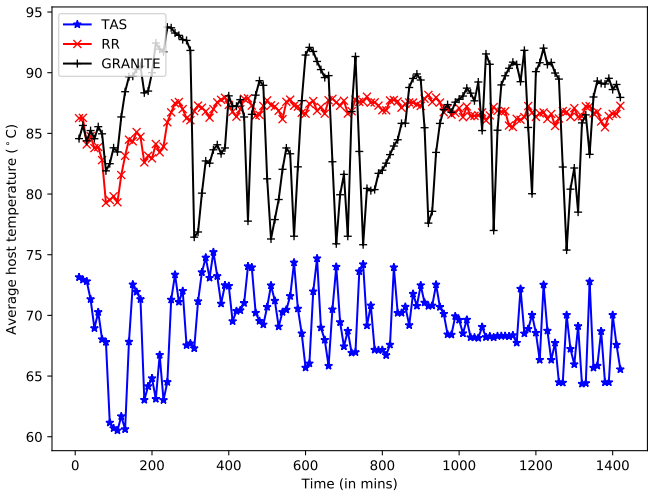


Figure 5.7: Average temperature in each scheduling interval (total experiment time of 24 hours, with scheduling interval of 10 minute)

We use our prediction models to estimate the temperature in both RR and GRANITE algorithms. For GRANITE, the required parameters are set similar to their algorithm in [181] including overload and underload detection methods. The comparison of the average temperature from all hosts in each scheduling interval by all three algorithms is shown in Figure 5.7. Our Thermal-Aware Scheduling (TAS) has the lowest average temperature compared to RR and GRANITE. The RR algorithms' equal workload distribution policy results in less variation in average temperature. However, this will not help to reduce the peak temperature in the data centre irrespective of its intuitive equal distribution behavior as it doesn't consider the thermal behavior of individual hosts and its decisions are completely thermal agnostic. The GRANITE policy has a high average temperature and large variations between scheduling intervals due to its inherent dynamic threshold policies. To further analyze the distribution of temperature due to two scheduling approaches, we draw a histogram with Kernel Density Estimation (KDE) by collecting temperature data from all the hosts in each scheduling interval as shown in Figures 5.8, 5.9, and 5.10. Most of the hosts in the data centre operate around 70 to 80 °C in TAS (Figure 5.8), well below the threshold due to its expected peak temperature minimizing objective. However, the RR approach results in more thermal varia-

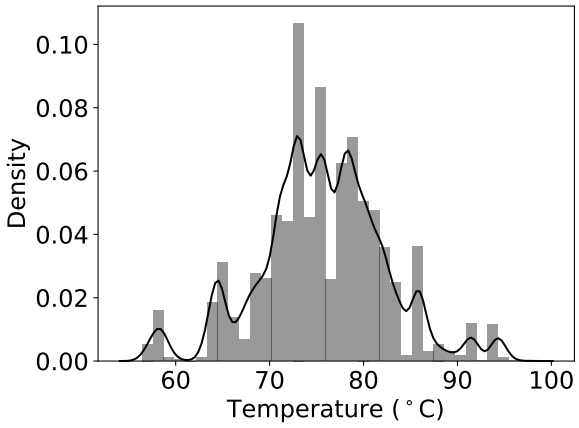


Figure 5.8: TAS

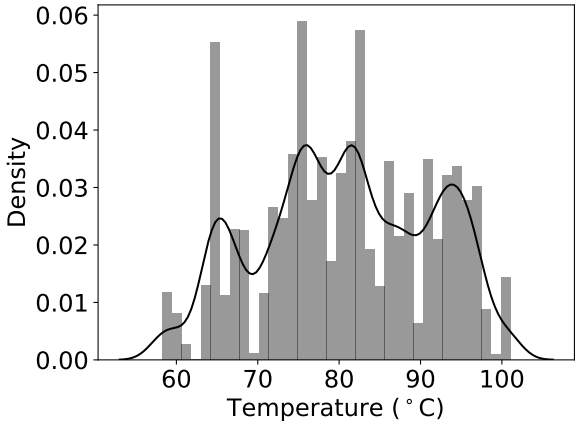


Figure 5.9: RR

tions with sustained high temperatures (Figure 5.9). The GRANITE also has significant distributions around the peak temperature (5.10). This temperature distribution is effectively summarized using the Cumulative Distribution Function (CDF) between three approaches (Figure 5.11). As we can see in Figure 5.11, TAS reaches the probability density value of 1 well below 100 °C, indicating most of the hosts operate in reduced temperature value. RR and GRANITE has a peak temperature of more than 100 °C with high cumulative probability. In addition, as depicted in Figure 5.11, the average and standard deviation of temperature in TAS ($\mu = 75.65$, $\sigma = 6.82$) is lesser compared to the other two approaches ($\mu = 80.69$, $\sigma = 10.49$ for RR and $\mu = 77.36$, $\sigma = 9.34$ for Granite), this is also evidenced by Figure 5.7.

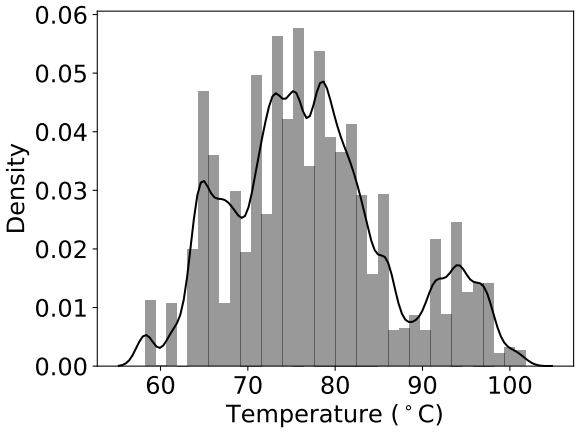


Figure 5.10: GRANITE

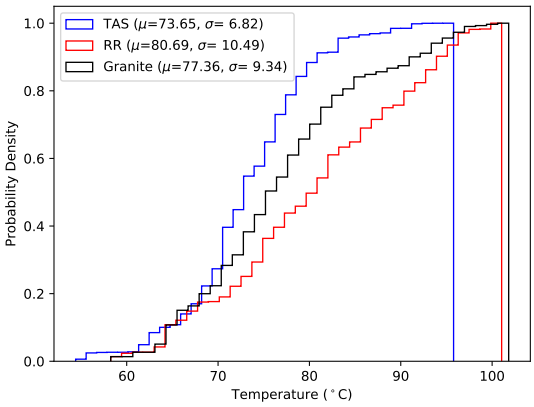


Figure 5.11: CDF between TAS and RR and GRANITE

Further results of the experiments are depicted in Table 5.5. The total energy consumption by TAS, RR, and GRANITE is 172.20 kWh, 391.57 kWh, and 263.20 kWh, respectively (the total energy is a combination of cooling and computing energy calculated as in [192]). Therefore, RR and GRANITE have 56 % and 34.5 % more energy consumption than TAS, respectively. This is because RR and GRANITE distribute workload into more hosts resulting in a high number of active hosts. In this experimented period, RR and GRANITE had 18 and 11 average number of active hosts while the TAS algorithm resulted in 4 active hosts. Furthermore, although RR distributes workload among many hosts, its thermal agnostic nature had a peak temperature of 101.44 °C, GRANITE had peak temperature of 101.80 °C and TAS had attained a maximum of 95.5

Table 5.5: Scheduling results compared with RR and GRANITE algorithm

Algorithm	Peak Temperature (°C)	Total Energy (kWh)	Active Hosts
TAS	95	172.20	4
RR	101.44	391.57	18
GRANITE	101.81	263.20	11

°C during the experimentation period which is 6.5 °C lower than the latter approaches. This demonstrates that the accurate prediction of host temperature with an effective scheduling strategy can reduce the peak temperature and also save a significant amount of energy in the data centre.

5.8.3 Evaluating Performance Overhead

It is important to estimate the overhead of dynamic scheduling caused due to migration and workload consolidation. In the context of scheduling in the Cloud, the expected performance is usually defined using Service Level Agreements (SLAs). In our approach, the scheduling is at a higher VM level, hence, we represent the SLA metrics using the VM level features. In this regard, we consider the following metrics [166, 181]:

Number of VM migrations: Virtual machines may experience degraded performance during migration. Hence, the number of migrations should be minimized to reduce the overhead and avoid SLA violations.

$SLA_{violation}$: Due to oversubscription and consolidation, hosts may reach full utilization level (100%), in such cases, the VMs on such host experiences degraded performance. This is expressed using SLA violation Time per Active Host (SLA_{TAH}) metric as shown in Equation 5.6. Furthermore, the consolidation of VMs comes with performance overhead caused due to live VM migration [189], this Performance Degradation due to Migration (PDM) is defined as in Equation 5.7.

$$SLA_{TAH} = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{max}}{T_{active}} \tag{5.6}$$

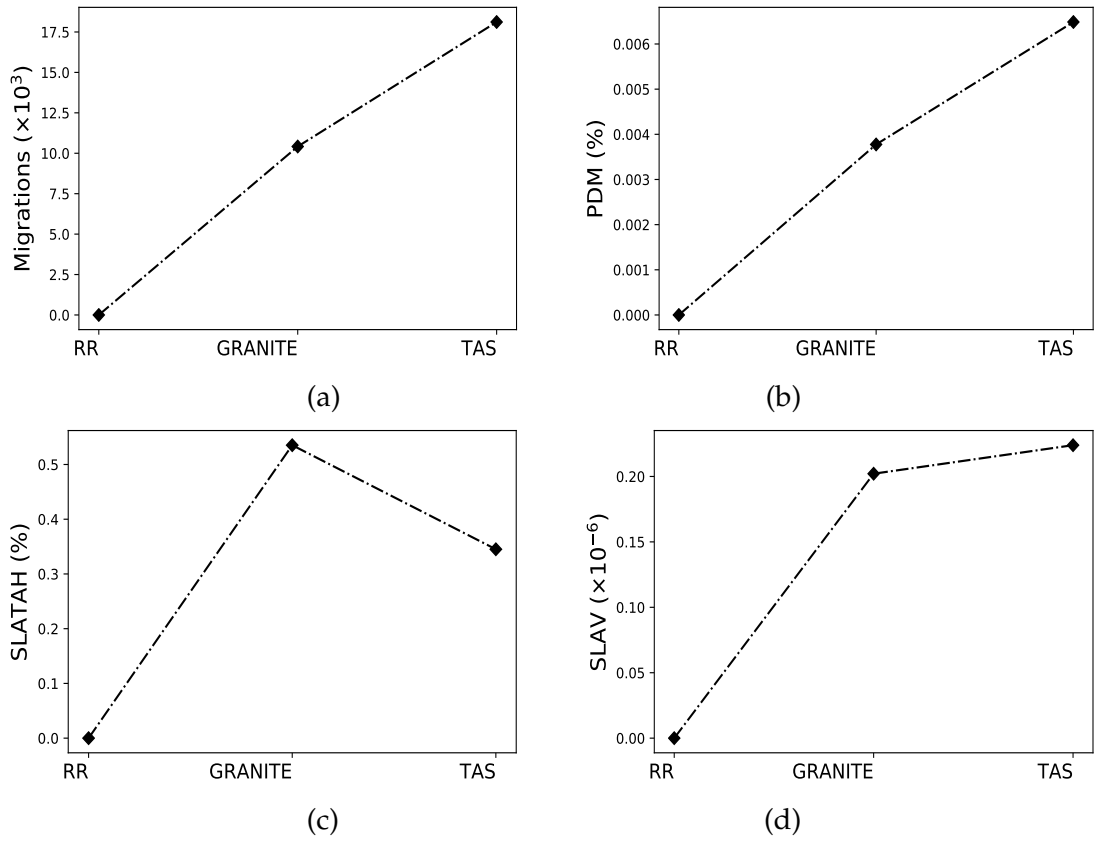


Figure 5.12: Performance Overhead Metrics (a) Number of VM migrations (b) The PDM metric (c) The SLATAH metric (d) The $SLA_{violation}$ metric

$$PDM = \frac{1}{M}\sum_{j=1}^{M}\frac{C_{Aj} - C_{Rj}}{C_{Rj}} \tag{5.7}$$

$$SLA_{violation} = SLA_{TAH} \times PDM \tag{5.8}$$

Here, N is total number of hosts, T_{max} is amount of time $Host_i$ has experienced 100% of utilization and T_{active} is total active time of $Host_i$. M is the total number of VMs. The C_{Aj} is the total amount of CPU capacity allocated and C_{Rj} is the total amount of CPU capacity requested by VM_j while in migration during its lifetime, this captures the under allocation of VMs during live migration. The overall SLA violation of Cloud

infrastructure ($SLA_{violation}$) can be defined by combining both SLA_{TAH} and PDM metrics as shown in Equation 5.8.

The results of overhead metrics for different algorithms are shown in Figure 5.12. As shown in Figure 5.12.a, the number of migrations is 10417 and 18117 for GRANITE and TAS, respectively. The RR has zero migrations. It is expected as RR distributes workload equally among the required number of hosts from the initial step and is not concerned about dynamic optimizations in runtime. For the PDM metric (Figure 5.12.b), GRANITE and TAS have 0.0037 % and 0.0064%, respectively. This is because to TAS has a higher number of migrations compared to GRANITE. As TAS continuously tries to minimize the peak temperature among active hosts based on workload level, it performs aggressive consolidation in each scheduling interval. However, the proactive approach of TAS trying to reduce the host peak of temperature also results in reduced CPU overload of hosts. This is evidenced as the TAS has a lower value of SLA_{TAH} metric (0.34%) compared to the GRANITE (0.53%). Furthermore, for the overall $SLA_{violation}$ metric (Figure 5.12.d), TAS has increased value (0.22×10^{-6}) compared to GRANITE (0.20×10^{-6}). This little increased value is due to the higher PDM value of TAS. However, TAS significantly outperforms both GRANITE and RR in reducing peak temperature and energy efficiency with this negligible overhead.

5.8.4 Dealing with False Predictions

In our scheduling experiments, we observed that a few of the temperature predictions have resulted in some large number which is beyond the boundaries of the expected value. A further close study into such cases has revealed that this happens with particularly three hosts which were almost idle in the data collection period of 3 months having a CPU load less than 1%, which means the models trained for these hosts have limited variations in their feature set. As the trained models did not have any instance close to the instance of prediction, prediction results in an extreme variant value. Such a false prediction in runtime results in an incorrect scheduling decision that affects the normal behavior of the system. In this regard, the scheduling process should consider such adverse edge cases. To tackle this problem, we set minimum and maximum bound

for expected prediction value based on our observations in the dataset. For any prediction beyond these boundaries, we pass the input vector to all remaining hosts' models and take an average of predicted value as a final prediction value. In this way, we try to avoid the bias influenced by a particular host and also get a reasonably good prediction result. In the case of a huge number of hosts, subsets of hosts can be used for this.

This also suggests that, to effectively use the prediction models, the training data should have a distribution of values of all hosts covering all possible ranges of values. Deploying such models in a real-world data centre requires good coverage of data to handle all possible operating points of the data centre so that when ML models are trained they will not be overfitted for a skewed range of data and thus perform poorly.

