Energy-Efficient Resource Scheduling Techniques For Cloud Computing

*A **Thesis***

submitted for the award of degree of

DOCTOR OF PHILOSOPHY

by

Tarandeep Kaur

(901203005)

Under the guidance of

Dr. Inderveer Chana

Professor

Computer Science and Engineering Department

Thapar Institute of Engineering and Technology, PATIALA

May, 2019

Contents

List of Figures

List of Tables

Dedicated to
My Loving Family

Certificate

I hereby certify that the work which is being presented in this thesis titled **"Energy-Efficient Resource Scheduling Techniques For Cloud Computing"**, for the award of degree of **"Doctor of Philosphy"** submitted in Computer Science and Engineering Department of Thapar Institute of Engineering and Technology, Patiala, is an authentic record of my own work carried out under the supervision of Dr. Inderveer Chana and refers other researchers works which are duly listed in the reference section.

The matter presented in this thesis has not been submitted for the award of any other degree of this or any other university.

Tarandeep Kaur

(Tarandeep Kaur)
901203005

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

Inderveer 08/09/15

(Inderveer Chana)
Professor
Computer Science and Engineering Department
TIET, Patiala

Acknowledgments

At the outset, I express my gratitude to the Almighty, who was like a lamp to my feet and filled me with the zeal,enthusiasm, wisdom and perseverance to complete the challenging journey of this PhD thesis successfully. I am eternally grateful to Him for his continuum of blessings.

I would like to express my heartfelt gratitude to my worthy supervisor, Dr. Inderveer Chana, Professor, Thapar Institute of Engineering and Technology, Patiala, for being a beacon of light and continuously guiding me through thick and thin. She indeed is a treasure trove of all the qualities that a pupil could have asked for in a mentor. Her constant support, encouragement, immense knowledge and scientic perception, lit up my way in the darkest times. I am eternally indebted to her for her insightful discussions and path breaking suggestions that assisted me in shaping up the direction of this research. The time and energy that she devoted on careful and the patient proof readings of this manuscript helped me to present this thesis with the desired quality. I cannot imagine a better mentor for my PhD study. Her belief in me was always re-energizing and rejuvenating.

I extend my sincere thanks to the members of my Doctoral committee - Dr. Seema Bawa, Dr. Neeraj Kumar and Dr. M. D. Singh for their academic support and invaluable comments. I sincerely concede their valuable feedback and constructive comments while ensuring the progress of my research work. I will forever be thankful to Dr. Maninder Singh, Head, Computer Science & Engineering Department, Thapar Institute of Engineering and Technology, Patiala, for his whole hearted support & for providing productive and stimulating ideas that motivated me to explore the tools and technologies for carrying out the experiments during this research process. My deepest regards and gratitude to Dr. O.P.Pandey, Dean of Research & Sponsored Projects and Dr. Prakash Gopalan, Director, Thapar Institute of Engineering and Technology for all the facilities that have been instrumental in the creation of a

healthy research environment in the university and have been immensely helpful for the completion of my work.

I wish to further extend my thanks to all the members of the faculty of the Computer Science & Engineering Department, Thapar Institute of Engineering and Technology, Patiala, who helped me in one way or the other to carry out my research work successfully. I am grateful to them for their co-operation and support. I also acknowledge the cooperation rendered to me by the office and the laboratory staff of this department. I would also like to thank all my friends and lab mates for their continuous motivation and moral support. Also, I am thankful to them for all the great times that we have shared and for making this research experience enjoyable and memorable.

I gratefully acknowledge the generous assistance provided by Corporation Bank, Bakkarwala branch employees, India. I also express my appreciation to the organization for granting me an opportunity to work on its infrastructure.

This thesis would never have been conceived or borne fruit without the unconditional support of my family. My parents deserve a special mention here, who inculcated moral, ethical and religious values in me. They made me smile and realize their faith in me during the rough road of this journey. They have passed onto me a wonderful humanitarian lineage and a good foundation to face the challenges of life.

I would extend my deep sense of gratitude to my sister, brother-in-law and my little niece who extended their cooperation and encouragement to me. Collectively, my family provided me an excellent conducive environment for continuing my research.

Last but not the least, my warmest thanks to my dear husband, whose continuous motivation is beyond expression in words. He has entered my life just a year ago but his are the encouraging words that made sure that I concluded what I began. He instilled condence in me to successfully complete this journey. I look forward for a life full of happiness and togetherness.

Tarandeep kaur

Abstract

Cloud computing, a form of distributed computing, assures steadfast and convoluted service delivery. The emergence of cloud has revolutionized the style of consumption and delivery of computing and information technology by triggering relocation of computing and data capabilities from personalized computers to big data centers. The setting up of cloud data centers has embellished the delivery of extensive, reliable and intricate technical developments while assuring extremely networked, scalable, and virtualized environment to the users.

The data centers facilitating cloud services have undoubtedly ensured swift service delivery but have significantly exaggerated the energy demands and caused severe energy crisis. A large amount of energy is wasted in these data centers whilst posing performance abasements due to irregular resource utility and energy consumption levels. The trepidations related to energy crisis have further rooted serious environmental concerns and danger to ecological sustainability by increasing atmospheric Green House Gas and Carbon Dioxide (CO_2) emissions. Besides this, the high energy consumption and demand hampers processing capabilities in terms of execution time, Quality of Service (QoS) parameters etc. and elevates energy-related expenditures. Consequently, the realization of energy efficiency and surmountability of high energy demands has become prime concern for the computing sector.

The curtailment of the energy demands becomes more important when scarcity, location-dependency and other restrictions are associated with the renewable and non-renewable sources of energy. Thus, the constraints related to such explicit measures and mounting energy demands have called for the implicit management of the energy crisis within the data centers. For this, energy optimization initiatives have been made at both the hardware as well as software levels, out of which, the software-oriented measures have proved more significant in terms of accomplishment and cost. The energy-aware allocation and scheduling schemes form part of such software-based energy efficiency measures.

The research work presented in this thesis proposes energy-efficient resource scheduling techniques for cloud. The techniques are offered through two energy-efficient allocation and scheduling models in cloud computing. These models offer efficient resource utilizations, proficient energy usages, improvement in the performance while minimizing energy costs. The first model is Green Cloud Scheduling Model (GCSM) that performs energy-efficient allocation and scheduling of tasks on the cloud nodes. It principally tends to optimize the energy consumption and thus referred to as green model. The Green Cloud Scheduling (GCS) technique of GCSM tends to optimally provision and schedule the tasks entering the system while maintaining minimal energy consumption, maximizing utilization of resources, fulfilling task deadlines and averting degradation in the overall performance. Exclusively, GCSM is an energy and deadline optimization model in cloud computing.

Thereafter, a GreenSched model, an extended model of GCSM has been proposed. GreenSched implements an intelligent machine learning technique, Forward-only Counter Propagation Network (CPN)-based scheduling (FCS), that proactively allocates and schedules deadline-and-budget constrained tasks to identified energy-aware cloud nodes. GreenSched primarily achieves energy efficiency along with optimizing variable QoS metrics such as, deadline, budget and performance. Contrary to GCSM, GreenSched model is an intelligent model that apart from offering energy efficiency also assures high QoS for performance and cost parameters. Unlike GCSM, it assures contentment of both user-imposed deadline and budget restrictions particularly when energy, deadline and budget are closely inter-related parameters.

The performance of the proposed models with variation in time, frequency of tasks and number of nodes has been analyzed which vary in the cloud environment. Also, all the proposed techniques have been evaluated with respect to the energy consumption levels and performance. In order to validate GCSM, a prototype environment as close as possible to a cloud data centre has been simulated. A heterogeneous cloud environment consisting of different virtualized servers to exploit the power of cloud computing has been created. The experimental results predict that GCSM achieves energy savings up to 71% and almost 82% tasks are completed within the deadline constraints as imposed by users pertaining to each task. For evaluation of

GreenSched model, a simulated data center in CloudSim toolkit has been created consisting of variable nodes, running multiple VMs using the customizable policies and methods available in the CloudSim toolkit. The experimental observations indicate that the scheduling technique in GreenSched model achieves an overall energy savings up to 64.21% as compared to two other techniques.

Thereafter, the proposed GCS scheduling technique of GCSM has been experimentally verified through a case study conducted at Corporation Bank, Bakkarwala branch, New Delhi. GCS has been experimentally compared for energy consumption levels, resource utilizations (both CPU and memory) and performance achieved with cloud-based Finacle application scheduler implemented by the Corporation Bank. The experimental results demonstrate the efficacy of GCS by saving an average of 76.58% energy with 80% CPU and 67% memory utility levels. GCS achieves 81.3% performance while minimizing the energy and thus achieving green banking operation at the bank.The testing results demonstrate that the proposed solutions are working efficiently and can be effectively used to address the low resource utility and high energy consumption challenges to realize energy efficiency.

By minimizing the energy consumption, the proposed models and their techniques circuitously reduce energy demands, carbon emissions and tend to overcome cooling requirements, energy expenditures of the cloud data centers and help in accomplishing green computing.

Chapter 1

Introduction

Cloud computing has emerged as a boon for the Information and Communication (ICT) industry owing to its rich, reliable and intricate stock of services. Presently, the extended capability of cloud computing in also offering hardware-, data-, metal-as-a-service apart from software-, platform- and infrastructure-as-a-service makes it ideal for today's comprehensive and complex computational requirements. Significantly, the current drift in setting up large computing data centers has undoubtedly embellished the delivery of all-embracing, and steadfast technical services but has also overstated the energy demands and posed stern energy crisis.

Comparatively, the ICT sector has emerged as major culprit in bumping up the energy crisis over other sectors (agriculture, transportation, manufacturing etc.). These energy crisis apart from boosting the energy demands, trigger severe climatic instabilities by increasing atmospheric Green House Gas (GHG) and Carbon Dioxide (CO_2) emissions. Besides this, the high energy consumption and demand augments energy-related expenditures, thereby, fiscally upsetting the ICT vendors. Accordingly, a large number of ICT firms have initiated energy saving measures at both the hardware and software levels. Particularly, the software-oriented measures including energy-aware resource allocation; management; and scheduling schemes have proved more significant. Additionally, the accomplishment of these optimization measures via a strong and effective cloud platform undoubtedly actuates achievement of energy efficiency. Generally, the energy efficiency realized through cloud computing has been coined as Green cloud computing and implementing resource or task scheduling can be called as Green Resource Scheduling or Green Task Scheduling respectively.

This chapter provides a high level view of the thesis. The initial Section 1.1 throws light on the fundamental concepts of cloud computing and discusses its applications. Section 1.2 analyses the necessity of energy efficiency and investigates the impact of rising energy consumption. Section 1.3 unfolds the growth and development of the concept of Green cloud computing identifying the underpinning technologies facilitating its successful implementation. Section 1.4 discusses the motivation behind proposing a novel cloud-based energy efficient scheduling technique and acuminates the contributions and organization of the rest of the thesis in subsequent Section 1.5.

1.1 Cloud Computing- An Overview

Cloud computing has embarked a revolution in accessing, provisioning and consumption of the information and computing in the ICT industry. It has emerged as a novel paradigm of high-performance and large scale computing that actuates relocation of computing and data from desktops and personal computers to big data centers. These cloud data houses or centers offer an extremely networked, scalable, and virtualized environment to the users who do not have to own and manage systems [1,2]. Instead, the users rely on the vast amount of space and computing capability offered by the cloud service providers [3].

Cloud computing tends to exploit the capability of IT infrastructure "as a service" to the users in an economically sustainable and pay-per-use way [4]. By adopting cloud services, companies and simple users are enabled to externalize their hardware resources, services, applications and their IT functions. This facilitates scalability and consistency accompanied with economic advantage for both consumers and providers. From the consumer point of view, they are required only to pay for what resources they need whereas from the providers point of view, the resource utilization can be maximized in addition to the increase in the profits which is indeed their biggest priority. The National Institute of Standards and Technology (NIST) has defined cloud computing as, "a model that facilitates expedient and dynamic access to a large pool of computing resources that can be shared, dynamically allocated, and discharged without much managerial involvements or service provider assistance" [5–9].

The growth of cloud computing is not an instantaneous task but has transited from several intermediate stages beginning from an era of mainframe computing where huge and powerful mainframe systems supported many users connected through dummy terminals and running data management applications. Figure 1.1 depicts the five phases of the intermediary stages from mainframe computing to the use of personal stand-alone computers running desktop applications deriving personal computing to the influx of interconnected computers converging to networking computing. This stage saw the growth of local networks connected to other networks creating a globally interconnected network such as the Internet for utilizing remote applications and resources. The networked computers usually functioned in autonomic fashion resulting in autonomic computing or followed client-server architectures resulting in client-server computing. The development of grid computing offered sharing of computing power and resources spread across multiple geographical domains. The recent stage involves rise of cloud computing where service-oriented, market-based computing applications are predominant.