5.9 Feature Set Analysis

We carried out a feature analysis to identify the importance of each feature towards the model performance. This analysis can also be used in the feature selection process to remove the redundant features, reduce the computational cost, and increase the performance. Figure 5.13 shows the importance of each feature in the constructed XGBoost model. Here, the weight metric associated with each feature corresponds to its respective number of occurrences in the constructed tree which indirectly notifies its importance. Based on the results, host power (P_c), fanspeed1 (fs_1) and number of VMs (N_vm) are the most important features towards accurate prediction. It is important to note that, though we have 4 fan speeds, the model intuitively selects one fan speed with more weight, this is since all four fans operate almost at the same rpm, which is observed in our data set. The least important feature is network metrics (N_{rx}, N_{tx}) along with the remaining three fan speed readings. The crucial observation is that the model gives high importance to power instead of CPU load, indicating, the high correlation between temperature and power. The number of cores (NC) is not included in the tree as it has constant value across hosts introducing no variation in the data.

The performance of temperature prediction with different thresholds can be observed in Figure 5.14. We start with the most important feature and recursively add more features according to their importance to the model. The y axis indicates RMSE

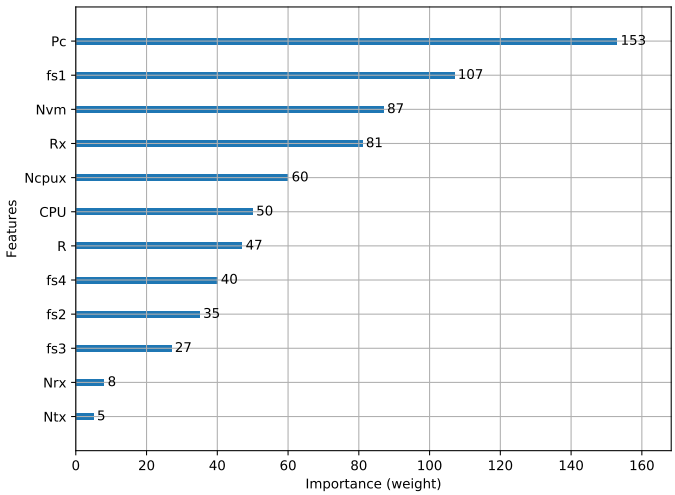


Figure 5.13: Feature importance (weight)- number of times a particular feature occurs in the trees

value and the x axis shows a number of features. The first three features (P_c,fs_1,N_{vm}) significantly contribute to prediction accuracy and the accuracy gain is little as we add more features to the model. Therefore, based on the required accuracy or RMSE value, we can select top n features to effectively train the model with less complexity.

5.9.1 Assumptions and Applicability

The scheduling algorithm and prediction models proposed in this chapter have the following assumptions and applicabilities. The scheduling algorithm is applicable for workloads that run in VMs for a long period without any interruptions (such as web and enterprise applications). Our policy tries to monitor the utilisation level of such workloads and consolidate them at regular intervals for energy efficiency while minimising the data centre's peak temperature. The workload independent performance metrics in section 5.8.3 indirectly captures the overhead of the scheduling algorithm. For other types of workloads such as tasks with predefined completion time, this algorithm is not directly applicable. In addition, the models trained from the particular data centre should only be used in that data centre. This is required to capture the unique

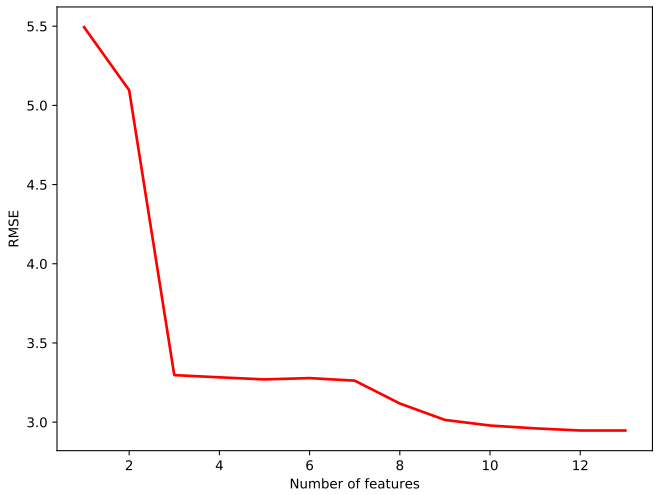


Figure 5.14: Feature threshold analysis

characteristics and configuration of a data centre that influences temperature variations in it. They include data centre physical rack-layout, air circulation pattern, and server heat dissipation rate that directly affects the sensor readings and thus ambient temperature of server [34, 181, 201]. Hence, it is essential to train prediction models with data collected from a individual data centre to capture its characteristics. However, our proposed techniques are still applicable in building such models. Therefore, the scheduling algorithm and prediction models are only suitable for a specific workloads, in a particular data centre.

5.10 Summary

Estimating the temperature in the data centre is a complex and non-trivial problem. Existing approaches for temperature prediction are inaccurate and computationally expensive. Optimal thermal management with accurate temperature prediction can reduce the operational cost of a data centre and increase reliability. Data-driven temperature estimation of hosts in a data centre can give us a more accurate prediction than simple mathematical models as we were able to take into consideration CPU and inlet airflow

temperature variations through measurements. Our study which is based on physical host-level data collected from our University's private Cloud has shown a large thermal variation present between hosts including CPU and inlet temperature. To accurately predict the host temperature, we explored several machine learning algorithms. Based on the results, we found a gradient boosting based XGBoost model for temperature prediction is the best. Our extensive empirical evaluation has achieved high prediction accuracy with the average RMSE value of 0.05. In other words, our prediction model has an average error of 2.38 °C. Compared to an existing theoretical model, it reduces the prediction error of 7 °C.

Guided by these prediction models, we proposed a dynamic scheduling algorithm for Cloud workloads to minimize the peak temperature. The proposed algorithm is able to save up to 34.5% more of energy and reduce up to 6.5 °C of average peak temperature compared to the best baseline algorithm. It is important to note that, though the models built for one data centre are optimized for its own (as each data centre's physical environment and parameters vastly change), the methodology presented in this chapter is generic and can be applied to any Cloud data centre given the sufficient amount of data collected from the respective data centres.

While this chapter presented predictive ML models coupled with a heuristic scheduling algorithm, in the next chapter, we explore building a complete learning-based scheduling model using the RL framework.

Chapter 6

DRL-based Scheduling for Integrated Energy and Thermal Efficiency

Cloud data centres need to be managed both energy and thermally efficient to provide reliable services and reduce their energy consumption. Optimising both computing and cooling systems is challenging due to the complexity of the data centre infrastructure and diverse workload characteristics. This chapter proposes a Thermal and Energy-aware workload scheduling based on Deep Reinforcement Learning (TEDRL) in cloud data centres. We leverage the DRL framework to manage the complexity of data centre infrastructures and workload characteristics and achieve energy and thermal efficiency through its scheduling decisions. We design adequate state space, action space and rewards for our DRL agent. The policy gradient based DRL agent is trained for multi-objective optimisation that reduces the data centre's temperature and minimises its energy consumption. Experiments conducted in simulation using real workload traces and data centre logs have shown that TEDRL outperforms in terms of energy consumption and peak temperature compared to the baseline algorithms.

6.1 Introduction

Thermal and Energy-aware workload management is an extremely challenging task due to conflicting trade-offs between two competing computing and cooling subsystems. . For instance, decreasing power consumption in the computing system would reduce the CPU's temperature dissipation but compromise with the application SLAs since

This chapter is derived from:

- **Shashikant Ilager**, Rajkumar Buyya, "TEDRL: Thermal and Energy-aware Deep Reinforcement Learning approach for Workload Scheduling in Cloud Data Centres", *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, USA, 2021 (in review).

reducing power slows down the performance (CPU frequency/speed) and results in lower resource utilisation. In contrast, higher power consumption of the computing system increases utilisation level, potentially violating the temperature threshold and increasing the cooling cost. Therefore, due to the complex interactions between multiple subsystems and dynamic workloads, the data centre environment exhibits non-linear relationships between different parameters. For instance, a host with a similar state often exhibits non-stationary in their thermal response. Its temperature is affected by heat recirculating, physical position, workload type and many other parameters. Similarly, resource utilisation, power consumption and corresponding temperature response will have a non-linear relationship between them. A simple rule-based solution fails to capture these intricacies in a data centre. Furthermore, most of current the studies in workload management focus on optimising solely energy [177] [209] [82] [74] or temperature aspects [129] [118] [113] [117]. However, these proposed solutions are based on static rules or fine-tuned heuristics which are often inefficient in complex and dynamic environments such as Cloud data centres [44]. Recent advancement in Deep Reinforcement Learning (DRL) has allowed learning-based policies to be applied to workload management of complex data centre systems which are suitable to deal with complexities of data centre systems and workload characteristics and also manage the trade-offs between subsystems [195]. Some studies have applied learning-based solutions for optimising energy through scheduling [200] [203] [36] and cooling [212] aspects in data centres. Very few studies have focused on joint optimisation of energy and thermal aspects of data centres.

In this chapter, we propose a Thermal and Energy-aware workload scheduling based on Deep Reinforcement Learning (TEDRL) technique, to address the above challenges. DRL is a computational approach for goal-oriented learning, and decision-making framework where a single or multiple agents can interact with the environment without requiring direct supervision [46]. It provides efficient state-space exploration methods while optimising the objective. In the general RL framework, an agent actively interacts with the environment and takes decisions to optimise towards achieving a designed goal. Although an agent does not initially know the environment, it continuously learns by partially observing each step. Upon observing an environment state, it takes action

based on designed policy. This action results in an environment to move to a new state and gives back a reward to the agent. RL agent's inherent goal is to take a set of actions that eventually maximise the expected reward, thus optimising the policy and its objective. RL's advancement has seen adoption into many critical domains, including network resource management [213], robotic, and games, among others. In this chapter, we model workload scheduling with a DRL-based approach to identify the best scheduling decision by interacting with the data centre environment. The agents can learn the energy and thermal properties of a data centre and take action to achieve overall thermal and energy efficient workload scheduling in data centres.

In summary, the key contributions of this chapter are:

- We propose energy and thermal aware DRL scheduling framework for Cloud data centres.
- We design a discrete state-action space and reward space for energy and thermal aware scheduling using DRL
- We propose a Deep Q-learning and a policy gradient based scheduling algorithm for DRL environments.
- We perform extensive experiments and evaluate the proposed system with real-world data collected from our University's private data centre and public workload traces.

The rest of the chapter is organised as follows. Section 6.2 highlights several relevant works form literature. Section 6.3 provides motivation for the study. Section 6.4 describes the system model and Section 6.5 explains the problem formulation. The background and preliminaries of RL are given in Section 6.6. The proposed DRL based energy and thermal-aware scheduling framework are described in Section 6.7. Section 6.8 presents the proposed solution's simulation environment and performance evaluation. Finally, Section 6.9 concludes the chapter.

Table 6.1: Related Work Comparisons with Our Work

Work	Algorithm	Approach
Ahmad et al. [23]	Heuristics and analytical model	Power budget shifting where excessive cooling power and idle power are traded between each other to optimise energy usage
Li et al. [21]	Analytical model	Actuating knobs of computing and cooling systems and accordingly adjusting the workload distribution across the rack thus achieving better throughput and peak CPU temperature reduction.
Wan et al. [20]	heuristic based on analytical framework	Minimising energy in both computing and cooling systems through an analytical framework based on heuristic algorithms to configure different knobs of two systems
Lee et al. [22]	GRANITE - A heuristic algorithm	Dynamic scheduling and migration problem to minimise data centres energy holistically by minimising computing power and optimising data centre temperature
Ran et al. [36]	DRL based algorithm	DRL based solution approach using a parameterised action space Deep Q-Network (PADQN) optimising scheduling and cooling parameters
Basu et al. [25]	Megh- Q-learning based algorithm	Live migration Virtual Machines (VMs) improving SLA and energy usage
[24, 26, 33]	DRL based algorithms	Computing system optimisation on cloud and edge environments optimising different metrics including latency and execution time
TEDRL (Our Work)	DRL algorithm (REINFORCE)	Complete learning based workload scheduling by optimising both computing and cooling energy and also reducing peak temperature.

6.2 Related Work

Many researchers have studied joint optimisation of the data centre's computing and cooling subsystems. Different studies have tackled the problem with various resource management techniques, including resource provisioning, scheduling, load balancing, and configuring various knobs of two subsystems [107].

The most popular technique for joint optimisation is configuring the different computing and cooling system parameters to achieve overall energy efficiency. In this regard, Wan et al. [123] studied minimising energy in both computing and cooling systems through an analytical framework. The proposed heuristic algorithms configure different knobs of two systems while managing workloads. The configuration parameters include server frequency, on-board fan speed, CRAC temperature set point. By varying these parameters, it tries to reduce the overall energy consumption of computing and cooling systems. A similar approach has been taken in [124] where by actuating knobs of computing and cooling systems and accordingly adjusting the workload distribution across the rack, it balances throughput and peak CPU temperature requirements.

Some techniques have also explored workload consolidation for joint optimisation. Lee et al. [181] studied dynamic scheduling and migration problem to minimise data centres energy holistically. The proposed algorithm GRANITE is a greedy heuristic that identifies overloaded hosts and migrates workload while also keeping reducing active

machines to minimise computing power.

The authors in [125] explored techniques like power budget shifting, where excessive cooling power and idle power are traded between each other to optimise energy usage. Also, they over-provision the resources to increase the number of active servers so that future load can be accommodated when the computing power budget increases. All of these techniques are based on analytical frameworks or heuristics algorithm tailored for particular problems. As cloud data centres and workload complexity increase, they often fail to perform better and provide accurate decisions.

Many recent studies have explored predictive optimisations [36, 83, 204, 214–218]. These works use different ML (Machine Learning) and DL (Deep Learning) techniques to optimise the data centres through different Resource Management System (RMS) techniques. Deep Neural Networks (DNN) and Deep Reinforcement Learning (DRL) approaches have also been widely used in this regard. In most of these works, optimising energy is a primary objective. Bui et al. [219] studied a predictive optimisation framework for energy efficiency of cloud computing. They predict the system's resource utilisation in the next scheduling period by the Gaussian process regression method. Based on this prediction, they choose a minimum number of servers to be active to reduce the overall system's energy consumption. However, their approach still uses many heuristics in scheduling decisions and hence do not adapt to dynamic Cloud environments or changing workload characteristics.

The RL-based methods have been explored by different studies in data centre optimisation. Zhang et al. [215] proposed a Double Q-Learning (DDQN) based method for energy-efficient edge computing. Initially, they proposed a hybrid dynamic voltage frequency scaling (DVFS) scheduling based on Q-learning. As a deep Q-learning model cannot distinguish the continuous system states, in an extended work [217], they investigated a Double Deep Q-learning (DDQN) model to optimise the solution further. Similarly, Xu et al. [216] proposed LASER, a Deep Neural Network (DNN) approach for speculative execution and replication of critical deadline jobs in the Cloud. They implemented this DNN based scheduling framework for the Hadoop framework. Basu et al. [83] have also investigated the live migration problem of Virtual Machines (VMs) using RL based Q-learning model. The proposed algorithms are aimed to improve over

existing heuristic-based live migration. Live migration is widely used for consolidating the VMs to reduce energy consumption. Their proposed RL model- Megh, continuously adapts and learns system changes to increase energy efficiency. Cheng et al. [36] have studied Deep reinforcement learning-based resource provisioning and task scheduling approach for cloud service providers. Their Q-learning based model is optimised to reduce the electricity price and task rejection rate. In addition, Mao et al.[204] and Li et al.[220] explored Resource Management with DDQN. They apply the DRL to scheduling jobs on multiple resources and analyse the reasons for achieving high gain compared to state-of-the-art heuristics. Rjoub et al. [221] also studied DRL (REINFORCE) based scheduling for edge only environments. They only consider the response time as a metric and do not exploit asynchronous or recurrent networks to optimise model adaptability and robustness. As described before, these Q-learning-based algorithms cannot adapt in stochastic environments such as Cloud data centres quickly. These solutions also solely try to optimise the computing system's energy.

The vast majority of existing joint optimisation solutions heavily rely on heuristics, and very few studies have explored DRL based method for joint optimisation of computing and cooling systems. Authors in [126] propose a DRL-based framework, specifically, a parameterised action space-based Deep Q-Network (PADQN) algorithm that builds an action space and configures different elements of systems to achieve energy efficiency through scheduling. In particular, they adjust the airflow rate of the cooling system. The comparisons with the most relevant works discussed in this section are presented in Table 6.1. In this chapter, we design a DRL based scheduling method for joint optimisation of the cooling and computing system, focusing on minimising peak temperature that directly reduces thermal load and also reduces total data centre energy.

6.3 Motivation

Holistic energy and thermal aware scheduling in a data centre is a complex resource management problem. The temperature behaviour of a host is a Spatio-temporal problem [35]. The server's temperature depends on multiple factors, including the workload level (utilisation), the server's physical location in the data centre, and the heat recircu-

lation effect within the data centre. Rack-layout based data centre usually experiences heat recirculation where heat dissipated by the server affects other servers in the vicinity. Capturing this heat recalculation behaviour is computationally expensive and extremely difficult [34] [170]. In addition, optimising energy and thermal behaviour jointly have conflicting trade-offs. A common approach to optimise computing energy is increasing utilisation, which increases the server temperature, thus requiring more cooling energy. Solving a multi-objective optimisation problem under such complex environments is a difficult task requiring hierarchical solutions from energy and temperature estimation modules to scheduling policies which are often designed based on heuristics. However, a poor or sub-optimal decision in one of the modules often has a cascading effect on the overall results.

Learning-based solutions are highly suitable in dynamic data centre environments [44]. For instance, data centres usually experience burst workloads [209] affecting the sudden energy and temperature rise. Such dynamic changes in the environment should be carefully perceived and handled to provide uninterrupted, reliable services yet safeguarding infrastructure safety. To that end, recent advancements in learning-based optimisation is a promising avenue. In particular, the Reinforcement Learning (RL) approach is suitable to learn the given data centre environment's energy and thermal characteristics provided the agents are trained using the environment's data. RL agents can be designed to perform resource management tasks such as scheduling along with learning the data centre's energy and thermal features. Unlike static heuristics, RL policies can continually improve over time by continuous goal-based learning from the environment and maximising its reward. Moreover, such a learning-based approach in a data centre eliminates complex multi-tier solution approaches often used by existing methods [222].

6.4 System Model

A high-level system model for the proposed TEDRL is shown in Figure 6.1. Here, the RL agent directly interacts with the data centre environment and receives the feedback in terms of new observations from the environment that constitutes a state space in each

Table 6.2: Definition of Symbols used in this Work

Features	Definition
CPU	CPU Load (%)
R	RAM- Random Access Memory (MB)
R_x	RAM in usage (MB)
N_{CPU}	Number of CPU cores
N_{CPUx}	Number of CPU cores in use
N_{Rx}	Network inbound traffic (Kbps)
N_{Tx}	Network outbound traffic (Kbps)
P_c	Power consumed by host (watts)
$P_{cooling}$	Power consumed by cooling system (watts)
T_{amb}	Host ambient temperature (°C)
f_s	Fan speed of host(RPM)
T_{in}	Inlet temperature (°C)
N_{vm}	Number of VMs running on host
n	Number of PMs
m	Number of VMs

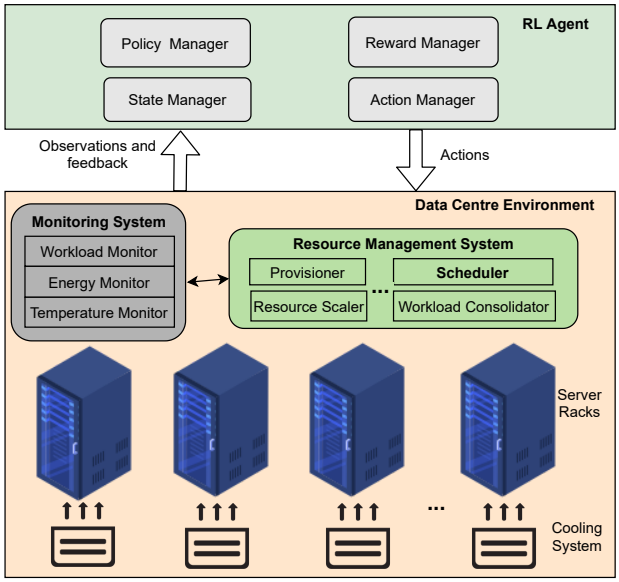


Figure 6.1: A High Level System Model for TEDRL

iteration. It provides actions based on the action space modelling. A Resource Management System (RMS) is responsible for implementing the action onto data centre infrastructure. In a data centre, RMS is usually responsible for managing resources and workloads in run time. It performs different operations such as resource provisioning, scheduling, workload consolidation. However, in this work, we focus on the scheduling problem. The data centre also consists of a Data centre Infrastructure Monitoring (DCIM) system that collects run-time data of workload levels on servers, and various sensor readings, including power and temperature. The RL agent interacts with the monitoring system to receive the information from the environment. RL agent itself has four main components. (1) State Manager- is responsible for managing the state space for the RL agent, representing the environment's current status. The state is nothing but the current values of different environment's parameters at time step t in the environment, which abides the Markov property [46]. (2) Policy Manager- RL agent is trained to optimise certain policy. By nature, it tries to increase the reward it receives, which can be transcribed into different objectives such as energy minimisation. (3) Reward Manager- is responsible for providing a reward for the agent's action based on design. Finally, (4) Action Manager- An agent in the RL framework is designed to provide di-

rectly performed action on the environment. Action can be a set of discrete steps or a continuous probability distribution over action space. In essence, the RL agent helps the resource management system take its decisions more accurately and achieve the desired objective.

6.4.1 Workload Model

Our workload model considers scheduling a VM or set of VMs in data centres. Public Clouds usually receive numerous requests to create new VMs from users. We represent set of VMs as $VM = \{VM1, VM2, VM3.., VMn\}$. Our workload model is independent of applications that run inside the VM. This is also feasible as Cloud users do not expose their isolated virtual environments for service providers, and resource management systems should rely on VM level metrics for their decisions. Here, each VM has various resource requirements including, CPU cores (N_{CPU}), memory size (R) network Input (N_{Rx}) and network Output (N_{Rx}) . This resource requirement of each VM is represented as tuple $\{N_{CPU}, R, N_{Rx}, N_{Tx}\}$.

6.5 Problem Formulation

In this chapter, we target scheduling in Infrastructure as a Service (IaaS) cloud services. In IaaS Cloud, requests for computing resources are provisioned as a number of isolated Virtual Machines (VMs). Service providers need to identify a suitable physical machine in a data centre and place the VM on it. While scheduling VM, one can optimise different parameters based on users and service providers requirements. Some Cloud service provider optimises resource utilisation [223], energy consumption [82] [74] and user's VM-affinities [224]. Scheduling decisions also affect the Service Level Agreements (SLAs) or Quality of Service (QoS) requirements of applications. Suppose a VM is placed onto a physical machine that regularly experiences higher utilisation or sudden bursts of load due to adjacent VM's workload. In that case, the newly placed VM observes degraded performance. Hence, it is necessary to take multiple factors into account while scheduling the VM.

Here, we aim to optimise VM scheduling for energy and thermal efficiency. Energy consumption of a data centre is mainly due to two computing and cooling systems. Computing energy is the total energy consumed by all the servers in the data centre, while cooling energy is energy spent to dissipate heat from the data centre. It is important to note that this cooling energy is specific to a data centre and independent of an individual server's cooling energy by operating their on-board fans and takes out heat from CPU to ambient environment. Thus, total energy consumption in a data centre is defined as:

$$P_{total} = P_c + P_{cooling} \tag{6.1}$$

In the above Equation 6.1, P_c is energy consumption of computing system and $P_{cooling}$ is energy consumption of cooling system. The computing system's energy is a combination of energy consumed by all the servers in a data centre, and it is defined as:

$$P_c = \sum_{i=1}^{n} P_c^i \tag{6.2}$$

The cooling energy in the data centre is energy spent by cooling infrastructures to take out heat from a data centre. Although there are multiple cooling technologies, Computer Room Air Condition (CRAC) is the most common cooling system in Cloud data centres. The energy consumption of the CRAC system [34] is defined as:

$$P_{cooling} = \frac{P_c}{CoP(T_{sup})} \tag{6.3}$$

In Equation 6.3, $CoP(T_{inlet})$ is Coefficient of Performance (CoP) indicates the efficiency of CRAC system. It is a function of cold air supply temperature [170] (T_{sup}), the ratio of total power consumed by the computing system to the total power consumed by the cooling system. Given data centre infrastructure, it can be calculated based on regression techniques. In this chapter, we use following coefficient based regression $CoP(T_{sup}) = 0.0068T_{sup}^2 + 0.0008T_{sup} + 0.458$ [170] . It indicates the ratio of total power consumed by the computing system to the cooling system's.

The data centre's peak temperature determines the cooling energy cost and affects

the system's reliability. Higher the peak temperature, the cooling system needs to pass lower supply air temperature that requires more energy. Peak temperature in a data centre is defined as:

$$T_{peak} = \max_{1 \leq i \leq n} T^i \tag{6.4}$$

VMs need to be scheduled as and when the user requests for a new VM to be created. Accordingly, a service provider can trigger scheduling mechanisms in their data centre. To accommodate a large number of requests, VMs are often scheduled in scheduling interval [181]. In each scheduling interval, a set of VMs is scheduled that arrive during that period and given a set of VMs with their workload level and PMs with their corresponding temperature response and power consumption. The aim is to find an optimal balance in the PM workload level so that overall data centre energy is minimised and the peak temperature is reduced. This problem necessitates online scheduling of VMs to avoid underutilisation of PMs and over utilisation to reduce the peak temperature in hosts. While Service Level Agreements are the common metric used to quantify the tolerable high utilisation level, we focus solely on temperature reduction that would eventually yield better SLA metrics [192] by maintaining an acceptable utilisation rate. Hence we define energy and thermal efficient scheduling objective as below:

$$\begin{aligned} \text{minimize} \quad & \forall_{t=0}^{T} \sum_{j=1}^{m} \sum_{i=1}^{n} \delta_{ji}^{t} (P_{total} + T_{peak}^{i}) \\ \text{subject to} \quad & \delta_{ji}^{t} = \{0, 1\}, \\ & \sum_{i=1}^{n} \delta_{ji}^{t} = 1 \end{aligned} \tag{6.5}$$

In the above Equation 6.5, at each scheduling interval t, we try to place all the m VMs from scheduling queue onto a host from n available machines in data centres. The δ_{ji}^{t} is a binary variable, its values is 1 when j^{th} VM is placed in i^{th} host, otherwise zero. Thus, this constraint ensure a VM is assigned to exactly one physical machine in all the time steps t. The overall aim is to reduce total energy consumption (P_{total}) and also peak temperature (T_{peak}) of a data centre.

Solving the above objective function is an NP-hard problem, which can be translated to constrained global optimisation problem. The existing solutions usually rely on

heuristics to find a reasonable solution within a time [35, 181]. However, in this chapter, we translate this into RL-based learning, where agents would try to achieve the objective by actively learning the environment and providing scheduling decisions accordingly.

6.6 Background and Preliminaries of Deep Reinforcement Learning

Reinforcement learning is a framework for goal-oriented learning and decision making. Here, agents actively interact with the environment without requiring direct supervision. The environment is formally defined by the Markov Decision Process (MDP), a discrete-time stochastic control process [46]. This process consists of four components. First, set of states - at each time step (t), agent observes the environment which is translated as a state s, from state space $S = \{s_1, s_2, ..., s_n\}$. Second, set of actions- agent chooses an action a from set of possible actions from action space $A = a_1, a_2, ..., a_r$ representing it's decision. Third- transition probability T, each agent's action moves state s to to new state s' in the next time step $(t + 1)$. Each combination of state s and action a and new state s' has a transition probability $T(s'|s, a)$. Finally, reward $R(s, a)$, given for action a from the set of actions A. An RL agent aims to maximise the expected reward it receives, thus optimising the long term desired objective. Increasing its reward in each step, it learns deterministic stationary policy π, which maps each state to the corresponding action. Its expected future reward is maximised from time step t. This state-dependent function of policy π is defined as:

$$V^{\pi}(s) = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\right\} \tag{6.6}$$

In Equation 6.6, $\gamma \in [0, 1]$ is called as a discount factor and used to tune the value for immediate and log-term future rewards. For a given state and action pair (s, a), a value that stores expected return i.e., reward by following policy π is represented as Q-function, $Q^{\pi}(s, a)$. The optimal Q value (Q^*) can be computed by solving Bellman equation.

$$Q^*(s,a) = R(s,a) + \gamma \sum_{s'} T(s'|s,a) \max_{a'} Q^*(s',a') \tag{6.7}$$

However, in environments where complete MDP has no known priory, a policy can be learnt from the environment by interacting with it instead of computing optimal policy. To that end, Q-learning is a widely adopted technique that learns the set of actions to receive maximal expected by estimating Q^* by sampling.

$$\hat{Q}(s,a) \leftarrow (1-\alpha_t)\hat{Q}(s,a) + \alpha_t(r + \gamma \max_{a'} \hat{Q}(s',a')) \tag{6.8}$$

In Equation 6.8, $\hat{Q}(s,a)$ is set of all possible Q-value vectors for all possible (s,a) combinations. α is the learning rate, and γ is the discount value, and r is the immediate reward. Learning rate can be used to fine-tune the ability to obtain new information. If α = 0, the agent does not store any new value obtained at current step, when $\alpha = 1$, it stores new value for a given (s,a) and overwrites previous reward values. Setting value of α between 0 and 1 helps strike a balance between the new values and the old ones. The Q-Learning algorithm is the model-free algorithm as it does not use transition probability distribution as in Equation 6.7. This approach is useful when state-space is large, and it is computationally expensive to obtain the probability of all distributions.

The Q learning stores all possible state action combinations and corresponding reward into a vector, called Q-table. However, in the environments with a large number of (s, a) combination, the size of the table increases exponentially, making it infeasible in size and time complexity when updates are needed to be done to the table. Deep Reinforcement Learning (DRL) is an RL approach based on deep learning networks used as a function approximator for expected rewards. Neural networks are prominent function approximators, and they can be trained to output the expected value for given action on the environments.