However, cloud computing is similar to grid computing and autonomic computing in certain prospects but varies distinctively in other aspects. It offers distinguishing benefits including [7,10–12]:

- Multitenancy: Multitenancy implies the services offered by multiple cloud providers are simultaneously located in a single data center. It enables sharing of resources and costs across a large pool of users thus allowing for centralization

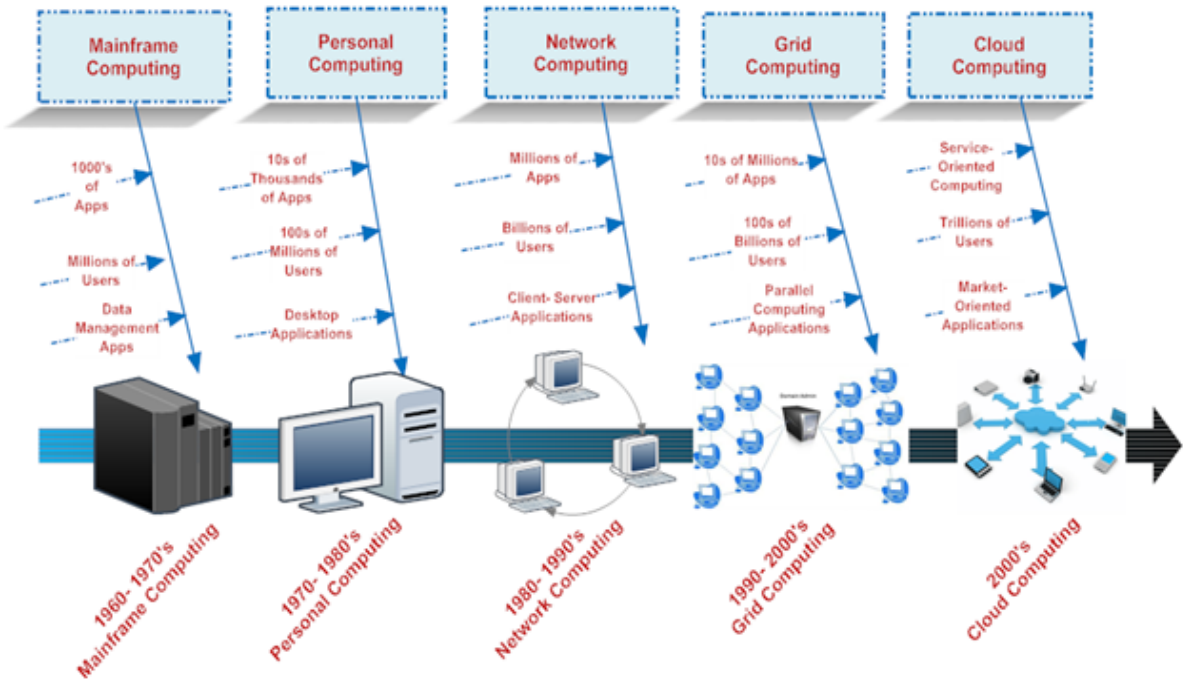

Figure 1.1: Computing Evolution

of infrastructure in locations with lower costs (such as real estate, electricity, etc.); utilization and efficiency improvements for systems that are often only 10-20% utilized.

- Geographically Distributed and Accessible: Cloud data centers, these days, are globally distributed and located at multiple locations. Different users possessing internet connectivity on their devices can easily access cloud services irrespective of their locations from these data centers. In other words, the device and location independence enables the users to access services regardless of their location or what device they are using. Comparatively, a service provider can benefit from the geo-diversity spread across multiple geographical domains to maximize the service utility.

- Affordable and Economical: From a user perspective, cloud computing offers independence from hosting vast and expensive infrastructure that could have been otherwise needed to access the IT services. Instead, cloud data centers offering services house huge infrastructure, monitored and maintained around the clock by the service providers. The availability of highly robust infrastructure typically provided by a third-party and not purchased by users reduces the incurrence costs.

- Large and Shared Pool of Resources: The large pool of resources is available with

cloud infrastructure where the resources are dynamically allocated to the multiple cloud users on demand. The dynamic resource allocation enables flexibility in managing of resource utilization and associated operating expenditure.

- Service- Driven Operability: Cloud computing is a service-driven operating model that stringently emphasizes on service delivery and management. The cloud services are regulated as per a negotiated Service Level Agreement (SLA) with the users which is one of the most critical concerns for a service provider.

- Dynamic Provisioning of Resources: The cloud resources can be dynamically allocated and de-allocated as and when required based on current demand, in contrary to traditional model where the resources were provisioned driven by the peak demand. The dynamic ("on-demand") resource provisioning aids in lowering the operational costs. Moreover, such a fine- grained, self-service based dynamic resource provisioning without users having to engineer for peak loads favors scalability and elasticity.

- Pay-per Use Pricing Model: Cloud computing is based on a pay-per- use pricing model. Even though, the scheme for incurring pricing is dependent on the type if service but on a whole, it lowers the service operating cost when users are charged on a per-use basis. However, it also introduces complexities in controlling the operating cost.

- Self Organization: The dynamic provisioning of resources empowers the self management capabilities of service providers. The automated management of resources enables them to rapidly respond to sudden demand of service delivery such as in surge computing.

1.1.1 Cloud Services, Deployment Models and Entities- An Insight

Cloud computing services and deployment models describe how the services delivery is carried out in cloud computing. They indicate the topological layouts for the cloud computing [13]. The entities basically correspond to the operational components in cloud computing.

I Cloud Service Models: Currently, cloud service models can be abstracted into two high levels of classifications, namely, archetypal and hybrid cloud service models. The classic or archetypal category pertains to the three original cloud service models, Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS) and Infrastructure-as-a-Service (IaaS) [7, 14]. Figure 1.2 depicts the classic cloud service models.

- SaaS is a software delivery model that helps users to access applications through a simple interface over the Internet. The providers of SaaS posses total control on the applications and they enable the users to access them.

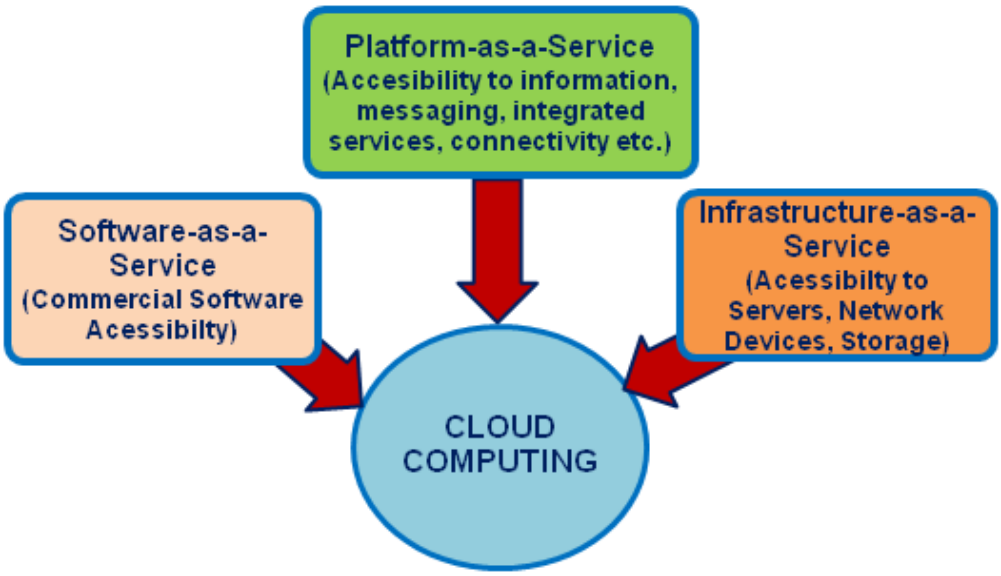

Figure 1.2: Cloud Service Models

The users have an illusion as if their applications are locally hosted without being bothered of the application background details. Typical SaaS examples are social media platforms, email boxes, Facebook, Google Apps etc.

- PaaS model is more of an urbane model that offers building, testing, deployment, and hosting environments for applications created by users or otherwise acquired from them. The prominent platforms of Microsoft Azure, Google App Engine are perfect PaaS cloud models.
- IaaS is an undemanding model for delivering the cloud service. It provides an actual physical infrastructure support that includes computing, storing, networking, and other primary resources to the users [15]. The users benefit by renting resources from the IaaS providers and using them on demand instead of incurring their own infrastructure. Example are Amazon EC2, Nimbus etc.

Although, the archetypal service models, SaaS, PaaS, and IaaS models have contributed significantly towards delivering cloud services but with the gradual progress of cloud and the users demanding more flexibility and control on the deployment of their applications, these models have often been speculated and considered for improvements. This resulted in consolidation of PaaS and IaaS models paving development and nurturing of new hybrid cloud service delivery models. The hybrid models have enabled the users to create and experience a single gamut of cloud services. Big IT tycoons such as Google [16], Microsoft [17] and Amazon [18] have initiated efforts to concurrently blend IaaS and PaaS. Kubernetes [19], a Google initiative is a hybrid cloud-based solution

merging IaaS and PaaS.

Also, with the advent of new and innovative computing strategies, the researchers have further classified the cloud services into Hardware-as-a-Service (HaaS) and Data-as-a-Service (DaaS) [20–22]. A very recent of new cloud delivery model is Metal-as-a-Service model (MaaS). It is provisioning construct designed to support the deployment and active provisioning of frenzied, high scale computing such as processing of big data applications and services [23].

II Cloud Deployment Models: Cloud computing can be implemented through three models, namely, Public, Private and Hybrid cloud models. Public clouds are suitable when a service provider wishes to make all the resources accessible to the users over a network. Private clouds, a type of proprietary computing architecture, are suitable when an organization wants to maintain control over its data. Business firms such as eBay and HP CloudStart have deployed the private clouds. The hybrid deployment model combines both public and private cloud models as well as local infrastructures. Vendors such as HP, IBM, Oracle, and VMware are striving to create and utilize a hybrid environment for an efficient delivery of services [15, 22]. Figure 1.3 shows the different cloud deployment models along with their potential service vendors.

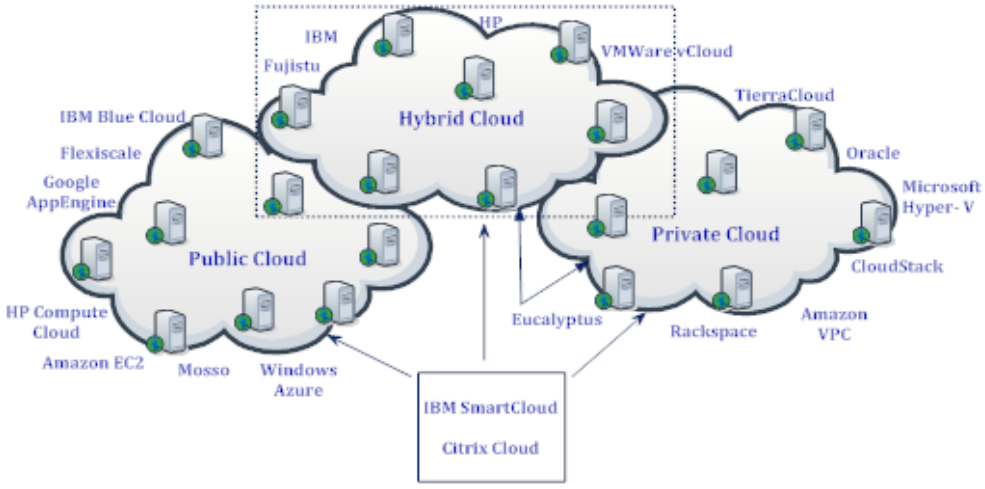

Figure 1.3: Service Vendors Offering Different Types of Clouds

III Cloud Entities: Cloud computing involves three principal entities or actors each having a distinctive role and interaction with the cloud environment (Figure 1.4) [24]. Some cloud models also consider an entity, "Cloud Auditor" depending upon the type of cloud model being practiced.

- Cloud Users: The cloud users correspond to the customers that are using and accessing cloud applications. These may be the users accessing a software

hosted on cloud or the application developers utilizing the cloud infrastructure for hosting their own created applications. Principally, cloud follows utility computing model where the users are billed for their usage of cloud services. Additionally, trust management plays an important role between users and cloud service providers and is governed under specific performance delivery assurances laid out in the form of Service Level Agreements (SLAs).

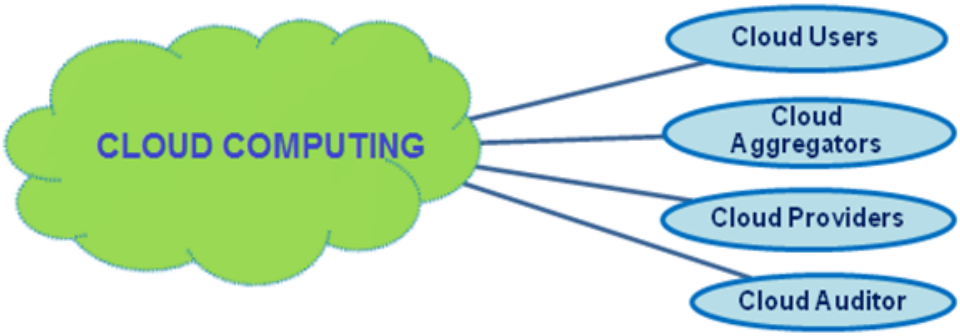

Figure 1.4: Cloud Entities

- Cloud Providers: The cloud providers handle the cloud infrastructure on which the cloud services dwell upon. This entity helps in managing and controlling the cloud resources while handling access to them as per the user requests. The allocation of resources is carried out dynamically and varies during the execution cycle of an application. In other words, the resources are created as per the demand and are finished in case of little usage.
- Cloud Aggregator: A cloud aggregator is an intermediary between the users and service providers and controls the distribution of the user requests amongst multiple cloud providers depending upon the user demands. Basically, it acts as an interface or interaction medium with service providers on behalf of the cloud users. They also play the role of a broker by negotiating SLAs between the users and providers.
- Cloud Auditor: A cloud auditor is chiefly ensures the applicability of proper security controls and analyses the reliability of cloud services . The auditor majorly regulates and verifies the security policies.

1.1.2 Cloud Computing Applications

Cloud computing offers distributed; agile; transparent; seamless integration and delivery of services to the users and thus extends a wide horizon of opportunities for supporting highly pervasive technological progress and innovations. Moreover, by facilitating technological advancements that empower interactions across the globe,

cloud computing generously proffers ample and flawless delivery of large number of services and applications [25].

The capability to host large number of applications that can be delivered to the users speedily at minimal cost has augmented and attracted cloud computing as a favorable paradigm for the IT industry [26]. The cloud services and applications are distributed across multiple domains and are accessible to the users without any location confinements. Consequently, big IT giants such as Amazon, Google, Microsoft, Salesforce, Facebook have significantly adopted cloud computing as principal technological platform for their operation. Owing to the extensive implementation of virtualized servers, consolidation and useful service provisioning, cloud computing has unexpectedly revolutionized and dominated the business and service delivery patterns of the IT applications. The prominent ventures include:

- Amazon Web Services (AWS): AWS is a platform supporting design and development of highly flexible applications that can sustain scalability, storage; messaging across elastic infrastructure vested by Amazon. The prominent compute services include: Amazon Elastic Compute Cloud (EC2), Amazon Elastic MapReduce etc. Similarly the storage services include: Amazon Simple Storage Service (S3), Amazon SimpleDB etc. Other applications include: Amazon CloudWatch, Amazon Simple Storage Service (SES) etc [27].

- Google AppEngine: IT bigwigs like Google largely offers PaaS through its venture, Google AppEngine. Google AppEngine provides services for development and hosting of web applications on the distributed, dynamic and scalable infrastructure of Google. The application developers can develop their applications in Java, Python and Go languages. It constantly monitors the application's usage of resources and services [28].

- Microsoft Windows Azure Platform Appliance: The Windows Azure platform is deployable as an appliance on third- party data centers and constitutes the infrastructure controlling the physical machines in the data center. The prime Windows Azure platform appliance products include Windows Azure, SQL Azure and Microsoft- assisted network; storage; and physical configurations [29].

- Salesforce.com: Salesforce is primarily a SaaS solution providing support for Customer Relationship Manager (CRM) applications [30]. Table 1.1 lists certain cloud computing offerings and their prominent features.

Table 1.1: Cloud Computing Offerings

S.No.	Product	Cloud Service Type	Description
1.	Amazon Web Services (AWS) [27]	IaaS, PaaS and SaaS	A collection of large number of web services facilitating computing, storage and other advanced services especially for IaaS services.
2.	Google AppEngine [28]	PaaS	Supports development of simple and scalable web applications
3.	Microsoft Azure [29]	SaaS through third party vendor, PaaS, IaaS	A kind of cloud operating system enabling development of scalable applications.
4.	SalesForce.com [30]	SaaS, PaaS	A SaaS solution for creating CRM applications vested on Force.com platform.
5.	Heroku [31]	PaaS	Scalable, dynamic environment for creating Ruby-based applications.
6.	RightScale [32]	IaaS	A management platform for cloud providing scalable infrastructural support.

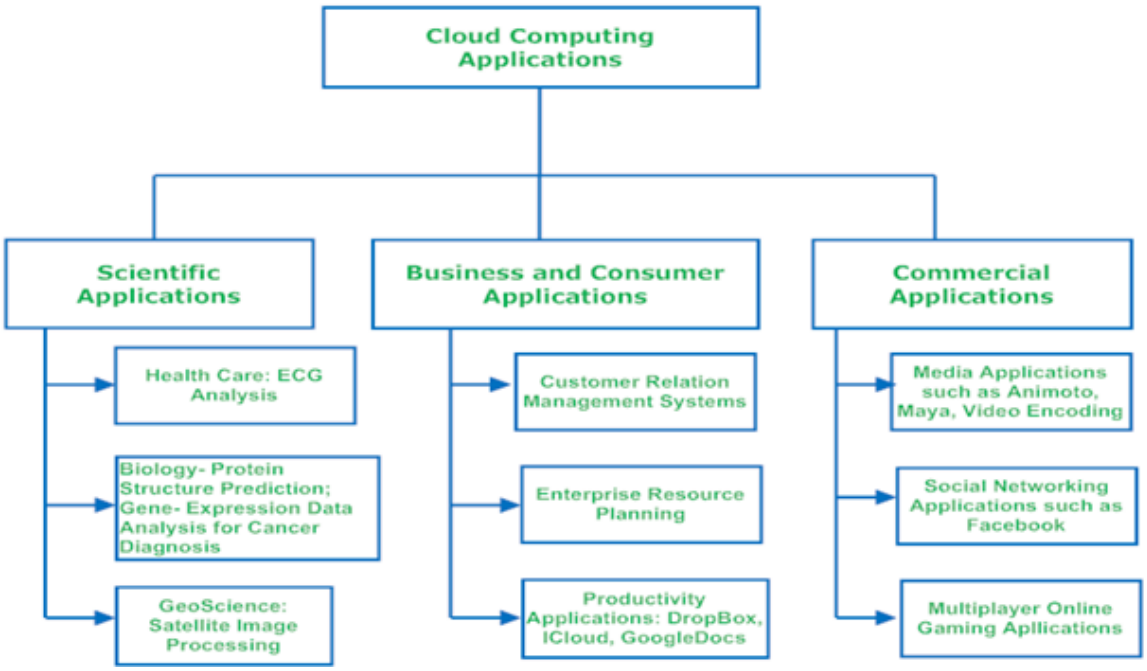

Figure 1.5: Cloud Applications

Cloud computing can lucratively host pervasive applications ranging from scientific,

consumer and business domains [33, 34] driven by its capability to liberally map existing applications to it without much alterations. Accordingly, several cloud-specific applications ranging from scientific to business domains and consumer applications. The scientific applications exploit the scalability and elasticity features of cloud environments that further facilitates higher degree of customization for deploying and executing scientific applications. Also, the economic benefits of cloud computing aid business and consumer applications in reducing or eliminating incurrence; operational; and maintenance costs for the infrastructure. The extended storage and easy accessibility offered by cloud makes it ideal for development of productive applications. Similarly, media applications and social networks benefit from continuous capacity extension and service reliability of cloud computing. Figure 1.5 represents different cloud applications and their subsequent sub-applications exploiting the power of cloud computing.

The increased applicability and dependency upon cloud computing for supple and incessant delivery of services and applications has prompted the IT industry to expand and extend their operational capacities. This has resulted in setting up of big data centers that are performing as service providers for agile delivery of several cloud applications and services. The data centers enable the data and services to be available uninterruptedly and irrespective of the location [35]. A wide range of IT companies, including Amazon, Yahoo, Salesforce, Facebook, Microsoft, and Google, have established their own data centers [15, 36].

However, ever since the growth of the data centers and further advancements in the computing industry have taken place, the consequent high energy demands have been noticed. Nevertheless, the data centers providing cloud services and running several cloud applications have been observed to be high energy consuming centers and also demand huge amounts of electrical energy for operations [37, 38]. This prompted the need for energy awareness within the data centers and energy efficiency. Priorly, it is important to analyze and understand the need for energy efficiency within the data centers from crux as discussed in further sections.

1.2 Energy Efficiency and Data Centers

The data centers established to facilitate scalability and easily accessible cloud services and applications have undoubtedly ensured agile service delivery but have induced high energy demands [39]. A large number of measures to pursue minimal energy demands and realize energy efficiency within them have been made. The following subsections report the growth of energy efficiency in data centers and attempt to investigate the impact of high energy consumption in the data centers.

1.2.1 Rise of Energy Consumption in Data Centers

As the ICT industry matured, infrastructural requirements also changed and gained specialization. Initially, internet servers were de-centrally confined to the office buildings. Subsequently, Internet Data Centers (IDCs) were constructed to centralize

and implement computation within the specialized centers. But, as the IDCs were set up, the power sector witnessed sudden rise in the electricity consumption levels [38, 40–47].

Particularly, the servers hosted in the data centers are the major energy consuming units and also dissipate a lot of heat. Thus, data centers require fully engineered and air- conditioned surroundings for operation [48, 49]. The high energy consumption can trigger performance degradation, causing application saturations and latencies and lowers the overall service delivery patterns [50]. A variety of additional impacts can arise as discussed below.

1.2.2 Impact of Growing Energy Consumption Levels

Increasing energy consumption is accompanied by higher carbon and GHG emissions to the environment. The present energy wastage and CO_2 emission rates are alarmingly high. Figure 1.6 shows the major culprits from different industrial sectors and services throughout the world that contribute to the amount and percentage level of GHG emissions. The values are in million tonnes of CO_2 emissions ($MTCO_2e$). By using these values, the contributory percentage levels have been calculated and displayed [51–58]. The energy sector is clearly the major GHG emitter throughout the world. Table 1.2 lists the contribution values of CO_2 emissions by different countries. The red cell values indicate the highest CO_2 emitting country in a particular year [59].

Table 1.2: Country-wise contribution in CO_2 emissions (values in $MTCO_2e$)

Country → Years ↓	**USA**	**Russia**	**China**	**India**	**United Kingdom**	**Canada**	**Australia**	**Japan**	**France**
1960	2888	888	780	121	584	193	88	233	271
1970	4325	1446	771	195	653	341	198	768	439
1980	4719	2140	1466	314	579	443	221	947	505
1990	5115	2590	2423	619	596	464	278	1156	398
2000	5992	1505	3615	1031	561	572	350	1274	414
2010	5689	1663	9019	1718	508	555	406	1213	389
2015	5414	1617	10357	2274	417	557	400	1237	340

The current onset of the rising power dissipations and the growing GHG and CO_2 emissions to the atmosphere have resulted in several severe climatic changes. Particularly, the growth rate of GHG emissions is constantly posing a danger to environmental sustainability. The contribution of ICT industry in increasing GHG emissions is also growing faster than the overall growth of the emission [60]. The Climate Action group recently mentioned that, “The year 2015 observed about 32

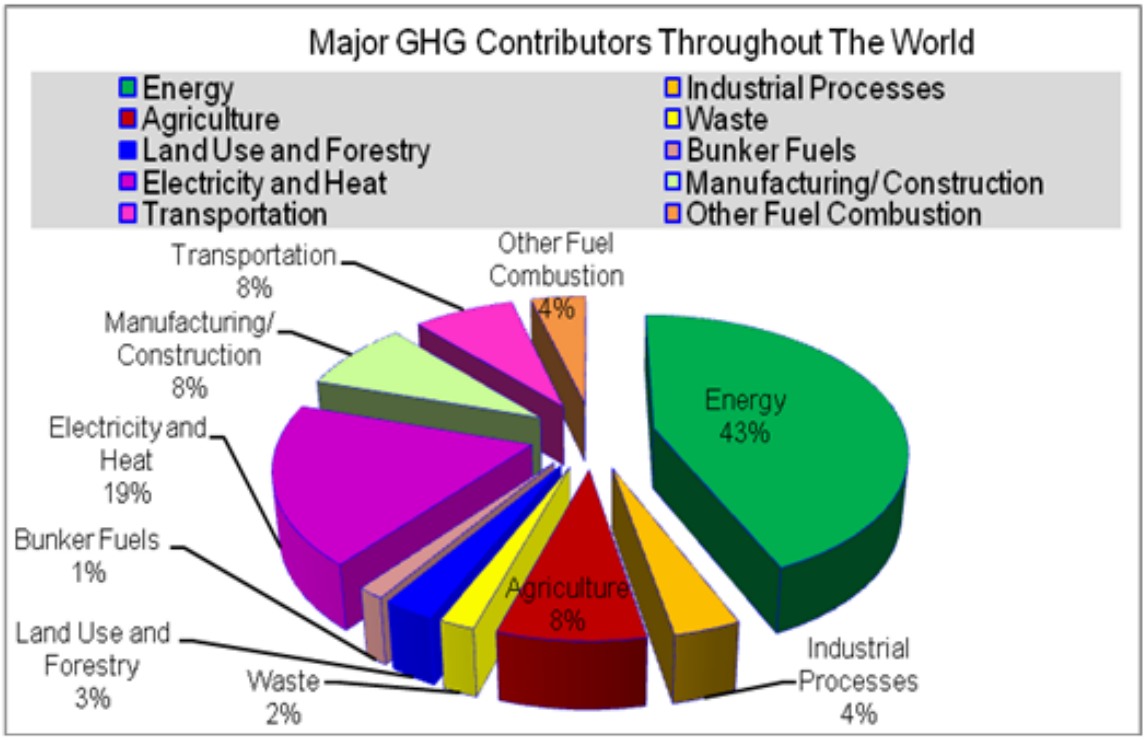

Figure 1.6: Percentage Contribution of Different Industrial Sectors and Services towards GHG Emissions throughout World

gigatonnes of CO_2 emissions across the globe" [61]. In further forecasts done by [62] an increase in the energy demand by 30 percent will occur between 2015 and 2035.

The energy consumption by the ICT industry is usually considered to be power consumption done by computers, equipment, and electrical devices excluding servers. However, it has been observed that the maximal power and thus energy consumed in the ICT firms is not only because of computations by processors and the memory of servers but also due to the large number of servers hosted in data centers. Such numerous servers remain idle for most of the time and leisurely consume power [63]. The server under utilization not only increases energy consumption but also increases over-provisioning expenses and, consequently, the Total Cost of Acquisition (TCA). Different servers operate at random power ranges because of which the power is wasted because of fluctuations. In addition, the rising energy consumption negatively impacts the Total Cost of Operation (TCO) [64–66] where a considerable amount of electricity is wasted in cooling the servers hosted in the data centers. Cooling equipment, such as ACs, fans, pumps, and dehumidifiers, consume almost one-fourth to half of the electricity consumed by the data centers [35, 67, 68].

Predominantly, the costs of delivering the desired energy, cooling the equipments,

maintaining the network connectivity, storage space, and real estate to underutilized servers increase considerably. Additionally, the construction and functioning costs of such huge data centers consumes a big portion of an organizations budget, leaving behind little for various other corporate projects [36, 69]. In other words, besides heightening the demand, the rising energy consumption affects the economic budget of the service providers [65, 66, 70]. Energy-related expenditures of Google are increasing, probably because Google's initial set ups comprised a few high-powered though costly servers. This has consequently posed a problem for Google since it runs approximately half a million servers at approximately a dozen or more physical sites. To decrease operating expenses, Google needs to implement both physical and virtual solutions in its data centers [71, 72].

Energy saving efforts using the latest technological innovations are the only possible means to meet increasing electricity demands and reduce costs. The recent analyses and observations make it clear that if proper implementation of technology, equipment, and control in the data centers is done then the energy consumption and related energy expenditures can be lowered, yielding higher economic savings for the industries [73].

1.3 Conceptualization and Arise of Energy Awareness Through Cloud

The dilemma of the energy crisis has always prompted the necessity to search for the causes of rising energy consumption and seek their elimination. Upon analysis of the high energy trends within computing sector and their visible consistent impacts, the fulfillment of current ICT energy demands through the renewable and non-renewable sources formerly seemed hazy [74]. Later on, due to severity and persistent arise in the energy demands and ever-decreasing levels of non-renewable energy sources, such as coal and fossil fuels, the need to develop certain optimal solutions for managing the growing levels of energy consumption came up [75].

Thus, in such stringent predicament of situations, the ICT energy saving endeavors included search for technological measures to be inflicted and adopted within data centers in order to cope with the energy demands. With the preliminary energy crisis apprehensions, cloud computing at the outset was observed to be principal energy causer but sooner or later, cloud's technical perspectives and strong hold of its implementations, prompted the ICT experts to substantially explore its capability as a possible energy efficiency driver. This section analyses the involvement of cloud in magnifying the energy demands and later on considerable reasons for adopting it as a strategy of obtaining energy efficiency are stated.

1.3.1 Cloud as a Contributor To Energy Wastage

Although cloud computing has revolutionized the manner of information usage and delivery, it is not as always as adept for IT solutions [4]. The successful and reliable

delivery of cloud services requires huge vested infrastructure comprising of high-power servers, air-conditioning systems to cool them etc. The data centers set up to provide agile cloud services require large amount of energy for the operation of vast equipment installed in them. Also, there is significant energy wastage during cloud operations [75] inducing tremendous increase in its consumption, escalating data center ownership and maintenance costs while augmenting carbon footprints and performance issues.