6.7 TEDRL: Thermal and Energy-aware DRL-based Scheduling

The environment in RL should abide by Markov property, i.e., we should be able to predict the next state and expected reward solely based on current state and action and not

depending on any previous states or actions. Therefore, constructing an environment with an appropriate state abiding by this property is essential. In addition to proper environment design, the DRL framework also needs accurate state, action and reward design to learn and take necessary actions efficiently. Accordingly, this section describes the design of our TEDRL environment, its state space, action space, and reward structure.

Environment As it is impractical to train and build RL agents in real Infrastructure due to their massive interactions with the environment during the training of agents, we built our RL environment using Tensorflow agents (TFAgents) [225]. This python based environment simulates the VMs, PMs and loads the workload from the real world traces in intervals. We mainly use the BitBrain's data set [210] with resource usage statistics of VMs collected for an extended period. The Hosts are modelled according to the data collected from a private Cloud [226].

State Space: Our intention is to reduce the data centre power (see Equation 6.1), and also minimize the peak temperature in data centre, in that regard, we constitute state space representing the physical machine, its workload, energy and temperature attributes. Our state space includes set of Physical Machine (PM) features $\{Id, CPU, R, R_x, N_{CPU}, N_{CPUx}, N_{Rx}, N_{Tx}, N_{vm}, P_c, fs, T_{amb}\}$ and Virtual Machine (VM) features $\{N_{CPU}, CPU, R, R_x, N_{Rx}, N_{Tx}\}$. These features represent the utilisation level of host and virtual machine along with power consumption and temperature dissipation values of physical machine. These features are accessible in data centre from Data Centre Infrastructure Monitoring (DCIM) systems. One can get these values using IPMI interface and monitoring tools of data centre. We represent features of PM as $F_{PM} = \{f_1^{PM}, f_2^{PM}, ...f_p^{PM}\}$ and features of VM as $F_{VM} = \{f_1^{vm}, f_2^{vm}, ...f_q^{vm}\}$. Hence, the total state space is large vector that includes above mentioned observation from all the PMs and VMs. i.e., $N \times F_{PM} \cup M \times F_{VM}$ with n PMs and m VMs in data centre.

Action Space: The major decision in our TEDRL environment is to find a suitable physical machine for the VM in our scheduling process. In that regard, for a given state, the action should output a PM's id so that VM in que can be placed into that machine. This decision should account for and ensure that the desired energy and thermal objective is achieved in the long term. To achieve this, action space includes discrete values,

where each value represents the PM id. For simplicity, we represent PM id as a vector of $[1, 2, 3...n]$. Hence, our set of action values is a vector of $[1..n]$, indicating chosen PM for a particular VM in scheduling.

reward : One of the RL framework's important aspect is to design the reward function such that agents should take decisions that helps to move them towards their goal or objective we want to achieve. By default, an agent wants to maximise the reward it receives. Since our intention is multi-objective, i.e., reducing peak temperature and data centre energy, we design our reward to capture both of these parameters. Assuming the peak temperature of a data centre is T^{peak} at time step t. For a given state and action, then we assign temperature reward as:

$$R_T = -T^{peak} \tag{6.9}$$

In Equation 6.9, temperature reward is a simple scalar value; it effectively indicates, if the lower the temperature value, the higher will be its reward. Suppose an agent's action results in a higher temperature. It gets a higher negative value, showing it received a low reward for its action, which is opposite to the objective we intend to achieve. It is important to note that each host has different temperature behaviour under the same workload conditions (as discussed in Section 3). Hence, an agent should learn to identify the best suitable machine in the given state from learning through multiple iterations.

To decrease the total data centre power, assuming the data centre power is defined as P_{total}, then we define power reward (R_P) as:

$$R_p = -P_{total} \tag{6.10}$$

Similar to time reward, if an action results in higher energy consumption, it receives a less reward (high negative value) and vice versa.

Hence the total reward is defined as:

$$R_{total} = R_T + R_P \tag{6.11}$$

The reward function defined in Equation 6.11 is an episodic reward. It tries to achieve

energy and thermal efficiency in scheduling and learns the optimal policies after many iterations.

TEDRL Agent: Traditionally, RL problems are solved using the Q-learning technique using methods like dynamic programming. However, Q-table size exponentially grows when state and action space size becomes large, hindering the performance. Hence, to avoid the curse of dimensionality associated with state and action spaces, instead of estimating Q value for each possible state and action pair, neural networks can be estimated as function approximators. Moreover, deep learning has enabled RL problems to be designed using deep neural networks to estimate these Q value. Such RL methods are known as Deep Q Learning (DQN) that effectively approximate Q value instead of storing it in a separate table. Similarly, Policy Gradient (PG) method is another popular DRL agent that aims to directly optimise the policy instead of separately estimating Q Values with value functions. It uses neural networks and models action probabilities. When an agent interacts with the environment, it observes or generates $(< s, a, r, \hat{s} >)$ values, it then updates weights of the neural network (θ) such that it maximises the expected return, allowing better actions are likely chosen in the future steps. In our work, we use policy gradient based REINFORCE algorithm as our DRL agent [227].

6.7.1 Energy and Thermal-aware DRL-based scheduler

The overview of the scheduling algorithm is presented in Algorithm 6. This algorithm logic is embedded inside our designed DRL environment. The scheduler's input consists of $VMLIST$ and $HOSTLIST$ and designed state and action spaces for the environment. The run-time state is generated based on the current status of the data centre environment. In each step, a VM that needs to be scheduled is taken from scheduling Que. The goal is to find a corresponding host for this VM. Our DRL agent's action provides $HOST_{id}$ on which this VM is to be scheduled. Once the VM s is placed onto that host, relevant usage metrics, power, and temperature readings are updated. In addition, if the host temperature is more than the current peak temperature of the data centre, then peak temperature is updated accordingly (which later affects reward for the agent). Once all the VMs in que are scheduled, the episode is considered to be finished. Then,

Algorithm 6 TEDRL: ENERGY AND THERMAL-AWARE DRL-BASED SCHEDULER

Inputs: $VMLIST$, $HOSTLIST$, (S, A)
Output: SCHEDULING MAPS
for each VM in $VMLIST.scheduleQue$ **do**
  $Host_{id} \leftarrow$ getDRLAgentAction()
  allocatedHost $\leftarrow HOSTLIST.Host_{id}$
  /*Update resource usage metrics */
  allocatedHost.Temperature $\leftarrow$ getHostTemperature()
  allocatedHost.Power $\leftarrow$ getHostPower()
  if allocatedHost.Temperature $> T^{peak}$ **then**
    $T^{peak} \leftarrow$ allocatedHost.Temperature
  end if
end for
if episodeEnd == True **then**
  $R_{total} \leftarrow$ getReward()
  /* Reward is calculated based on Equation 6.11 */
  Update reward, sate to DRL agent and transition to next episode
end if
End

the DRL agent receives a reward based on the performance of its actions. If its scheduling decisions have resulted in lower peak temperature and energy consumption, it receives higher reward and vice versa. In the training phase, the DRL agent learns the scheduling policy by identifying the hosts and workload characteristics from iterating over many episodes and optimising its decisions.

6.8 Performance Evaluation

In this section, we evaluate our proposed approach and compared it with Round Robin and GRANITE [181] approaches through simulation experiments.

6.8.1 Experimental Setup

We implement the proposed TEDRL in a simulation environment based on realistic data sets and workload traces. We used a real-world dataset from Bitbrain [210]. It contains resource usage metrics of VMs from business-critical applications hosted on Bitbrain's

data centre infrastructure. It includes logs of over a thousand VMs on two types of physical machines. We use physical machine data collected from our Universities' private Cloud data centre to represent the data centre environment. This data has resource usage and corresponding power and thermal sensor data. More information about this data can be found in [226]. The Bitbrain's workload traces representing VMs utilisation and our data centre's data set representing physical machines and environments (temperature sensor data) helps to model realistic data centre environments. This is crucial in generating accurate state spaces in RL environments.

The total experiment period is set to 24 hours and the scheduling interval to 10 minutes, which is similar to our data collection interval. The data centre entities are configured as follows. The physical hosts are configured similar to the hosts in our data centre, i.e., DELL C6320 machines. These hosts have an Intel Xeon E5-2600 processor with dual CPUs (32 cores each) and primary memory of 512 GB RAM. Similarly, VMs are configured according to Bitbrains dataset's resource subscriptions. The number of VMs are around 750 based equally to Bitbain's dataset (fast storage). The number of hosts in the data centre configuration is 75, similar to the hosts' number in our private Cloud collected data. The workload is generated to these VMs based on Bitbrain's traces. The scheduling Que is populated based on Gaussian distribution across all scheduling intervals.

- Round Robin (RR) - It tries to distribute the workload across data centres hosts equally. Although this simple heuristic tries to minimise peak temperature, we show that its thermal agnostic nature often violates the threshold and harms energy consumption.

- GRANITE- It is an energy and thermal-aware heuristic scheduling algorithm proposed in [181], which tries to reduce computing and cooling energy together.

Based on the empirical evaluation, TEDRL REINFORCE agent's parameters are configured as follows. The batch size is set to 128, and the learning rate is set to 0.001, the discount factor is set to 0.9, the replay buffer size is configured as 10000. We have chosen "Adam" optimiser, and finally, the number of iterations is set to 25000.

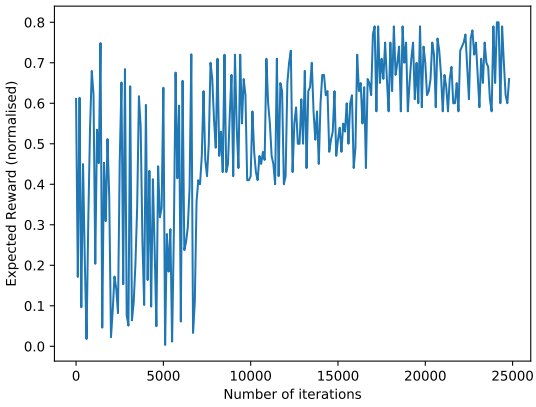


Figure 6.2: Reward Convergence for TEDRL

6.8.2 DRL Environment Implementation and State Space Generation

We implement the python-based RL environment using TF-Agents [225]. RL environments' important aspect is to generate accurate state spaces in run-time to depict the real-time environments, in our case, Cloud data centres. In our state space, physical machine and virtual machine usage metrics are accessed based on workload traces. The Bitbrain's workload traces contain utilisation value of different resource types; when multiple VMs are placed onto a PM, aggregated VM utilisation at time step t represents the PM utilisation. To estimate the power consumption of a PM, we use the SPEC benchmark [211] for our configured server. The cooling power is calculated based on lumped RC model as defined in Equation 6.3. The computing (P_C) and cooling system ($P_{cooling}$) together represent total energy (P_{total}) as defined in 6.1. The server's temperature is predicted based on XGboost based machine learning model built for our data centre characteristics, as proposed in [128]. Finally, fan speeds are based on simple linear regression models built using the same data. Hence, this realistic generation of state parameter values helps RL models to learn the complexities of data centre environments accurately.

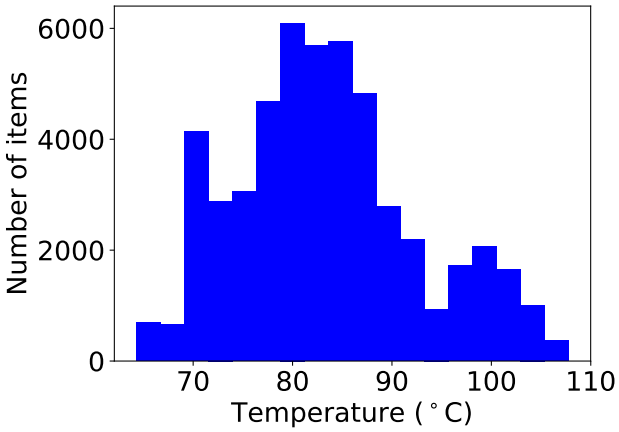


Figure 6.3: RR Temperature Histogram

6.8.3 Analysis of Results

The main objective of TEDRL is to reduce total energy consumption and the peak temperature of the data centre while scheduling the workloads in a data centre. The performance of our REINFORCE based DRL agent can be observed in Figure 6.2. This figure includes sampled points of normalised reward received among all iterations. The agent converges after 20000 iterations and receives almost similar reward afterwards with minor variance.

The performance of TEDRL in peak temperature minimisation can be observed in Figure 6.5. This histogram represents hosts experiencing different temperatures in all scheduling intervals. Most of the data centre hosts operate around 70-80 °C using TEDRL scheduling policy, much below the red-line threshold of 105 °C. This shows that the DRL agent learns the workload and data centre characteristics and places VMs to reduce its peak temperature.

The thermal efficiency of the RR policy can be seen in Figure 6.3, which demonstrates that RR policy has few hosts reaching peak temperature. Although RR tries to distribute workloads across all machines in a circular fashion, its thermal agnostic nature may choose some machines experiencing overusage. In contrast, TEDRL can dynamically observe hosts' utilisation and temperature status before deciding scheduling on it, thus mitigating potential peak temperature violations. It is important to note that many hosts also reside in a lower temperature zone in RR. Nevertheless, it does account for higher

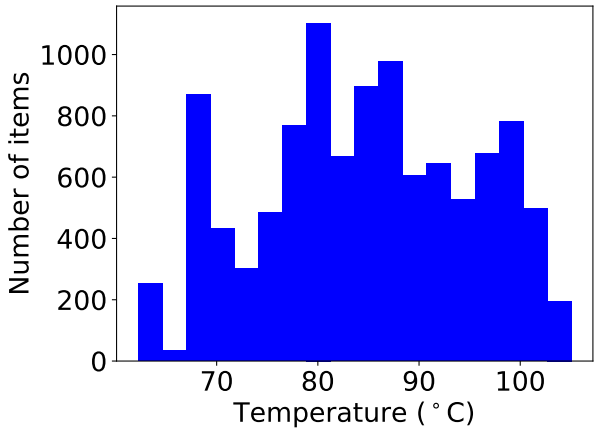


Figure 6.4: GRANITE Temperature Histogram

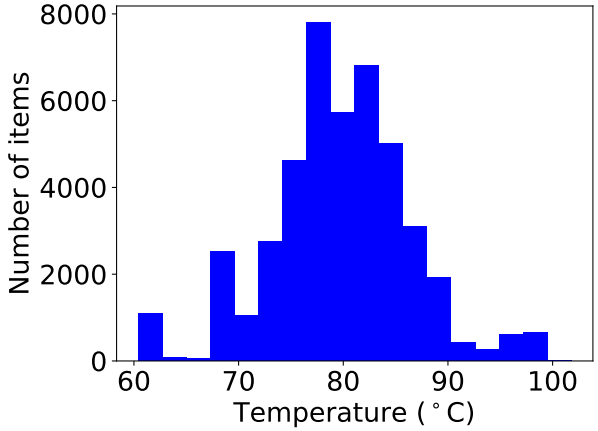


Figure 6.5: TEDRL Temperature Histogram

energy consumption and lower resource utilisation. An acceptable policy should increase resource utilisation and avoid peak temperature, requiring hosts to run near to thresholds, which our TEDRL policy does.