The ever-increasing energy wastage, and higher carbon footprints cause a halt to the performance growth of the data centers [76]. Whenever the equipment installed gets over-heated, the performance degradation can take place. There is also a possibility for generation of wrong calculations due to high temperature rise and fluctuations in the power supplied. Different coolants are used to lower down the rise in temperature and reduce the excessive flow of heat. Using the coolants is risky due to chances of a coolant leak. The leaked coolants can spoil the electronic components. Some tests and repairs are often needed for maintenance which results in the increase in the cost of operation along with cost of using the coolants.

Another factor responsible for posing high energy consumption via clouds is excessive deployment of virtualized resources. By creating a virtualized environment, the cloud providers have tried to overcome the need of vast infrastructure that would have been otherwise needed. Even though virtualized resources offer faster performance and efficiency but it misses out the high levels of carbon footprint generated. Consequently, the data center infrastructure has to provide an optimal support for such an energetic and extensive virtualized environment which actually results in the involvement of more energy [36].

Clearly, cloud computing contributes to the energy wastage issue though without any intent. It generates certain adverse effects associated with energy production and consumption. This provokes the need for in-depth investigation and reforming schemes to explore the role of cloud computing and thereby develop energy management strategies which are not only energy saving but sustainable in the long run.

1.3.2 Cloud as a Solution for Realizing Energy Efficiency

Several research analysts and ICT experts have attempted to thoughtfully and practically explore the role of cloud computing as a solution to instill energy efficiency. Eventually, cloud computing been observed to possess the ability to be more energy efficient compared to the previous computing paradigms [77]. Apart from offering on-demand, broader network access and rapid elastic services to both IT and business sector, it helps to efficiently manage energy wastage problems [78,79]. In due course, ICT vendors have understood the true potential of cloud computing and predicted it as the most preferable technology to overcome the energy crises. In addition, several researchers have investigated the role of cloud computing in realizing energy efficiency. One of a renowned IT expert Jonathan Koomey explored the role of cloud computing and called it as an energy saving tool [40] and can help to overcome GHG emissions by 28% by 2020 [80]. Big firms like Google and Intel have started

monitoring the amount of energy consumed by their PCs, printers, and other equipments installed. Intel is trying to develop a Personal Office Energy Monitor than can monitor the amount of the energy consumed by the installed equipment [81].

However, the key question is what makes cloud computing energy efficient? The answer concerns implementation technologies for the cloud environment. Virtualization is extensively utilized in the cloud paradigm and it helps to abstract infinite number of resources from a large shared collection of the resources, resulting in high resource utilizations and energy savings [80, 82]. The energy efficiency support provided by virtualization and the concept of consolidation (VM consolidation, server consolidation, and task consolidation), facilitated through virtualization, are discussed comprehensively in the section 1.3.3.

Nowadays, besides minimizing energy consumption through the support of virtualization and multi core architectures, cloud computing can be used to achieve energy efficiency through efficient scheduling in the cloud environment. This proficient scheduling helps to improve resource utilization, which, in turn, helps to minimize energy consumption. The role of resource scheduling for achieving energy efficiency is discussed in detail in next chapter. However, it is with the use of underlying perspectives of cloud computing that energy efficiency can be accomplished successfully.

1.3.3 Energy Efficient Cloud: Enabling Technologies

The extensive espousal of virtualized resources, consolidation schemes have embroidered the growth and adoption of cloud computing as a principal technological buzz and advancement in the current IT industry. Cloud computing has emerged as a powerful and preferable paradigm for realizing energy efficiency particularly backed up by the support of [83]:

A. Virtualization

Virtualization provides the abstraction of applications from the underlying hardware and software complexities of the operating environment [84, 85]. In a virtual scenario, multiple OSs (operating on virtual servers) running concurrently on the same hardware platform helps to reduce the number of machines to be installed and lower the associated incurrence, installation, and management costs [25, 86]. Because of the reduction in the number of machines, energy requirement reduces and management becomes simple and easier. The running of virtual servers also minimizes the CPU idle time [71], which helps to further cut down the energy required for operations [87].

Figure 1.7 shows an actual virtualized environment created from the combination of a server hosting some OS and running an application with a virtual layer and multiple OS layers. Virtualization actually shields the working of applications from the infrastructure, thus enabling server sharing among multiple, geographically distributed applications [88]. This helps to improve resource utilization levels. The improved usage of underutilized resources low-

ers the amount of energy consumed that can otherwise consume considerable energy than the completely utilized resources [89–91].

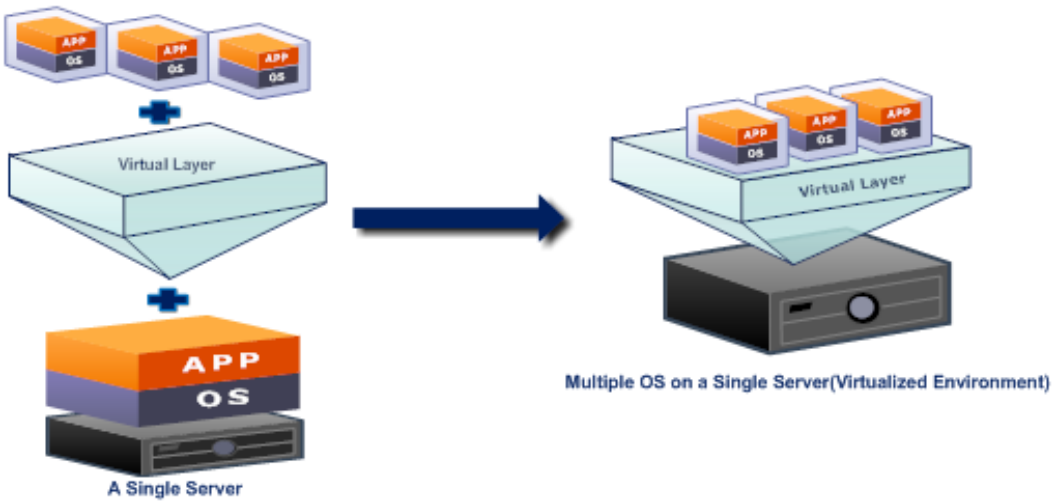

Figure 1.7: Virtualization Scenario

The other key aspects of virtualization that enable energy efficiency include VM migration and consolidation techniques. The virtualized cloud environment minimizes the number of active physical machines through Virtual Machine (VM) migrations. In addition, the VM migrations facilitate workload balancing among all machines and prevent the over-usage of certain machines. The improvements thereby obtained in the resource utility and performance because of workload balancing can reduce energy consumption and carbon footprints [92–94].

B. Consolidation

Besides performing the VM migrations, cloud computing implements VM consolidation depending upon the present resource requirements of VMs [95, 96]. VM consolidation is performed through VM live migration where VMs hosted on the lowly underutilized servers are consolidated onto a single server to prevent the remaining servers from drawing unnecessary electrical power [84, 97, 98].

Apart from this, cloud computing also performs task and server consolidation. Server consolidation helps to enhance the utilization of the resources and decrease the levels of consumed energy [99]. Particularly, server consolidation improves the server utilization levels which further helps in curtailing the energy consumption levels of the servers. Figure 1.8 represents a server consolidation scenario where a single physical server can house multiple server applications or servers. In certain cases, the underutilized servers are consolidated to a centralized location to make data center efficiency measures effective [100].

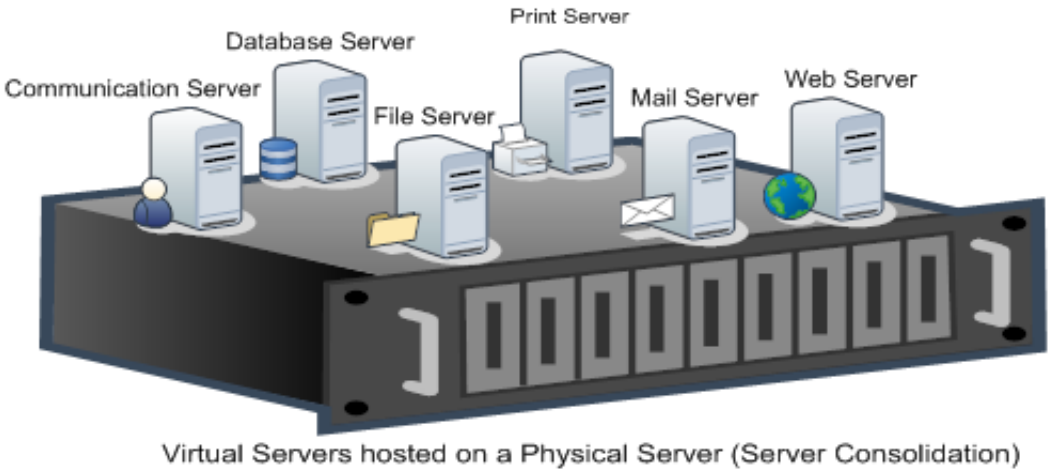

Figure 1.8: Server Consolidation Representation

Through task consolidation, unmatched processing can be avoided and energy wastage can be prevented by enabling servers to execute similar types of tasks and avoid add-on task complexities [101–103].

C. Multicore Architectures

The current computing technology is witnessing a drift toward the use of multicore architectures. The addition of the multiple processing cores on a same chip increases efficiency by facilitating concurrent processing of multiple tasks [35, 104, 105]. Multicore processing expedites task performance by isolating task workload on separate cores instead of using a single-core processor [[106]. As compared with single-core processors, multiple cores provide improved performance by using the same amount of electrical power and cooling load and help to consolidate shared devices on a single-core processor [67]. Moreover, multicore processors help to offset the drawbacks of single-core processors by enhancing data transmission rates and diminishing energy consumed by them [104, 107].

For most of the organizations that operate servers within power and thermal constraints, achieving more performance without increasing energy consumption and temperature of the servers is challenging [108]. The contemporary multi-core processors reasonably consume electrical power and dissipate lesser heat as compared to previous generation multicore processors [104, 109]. Also, the they are engineered to be more energy efficient and thus have a strong ecological impact. Therefore, exploring the role of multicore processors in realizing energy efficiency, particularly in cloud computing, is necessary. Cloud computing platforms can consolidate multiple applications in a server by exploiting the increased computational capacity of multicore CPUs, which can further result in increased overall server utilization and reduced infrastructure

costs [110]. Virtualization is facilitated with the help and support of the multiple cores. The task of running multiple OSs on a single hardware platform is extensively supported by multicore architectures. The dual implementation of the multicore architectures and virtualization techniques facilitates parallel processing and offers energy-efficient cloud computing [109].

1.4 Energy-aware Computing: Research Motivation

The persistent, severe and prevalent environmental changes due to high levels of GHG emissions and carbon footprints in the atmosphere are continuously daunting and endangering the environmental sustainability. The current onset of the rising energy demands and the growing GHG and CO_2 emissions have posed danger to the very survival of humans. With growth of ICT services and applications in digital world leading to growth in energy demand, the energy management has emerged as a principal design restraint for the current computing industry. Presently, ICT data centers are struggling with growing energy demand and energy consumption issues.

The need is to develop an energy-aware technique that can control and minimize overall energy consumption and offer high performance in the ICT data centers. Such endeavors become more important when the renewable and non-renewable sources of energy are already scarce or location-specific and nuclear power for electricity generation being non disposable poses other serious environmental implications. Therefore, the constraints associated with using renewable or non-renewable energy sources and continually increasing energy demands have called for the management of the energy crisis.

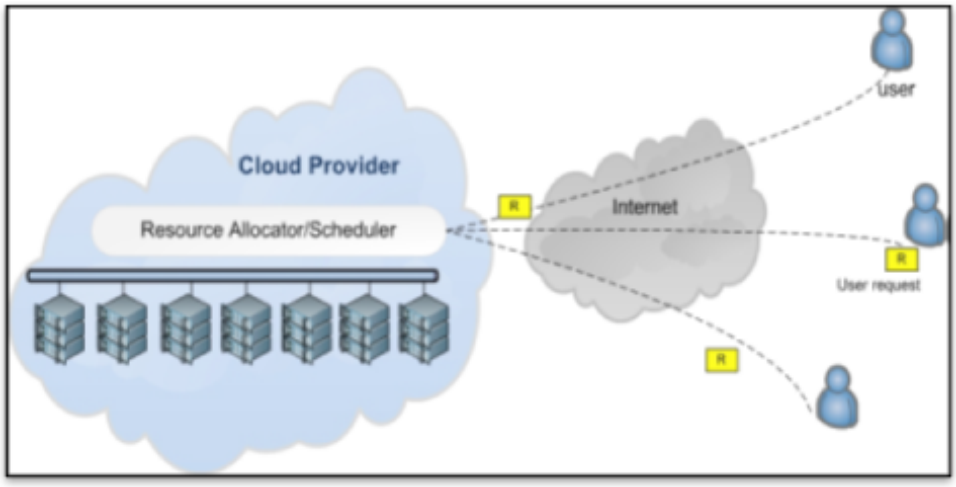

Figure 1.9: Allocation and Scheduling in Cloud Computing

Numerous energy management techniques have emerged as a solution to overcome

the energy crisis. Over the period of time and with consistent growth in the stern energy crisis, the need for energy management and scheduling techniques that can concurrently maximize resource utilization and improve performance has surfaced. Recently, the ubiquitous multiple processing capability extended through multicore architectures also plays significant role. Besides better performance, multi-cores have been observed to be low power consumers and less heat generators. Thus, it becomes highly important to explore their role to achieve energy efficiency especially in cloud computing.

Within ICT, the energy aware management can be done through either hardware optimizations (using energy-star rating equipment, setting data centers in cooler locations or incurring high energy aware servers etc.) or using software management schemes(resource scheduling and provisioning mechanisms). The hardware optimizations are hard for economic sustainability in the long run and require implementation prior to setting up data centers whereas software measures appear favorable in realizing energy efficiency in already vested data centers.