Similarly, temperature management of GRANITE in scheduling can be observed in Figure 6.4. GRANITE tries to identify over utilised machines (among the top 10% of machines in a data centre) and migrate VMs from them to other machines to reduce peak temperature. This thermal aware nature avoids a massive number of peak temperature violations, as seen in Figure 6.4. Compared to RR, it achieves better temperature distribution. However, its reactive nature and failure to observe complex workload dynamics

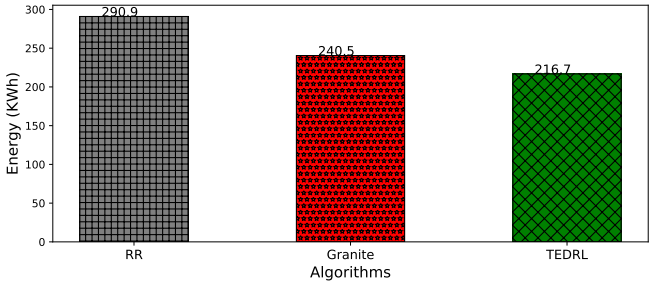


Figure 6.6: Energy Consumption

make some hosts exceed peak temperature. Compared to RR, fewer machines are in a low-temperature zone due to its holistic energy minimisation formulation that ensures fewer machines in data entre are underutilised.

Along with optimising thermal behaviour through workload scheduling, TEDRL also optimise data centre energy consumption. Energy inefficiencies can be eliminated in a data centre by avoiding resource underutilisation and better temperature management. This requires increasing host utilisaiton while satisfying thermal constraints and SLA requirements. In that regard, TEDRL consumes the lowest energy compared to the other two approaches, as shown in Figure 6.6. RR consumes the highest amount of energy since many of its hosts are underutilised and keep many active machines to accommodate workloads. Similarly, the GRANITE policy performs slightly better than RR and consumes less energy. However, TEDRL's multi-objective policy and DRL agent's scheduling decision results in the lowest energy consumption, i.e., 11.09% and 13.40% compared to RR and GRANITE policy, respectively.

In summary, our proposed TEDRL method achieves both energy and thermal efficiency. It learns the data centre and workload complexities and accordingly manages the workloads.

6.9 Summary

AThe massive energy consumption of data centre is a limiting factor for the sustainable growth of cloud data centres. Computing and cooling are two main subsystems of data centres that contribute to major energy consumption. Therefore, joint optimisation of

both these systems is necessary to reduce a significant amount of energy consumption. In this chapter, we proposed an energy and thermally efficient scheduling method using the DRL technique called TEDRL. The proposed solution decreases peak temperature in a data centre and simultaneously reduces energy consumption by managing workloads efficiently. Unlike existing static heuristics-based solutions, TDERL learns data centre energy and thermal characteristics using realistic temperature and energy models and converges to an optimal scheduling policy. To demonstrate the feasibility of our proposed TEDRL policy, we conducted experiments in a simulated environment with TFAgents using real workload traces and data sets and compared its performance with baseline scheduling policies. The experimental results show improvements in reducing the peak temperature of the data centre and reducing total energy consumption. Moreover, it demonstrates the feasibility of using learning-based methods in the data centre's resource management systems.

Chapter 7
Conclusions and Future Directions

This chapter concludes the thesis and discusses a summary of works and key contributions. It also highlights several future research directions in resource management of Cloud and emerging distributed systems.

7.1 Summary and Conclusions

Cloud computing platforms offer on-demand and flexible access to elastic resources, enabling highly connected resource-intensive business, scientific, and personal applications. Cloud computing demand has accelerated growth in data centres that are distributed, large scale, and heterogeneous. Managing resources energy efficiently in such infrastructure is essential to achieve sustainability in Cloud computing. Furthermore, providing reliable services to application users by meeting their SLA requirements is also necessary. The state-of-the-art rule-based or heuristics based Resource Management Systems (RMS) solutions have become inadequate in modern Cloud systems. The RMS policies need to deal with massive scale, heterogeneity, varying workload requirements, and data centre resources complexity. To that end, machine-learning-based techniques and tools can be widely utilised in numerous RMS tasks, including monitoring, resource provisioning, scheduling, and many others. Such approaches are highly adaptive and better suited to deal with the resource management complexities, enabling optimised resource management from processor to middleware platforms and application management. In this thesis, we investigated various resource management techniques to achieve energy efficiency across computing and cooling systems of data centre and satisfy SLA requirements using ML techniques.

Chapter 1 presented the concept of Cloud computing and highlighted Cloud data centres' energy consumption problem and its challenges. It also described motivations for this thesis, outlined the research questions addressed in this thesis, and presented the research methodology adopted in this thesis.

Chapter 2 investigated the existing resource management techniques in energy and thermal efficient management of Cloud data centres and presented a taxonomy of different methods. Finally, a conceptual AI-centric resource management system has been presented for future distributed systems, including Cloud computing.

Chapter 3 investigated data-driven frequency scaling and deadline aware application scheduling in GPUs. Dynamic voltage and frequency scaling (DVFS) is a popular technique to reduce dynamic power in computing elements. However, different workload exhibits different energy consumption behaviour using DVFS. Hence, it is essential to identify the energy-efficient frequency for application and configure it accordingly to reduce energy consumption. Proposed ML prediction models estimate execution time and energy consumption for different frequency ranges, which helps RMS to configure the processing elements optimally. Moreover, such techniques are highly feasible in application scheduling when an application with varied QoS requirements need to be executed. Scheduling algorithm guided prediction model performs frequency scaling in runtime based on application deadline requirements to achieve energy efficiency.

Chapter 4 presented energy and thermal aware dynamic Virtual Machine (VM) consolidation technique. The workload consolidation technique is employed in the data centre to reduce computing system energy. However, aggressive consolidation creates local hotspots, thus increasing cooling energy cost and affecting system reliability. Our solution considers both computing and cooling energy optimisation in consolidation. We formulate the consolidation problem as an integrated energy minimisation problem and propose a GRASP metaheuristic algorithm to solve it. Our policy performs consolidation by proactively mitigating the hotspots with negligible SLA violations. It identifies efficient consolidation level and minimises computing and cooling systems' energy consumptions.

Chapter 5 built thermal prediction models and proposed scheduling algorithm to minimise peak temperature in Cloud data centre environments. Existing Computational

fluid dynamics (CFD) and analytical models are either computationally expensive or inaccurate. Hence, we proposed data-driven prediction models for this. In order to build prediction models, we collected data centre data from monitoring infrastructure for a long time containing the different sensor and physical machine-level data. Prediction models built using this data have shown that they can accurately predict ambient server temperature with fast inference capabilities in runtime. Furthermore, a workload scheduling algorithm is proposed, which takes the assistance of these prediction models. It minimises peak temperature across the data centre and helps the cooling system energy requirements. It also ensures that computing systems' energy is minimised by avoiding underutilisation of resources and guarantees SLAs for workloads. Therefore, proposed methods help accurate and fast thermal modelling in a data centre and efficient management of Cloud workloads.

Chapter 6 proposed a DRL-based scheduling framework for energy and thermal efficiency in data centres. The manual configurations and simple heuristics often perform poorly in complex data centre environments. Hence, new resource management methods are required that learn the complexities of environments and accordingly take RMS actions. The proposed DRL-based scheduling method adequately designs state space, action space, and reward and accurately depicts data centre environments. The proposed policy-gradient based agent learns the environments and takes scheduling decisions, thus, eliminating the need for external heuristics. The agent interacts with cooling and computing subsystems of the environment, identifies parameter's tradeoffs, and optimises two systems together while scheduling workloads.

The chapters mentioned above collectively present energy and thermal efficient resource management algorithms and systems in Cloud data centres while simultaneously providing required SLAs or QoS for applications. The machine-learning-based solutions are new directions in data centre resource management systems. It is indeed a timely contribution to the state-of-the-art.

7.2 Future Research Directions

This thesis addressed several challenges of energy and thermal efficient resource management in Cloud data centres. However, Cloud computing can be further improved by addressing several key issues that require detailed investigation and solutions. This section gives some insights into these challenges for future work in this area.

7.2.1 Moving from ”time-to-solution” to ”Kw-to-solution”

The current software development paradigms, platforms, and algorithms focus on improving applications' execution speed, neglecting its energy footprints. Hence, a paradigm shift is required to move from “time-to-solution” to “Kw-to-Solution” in software development and deployments. New tools and programming constructs are needed to facilitate software developers to analyse the energy footprints of application logic so that developers can optimise software applications to minimise energy and improve execution speed.

7.2.2 Standardisation and Tools for AI-centric RMS

One of the important obstacles in adopting AI or ML solutions in data centre RMS is the lack of standardisation and tools. ML solutions need a good amount of data. Currently, distributed systems, including Cloud systems, produce vast amounts of data belonging to different computing layers. Standard methods and semantics are needed to collect, monitor, and interpret these data to adopt AI-centric models faster. Moreover, software tools and libraries need to be built specifically to resource management systems, which will easily integrate policies into existing systems.

7.2.3 Cross Layer Coordination Methods in Cloud Computing Stack

The total energy efficiency is achieved when resources are managed efficiently across different computing layers from on-chip microprocessor, data centre level platforms. Current approaches are limited to the individual computing layer due to a lack of coor-

dination and heterogeneity among different computing layer. New interfaces and APIs can be built that easily facilitates and allows configuration across different computing layers.

7.2.4 Resource Management in Emerging Cloud Execution Models

As Cloud computing is evolving, it is moving from partially managed services to fully-managed services through application execution models such as Serverless computing. Serverless computing allows an application to be built based on multiple stateless microservices. Cloud service providers manage these microservices or stateless functions lifecycle completely with an assurance of autonomic scalability. It brings new challenges in pricing and the management of thousands of stateless application services. This requires new resource management approaches in these fine-grained, network-accessed hardware resources shared by different containerised applications belonging to other users.

7.2.5 Holisitic Resource Management

Cloud data centres host closely interconnected systems, including computing, networking, storage and cooling systems. All these systems are closely interconnected and play an essential role in reliable service delivery. The resource management system should identify the dependencies and manage the resources holistically to achieve higher energy efficiency. It requires the development of new algorithms and platforms that configure parameters across different systems managing tradeoffs.

7.2.6 Efficiency Across Multi-tier Computing Platforms

The emergence of multi-tier (distributed computations from the network edge to remote clouds) computing paradigms such as Edge/Fog computing to support IoT applications has created new resource and application management challenges. Applications in such environments require low latency response, which requires Cloud services to move from centralised remote locations to the network's edge. Such environments are

highly heterogeneous than remote Clouds and are powered through battery or limited energy sources. Hence, application and resource management under these resource-constrained environments is challenging, requiring new solutions and approaches.

7.2.7 Decarbonising Cloud Computing

Cloud data centres contribute significant CO_2 emissions due to their heavy reliance on brown or fossil fuel-based energy sources. Many service providers are procuring renewable energy to decarbonise Cloud systems. However, intermittent availability has hindered the adoption of renewable energy sources. New solutions shall explore addressing energy storage infrastructures and workload management in uncertain energy availability. Moreover, policymakers need to enforce new regulations for Cloud service providers to adopt greener energy sources to power their data centres.

7.2.8 Privacy-aware Resource Management

The increasing security threats to internet services have brought new challenges in managing digital platforms. The new regulations, such as the General Data Protection Regulation (GDPR), require the data to be stored within the data source's geographic jurisdiction. This necessitates resource management solutions to be privacy-aware, requiring distributed storage and multi-part computation or computation over partial data. Hence, resource management platforms should be built considering such privacy and security requirements of applications.

7.3 Final Remarks

Cloud computing has become a backbone infrastructure of modern digital society. Solving the massive energy consumption problem of Cloud data centre is a significant issue of Cloud platforms. This thesis investigated how to leverage machine learning techniques to make Cloud data centres energy and thermal efficiency through various resource management techniques, including frequency scaling, workload consolidation,

thermal modelling, and scheduling. The proposed methods reduce the energy consumption of computing and cooling systems, minimise peak temperature in data centres, mitigate hotspots, meet application QoS such as deadlines, and minimise SLA violations for users. This thesis presents system models, designs mathematical models, proposes algorithms, builds ML prediction models, develops software systems, and produces opensource data sets. Moreover, these research outcomes provide opportunities for further innovation and development in Cloud computing platforms.

http://clouds.cis.unimelb.edu.au/ai4clouds/

Bibliography

[1] A. S. Andrae and T. Edler, "On global electricity usage of communication technology: trends to 2030," Challenges, vol. 6, no. 1, pp. 117–157, 2015.

[2] P. Johnson and T. Marker, "Data centre energy efficiency product profile," Pitt & Sherry, report to equipment energy efficiency committee (E3) of The Australian Government Department of the Environment, Water, Heritage and the Arts (DEWHA), 2009.

[3] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility," Future Generation computer systems, vol. 25, no. 6, pp. 599–616, 2009.

[4] A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica et al., "Above the clouds: A berkeley view of cloud computing," Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, vol. 28, no. 13, 2009.

[5] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," Future Generation Computer Systems, vol. 29, no. 7, pp. 1645–1660, 2013.

[6] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," Computer networks, vol. 54, no. 15, pp. 2787–2805, 2010.

[7] R. Mahmud, R. Kotagiri, and R. Buyya, "Fog computing: A taxonomy, survey and future directions," in Internet of everything. Springer, 2018, pp. 103–130.

[8] M. Satyanarayanan, "The emergence of edge computing," Computer, vol. 50, no. 1, pp. 30–39, 2017.

[9] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," Computer networks and ISDN systems, vol. 30, no. 1-7, pp. 107–117, 1998.

[10] Amazon, " Amazon Web Services." [Online]. Available: https://aws.amazon.com/

[11] Gartner, "Gartner forecasts worldwide public cloud revenue to grow 17 percent in 2020," https://www.gartner.com/en/newsroom/press-releases/2019-11-13-gartner-forecasts-worldwide-public-cloud-revenue-to-grow-17-percent-in-2020, 2019, [Online; accessed 10-Jan-2021].

[12] ——, "Gartner forecasts worldwide public cloud end-user spending to grow 18 percent in 2021," https://www.gartner.com/en/newsroom/press-releases/2020-11-17-gartner-forecasts-worldwide-public-cloud-end-user-spending-to-grow-18-percent-in-2021, 2020, [Online; accessed 10-Jan-2021].

[13] jachoos, " Microsoft Azure DataCenter Locations." [Online]. Available: https://jachoos.net/microsoft-azure-datacenter/

[14] S. Maybury., "How much Energy does your Data Centre Use?." https://www.metronode.com.au/energy_usage/, 2017, [Online; accessed 05-Jan-2021].

[15] A. Shehabi, S. Smith, D. Sartor, R. Brown, M. Herrlin, J. Koomey, E. Masanet, N. Horner, I. Azevedo, and W. Lintner, "United states data center energy usage report," 2016.

[16] J. Koomey, "Growth in data center electricity use 2005 to 2010," A report by Analytical Press, completed at the request of The New York Times, vol. 9, 2011.

[17] R. Pereira, M. Couto, F. Ribeiro, R. Rua, J. Cunha, J. P. Fernandes, and J. Saraiva, "Energy efficiency across programming languages: how do energy, time, and memory relate?" in Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering, 2017, pp. 256–267.

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

[19] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green ai," arXiv preprint arXiv:1907.10597, 2019.

[20] D. Amodei and D. Hernandez, "Ai and compute," 2018, https://blog.openai.com/ai-and-compute.

[21] J. Gao, "Machine learning applications for data center optimization," Google White Paper, 2014.