The energy aware allocation and scheduling consists of identification of the energy efficient nodes within the cloud system and assignment of the user tasks to these nodes performed by the resource allocator or scheduler unit (Figure 1.9 [111]). It is important to identify high energy consuming nodes within the cloud system and thwart their surfeit energy consumption for achieving energy efficiency and thereby avert performance squalor. The identification of high energy consuming nodes aids in scheduling decisions where tasks are thereafter allocated to the nodes in the cloud system which are less energy consuming. This keeps the energy consumption within the cloud system within acceptable limits. Consequently, this facilitates high node utilization levels and thus elevates performance degradation issues. In addition, the energy aware task allocation process catering to the user requirements, that is, optimizing the energy demand as well fulfilling the user-specified constraints while thwarting performance and delivering improved QoS is highly beneficial for both the service providers as well as the users.

Expectedly, a QoS optimized energy-efficient resource scheduling technique for cloud computing requires efficient resource utilization and proficient energy usages while keeping the energy costs to minimum. The technique is required to improve the performance of cloud computing by appropriately managing the energy across all the nodes in the cloud system, in turn lowering the carbon emission rates and cooling requirements of the cloud data centers thereby realizing green computing.

Due to the factors stated above, an efficient energy-aware resource scheduling technique in cloud computing has been the motivation behind the work carried out in this thesis. The following section outlines the contributions of the thesis.

1.5 Thesis Contributions

This thesis contributes in the following manner:

- It presents a comprehensive study of the factors leading to the persistent energy

crisis across the globe along with the search for sector-wise energy contribution and their impact.

- An analysis of the possible solutions for overcoming the high energy demand has been conducted and the role of ICT data centers and cloud computing has been investigated. It critically analyzes the evolution of the concept of energy awareness through cloud. It provides a cross-sectional view of the present trends in energy-efficient clouds like virtualization, multicore paradigm, and consolidation.

- A survey and taxonomical study of the state of the art on energy efficiency techniques in the cloud computing has been conducted that also provides pathways for future research in this area. An exhaustive survey of the existing energy efficiency techniques in cloud computing was conducted, and the techniques were compared using different metrics.

- An energy-aware scheduling model for deadline-constrained tasks in cloud computing has been proposed, designed and developed. The proposed model controls and minimizes overall energy consumption and offers high performance and QoS in the ICT data centers.

- The proposed GCSM model performs heterogeneous task allocations for the heterogeneous cloud resources in an energy aware manner without affecting the system performance and within the task deadlines imposed by the users. The model has been experimentally compared with two other techniques and First-Come-First-Serve scheduling scheme too. The obtained results illustrate the efficacy of both the approaches.

- An Intelligent model, termed as, GreenSched model has been designed and implemented. It has a soft computing network based scheduler unit that performs energy aware scheduling of deadline-and-budget constrained cloud tasks.

- GreenSched offers energy efficiency and higher resource utilizations in heterogeneous data center and facilitates autonomic provisioning and scheduling on the resources handling heterogeneous workloads. It is viable and demanding in the present intensely dominating era of distributed computing systems and with the spiraling of heterogeneous Clusters and Grid systems. Moreover, it is favorable for both the users and service providers. From the provider perspective, the minimization in energy consumption averts performance degradation and improves resource utilization whereas from the user context, the execution of the tasks within the specified deadlines and budget fastens the task execution process.

- An empirical evaluation of the proposed GCSM model has been conducted in a public banking firm, Corporation Bank. The case study of the bank for the existing energy measures implemented and its comparative analysis with the proposed algorithm has been conducted. The experimental results demonstrate the efficacy of proposed GCSM model that saves an average of 76.58 % energy with 80%

CPU and 67% memory utility levels. It also achieves 81.3% performance while minimizing the energy and thus accomplishing greener banking operations.

1.6 Thesis Outline

This thesis is organized into six chapters. Chapter 1 as already discussed above provides an introduction to both cloud computing and energy efficiency concepts. It indicates how the sustainability through energy efficiency is becoming increasingly important for cloud data centers and highlights the transformation of cloud from being an energy contributor to being an energy efficiency driver for the IT industry. It showcases the need and growth of energy efficiency in data centers and highlights the rise of the Green Cloud Computing concept. Chapter 1 derives from:

- Tarandeep Kaur and Inderveer Chana, "Energy Efficient Cloud: Trends, Challenges and Future Directions", International Conference on Next Generation Computing and Communication Technologies 2014 (ICNGCCT, 14), Dubai, UAE.
- Inderveer Chana and Tarandeep Kaur, "Delivering IT as a Utility- A Systematic Review", International Journal in Foundations of Computer Science Technology, Volume 3, No. 3, May 2013. 10.5121/ijfcst.2013.3302

Chapter 2 presents an exhaustive and comprehensive study of the existing energy efficiency techniques in cloud computing. The literature review also expands into comparative study of the state-of-art techniques. It presents taxonomies for categorization of the different technologically variant energy efficiency techniques. Chapter 2 partially derives from:

- Tarandeep Kaur and Inderveer Chana, "Energy efficiency techniques in cloud computing: A survey and taxonomy", ACM Computing Surveys, Volume 48, No. 2, Article 22, 2015. http://dx.doi.org/10.1145/2742488, SCIE (IF- 6.7).
- Inderveer Chana and Tarandeep Kaur, "Resource Scheduling Techniques in Utility Computing- A Survey", International Journal of Systems and Service-Oriented Engineering (IJSSOE), IGI Global, USA, Volume 4, No. 2, pp. 44-65, (2014). http://dx.doi.org/10.4018/ijssoe.2014040104 (IF- 0.37).

In Chapter 3, an energy aware cloud-based model termed as Green Cloud Scheduling Model (GCSM) has been proposed. The proposed GCSM implements a scheduler unit called as Green Cloud Scheduler that makes task allocation and scheduling decisions considering the energy aware capability of the heterogeneous cloud nodes and also considers the nodes that can fulfill the task completion time limits as imposed by the users. In other words, GCSM not only achieves energy efficiency but also tends to offer desired QoS to the users in terms of task deadline metric satisfaction. The experimental results predict that GCSM achieves higher energy savings and large number of tasks are completed within the deadline constraints as imposed by users. The contents in Chapter-3 derive from:

- Tarandeep Kaur and Inderveer Chana, "Energy Aware Scheduling of Deadline-Constrained Tasks in Cloud Computing", Cluster Computing, Volume 19, No. 2, 2016, pp. 679-698. http://dx.doi.org/10.1007/s10586-016-0566-9- SCIE (IF-2.04).

- Tarandeep Kaur and Inderveer Chana, "Energy Conscious Allocation and Scheduling of Tasks in ICT Cloud Paradigm", Advances in Intelligent Systems, Computing, Vol. 409, Proceedings of International Conference on ICT for Sustainable Development (ICT4SD, 2015), Ahmadabad, Gujarat. Springer International Publishing. Retrieved from http://www.springer.com/in/book/9789811001277aboutBook https://doi.org/10.1007/978-981-10-0135-2_57

- Tarandeep Kaur, "Energy Aware Scheduling of Deadline-Constrained Tasks in Cloud Computing", Grace Hopper Conference of Women in Computing (GHC), Poster Presentation, Houston, Texas, USA (2015).

In Chapter 4, the proposed GCSM has been optimized to deal collectively with QoS metrics such as deadline and budget constraints while maintaining the energy efficiency. An intelligent energy aware task allocation and resource provisioning technique running in GreenSched model has been proposed. The GreenSched model tends to exploit the heterogeneity of tasks and multi-core capacity of the varied nodes in the cloud environment and attempts to proactively schedule deadline-and-budget constrained tasks to only energy aware nodes. It implements a Forward-only Counter Propagation Network based intelligent scheduler unit that runs a scheduling technique to identify the best nodes with least energy consumption values for the task allocation. The experimental results exhibit that the proposed technique reduces the energy consumption along with an overall improvement in the performance while meeting the deadline-and-budget constraints imposed by the users. This chapter is derived from:

- Tarandeep Kaur and Inderveer Chana, "GreenSched: An Intelligent Energy Aware Scheduling For Deadline-and-Budget Constrained Cloud Tasks", Simulation Modeling Practice and Theory, Elsevier, 82, 2017, pp. 55-83. https://doi.org/10.1016/j.simpat.2017.11.008- SCIE (IF- 1.9).

- Tarandeep Kaur, "IntelGreen- An Intelligent Model for Allocation and Scheduling of Tasks in Energy aware Cloud", Grace Hopper Conference of Women in Computing (GHC), Poster Presentation, Orlando, Florida, USA (2017).

Chapter 5 presents a case study conducted for the empirical evaluation of the proposed GCSM model in a banking firm and the corresponding results obtained.

Chapter 6 concludes, summarizes our major contributions and explores perspectives, challenges and future work directions.

Chapter 2

Literature Survey

The previous chapter conceptually introduced cloud computing, its service and deployment models, entities and cloud applications. Besides discussing cloud concepts and entities, it presented an in-depth investigation into growth of cloud data centers and consequent increase in the energy demands and their impact. It highlighted the role of cloud computing in energy management and the supporting technologies aiding energy efficiency through cloud as a feasible solution.

Owing to the extensive implementation of virtualized servers, consolidation, supple and useful service provisioning, cloud computing proves as favorable paradigm to realize energy efficiency. The ICT firms adopted several energy management measures before drifting towards green cloud. The initial measures included fulfillment of energy demands through scarcely and restrictively available renewable/non-renewable energy sources. Later on, the emphasis was laid on supplying adequate energy to the implementation of high efficiency and low energy consuming hardware equipments in the data centers. Under such initiatives, greener hardware-oriented paraphernalia were vested within the data centers but proved insignificant due to high incurrence, alteration and operational costs. Consequently, software-optimization measures such as resource management and scheduling schemes gained prominence for realizing energy efficiency.

This chapter comprehensively and comparatively analyses and reviews the preliminary data center energy management measures (in Section 2.2 and 2.3). Initially, section 2.1 expansively investigates a few recent surveys encompassing similar topics. The existing surveys in this field have stressed on hardware-based optimization methods while the prime focus of survey conducted in this chapter is to explore existing software-oriented, cloud-based energy-aware techniques. Accordingly, section 2.4 unfolds the role of resource provisioning and scheduling in cloud computing for accomplishing energy efficiency. Section 2.5 discusses the existing software-oriented energy efficiency techniques in cloud computing. In section 2.6, the techniques have been classified and categorized to clearly indicate their levels, mechanisms and performance metrics while section 2.7 provides taxonomies for them. Section 2.8 pens down the objectives of the thesis. Finally, section 2.9 concludes the chapter.

2.1 Notable Research Reviews on Worldwide Energy Crisis

With ICT sector rigorously attempting to overcome the complex and stringent task of meeting their inexorable energy requirements within data centers, the persistent growth in scale and intensity of ICT data and applications is continuously adding to the energy demands. As per a recent survey conducted in 2016, The total world energy consumption will grow by 48 percent between 2010 and 2040, and ICT industry will be the major culprit. The report forecasted 34 percent increase in worldwide energy-related CO_2 emissions from 2020 to 2040 [60]. A large number of initial predictions on the problems and levels of energy consumption in the data centers have been made in some notable research studies conducted in [41, 44–46, 112, 113]. Amongst these, [113] offers a broader overview of the energy trends in the ICT sector.

The mid reports on energy crisis have also been analyzed. A prediction made in 2008 by Smart2020 [114] states, the overall emissions from all sources will increase by 30 percent, while the emissions by the ICT sector will reach up to 180 percent from 2002 to 2020 [115]. Also, a notable mid report on the energy-related expenditure has been conducted in [71] and concluded that almost fifty percent of data center expenditure is spent in using the cooling infrastructure [[71].

In continuity, the studies presented in [76, 116, 117] indicate the current trends in energy consumption in data centers and present a detailed review on the energy crisis troubling the world community. A large number of energy optimizations have been made by the IT industry focusing on minimizing energy crisis and attaining high performance and profits.

A recent measure to stabilize the service provider budgets and profits while constraining energy crisis, countries like UK introduced a Climate Change Act to set up a Low Carbon Transition Plan in order to realize carbon budget targets. The plan aims at achieving 34% energy savings by 2022 [118]. Recently, other countries like Australia, Ireland etc. have initiated Carbon tax or Carbon pricing scheme. The objective of levying such scheme is to devise regulations for the use and release of carbon and thereby restrict the carbon emissions to the environment [117].

The next section highlights several other prominent and contemporary energy efficiency measures implemented in the data centers.

2.2 Preliminary Energy Efficiency Measures in the Data Centers

Several efforts have been made to promote energy consciousness in ICT data centers. Upon analysis, the energy optimization measures in the data centers have been classified as location-based, hardware-based, software-based, and infrastructure-based optimizations. Figure 2.1 shows different optimization measures that have been taken at multiple levels for energy efficiency in the data centers.

2.2.1 Infrastructural Changes and Location Choices

Earlier, the power distribution in the data centers was improved by distributing the power reliably and efficiently among the data center equipments, which consumes power from Power Distribution Units (PDUs) and Power Management Modules (PMMs). Improvements in electrical power distribution could help to reduce energy consumption. Thus, attempts were made to improve flow of current and minimize the voltage drop, which causes undesired energy losses, in the PDUs and PMMs. In addition, managing electrical power distribution among the equipment included avoiding multiple alterations between alternating current and direct current. The key concept behind this was to perform a single conversion at a data center instead of performing multiple conversions at different servers. A single conversion prevents electrical power and thus energy that otherwise occurs while doing multiple alterations. The use of energy-efficient equipment within the data centers was also emphasized [41].

The initial attempts included the installation of the strict energy star rating-based non-IT equipment, such as lighting, and IT equipment, such as adapters, personal computers, mobile phones, keyboards, and monitors, in data centers [76]. Other efforts to minimize the consumed energy in the data centers were achieved through infrastructural changes, where emphasis was laid on constructing green buildings. These buildings consisted of flooring with perforated tiles and a raised floor plenum for cool air intake. The air-conditioning and cooling equipment, such as air-conditioners, fans, and pumps, are used for circulating cold air. However, the maintenance of internal data center temperature alone was insufficient. The problems associated with air flow inefficiencies (such as hot aisles and cold aisles) or the necessity to install extremely intelligent controllers to improve cold air delivery proved expensive for the data center managers [115, 119]. Thus, location dependencies involved in using air-cooling systems were overcome by developing strategies for the placement of high-power servers. The servers were stacked on racks by using different rack arrangements such as 2,-4 post open, perforated, or enclosed schemes [120]. The racks were further positioned in proper air-conditioned rooms to cool the placed servers.