[22] A. Mirhoseini, H. Pham, Q. V. Le, B. Steiner, R. Larsen, Y. Zhou, N. Kumar, M. Norouzi, S. Bengio, and J. Dean, "Device placement optimization with reinforcement learning," in Proceedings of the 34th International Conference on Machine Learning. JMLR. org, 2017, pp. 2430–2439.

[23] S. Tuli, S. Ilager, K. Ramamohanarao, and R. Buyya, "Dynamic scheduling for stochastic edge-cloud computing environments using a3c learning and residual recurrent neural networks," IEEE Transactions on Mobile Computing, 2020.

[24] G. Ayers, N. P. Nagendra, D. I. August, H. K. Cho, S. Kanev, C. Kozyrakis, T. Krishnamurthy, H. Litz, T. Moseley, and P. Ranganathan, "Asmdb: understanding and mitigating front-end stalls in warehouse-scale computers," in Proceedings of the 46th International Symposium on Computer Architecture, 2019, pp. 462–473.

[25] M. Dayarathna, Y. Wen, and R. Fan, "Data center energy consumption modeling: A survey," IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 732–794, 2015.

[26] N. T. Hieu, M. Di Francesco, and A. Ylä-Jääski, "Virtual machine consolidation with multiple usage prediction for energy-efficient cloud data centers," IEEE Transactions on Services Computing, vol. 13, no. 1, pp. 186–199, 2017.

[27] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," Future Generation Computer Systems, vol. 28, no. 5, pp. 755–768, 2012.

[28] S. S. Gill and R. Buyya, "A taxonomy and future directions for sustainable cloud computing: 360 degree view," ACM Computing Surveys (CSUR), vol. 51, no. 5, pp. 1–33, 2018.

[29] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya, "A taxonomy and survey of energy-efficient data centers and cloud computing systems," in Advances in Computers. Elsevier, 2011, vol. 82, pp. 47–111.

[30] N. El-Sayed, I. A. Stefanovici, G. Amvrosiadis, A. A. Hwang, and B. Schroeder, "Temperature management in data centers: Why some (might) like it hot," in Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems, 2012, pp. 163–174.

[31] W. Torell, K. Brown, and V. Avelar, "The unexpected impact of raising data center temperatures," Write paper 221, Revision, 2015.

[32] J. Choi, Y. Kim, A. Sivasubramaniam, J. Srebric, Q. Wang, and J. Lee, "A cfd-based tool for studying temperature in rack-mounted servers," IEEE Transaction on Computers, vol. 57, no. 8, pp. 1129–1142, 2008.

[33] A. Almoli, A. Thompson, N. Kapur, J. Summers, H. Thompson, and G. Hannah, "Computational fluid dynamic investigation of liquid rack cooling in data centres," Applied energy, vol. 89, no. 1, pp. 150–155, 2012.

[34] Q. Tang, S. K. S. Gupta, and G. Varsamopoulos, "Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach," IEEE Transactions on Parallel and Distributed Systems, vol. 19, no. 11, pp. 1458–1472, 2008.

[35] H. Sun, P. Stolf, and J.-M. Pierson, "Spatio-temporal thermal-aware scheduling for homogeneous high-performance computing datacenters," Future Generation Computer Systems, vol. 71, pp. 157–170, 2017.

[36] M. Cheng, J. Li, and S. Nazarian, "Drl-cloud: Deep reinforcement learning-based resource provisioning and task scheduling for cloud service providers," in Proceedings of the 23rd Asia and South Pacific Design Automation Conference (ASP-DAC). IEEE, 2018, pp. 129–134.

[37] "Grid 5000," 2019. [Online]. Available: https://www.grid5000.fr/w/Grid5000:Home

[38] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software: Practice and experience, vol. 41, no. 1, pp. 23–50, 2011.

[39] Norton, "The future of iot: 10 predictions about the internet of things," https://us.norton.com/internetsecurity-iot-5-predictions-for-the-future-of-iot.html, 2019.

[40] I. Baldini, P. Castro, K. Chang, P. Cheng, S. Fink, V. Ishakian, N. Mitchell, V. Muthusamy, R. Rabbah, A. Slominski et al., "Serverless computing: Current trends and open problems," in Proceedings of the Research Advances in Cloud Computing. Springer, 2017, pp. 1–20.

[41] Y. Gan, Y. Zhang, D. Cheng, A. Shetty, P. Rathi, N. Katarki, A. Bruno, J. Hu, B. Ritchken, B. Jackson et al., "An open source benchmark suite for microservices and their hardware software implications for cloud and edge systems," in Proceedings of the Twenty Fourth International Conference on Architectural Support for Programming Languages and Operating Systems, 2019, pp. 3–18.

[42] A. V. Dastjerdi and R. Buyya, "Fog computing: Helping the internet of things realize its potential," Computer, vol. 49, no. 8, pp. 112–116, 2016.

[43] D. Jeff, "ML for system, system for ML, keynote talk in Workshop on ML for Systems, NIPS," 2018. [Online]. Available: http://mlforsystems.org/

[44] R. Bianchini, M. Fontoura, E. Cortez, A. Bonde, A. Muzio, A.-M. Constantin, T. Moscibroda, G. Magalhaes, G. Bablani, and M. Russinovich, "Toward ml-centric cloud platforms," Communications of the ACM, vol. 63, no. 2, pp. 50–59, 2020.

[45] R. S. Sutton, A. G. Barto et al., Introduction to reinforcement learning. MIT press Cambridge, 1998, vol. 135.

[46] R. S. Sutton and A. G. Barto, Reinforcement learning: An introduction. MIT press, 2018.

[47] H. Viswanathan, E. K. Lee, and D. Pompili, "Self-organizing sensing infrastructure for autonomic management of green datacenters," IEEE Network, vol. 25, no. 4, pp. 34–40, 2011.

[48] D. Minoli, K. Sohraby, and B. Occhiogrosso, "Iot considerations, requirements, and architectures for smart buildingsenergy optimization and next-generation building management systems," IEEE Internet of Things Journal, vol. 4, no. 1, pp. 269–283, 2017.

[49] Q. Liu, Y. Ma, M. Alhussein, Y. Zhang, and L. Peng, "Green data center with iot sensing and cloud-assisted smart temperature control system," Computer Networks, vol. 101, pp. 104–112, 2016.

[50] S. Saha and A. Majumdar, "Data centre temperature monitoring with esp8266 based wireless sensor network and cloud based dashboard with real time alert system," in Proceedings of the Devices for Integrated Circuit (DevIC). IEEE, 2017, pp. 307–310.

[51] C. Cummins, P. Petoumenos, Z. Wang, and H. Leather, "End-to-end deep learning of optimization heuristics," in Proceedings of the 26th International Conference on Parallel Architectures and Compilation Techniques (PACT). IEEE, 2017, pp. 219–232.

[52] I. A. Cano, "Optimizing distributed systems using machine learning," Ph.D. dissertation, University of Washington, Seattle, USA, 2019.

[53] I. Portugal, P. Alencar, and D. Cowan, "A survey on domain-specific languages for machine learning in big data," arXiv preprint arXiv:1602.07637, 2016.

[54] A. Toma, J. Wenner, J. E. Lenssen, and J.-J. Chen, "Adaptive quality optimization of computer vision tasks in resource-constrained devices using edge computing," in Proceedings of the 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), 2019, pp. 469–477.

[55] S. Russell and P. Norvig, Artificial intelligence: a modern approach. Prentice Hall, 2002.

[56] J. Y. Lee, S. V. Mehta, M. Wick, J.-B. Tristan, and J. Carbonell, "Gradient-based inference for networks with output constraints," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 4147–4154.

[57] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins et al., "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," Information Fusion, vol. 58, pp. 82–115, 2020.

[58] D. Gunning, "Explainable artificial intelligence (xai)," Defense Advanced Research Projects Agency (DARPA), nd Web, vol. 2, 2017.

[59] M. Bermudez-Edo, T. Elsaleh, P. Barnaghi, and K. Taylor, "Iot-lite: a lightweight semantic model for the internet of things and its use with dynamic semantics," Personal and Ubiquitous Computing, vol. 21, no. 3, pp. 475–487, 2017.

[60] H. Zhang, N. Liu, X. Chu, K. Long, A.-H. Aghvami, and V. C. Leung, "Network slicing based 5g and future mobile networks: mobility, resource management, and challenges," IEEE communications magazine, vol. 55, no. 8, pp. 138–145, 2017.

[61] G. Dulac-Arnold, D. Mankowitz, and T. Hester, "Challenges of real-world reinforcement learning," arXiv preprint arXiv:1904.12901, 2019.

[62] V. Venkatachalam and M. Franz, "Power reduction techniques for microprocessor systems," ACM Computing Surveys (CSUR), vol. 37, no. 3, pp. 195–237, 2005.

[63] R. A. Bridges, N. Imam, and T. M. Mintz, "Understanding GPU power: A survey of profiling, modeling, and simulation methods," ACM Computing Surveys, vol. 49, no. 3, 2016.

[64] G. Von Laszewski, L. Wang, A. J. Younge, and X. He, "Power-aware scheduling of virtual machines in dvfs-enabled clusters," in Proceedings of the IEEE International Conference on Cluster Computing and Workshops. IEEE, 2009, pp. 1–10.

[65] 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, vol. 29, no. 10, p. e4067, 2017.

[66] M. Safari and R. Khorsand, "Energy-aware scheduling algorithm for time-constrained workflow tasks in dvfs-enabled cloud environment," Simulation Modelling Practice and Theory, vol. 87, pp. 311–326, 2018.

[67] S. Wang, Z. Qian, J. Yuan, and I. You, "A dvfs based energy-efficient tasks scheduling in a data center," IEEE Access, vol. 5, pp. 13 090–13 102, 2017.

[68] J.-G. Park, N. Dutt, and S.-S. Lim, "Ml-gov: A machine learning enhanced integrated cpu-gpu dvfs governor for mobile gaming," in Proceedings of the 15th IEEE/ACM Symposium on Embedded Systems for Real-Time Multimedia, 2017, pp. 12–21.

[69] 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, vol. 4, no. 2, pp. 166–179, 2015.

[70] H. Li, J. Li, W. Yao, S. Nazarian, X. Lin, and Y. Wang, "Fast and energy-aware resource provisioning and task scheduling for cloud systems," in Proceedings of the 18th International Symposium on Quality Electronic Design (ISQED). IEEE, 2017, pp. 174–179.

[71] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Energy-efficient resource allocation and provisioning framework for cloud data centers," IEEE Transactions on Network and Service Management, vol. 12, no. 3, pp. 377–391, 2015.

[72] 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, vol. 99, pp. 20–35, 2015.

[73] Q. Huang, S. Su, J. Li, P. Xu, K. Shuang, and X. Huang, "Enhanced energy-efficient scheduling for parallel applications in cloud," in Proceedings of the 12th

IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid 2012). IEEE, 2012, pp. 781–786.

[74] Y. Ding, X. Qin, L. Liu, and T. Wang, "Energy efficient scheduling of virtual machines in cloud with deadline constraint," Future Generation Computer Systems, vol. 50, pp. 62–74, 2015.

[75] R. N. Calheiros and R. Buyya, "Energy-efficient scheduling of urgent bag-of-tasks applications in clouds through dvfs," in Proceedings of the IEEE 6th international conference on cloud computing technology and science. IEEE, 2014, pp. 342–349.

[76] C. Ghribi, M. Hadji, and D. Zeghlache, "Energy efficient vm scheduling for cloud data centers: Exact allocation and migration algorithms," in Proceedings of the 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing. IEEE, 2013, pp. 671–678.

[77] J. L. Berral, Í. Goiri, R. Nou, F. Julià, J. Guitart, R. Gavaldà, and J. Torres, "Towards energy-aware scheduling in data centers using machine learning," in Proceedings of the 1st International Conference on energy-Efficient Computing and Networking, 2010, pp. 215–224.

[78] S. F. Piraghaj, A. V. Dastjerdi, R. N. Calheiros, and R. Buyya, "A framework and algorithm for energy efficient container consolidation in cloud data centers," in Proceedings of the IEEE International Conference on Data Science and Data Intensive Systems. IEEE, 2015, pp. 368–375.

[79] M. H. Ferdaus, M. Murshed, R. N. Calheiros, and R. Buyya, "Virtual machine consolidation in cloud data centers using aco metaheuristic," in Proceedings of the European conference on parallel processing. Springer, 2014, pp. 306–317.

[80] Y. Sharma, W. Si, D. Sun, and B. Javadi, "Failure-aware energy-efficient vm consolidation in cloud computing systems," Future Generation Computer Systems, vol. 94, pp. 620–633, 2019.

[81] S.-Y. Hsieh, C.-S. Liu, R. Buyya, and A. Y. Zomaya, "Utilization-prediction-aware

virtual machine consolidation approach for energy-efficient cloud data centers," Journal of Parallel and Distributed Computing, vol. 139, pp. 99–109, 2020.

[82] F. Farahnakian, P. Liljeberg, and J. Plosila, "Energy-efficient virtual machines consolidation in cloud data centers using reinforcement learning," in Proceedings of the 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing. IEEE, 2014, pp. 500–507.

[83] D. Basu, X. Wang, Y. Hong, H. Chen, and S. Bressan, "Learn-as-you-go with megh: Efficient live migration of virtual machines," IEEE Transactions on Parallel and Distributed Systems, vol. 30, no. 8, pp. 1786–1801, 2019.

[84] A. A. Bhattacharya, D. Culler, A. Kansal, S. Govindan, and S. Sankar, "The need for speed and stability in data center power capping," Sustainable Computing: Informatics and Systems, vol. 3, no. 3, pp. 183–193, 2013.

[85] P. Petoumenos, L. Mukhanov, Z. Wang, H. Leather, and D. S. Nikolopoulos, "Power capping: What works, what does not," in Proceedings of the IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS). IEEE, 2015, pp. 525–534.

[86] R. Azimi, M. Badiei, X. Zhan, N. Li, and S. Reda, "Fast decentralized power capping for server clusters," in Proceedings of the IEEE International Symposium on High Performance Computer Architecture (HPCA), 2017, pp. 181–192.

[87] 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," ACM SIGARCH Computer Architecture News, vol. 44, no. 3, pp. 469–480, 2016.

[88] C. Lefurgy, X. Wang, and M. Ware, "Power capping: a prelude to power shifting," Cluster Computing, vol. 11, no. 2, pp. 183–195, 2008.

[89] A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy, "Optimal power allocation in server farms," ACM SIGMETRICS Performance Evaluation Review, vol. 37, no. 1, pp. 157–168, 2009.

[90] H. Chen, C. Hankendi, M. C. Caramanis, and A. K. Coskun, "Dynamic server power capping for enabling data center participation in power markets," in Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD). IEEE, 2013, pp. 122–129.

[91] H. Zhang and H. Hoffmann, "Maximizing performance under a power cap: A comparison of hardware, software, and hybrid techniques," ACM SIGPLAN Notices, vol. 51, no. 4, pp. 545–559, 2016.

[92] M. Xu and R. Buyya, "Managing renewable energy and carbon footprint in multi-cloud computing environments," Journal of Parallel and Distributed Computing, vol. 135, pp. 191–202, 2020.

[93] A. Khosravi, L. L. Andrew, and R. Buyya, "Dynamic vm placement method for minimizing energy and carbon cost in geographically distributed cloud data centers," IEEE Transactions on Sustainable Computing, vol. 2, no. 2, pp. 183–196, 2017.