The recent efforts toward energy optimizations have been made by the big ICT giants such as Microsoft, IBM, and Google in choosing favorable sites for their data center establishments. The location choices have been based on setting up the data centers near the locations with inexpensive and readily available hydro electric power and external temperature conditions around the data centers. Google built a huge data center on the banks of Columbia River in Pacific Northwest, in proximity to cheap and readily available hydroelectric energy [71] with complementary weather conditions.

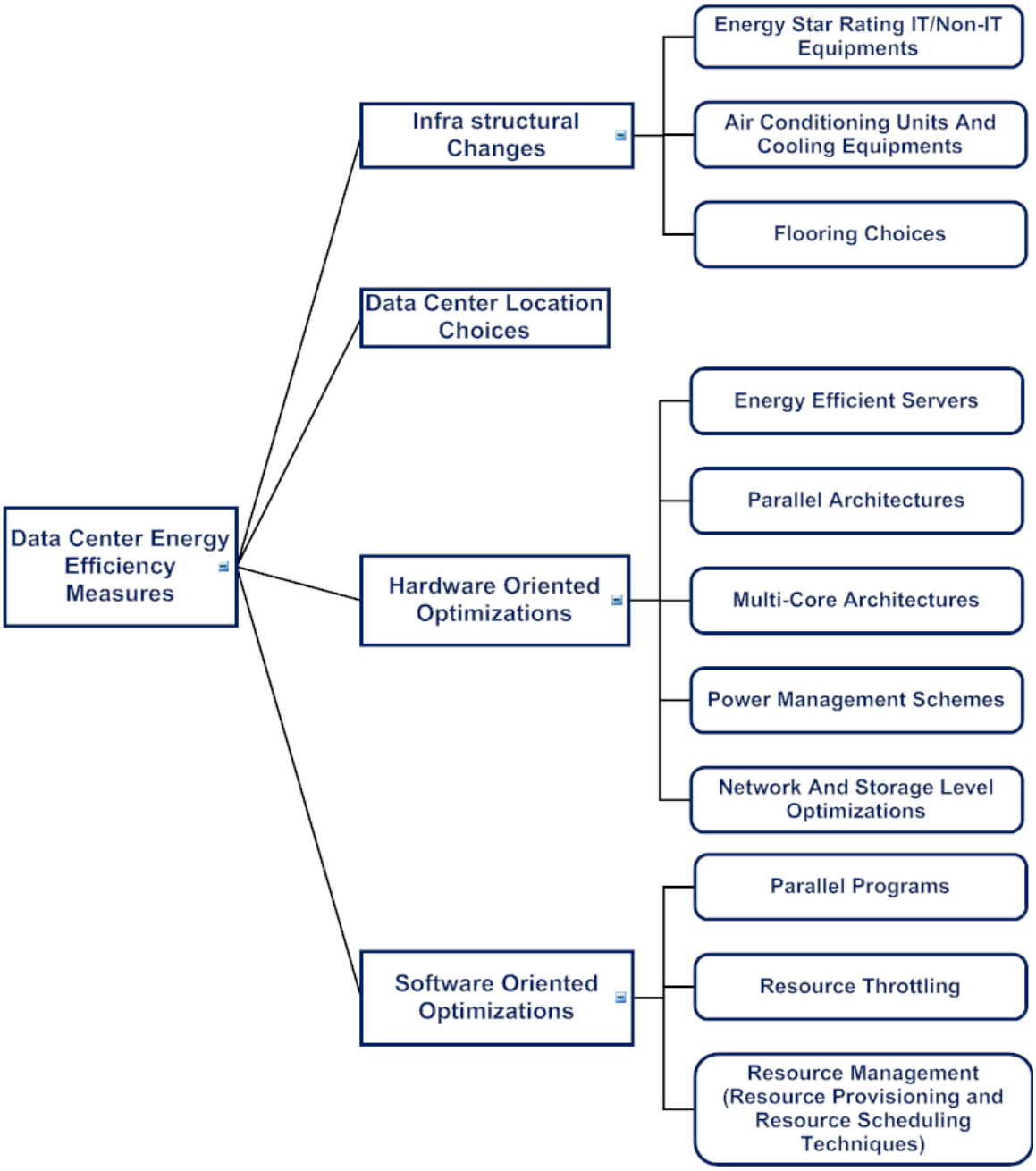

Figure 2.1: Energy Management Measures at Different Optimization Levels in Data Centres

2.2.2 Hardware-oriented Optimization Measures

Despite the infrastructural alterations, energy consumption levels have not dropped. Consequently, the focus has shifted from building new green data centers to energy efficient management of the already established data centers. Thus, new techniques incorporating energy management have been devised as re-locating and re-establishing previously established data centers is difficult task [121]. Subsequently, data center operators implemented various hardware-based energy optimizations, such as using energy-efficient servers, running multi-core and parallel architectures, utilizing power management and scaling techniques, and minimizing the energy consumption of network and storage components (such as solid-state hard drives). The energy-efficient servers included Cooperative Expendable Micro-Slice Servers (CEMS) [122], Fast Array of Wimpy Nodes (FAWN) [123–126], Amdahl blades [127], and MicroBlades [128].

Different optimizations at the circuit, chip, and architectural levels have been executed. The new chip designs that reduced standby electric power consumption by almost 90% were devised [42, 129]. The circuit-level optimizations consisted of controlling and reducing the switching movement of the logic gates and transistor-level combinational circuits, while the logic-level optimizations involved controlling the switching activity of logic-level combinational and sequential circuits. The architecture-level optimizations initially analyzed the system design and then implemented power and energy optimization techniques [76]. However, energy efficiency has not soared regardless of using ideal hardware and hardware-based energy efficiency approaches. Moreover, the cost of incurring the high efficiency hardware rose and consequent operational costs also negatively impacted the financial budget of the service providers [65, 66].

2.2.3 From Hardware to Software-oriented Energy Management Measures

Predominantly, the attempts to fulfill the rising energy demands especially through hardware- based measures or; location alterations etc. fiscally hinders the service providers [37, 47]. The burden related to the monetary and operational constraints associated with hardware-optimization measures prompted the experts to search for other effective measures. In view of that, the researchers observed that energy consumption is not only limited to the efficiency of the physical resources but also depends on the resource management strategies.

Also, the drawbacks associated with operation of hardware resources were attributed to the poor software designs and programs [76]. As a result, software designers begun writing parallel programs by creating codes that could run faster [130]. A technique called resource throttling has also been used. The throttling process controls and monitors the server resources to enhance resource utilization and statically or dynamically regulate the application processing rate [131, 132].

Eventually, many ICT vendors have taken momentous steps particularly implementing software-based measures such as developing resource management poli-

cies and scheduling algorithms to curb the energy wastage and achieve energy efficiency [50,133]. The software-oriented energy-efficient algorithms involved effectively provisioning resources while accommodating available workload and performance requirements [76].

Even though the technological platforms progressively evolved, the attempts to minimize the energy consumption continued with the researchers implementing energy-efficient mechanisms in their respective domains. However, cloud computing gained attention when cloud implementations resulted in increased server utilization and efficient infrastructural organization. Consequently, the lower energy demands of the cloud environment attracted several researchers to foster the development of energy-efficient techniques through cloud computing [40]. The variable number of energy optimizations in cloud computing either at the levels of OS (in a single server), storage, resources (within multiple servers), or applications have proved beneficial in minimizing the energy consumption.

2.3 Energy Efficiency Measures Facilitated Through Cloud Computing

The attempts to simultaneously coerce cloud computing for energy consciousness [65] have been and are still being constantly made but beforehand, the sustainability of cloud computing has been properly analyzed and ensured. This is because cloud computing with an increasingly invasive frontend devices operating across the globe and communicating with backend data centers can pose extreme energy demands. For this reason, the resources within cloud are required to be provisioned and managed appropriately for energy efficiency [33,38]. Subsequently, a large number of resource allocation and scheduling techniques for accomplishing energy efficiency have been proposed so far. A very recent work presenting the advances in the resource allocation and scheduling in cloud environment is given in [134].

Also, detailed studies on energy optimizations at multiple levels in cloud data centers have been conducted by Beloglazov et al. [76], Jing et al. [135], Mastelic et al. [65] and Orgerie et al. [136]. These studies are the pioneering surveys of energy efficiency in cloud computing. However, both surveys in [76] and [135] have extensively explored energy efficiency techniques based on hardware optimization schemes. The techniques presented by [76] provide a review and taxonomy of optimizations, such as power management schemes, though the energy efficiency techniques implementing the virtualized cloud environment have not been studied in detail. The survey conducted in [65] has emphasized on the energy efficiency of the ICT equipment such as servers and networks and deeply analyzes energy of the infrastructure supporting the cloud. Similarly, [136] reviews the existing techniques and corresponding solutions for improving the energy efficiency of the computing and network resources while unfolding the techniques operating at the infrastructural level for inducing energy efficiency.

2.4 Significance of Resource Provisioning and Scheduling in Accomplishing Energy Efficient Cloud

Efficient resource allocation and scheduling techniques are essential in accomplishing energy efficiency through cloud computing [86]. The scheduling of the incoming service requests and provisioning them to resources for execution are very crucial tasks performed by resource management systems [134]. The scheduling process actually consists of allocating the system resources (VMs, network components, CPU, memory, and storage devices) among the service requests. Meanwhile it ensures satisfaction of performance while simultaneously considering global system constraints, such as thermal envelope and energy budget. A proficient allocation and scheduling technique should track the status of physical or virtual resources in the cloud environment for requirement-based resource allocation. In cloud computing, with the advent of the multicore era and the virtualization support, the role of scheduling has been simplified by providing more choices for resource provisioning and scheduling. These multicore architectures pioneer firmly attached resource sharing, which the scheduler can easily manage [137].

The scheduling of the computational resources helps to optimize the execution of a program and initiates the efficient infrastructure usage. The scheduling strategies enable to handle any service request within specified QoS constraints and help to achieve higher resource utilization levels and thus process multiple jobs in small time [138]. Consequently, improvements in resource utilization help to lower energy demand. In other words, the amount of energy consumed and the level of resource utilization are closely related. This is evident from the fact that the resources if under-utilized or over utilized demand more energy. Thus, the optimal utilization of the resources improves energy consumption levels. This is because the resources that are under-utilized stay idle for a long time, doing nothing, withdrawing electrical power at leisure, while the resources that are over utilized operate beyond their capacities and require more power for handling the workload available on them. Underutilization prevents a node from performing optimally, incurring idle time, whereas overutilization causes a node to function more, thereby, sometimes, degrading the nodes performance [101]. The prevention of possible performance degradation and achievement of high performance enhances the profit of particularly public cloud providers and also discerns energy related expenditures [139]. Besides this, for the private cloud providers, this tends to maximize the throughput and thus yield high productivity.

Thus, upon analyzing and considering the benefits associated with efficient resource scheduling and allocation techniques, several such strategies have been proposed. The focus of these techniques is to efficiently allocate and schedule the cloud resources while improvising the utility levels of resources [101]. It becomes imperative to understand and scrutinize the state-of-art-works in the energy efficiency techniques for cloud computing in order to develop some novel techniques that can be an enhancement to the existing techniques or can benefit from the previous tech-

niques. The existing literature is discussed in detail in the following section.

2.5 Energy Efficiency Techniques in Cloud Computing

A number of energy-efficient resource scheduling techniques have been developed to prevent resource under-utilization, which is possibly responsible for incurring high energy consumption [101]. Many of these techniques have collectively emphasized on meeting QoS requirements apart from ensuring energy efficiency.

As per the analysis performed, the existing techniques have been classified into seven main categories. Figure 2.2 shows the categorization and high-level taxonomy of existing energy efficiency techniques in cloud computing. The high level taxonomy shown in the Figure 2.2 groups the existing techniques into seven categories, namely, Virtualization-based techniques, Multicore-based techniques, Consolidation-based techniques, Power-aware techniques, Thermal-aware techniques, Bio-inspired techniques and Miscellaneous techniques.

Figure 2.2: High Level Taxonomy of Energy Efficiency Techniques in Cloud Computing

Basically, the categorizations is based on following criteria. The techniques

that are highly involved in accomplishing energy efficiency using VM allocation, scheduling or migrations are grouped under "Techniques based on Virtualization". The techniques that focus on implementing memory- based energy-aware scheduling or explore the role of multicore architectures are grouped under "Multi-core architecture-based Techniques". Likewise, the techniques incorporating consolidation (either server-based, task-based or VM-based) consolidation mechanism are listed under "Consolidation-based Techniques". Broadly, the techniques that involve voltage scaling and frequency scaling are clustered under the "Power aware Techniques". The techniques that involve manipulations in the external environment, heat- based analysis, thermal arrangement and cooling equipment alterations are listed under "Thermal-aware based Techniques". The techniques that involve implementation of bio-inspired algorithms for the energy management form the "Bio-Inspired Techniques" class. However, certain techniques that are based on miscellaneous algorithms are grouped under "Miscellaneous Techniques" category. The detailed description of all the techniques is as following:

2.5.1 Virtual Machine Allocation and Scheduling based Techniques

Virtualization is an essential and enabling technology of cloud computing. The biggest advantage of a virtualized cloud environment is the portability of high-level functions and the sharing and aggregation of actual physical resources [140]. [141] proposed an EnaCloud approach that provides dynamic and live placement of applications, and it is simulated as a bin-packing problem after considering the energy consumption of the applications. An energy-conscious algorithm has been used for efficiently and heuristically scheduling applications [142, 143]. The VMs encapsulate the applications, and VM live migration have been implemented for reducing energy consumption [144].

The authors in [82] have designed a resource provisioning scheme for achieving energy efficiency that uses centralized online clustering, followed by cluster-based VM allocation and resource organization. All jobs with similar requirements are grouped together and submitted for execution to the same host system, while any unused subsystems or other components are turned off. The energy consciousness has been added by minimizing over allocation and reallocation costs.

Ding et al. in [145] have proposed an algorithm for dynamically scheduling VMs on heterogeneous cloud infrastructure. The VMs are allocated priorly to the Physical Machines (PMs) having high performance-power ratio to save energy. The performance-power ratio prioritizes PMs within the data center. It can also support Dynamic Voltage Frequency Scaling (DVFS) method. The scheduling process is divisible across equivalent periods over which cloud is reconfigured.

An algorithm in [146] devises a bi-objective VM speed concerned energy-efficient task scheduling for heterogeneous cloud environment that attempts to efficiently lower the energy consumption and cease task processing within its deadline. It attempts to restrict energy consumption by manipulating the speeds at which VMs

operate.