[94] U. Mandal, M. F. Habib, S. Zhang, B. Mukherjee, and M. Tornatore, "Greening the cloud using renewable-energy-aware service migration," IEEE network, vol. 27, no. 6, pp. 36–43, 2013.

[95] Í. Goiri, K. Le, T. D. Nguyen, J. Guitart, J. Torres, and R. Bianchini, "Greenhadoop: leveraging green energy in data-processing frameworks," in Proceedings of the 7th ACM european conference on Computer Systems, 2012, pp. 57–70.

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

[97] J.-P. Lai, Y.-M. Chang, C.-H. Chen, and P.-F. Pai, "A survey of machine learning models in renewable energy predictions," Applied Sciences, vol. 10, no. 17, p. 5975, 2020.

[98] J. Gao, H. Wang, and H. Shen, "Smartly handling renewable energy instability in supporting a cloud datacenter," in Proceedings of the IEEE International Parallel and Distributed Processing Symposium (IPDPS). IEEE, 2020, pp. 769–778.

[99] E. Capra, C. Francalanci, and S. A. Slaughter, "Is software green? application development environments and energy efficiency in open source applications," Information and Software Technology, vol. 54, no. 1, pp. 60–71, 2012.

[100] G. Pinto and F. Castor, "Energy efficiency: a new concern for application software developers," Communications of the ACM, vol. 60, no. 12, pp. 68–75, 2017.

[101] D. Shin, S. W. Chung, E.-Y. Chung, and N. Chang, "Energy-optimal dynamic thermal management: Computation and cooling power co-optimization," IEEE Transactions on Industrial Informatics, vol. 6, no. 3, pp. 340–351, 2010.

[102] R. Ayoub, K. Indukuri, and T. S. Rosing, "Temperature aware dynamic workload scheduling in multisocket cpu servers," IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems, vol. 30, no. 9, pp. 1359–1372, 2011.

[103] H. F. Sheikh, I. Ahmad, Z. Wang, and S. Ranka, "An overview and classification of thermal-aware scheduling techniques for multi-core processing systems," Sustainable Computing: Informatics and Systems, vol. 2, no. 3, pp. 151–169, 2012.

[104] Z. Wang, C. Bash, N. Tolia, M. Marwah, X. Zhu, and P. Ranganathan, "Optimal fan speed control for thermal management of servers," in Proceedings of the International Electronic Packaging Technical Conference and Exhibition, vol. 43604, 2009, pp. 709–719.

[105] S. Pagani, P. S. Manoj, A. Jantsch, and J. Henkel, "Machine learning for power, energy, and thermal management on multicore processors: A survey," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no. 1, pp. 101–116, 2018.

[106] A. Iranfar, F. Terraneo, G. Csordas, M. Zapater, W. Fornaciari, and D. Atienza, "Dynamic thermal management with proactive fan speed control through rein-

forcement learning," in Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, 2020, pp. 418–423.

[107] W. Zhang, Y. Wen, Y. W. Wong, K. C. Toh, and C.-H. Chen, "Towards joint optimization over ict and cooling systems in data centre: A survey," IEEE Communications Surveys & Tutorials, vol. 18, no. 3, pp. 1596–1616, 2016.

[108] A. H. Khalaj and S. K. Halgamuge, "A review on efficient thermal management of air-and liquid-cooled data centers: From chip to the cooling system," Applied energy, vol. 205, pp. 1165–1188, 2017.

[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, vol. 19, pp. 14–28, 2018.

[110] M. Tian, "Energy optimization by fan speed control for data centers," Ph.D. dissertation, The George Washington University, 2019.

[111] ASHRAE, "American society of heating, refrigerating and air-conditioning engineers," 2018, URL. http://tc0909.ashraetcs.org/. [Online]. Available: http://tc0909.ashraetcs.org/

[112] R. Zhou, Z. Wang, C. E. Bash, A. McReynolds, C. Hoover, R. Shih, N. Kumari, and R. K. Sharma, "A holistic and optimal approach for data center cooling management," in Proceedings of the 2011 American Control Conference. IEEE, 2011, pp. 1346–1351.

[113] M. T. Chaudhry, T. C. Ling, A. Manzoor, S. A. Hussain, and J. Kim, "Thermal-aware scheduling in green data centers," ACM Computing Surveys (CSUR), vol. 47, no. 3, pp. 1–48, 2015.

[114] Y. Mhedheb, F. Jrad, J. Tao, J. Zhao, J. Kołodziej, and A. Streit, "Load and thermal-aware vm scheduling on the cloud," in Proceedings of the International Conference on Algorithms and Architectures for Parallel Processing. Springer, 2013, pp. 101–114.

[115] M. Polverini, A. Cianfrani, S. Ren, and A. V. Vasilakos, "Thermal-aware scheduling of batch jobs in geographically distributed data centers," IEEE Transactions on cloud computing, vol. 2, no. 1, pp. 71–84, 2013.

[116] L. Wang, G. von Laszewski, F. Huang, J. Dayal, T. Frulani, and G. Fox, "Task scheduling with ann-based temperature prediction in a data center: a simulation-based study," Engineering with Computers, vol. 27, no. 4, pp. 381–391, 2011.

[117] E. K. Lee, H. Viswanathan, and D. Pompili, "Vmap: Proactive thermal-aware virtual machine allocation in hpc cloud datacenters," in Proceedings of the 19th International Conference on High Performance Computing. IEEE, 2012, pp. 1–10.

[118] H. Shamalizadeh, L. Almeida, S. Wan, P. Amaral, S. Fu, and S. Prabh, "Optimized thermal-aware workload distribution considering allocation constraints in data centers," in Proceedings of the IEEE international conference on green computing and communications and ieee internet of things and IEEE cyber, physical and social computing. IEEE, 2013, pp. 208–214.

[119] R. Romadhon, M. Ali, A. M. Mahdzir, and Y. A. Abakr, "Optimization of cooling systems in data centre by computational fluid dynamics model and simulation," in Proceedings of the Innovative Technologies in Intelligent Systems and Industrial Applications. IEEE, 2009, pp. 322–327.

[120] Y. Tarutani, K. Hashimoto, G. Hasegawa, Y. Nakamura, T. Tamura, K. Matsuda, and M. Matsuoka, "Temperature distribution prediction in data centers for decreasing power consumption by machine learning," in Proceedings of the IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom). IEEE, 2015, pp. 635–642.

[121] R. Lloyd and M. Rebow, "Data driven prediction model (ddpm) for server inlet temperature prediction in raised-floor data centers," in Proceedings of the 17th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm). IEEE, 2018, pp. 716–725.

[122] W. Zhang, Y. Wen, Y. W. Wong, K. C. Toh, and C.-H. Chen, "Towards joint optimization over ict and cooling systems in data centre: A survey," IEEE Communications Surveys & Tutorials, vol. 18, no. 3, pp. 1596–1616, 2016.

[123] J. Wan, X. Gui, R. Zhang, and L. Fu, "Joint cooling and server control in data centers: A cross-layer framework for holistic energy minimization," IEEE Systems Journal, vol. 12, no. 3, pp. 2461–2472, 2017.

[124] S. Li, H. Le, N. Pham, J. Heo, and T. Abdelzaher, "Joint optimization of computing and cooling energy: Analytic model and a machine room case study," in Proceedings of the IEEE 32nd International Conference on Distributed Computing Systems. IEEE, 2012, pp. 396–405.

[125] F. Ahmad and T. Vijaykumar, "Joint optimization of idle and cooling power in data centers while maintaining response time," ACM Sigplan Notices, vol. 45, no. 3, pp. 243–256, 2010.

[126] Y. Ran, H. Hu, X. Zhou, and Y. Wen, "Deepee: Joint optimization of job scheduling and cooling control for data center energy efficiency using deep reinforcement learning," in Proceedings of the IEEE 39th International Conference on Distributed Computing Systems (ICDCS), 2019, pp. 645–655.

[127] B. Fakhim, M. Behnia, S. Armfield, and N. Srinarayana, "Cooling solutions in an operational data centre: A case study," Applied Thermal Engineering, vol. 31, no. 14-15, pp. 2279–2291, 2011.

[128] S. Ilager, K. Ramamohanarao, and R. Buyya, "Thermal prediction for efficient energy management of clouds using machine learning," IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 5, pp. 1044–1056, 2020.

[129] E. K. Lee, H. Viswanathan, and D. Pompili, "Proactive thermal-aware resource management in virtualized hpc cloud datacenters," IEEE Transactions on Cloud Computing, vol. 5, no. 2, pp. 234–248, 2015.

[130] B. Porumb, P. Ungureşan, L. F. Tutunaru, A. Şerban, and M. Bălan, "A review

of indirect evaporative cooling operating conditions and performances," Energy Procedia, vol. 85, pp. 452–460, 2016.

[131] T. Gao, S. Shao, Y. Cui, B. Espiritu, C. Ingalz, H. Tang, and A. Heydari, "A study of direct liquid cooling for high-density chips and accelerators," in Proceedings of the 16th IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (ITherm). IEEE, 2017, pp. 565–573.

[132] A. Capozzoli and G. Primiceri, "Cooling systems in data centers: state of art and emerging technologies," Energy Procedia, vol. 83, pp. 484–493, 2015.

[133] B. Cutler, S. Fowers, J. Kramer, and E. Peterson, "Dunking the data center," IEEE Spectrum, vol. 54, no. 3, pp. 26–31, 2017.

[134] D. Luebke, M. Harris, N. Govindaraju, A. Lefohn, M. Houston, J. Owens, M. Segal, M. Papakipos, and I. Buck, "Gpgpu: general-purpose computation on graphics hardware," in Proceedings of the 2006 ACM/IEEE conference on Supercomputing. ACM, 2006, p. 208.

[135] M. Jeon, S. Venkataraman, J. Qian, A. Phanishayee, W. Xiao, and F. Yang, "Multi-tenant gpu clusters for deep learning workloads: Analysis and implications," Technical report, Microsoft Research, 2018., Tech. Rep., 2018.

[136] top500, "top500 supercomputer list," 2019. [Online]. Available: https://www.top500.org/lists/2019/11/

[137] S. W. Keckler, W. J. Dally, B. Khailany, M. Garland, and D. Glasco, "Gpus and the future of parallel computing," IEEE Micro, 2011.

[138] R. R. Exposito, G. L. Taboada, S. Ramos, J. Touriño, and R. Doallo, "General-purpose computation on gpus for high performance cloud computing," Concurrency and Computation: Practice and Experience, vol. 25, no. 12, pp. 1628–1642, 2013.

[139] S. Ilager, R. Wankar, R. Kune, and R. Buyya, "Gpu paas computation model in aneka cloud computing environments," Smart Data: State-of-the-Art Perspectives in Computing and Applications, p. 19, 2019.

[140] R. A. Bridges, N. Imam, and T. M. Mintz, "Understanding gpu power: A survey of profiling, modeling, and simulation methods," ACM Computing Surveys (CSUR), vol. 49, no. 3, p. 41, 2016.

[141] K. Ebrahimi, G. F. Jones, and A. S. Fleischer, "A review of data center cooling technology, operating conditions and the corresponding low-grade waste heat recovery opportunities," Renewable and Sustainable Energy Reviews, vol. 31, pp. 622–638, 2014.

[142] X. Mei, L. S. Yung, K. Zhao, and X. Chu, "A measurement study of gpu dvfs on energy conservation," in Proceedings of the Workshop on Power-Aware Computing and Systems. ACM, 2013.

[143] R. Ge, R. Vogt, J. Majumder, A. Alam, M. Burtscher, and Z. Zong, "Effects of dynamic voltage and frequency scaling on a k20 gpu," in Proceedings of the 42nd International Conference on Parallel Processing, 2013.

[144] J. Guerreiro, A. Ilic, N. Roma, and P. Tomas, "Modeling and decoupling the gpu power consumption for cross-domain dvfs," IEEE Transactions on Parallel and Distributed Systems, 2019.

[145] J. Guerreiro, A. Ilic, N. Roma, and P. Toms, "Dvfs-aware application classification to improve gpgpus energy efficiency." Parallel Computing, vol. 83, pp. 93–117, 2019.

[146] G. Lee and R. H. Katz, "Heterogeneity-aware resource allocation and scheduling in the cloud." in Proceedings of the HotCloud, 2011.

[147] N. Capodieci, R. Cavicchioli, M. Bertogna, and A. Paramakuru, "Deadline-based scheduling for gpu with preemption support," in Proceedings of the IEEE Real-Time Systems Symposium (RTSS). IEEE, 2018, pp. 119–130.

[148] Q. Wang and X. Chu, "GPGPU Performance Estimation with Core and Memory Frequency Scaling," Proceedings of the International Conference on Parallel and Distributed Systems, 2019.

[149] K. Fan, B. Cosenza, and B. Juurlink, "Predictable gpus frequency scaling for energy and performance," in Proceedings of the 48th International Conference on Parallel Processing, 2019, pp. 1–10.

[150] S. Che, M. Boyer, J. Meng, D. Tarjan, J. W. Sheaffer, S.-H. Lee, and K. Skadron, "Rodinia: A benchmark suite for heterogeneous computing," in Proceedings of the IEEE international symposium on workload characterization (IISWC). Ieee, 2009, pp. 44–54.

[151] S. Grauer-Gray, L. Xu, R. Searles, S. Ayalasomayajula, and J. Cavazos, "Auto-tuning a high-level language targeted to gpu codes," in Proceedings of the Innovative Parallel Computing (InPar). IEEE, 2012, pp. 1–10.

[152] Y. M. Teo, "A model-driven approach for time-energy performance of parallel applications," in Proceedings of the 6th International Conference on Intelligent Systems, Modelling and Simulation, pp. 3–4, 2015.

[153] V. Chau, X. Chu, H. Liu, and Y. W. Leung, "Energy efficient job scheduling with DVFS for CPU-GPU heterogeneous systems," in Proceedings of the 8th International Conference on Future Energy Systems, 2017.

[154] K. Ma, X. Li, W. Chen, C. Zhang, and X. Wang, "GreenGPU: A holistic approach to energy efficiency in GPU-CPU heterogeneous architectures," in Proceedings of the International Conference on Parallel Processing, pp. 48–57, 2012.

[155] G. Wu, J. L. Greathouse, A. Lyashevsky, N. Jayasena, and D. Chiou, "Gpgpu performance and power estimation using machine learning," in Proceedings of the IEEE 21st International Symposium on High Performance Computer Architecture (HPCA). IEEE, 2015, pp. 564–576.

[156] A. Lsch and M. Platzner, "A highly accurate energy model for task execution on heterogeneous compute nodes," in Proceedings of the IEEE 29th International Conference on Application-specific Systems, Architectures and Processors (ASAP), 2018, pp. 1–8.

[157] T. Komoda, S. Hayashi, T. Nakada, S. Miwa, and H. Nakamura, "Power capping of cpu-gpu heterogeneous systems through coordinating dvfs and task mapping," in Proceedings of the IEEE 31st International Conference on Computer Design (ICCD). IEEE, 2013, pp. 349–356.

[158] Z. Tang, Y. Wang, Q. Wang, and X. Chu, "The impact of gpu dvfs on the energy and performance of deep learning: An empirical study," in Proceedings of the Tenth ACM International Conference on Future Energy Systems, 2019, pp. 315–325.