Consolidating VMs helps to reduce the total number of running servers that are sitting idle and drawing electrical power unnecessarily. [147, 148] developed a technique that dynamically consolidates VMs based on adjustable utilization thresholds while satisfying SLAs [149]. It proposes a technique for automatically adjusting utilization thresholds by statistically analyzing historical data collected throughout the lifetime of VMs. The technique optimizes allocation by selecting appropriate VMs to be migrated. Subsequently, it places the selected VMs on physical nodes along with new VMs requested by users.

Liao et al., after considering the SLA constraints, proposed a technique for managing the energy wastage problem. This energy-efficient resource provisioning technique offers energy-optimization through a scheduling algorithm. The technique stresses on user-imposed service quality constraints. It implements an energy estimator, SLA monitoring, VM scheduling, resource monitoring, resource discovery, and task scheduler and applies an algorithmic approach for allocating resources onto least physical machines [150]. Similarly, [151] proposes a multi-objective resource allocation algorithm involving a hybrid of Genetic Algorithm and Particle Swarm Optimization (HGAPSO). It consists of an Euclidean distance based multiple objective optimization and resource allocation to VMs. HGAPSO lowers the data center energy and resource wastage levels apart from preventing SLA violation.

In addition to realizing energy efficiency, optimizing techniques for performance are also necessary. The strategy presented by Li et al. in [152] explores the different features of VM workflow scheduling in private clouds and proposes a hybrid energy-efficient scheduling approach based on pre-power technique and least-load-first algorithm. It aims at finding an appropriate solution for two main problems, the reduction of incoming request response time and workload balancing in a data center. For minimizing the response time, it uses a pre-power technique. In order to conserve the energy and workload balancing, it uses the least-load-first algorithm.

Deore et al. developed an Energy-efficient Scheduling Scheme (EESS) that implements migration, clone, pause, and resume concepts. It uses minimum load distribution; first-come, first-serve; and hybrid energy-efficient scheduling algorithms. The incoming VM requests are routed to the appropriate VMs; however, if an allocated request increases beyond capacity, the scheme uses migration for workload distribution. In case the VM request is platform-or software-based, then VM cloning is carried out [153].

[154, 155] proposed a similar technique, where an optimization algorithm collects statistics and data, particularly energy consumption and CO_2 emission data, of servers operating in a data center. Based on the data collected, the algorithm predicts the server with lowest energy consumption and CO_2 emission. After an appropriate selection, all heavily loaded VMs are moved to the servers with the best CO_2 emission values or the highest computational energy. Server selection is also characterized after the user-imposed SLAs are considered. The optimization algorithms are based on Power Usage Effectiveness and Carbon Usage Effectiveness.

Similarly, the authors in [156] have adopted a workload model and proposed a resource provisioning framework that supports energy-aware cost-based versatile

scheduling positively affecting SLAs and thus aiding service providers. It also handles deadline- constrained VM placement and task scheduling while optimizing energy related costs in holistic manner. It can efficiently handle data parallelism in workloads, thereby managing multiple user requests and can apply customizable optimizations for creating global energy, cost and deadline-aware cloud platform.

Certain techniques are based on the heuristics for achieving lower energy consumption. [90, 148] have developed one such technique based on the reallocation heuristics. It uses a live migration technique for VM migrations utilizing certain reallocation heuristics. These heuristics are based on the current resource requirements by tracking the desired QoS. Reallocation is carried out to minimize certain nodes that are handling the current workload and switch off unused physical nodes to save energy.

Similarly, an algorithm based on heuristics for energy efficiency in cloud computing has been proposed in [157]. By using the traditional computation approach, this algorithm implements an OpenNebula-based cloud, which offers a geographically dispersed cloud infrastructure for decreasing energy consumed and amount of carbon emitted. The scheduler in the algorithm works using a multi-start local search heuristic to reduce energy consumption by dispatching incoming VMs according to their energy consumption values. The Virtual Machine Manager (VMM) maintains the information of hosts and their current hardware usage details as these considerably affect the amount of energy consumed. The information of each host provided by the VMM is used by the algorithm to find the best scheduling option.

A meta-heuristic technique for energy awareness has been proposed in [158]. It involves live migration of the VMs from one node to another using a Firefly optimization technique. It tends to improve resource utilization levels and maximize energy efficiency through optimum migration of VMs. It identifies highly loaded VMs and migrates them to least loaded VMs while maintaining energy efficiency and performance. Another technique proposed in [159] presents a novel scheduling scheme called as PreAntPolicy that follows an efficient prediction model based on fractal mathematics and an algorithm based on an improved ACO to optimize energy and QoS. The prediction model predicts the load trends and triggers scheduler for resource scheduling in order to maintain minimal energy requirements.

The technique [155, 160] presented an energy-efficient resource allocation strategy that reallocates resources at runtime by exploiting the ability of the VM live migration technique. The main idea is to use a heuristic based on consolidation and rearrangement of allocation in an energy-efficient manner. It uses a Traditional Single algorithm to locate the computing nodes with least energy consumption. In the Cloud Global Optimization Algorithm, it first moves VMs from low-loaded servers to heavily loaded servers. This helps to liberate all the underutilized servers that unnecessarily consume electrical energy. Next, it migrates the VMs from old servers to the modern servers. The old servers that have been freed can be then turned off, and their workload is handled by the modern servers, which are more energy efficient as compared to the old servers.

The other two techniques using VM migration are actually capable of handling infrastructure workloads. [161] developed a technique to manage scheduling of VM

instances in the data centers and handle the IaaS workloads. The technique provides an optimized scheduler, OptSched, which uses the stipulated time-span of the requests for the optimization of VMM [161]. The scheduling algorithm focuses on minimizing the number of servers required for servicing any workload and decreasing the Cumulative Machine Uptime (CMU) [162]. CMU in a data center indicates the total uptime of all machines. A reduction in CMU directly reduces the energy consumed by a data center.

Like-wise, the authors in [163] proposed an energy-aware application scaling and resource management algorithm that operates on the IaaS environment, which is actually a type of clustered organization. The main aim of the algorithm and system model is to ensure that least number of operating servers go beyond their capacities or violate their operating regions. It also involves application migrations from low-loaded servers to other servers and setting the low-loaded servers to the sleep state or switch-off mode to prevent them from unnecessary power consumption and thereby reduce the costs involved.

A workload specific technique has been designed in [164] where an Energy-aware Resource Allocation method, named EnReal, for dynamically deploying VMs for scientific workflows has been designed. Firstly, it devises an energy consumption model for cloud applications and then proposes an energy-aware resource allocation scheme for allocating VMs based on the initially devised energy consumption model while supporting scientific workflows.

Recently, a Task Placing and Resource Allocation (TaPRA) Algorithm for energy aware scheduling of embarrassingly parallel tasks such as scientific workloads has been proposed in [165]. It presents a task placement and resource allocation plan to minimize the Job Completion Time (JCT). It further extends to its more simplified version: TaPRA-fast. It is applicable to large number of applications such as distributed database queries, Monte Carlo simulations, and image processing etc.

All VM allocation and scheduling based techniques are shown in Table 2.1. The techniques are compared according to their operating environment, scheduling technique used, and primary advantages and disadvantages.

Table 2.1: Comparison of Virtual Machine Allocation and Scheduling based Techniques

Techniques	Basis	Benefits	Drawbacks	Scheduling Technique/ Algorithm Used
Li et al. 2009 [141]	Application placement is modeled as a bin packing problem	Significantly reduces energy consumption	No cost optimization	Energy aware heuristic algorithm
Rodero et al. 2010 [82]	Uses centralized online clustering	Provides the desired QoS	No cost optimization	Clustering algorithm, followed by energy aware provisioning

Ding et al. 2015 [145]	Defines an optimal performancepower ratio to credence heterogeneous PMs	• Reduces Cost • Promotes heterogeneity • Fulfills deadlines	Ignores performance and power consequences of processor status transitions and VM migrations	Energy efficient scheduling of VMs algorithm (EEVS)
Du et al. 2014 [146]	Minimizes energy with schedule length constriction on VMs	• Supports deadline satisfactions while handling heterogeneity, • Improves resource utilizations	Significantly compromises the overall performance	Energy-Efficient Scheduling with Makespan Constraint
Beloglazov and Buyya 2010a; Beloglazov and Buyya 2012 [147, 148]	Involves dynamic VM consolidation by using adaptive utilization thresholds	• Realizes energy efficiency while keeping SLA violations to as low as 1% • Reduces operating costs	Does not include power consumed by memory and network resources	Reallocation algorithm for VM selection and Modified best fit decreasing (MBFD) algorithm
Liao et al. 2012 [150]	Evaluates the energy consumed by VMs and then chooses those with the shortest response time to fulfill SLA	Supports resource provisioning; that is, energy efficiency and guarantees SLA fulfillment	• No support for VM live migration • Ignores power consumption by the network, I/O devices, and GPU	SLA-based resource constraint VM scheduling
Sharma and Guddeti 2016 [151]	An energy aware, SLA aware VM allocation and migration policy	• Improves resource utilization levels • Offers SLA compliance	Constrained by space limitations	Hybrid of Genetic Algorithm and Particle Swarm Optimization (HGAPSO)
Li et al. 2011 [152]	Multiobjective scheduling on private clouds	• Saves more time and energy • Achieves a high level of load balancing	Slightly complex and challenging to accomplish multiple objectives	Least-load-first algorithm
Deore et al. 2012 [153]	Applies VM migrations for workload distribution and uses a lease management system	• Aims to minimize the number of VMs • Distributes workloads evenly among VMs	High response time and low workload utilization level	Energy-efficient Scheduling Scheme
Quan et al. 2012a; Quan et al. 2012c [154, 155]	Based on the traditional mode of computation for reducing the energy consumed and carbon emissions	• Follows SLAs • Reduces energy consumption and carbon emission	High complexity of implementation and operation	Algorithms for single allocation request and global optimization request

Yue et al. 2013 [156]	Can support multi-user large scale workloads while handling task dependencies, server heterogeneities and server communication bottlenecks	• Follows SLA • Offers deadline, energy-related cost optimizations	Only offers flexibility in non-adaptive search space procedure	Guided Migrate and Pack (GMaP) scheduling & optimization
Beloglazov and Buyya 2010b; Beloglazov and Buyya 2012 [90, 148]	Uses Minimization of Migrations (MM) or Highest Potential Growth or Random Choice policies for selecting VMs for migration	• Works well with a heterogeneous infrastructure and VMs. • Does not depend on the workload type	Lacks implementation on a real-world cloud platform	Optimal Online Deterministic Algorithms, VM Placement Optimization Algorithm, and MBFD Algorithm
[Kessaci et al. 2012 [157]	Based on the IaaS cloud model and multi-start local search heuristic	Offers the desired QoS and minimizes the energy consumed	• More complex and provisioning costs • Does not consider GHG emissions	Multi-start local search algorithm
Kansal and Chana 2016 [158]	An energy-aware VM migration technique for workload balancing	• Supports scalability • heterogeneity • Facilitates load balancing • Improves resource utilization	Requires significant work on robustness of the proposed technique	Firefly Optimization (FFO)- Energy-Aware Virtual Machine Migration (FFO-EVMM) Technique
Duan et al. 2017 [159]	Based on fractal mathematics and improved ACO	• Reduce number of active nodes, • Offers high QoS	Lacks mapping to real-world resource-intensive applications	Fractal Prediction Model and scheduling algorithm
Quan et al. 2012b; Quan et al. 2012c [155, 160]	Focuses on IaaS	Enhances performance	No cost optimization involved	Traditional single algorithm and Cloud global optimization algorithm
Knauth and Fetzer 2012 [161]	Handles IaaS workloads	Efficiently handles the data center and VM heterogeneity and timed instances	Lacks implementation on publicly available workloads	Energy-aware scheduler for infrastructure clouds called OptSched
Paya and Marinescu 2013 [163]	Operates in the IaaS environment comprising a clustered cloud organization	• Applicable to SaaS and PaaS as well as private and hybrid clouds • Applicable to processors using DVFS techniques	• Communication complexity • System not very capable of handling sudden increase in the system load	Borrowed-virtual-time scheduling algorithm, Server application management algorithms, Reallocation algorithm, and Cluster leader algorithms
Xu et al. 2016 [164]	Applicable for scientific workflows	Minimizes energy for scientific workflow executions	Ignores energy oriented resource allocation issues (like network bandwidths, storage and caches)	Migration Based Resource Allocation MBRA(tpz) and Energy-Aware Global Resource Allocation (R)

Shi et al. 2016 [165]	Performs online scheduling of multiple jobs using its own scheduler, OnTaPRA	• Reduces response time • Enhances profit and productivity while improving resource utilization	Non-compliant with minimum budgetary constraints	TaPRA scheduling algorithm running on OnTaPRA online scheduler

2.5.2 Multi-core Architecture based Techniques

New technological innovations for handling energy consumption issues and high heat dissipation problems faced by the data centers can be used both at the chip level and at a much higher scale. Various techniques have been designed for handling rising efficiency, temperature, and energy wastage concerns with the use of multicore processors. Multi-core processors have the advantage of being low electrical power consumers, less heat, generators and offer better performance [35]. Blake et al. [107] conducted a detailed study of multicores and their related elements. The authors focused on understanding the key design characteristics common to all multicore processors. Five essential attributes common to all multicore architectures and tradeoffs associated with them have been discussed. The study analyzes the domains of applications, power requirement, performance achieved, processing elements, memory, and integrated peripheral features of multicore architectures.

Similarly, Wells et al. have comprehensively explored the multicore paradigm with regard to the virtualized infrastructure [166]. Valle et al. used another approach based on the implementation of the virtualized infrastructure to overcome problems with the multicore processors [167].

The technique proposed in [168–170] is based on multi-core processors. This scheduling technique tends to reduce the memory contention and investigates the usefulness of frequency scaling in the energy savings. It operates by sorting each task entering the system in the running queue of the core after analyzing its memory characteristics for effectively co-scheduling the memory- and calculation-bound tasks and to reduce memory contention. In case of only memory-bound tasks, a heuristic that lowers the frequency is used. Another technique that has explored the ability of multi-core architectures has been developed by Kivilcim and Rosing [35]. The technique adjusts to the existing environment through the implementation of a strategy, which adapts to the system as well as workload characteristics and helps to handle variable workloads. It works by predicting the temperature and workload in a system. This temperature forecast data is used by the scheduler for retroactively reallocating the workload. Proactive scheduling shuns halting and migration of the tasks so as to decrease the performance-associated cost.