[159] A. Basu, J. L. Greathouse, G. Venkataramani, and J. Veselỳ, "Interference from gpu system service requests." in Proceedings of the IEEE international symposium on workload characterization (IISWC), 2018, pp. 179–190.

[160] Z. Allen-Zhu and E. Hazan, "Variance reduction for faster non-convex optimization," in Proceedings of the International conference on machine learning, 2016.

[161] H.-P. Kriegel, P. Kroger, and A. Zimek, "Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering," ACM Transactions on Knowledge Discovery from Data (TKDD), vol. 3, no. 1, p. 1, 2009.

[162] P. M. Pardalos and J. B. Rosen, Constrained global optimization: algorithms and applications. Springer, 1987, vol. 268.

[163] P. Basu, H. Chen, S. Saha, K. Chakraborty, and S. Roy, "SwiftGPU : Fostering Energy Efficiency in a Near-Threshold GPU Through a Tactical Performance Boost," Proceedings of the 2016 53rd ACM/EDAC/IEEE Design Automation Conference (DAC), 2016.

[164] C. D. Patel, C. E. Bash, and A. H. Beitelmal, "Smart cooling of data centers," Jun. 3 2003, uS Patent 6,574,104.

[165] R. W. Ahmad, A. Gani, S. H. A. Hamid, M. Shiraz, A. Yousafzai, and F. Xia, "A survey on virtual machine migration and server consolidation frameworks for cloud data centers." Journal of Network and Computer Applications, vol. 52, pp. 11–25, 2015.

[166] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers," Concurrency Computation Practice and Experience, vol. 24, no. 13, pp. 1397–1420, 2012.

[167] A. Verma, G. Dasgupta, T. K. Nayak, P. De, and R. Kothari, "Server workload analysis for power minimization using consolidation," in Proceedings of the USENIX Annual technical conference. USENIX Association, 2009, pp. 28–28.

[168] N. Bobroff, A. Kochut, and K. Beaty, "Dynamic placement of virtual machines for managing sla violations," in Proceedings of the Integrated Network Management. IEEE, 2007, pp. 119–128.

[169] R. Zhou, Z. Wang, C. E. Bash, and A. McReynolds, "Data center cooling management and analysis-a model based approach," in Proceedings of the Semiconductor Thermal Measurement and Management Symposium. IEEE, 2012, pp. 98–103.

[170] J. D. Moore, J. S. Chase, P. Ranganathan, and R. K. Sharma, "Making scheduling "cool": Temperature-aware workload placement in data centers." in Proceedings of the USENIX Annual Technical Conference. USENIX Association, 2005, pp. 61–75.

[171] G. Dhiman, G. Marchetti, and T. Rosing, "vgreen: a system for energy efficient computing in virtualized environments," in Proceedings of the 2009 ACM/IEEE International Symposium on Low Power Electronics and Design. ACM, 2009, pp. 243–248.

[172] A. Al-Qawasmeh, S. Pasricha, A. A. Maciejewski, and H. J. Siegel, "Power and thermal-aware workload allocation in heterogeneous data centers." IEEE Transaction on Computers, vol. 64, no. 2, pp. 477–491, 2015.

[173] S. Ben-David, A. Borodin, R. Karp, G. Tardos, and A. Wigderson, "On the power of randomization in on-line algorithms," Algorithmica, vol. 11, no. 1, pp. 2–14, 1994.

[174] T. A. Feo and M. G. C. Resende, "Greedy randomized adaptive search procedures," Journal of Global Optimization, vol. 6, no. 2, pp. 109–133, 1995.

[175] 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 Proceedings of the 14th International Symposium on High Performance Computer Architecture. IEEE, 2008, pp. 123–134.

[176] N. J. Kansal and I. Chana, "Energy-aware Virtual Machine Migration for Cloud Computing - A Firefly Optimization Approach," Journal of Grid Computing, vol. 14, no. 2, pp. 327–345, 2016.

[177] H. Li, G. Zhu, C. Cui, H. Tang, Y. Dou, and C. He, "Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing," Computing, vol. 98, no. 3, pp. 303–317, 2016.

[178] E. K. Lee, H. Viswanathan, and D. Pompili, "Proactive thermal-aware resource management in virtualized hpc cloud datacenters," IEEE Transactions on Cloud Computing, vol. 5, no. 2, pp. 234–248, 2017.

[179] X. Li, X. Jiang, P. Garraghan, and Z. Wu, "Holistic energy and failure aware workload scheduling in cloud datacenters." Future Generation Computer Systems, vol. 78, pp. 887–900, 2018.

[180] T. C. Ferreto, M. A. Netto, R. N. Calheiros, and C. A. De Rose, "Server consolidation with migration control for virtualized data centers," Future Generation Computer Systems, vol. 27, no. 8, pp. 1027–1034, 2011.

[181] 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, vol. 29, no. 6, pp. 1–1, 2017.

[182] A. Beloglazov and R. Buyya, "Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints," IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 7, pp. 1366–1379, 2013.

[183] K. Zheng, X. Wang, L. Li, and X. Wang, "Joint power optimization of data center network and servers with correlation analysis," in Proceedings of INFOCOM. IEEE, 2014, pp. 2598–2606.

[184] S. Zhang and K. S. Chatha, "Approximation algorithm for the temperature aware scheduling problem," in Proceedings of the International Conference on Computer-Aided Design, pp. 281–288, 2007.

[185] N. Rasmussen, "Avoidable mistakes that compromise cooling performance in data centers and network rooms," White paper, vol. 49, pp. 2003–0, 2003.

[186] SPEC, "Standard performance evaluation corporation," 2008. [Online]. Available: https://www.spec.org/power_ssj2008/index.html

[187] K. Park and V. S. Pai, "Comon: a mostly-scalable monitoring system for planetlab." Operating Systems Review, vol. 40, no. 1, pp. 65–74, 2006.

[188] K. Ebrahimi, G. F. Jones, and A. S. Fleischer, "A review of data center cooling technology, operating conditions and the corresponding low-grade waste heat recovery opportunities," Renewable and Sustainable Energy Reviews, vol. 31, pp. 622–638, 2014.

[189] W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, "Cost of virtual machine live migration in clouds: A performance evaluation," in Proceedings of the IEEE International Conference on Cloud Computing. Springer, 2009, pp. 254–265.

[190] U. plc, "A guide to ensuring your ups batteries do not fail from ups systems," 2018. [Online]. Available: http://www.upssystems.co.uk/knowledge-base/the-it-professionals-guide-to-standby-power/part-8-how-to-ensure-your-batteries-dont-fail/

[191] K. Zhang, A. Guliani, S. Ogrenci-Memik, G. Memik, K. Yoshii, R. Sankaran, and P. Beckman, "Machine learning-based temperature prediction for runtime thermal management across system components," IEEE Transactions on Parallel and Distributed Systems, vol. 29, no. 2, pp. 405–419, Feb 2018.

[192] S. Ilager, K. Ramamohanarao, and R. Buyya, "Etas: Energy and thermal-aware dynamic virtual machine consolidation in cloud data center with proactive hotspot mitigation," Concurrency and Computation: Practice and Experience, vol. 0, no. 0, p. e5221, 2019.

[193] J. Moore, J. S. Chase, and P. Ranganathan, "Weatherman: Automated, online and predictive thermal mapping and management for data centers," in Proceedings of the IEEE International Conference on Autonomic Computing, June 2006, pp. 155–164.

[194] M. Zapater, J. L. Risco-Martín, P. Arroba, J. L. Ayala, J. M. Moya, and R. Hermida, "Runtime data center temperature prediction using grammatical evolution techniques," Applied Soft Computing, vol. 49, pp. 94–107, 2016.

[195] G. Fox, J. A. Glazier, J. Kadupitiya, V. Jadhao, M. Kim, J. Qiu, J. P. Sluka, E. Somogyi, M. Marathe, A. Adiga et al., "Learning everywhere: Pervasive machine learning for effective high-performance computation," in Proceedings of the IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). IEEE, 2019, pp. 422–429.

[196] H. Mao, M. Schwarzkopf, S. B. Venkatakrishnan, Z. Meng, and M. Alizadeh, "Learning scheduling algorithms for data processing clusters," in Proceedings of the ACM Special Interest Group on Data Communication. New York, NY, USA: ACM, 2019, pp. 270–288.

[197] Y. Luo, X. Wang, S. Ogrenci-Memik, G. Memik, K. Yoshii, and P. Beckman, "Minimizing thermal variation in heterogeneous hpc systems with fpga nodes," in Proceedings of the 2018 IEEE 36th International Conference on Computer Design (ICCD), Oct 2018, pp. 537–544.

[198] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in Proceedings of the 22nd International Conference on Knowledge Discovery and Data Mining. ACM, 2016, pp. 785–794.

[199] T. Cao, W. Huang, Y. He, and M. Kondo, "Cooling-aware job scheduling and node

allocation for overprovisioned hpc systems," in Proceedings of the Parallel and Distributed Processing Symposium (IPDPS), 2017 IEEE International. IEEE, 2017, pp. 728–737.

[200] C. Imes, S. Hofmeyr, and H. Hoffmann, "Energy-efficient application resource scheduling using machine learning classifiers," in Proceedings of the 47th International Conference on Parallel Processing. New York, NY, USA: ACM, 2018, pp. 45:1–45:11.

[201] K. Zhang, S. O. Memik, G. Memik, K. Yoshii, R. Sankaran, and P. H. Beckman, "Minimizing thermal variation across system components." in Proceedings of International Parallel and Distributed Processing Symposium. IEEE, 2015, pp. 1139–1148.

[202] I. Aransay, M. Z. Sancho, P. A. García, and J. M. M. Fernández, "Self-Organizing maps for detecting abnormal thermal behavior in data centers," in Proceedings of the 8th IEEE International Conference on Cloud Computing (CLOUD), pp. 138–145, 2015.

[203] A. I. Orhean, F. Pop, and I. Raicu, "New scheduling approach using reinforcement learning for heterogeneous distributed systems," Journal of Parallel and Distributed Computing, vol. 117, pp. 292–302, 2018.

[204] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in Proceedings of the 15th ACM Workshop on Hot Topics in Networks, ser. HotNets '16. New York, NY, USA: ACM, 2016, pp. 50–56.

[205] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg et al., "Scikit-learn: Machine learning in python," Journal of Machine Learning Research, vol. 12, no. Oct, pp. 2825–2830, 2011.

[206] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in Proceedings of the 23rd International Conference on

Machine Learning, ser. ICML '06. New York, NY, USA: ACM, 2006, pp. 161–168.

[207] R. Bekkerman, "The present and the future of the kdd cup competition," 2015.

[208] A. Mangal and N. Kumar, "Using big data to enhance the bosch production line performance: A kaggle challenge," in Proceedings of the IEEE International Conference on Big Data (Big Data). IEEE, 2016, pp. 2029–2035.

[209] M. Xu, A. V. Dastjerdi, and R. Buyya, "Energy efficient scheduling of cloud application components with brownout," IEEE Transactions on Sustainable Computing, vol. 1, no. 2, pp. 40–53, 2016.

[210] S. Shen, V. van Beek, and A. Iosup, "Statistical characterization of business-critical workloads hosted in cloud datacenters," in Proceedings of the 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid). IEEE, 2015, pp. 465–474.

[211] SPEC, "Standard performance evaluation corporation," 2018. [Online]. Available: https://www.spec.org/benchmarks.html

[212] Y. Li, Y. Wen, D. Tao, and K. Guan, "Transforming cooling optimization for green data center via deep reinforcement learning," IEEE Transactions on Cybernetics, vol. 50, no. 5, pp. 2002–2013, 2020.

[213] Y. Li, X. Hu, Y. Zhuang, Z. Gao, P. Zhang, and N. El-Sheimy, "Deep reinforcement learning (drl): Another perspective for unsupervised wireless localization," IEEE Internet of Things Journal, vol. 7, no. 7, pp. 6279–6287, 2020.

[214] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: Deep learning for the Internet of Things with edge computing," IEEE Network, vol. 32, no. 1, pp. 96–101, 2018.

[215] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, and P. Li, "Energy-efficient scheduling for real-time systems based on deep q-learning model," IEEE Transactions on Sustainable Computing, vol. 4, no. 1, pp. 132–141, 2017.

[216] M. Xu, S. Alamro, T. Lan, and S. Subramaniam, "Laser: A deep learning approach for speculative execution and replication of deadline-critical jobs in cloud," in Proceedings of the 26th International Conference on Computer Communication and Networks (ICCCN). IEEE, 2017, pp. 1–8.

[217] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, S. U. Khan, and P. Li, "A double deep Q-learning model for energy-efficient edge scheduling," IEEE Transactions on Services Computing, 2018, (preprint).

[218] L. Huang, S. Bi, and Y. J. Zhang, "Deep reinforcement learning for online computation offloading in wireless powered mobile-edge computing networks," IEEE Transactions on Mobile Computing, pp. 1–1, 2019.

[219] D.-M. Bui, Y. Yoon, E.-N. Huh, S. Jun, and S. Lee, "Energy efficiency for cloud computing system based on predictive optimization," Journal of Parallel and Distributed Computing, vol. 102, pp. 103–114, 2017.

[220] F. Li and B. Hu, "Deepjs: Job scheduling based on deep reinforcement learning in cloud data center," in Proceedings of the 4th International Conference on Big Data and Computing, 2019, pp. 48–53.

[221] G. Rjoub, J. Bentahar, O. Abdel Wahab, and A. Saleh Bataineh, "Deep and reinforcement learning for automated task scheduling in large-scale cloud computing systems," Concurrency and Computation: Practice and Experience, p. e5919, 2020, (in press).

[222] S. Ilager, R. Muralidhar, and R. Buyya, "Artificial intelligence (ai)-centric management of resources in modern distributed computing systems," in Proceedings of the IEEE Cloud Summit. IEEE, 2020, pp. 1–10.

[223] A. N. Toosi, R. N. Calheiros, R. K. Thulasiram, and R. Buyya, "Resource provisioning policies to increase iaas provider's profit in a federated cloud environment," in Proceedings of the IEEE International Conference on High Performance Computing and Communications, 2011, pp. 279–287.

[224] X. Fu and C. Zhou, "Predicted affinity based virtual machine placement in cloud computing environments," IEEE Transactions on Cloud Computing, vol. 8, no. 1, pp. 246–255, 2017.

[225] S. Guadarrama, A. Korattikara, O. Ramirez, P. Castro, E. Holly, S. Fishman, K. Wang, E. Gonina, N. Wu, E. Kokiopoulou, L. Sbaiz, J. Smith, G. Bartk, J. Berent, C. Harris, V. Vanhoucke, and E. Brevdo, "TF-Agents: A library for reinforcement learning in tensorflow," https://github.com/tensorflow/agents, 2018, [Online; accessed 25-June-2019]. [Online]. Available: https://github.com/tensorflow/agents

[226] S. Ilager, K. Ramamohanarao, and R. Buyya, "Thermal prediction for efficient energy management of clouds using machine learning," IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 5, pp. 1044–1056, 2020.

[227] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in Proceedings of the 12th International Conference on Neural Information Processing Systems, ser. NIPS'99. Cambridge, MA, USA: MIT Press, 1999, pp. 1057–1063.