Hussin et al. [171] addressed the problem of scheduling tasks with variable priorities. The authors have primarily focused on accomplishing energy efficiency in LSDs by balancing the workload to attain maximum utilization levels and manage energy consumption through task execution control. The technique uses a hierarchically designed scheduler for exploiting multiple cores. It considers the priority of

tasks to achieve a higher performance and energy reduction via an active scheduling method. Table 2.2 compares the techniques offering energy efficiency by exploiting the capability of multi-core architectures in cloud computing.

Table 2.2: Comparison of Multi-core Architecture based Techniques

Techniques	Basis	Benefits	Drawbacks	Scheduling Technique/ Algorithm Used
Merkel and Bellosa 2006; Merkel and Bellosa 2008; Merkel 2010 [168–170]	Combines tasks with different characteristics to avoid contention among the tasks	• Can be scaled up to a large number of cores • Helps to reduce memory contention	Memory contention increases with the increase in the number of cores	Timeslice-based multitasking, multiprocessor scheduling
Coskun and Rosing 2008 [35]	Intelligently schedules the idle cycles for achieving higher performance	• Energy efficient and low costs • Reliable	Lacks efficient chip, server, and data center managing schemes	Proactive temperature-aware workload scheduling
Hussin et al. 2011 [171]	Exploits diverse task priorities across multiple diverse processors	• Handles resource heterogeneity • Improves energy efficiency • Robust	Difficult to obtain optimal scheduling decisions with dynamic workloads	Priority-based scheduling scheme

2.5.3 Consolidation based Techniques

Consolidation is an important technique for realizing the energy efficiency through cloud computing, and it achieves energy reduction with the support of the virtualized cloud environment and multi-core architectures. A task consolidation-based algorithm that uses energy-conscious task consolidation heuristics has been proposed in [101]. The energy consumed by a resource is precisely or inherently reduced, and the tasks are allotted for execution without affecting their performance. The approach is to exploit task consolidation to manage resources in small and extended terms. In small terms, the variation in the number of incoming tasks is "energy efficiently" treated by lowering the number of active resources, whereas in extended terms, IaaS providers can set unused resources to the power-saver or switched-off mode. This can alleviate any overhead running costs in a long-term scenario.

A methodology called Data Center Energy-Efficient Network-Aware Scheduling (DENS) has been proposed in [172, 173] for balancing the energy consumption in a data center while considering the network awareness and performance. It operates by distributing jobs between computing servers according to their workload or thermal profiles and communication potential. It receives and analyzes response obtained from data center network components and makes decisions and subsequently takes necessary actions. Based on workload analysis and feedback data collected from the network components, the DENS scheduler evades data center hotspots while

restricting computing servers vital for executing jobs.

A VM consolidation-based technique presented by Salehi et al. in [174] is an adaptive energy management policy that pre-empts VMs to reduce the energy consumed according to user-specified performance constraints and uses fuzzy logic for obtaining appropriate decisions. It also includes preemption of low-priority requests to save energy. Initially, the technique determines whether it can service any newly entered high priority request by preempting other requests or turning on the resources that had been previously turned off.

A VM migration and consolidation algorithm for dynamically migrating VMs based on a multi-resource energy efficient model has been proposed in [175]. It uses a MPSO method based MPSO algorithm for VMs reallocation decisions in order to improve energy efficiency within the data center. Various consolidation-based techniques are compared according to different metrics and the primary differences are listed in Table 2.3.

Table 2.3: Comparison of Consolidation based Techniques

Techniques	Basis	Benefits	Drawbacks	Scheduling Technique/ Algorithm Used
Lee and Zomaya 2012 [101]	Based on energy-conscious task consolidation heuristics	Offers cost optimization	Offers no high performance guarantee	Two energy-conscious task consolidation algorithms, ECTC and MaxUtil
Kliazovich et al. 2013; Kocaoglu et al. 2012 [172, 173]	A hierarchical model that can well adjust with existing data center topologies	• Optimizes energy consumption and performance according to QoS parameters and traffic demands • Minimizes computational and memory overheads	Lacks practical implementation in realistic setups	Data center energy-efficient network-aware scheduling
Salehi et al. 2012 [174]	Based on an energy management policy to reduce energy while fulfilling performance demands	Can perform ondemand switching during starvation	Does not consider the impact of VM migrations and consolidation on a preemption policy	Preemption aware energy management policy (PEMP)
Hongjian Li et al. 2016 [175]	Uses MPSO for consolidating VMs to prevent occurrence of local optima, a common issue in traditional heuristic algorithms	• Reduce number of active machines • Improves resource utilizations while satisfying SLA and QoS	Lacks implementation in real time environment	Modified Particle Swarm Optimization Algorithm (MPSO)

2.5.4 Power-aware Techniques

Most of the aforementioned VM migration- and consolidation-based techniques have focused on providing energy efficiency at the internal logical level. The Multicore-based techniques offer hardware-based energy efficiency optimization. However, en-

ergy consumption at the external level can be reduced using different power-scaling techniques [176].

Laszewski et al. have proposed energy-conscious scheduling for VMs in DVFS-supported clusters [177–179]. The cluster jobs are executed by the VMs in dynamic manner. The scheduling algorithm allocates the VMs in DVFS-based clusters and scales the available electrical power in dynamic fashion. The algorithm implementation is performed on a simulator for DVFS and multi-core cluster [177]. Similarly, in another power-aware scheduling technique, job scheduling is done to reduce the overall power and thus energy of the servers. It focuses on the server's power consumption and operates on a power-aware cluster system, which can support manifold power modes while turning on and off processor frequencies. Thus, the scheduler in the cluster system handles the energy-performance tradeoff [84].

A multiprocessor-based energy-aware scheduling technique in [75,180] presents an algorithm uses a methodology, which is energy-efficient and which schedules various real-time tasks in a Dynamic Voltage Scheme-based multiprocessor system. The indistinguishable multiprocessor platform is used for executing tasks by dividing their execution time while minimizing energy wastage through the application of the DVS scheme. The complete problem is modeled as NP-Hard and solved using a bin-packing heuristic.

A cloud service framework has been presented in [181] for accessing VMs in real-time through a power-conscious VM purveying method. Initially, the method preliminarily investigates all such different power-conscious DVFS-based schemes and then uses a purveying algorithm. This approach models any real-time jobs, such as real-time requests, and helps to purvey VMs by using DVFS schemes.

An energy-aware VM scheduling scheme in IaaS clouds provides energy efficiency through energy-aware VM scheduling. This scheme uses two sub-schedulers for realizing energy efficiency. These schedulers, namely placement scheduler and migration scheduler, work complementarily. The placement scheduler works according to the energy-aware placement scheduling algorithm for distributing the arriving provisioning requests to the currently active nodes or newly activated nodes from the resource pool. By contrast, the migration scheduler attempts to discover any energy-reduction opportunity through live migration according to the energy-aware migration scheduling algorithm [182].

The authors in [183] have implemented and validated a virtual and active resource provisioning framework. Limited Lookahead Control (LLC) methodology has been employed for resolving the resource allocation problem. LLC provides an optimal multi objective solution that can be applied to a nonlinearly active computing system. It keeps track of the costs involved in turning machines on or off; that is, the switching costs involved in VM provisioning. It uses a hierarchical LLC structure for performing operations quickly.

Ahmad and Vijaykumar proposed two schemes for the optimization of idle time as well as cooling power and minimization of the response time. For overcoming the tradeoff between the idle time and cooling power, a PowerTrade scheme has been used, which helps to reduce the overall power consumption; that is, the summation of the cooling power and inactivity power of servers. Another scheme called SurgeGuard

is used to minimize the response time. PowerTrade and SurgeGuard are combined to collectively control both total power and energy and response time degradation [68, 184]. Table 2.4 shows the comparison of the existing power-aware techniques.

Table 2.4: Comparison of Power-aware Techniques

Techniques	Basis	Benefits	Drawbacks	Scheduling Technique/ Algorithm Used
Laszewski et al. 2009 [177]	Based on DVFS-enabled cluster	• Applicable to high-performance clusters • Aims at maximizing system efficiency	Offers no cost optimization	Power-aware scheduling algorithm
Younge et al. 2010 [84]	A greedy-based algorithm to minimize energy consumption	Aims at reducing energy-performance tradeoff	Demands a better data center design with regard to server and cooling system placement	Power-aware VM scheduling algorithm
Berl et al. 2010; Xian et al. 2007 [75, 180]	Based on DVS	• Offers workload balancing • Possesses simplicity and ease of implementation	Considers no task dependencies and offers no cost optimization	Multiprocessor scheduling by using a polynomial time heuristic algorithm
Kim et al. 2011 [181]	Based on adaptive provisioning schemes, Adaptive-DVFS, and -advanced DVFS	Reduces power consumption and increases the profits of a data center	Works independently without considering the workload on the system	PowerTimeShared VM Scheduler
Chen 2011 [182]	Based on VM migrations and power models for CPU- and memory-intensive applications	Implements workload balancing and green sorting algorithm to optimize energy- performance trade-off	Involves several VM migrations, resulting in increased complexity and high operating costs	Energy-aware placement scheduling algorithm, Replacement scheduling algorithm, Energy-aware migration scheduling algorithm
Kusic et al. 2009 [183]	Power and performance management by using a look ahead control scheme	• Manages power consumption levels and maintains QoS and profit levels • Offers better online performance because of low execution- time overhead	Operational complexity	Dynamic resource provisioning scheme by using LLC
Ahmad and Vijaykumar 2010 [68, 184]	Applies overallocations by SurgeGuard to servers available at PowerTrade	Attains energy efficiency as well as reduces the response time	Difficult to track dynamic load variations	PowerTrade and SurgeGuard schemes

2.5.5 Thermal-aware Techniques

Thermal-aware techniques emphasize on implementing certain methods that are not only restricted to reducing heat dissipated by servers but also work for improving

the quality of cooling equipment used in the data centers. One such technique using thermal-aware scheduling strategy has been designed for homogeneous HPC data centers. In this scheduling technique, the temperature of an entire data center is lowered through an effective plan for scheduling the jobs. In general, energy savings are not only limited to the reduction in the energy consumption of a server but also towards reduction in the energy of the cooling infrastructure installed [15, 84, 185].

Another such technique based on the thermal awareness has been proposed by Rodero et.al. The technique proposes a VM allocation strategy to track application information such as CPU usage, memory requirements, and network bandwidth utilization levels. It achieves energy efficiency by consolidating different VMs and employs techniques for analyzing the thermal behavior of a data center to acquire self heat management capabilities in the data center management systems. The technique considers a layered architecture, where the first layer identifies, concentrates, and illustrates the thermal hotspots via some sensing infrastructure. The second layer supervises a servers hardware as well as software parts, and the last layer manages the VMs. The last layer, the application layer, tracks workload and application feature [186].

The need for a thermal management system for reducing the energy consumption prompted Pakbaznia et al. to develop a Power and Thermal Management (PTM) solution. The proposed solution consists of a Temperature-aware Dynamic Resource Provisioning (TA-DRP) module with 2 subunits: a Workload Monitor unit and a Power-Thermal Manager. The PTM makes different decisions regarding active systems. The Workload Monitor unit predicts the incoming workload in the form of requests entering per second through a workload forecasting method, and then implements efficient dynamic strategies for activating other servers in case of heavy workload levels or shutting down servers in case of lower workload levels [187].

The authors in [188] have proposed an adaptive VM placement and consolidation method for achieving energy efficiency. It considers server heterogeneity and low-power SLEEP states of servers, state transition latencies. It also integrates controls for maintaining operational temperature within datacenter. It consolidates VMs between servers based on utilization and thermal temperature constraints. Table 2.5 compares the above mentioned techniques that are based on Thermal-aware strategies for realizing energy efficiency in the cloud environment.

Table 2.5: Comparison of Thermal Management Scheme Techniques

Techniques	Basis	Benefits	Drawbacks	Scheduling Technique/ Algorithm Used
Teng 2011; Younge et al. 2010; Tang et al. 2008 [15, 84, 185]	Aims at reducing data center temperatures	Reduces power for running the cooling infrastructure installed in a data center	Suitable for homogeneous data centers only	Thermal-aware scheduling algorithm

Rodero et al. 2012 [186]	Offers an energy-aware thermal management Solution	• Maximizes resource utilization and satisfies application QoS requirements • Reduces the Total Cost of Ownership (TCO)	• Demands careful consideration of resources • Needs to appropriately deal with undesired thermal behavior and QoS guarantees	VM allocation algorithm
Pakbaznia et al. 2010 [187]	Power saving realized by combining chassis consolidation and cold temperature control	Maximizes overall power consumed by servers and AC units whilst keeping maximum temperature in any server chassis	Ignores power consumption by network components in a data center	TA-DRP algorithm, Server retirement policy, and Server employment policy
Raj et al. 2016 [188]	Based on a StaticPPMMax metric for VM allocation algorithm for ranking servers in IaaS cloud	Prevents performance degradation	Lacks implementation on real-world cloud platforms such as OpenStack	VM placement optimization approach; VM allocation to servers using StaticPPMMaxorStaticPPMMinmetric getNewVmPlacement approach, Power-MinModified getNewVmPlacement approach, Hybrid PPMMax with PowerMingetNewVmPlacement approach, Hybrid PPMMax with maximum utilizationgetNewVmPlacement approach, serverTransitionDowngrade()-Server power state downgrade transition approach, and serverTransitionUpgrade()-Server power state upgrade transition approach

2.5.6 Bio-inspired Computing based Techniques

The rising inclination of the users toward diverse and complex services has led to the development of more scalable, heterogeneous, and sustainable communication systems. The limited extendibility and growing complexity of present communication systems have prompted the need to shift toward novel paradigms that are more capable in equipment design and development. Although scalability, heterogeneity, and complexity have gained importance, various nature-based computing techniques are used to handle increased sophistication and proficiency [189,190]. Thus, bio-inspired computing is a fast-growing novel paradigm in communication and networking.

Many researchers have been working toward using the role of biological systems, such as swarm intelligence systems including Ant Colony Optimization, Glow (ACO), Glowworm Swarm Optimization, Fast bacterial swarming algorithm, and Bee Colony Optimization techniques to manage the energy problems. Several techniques are proposed for achieving energy efficiency in cloud computing by utilizing the abilities of these bio-inspired systems.

Kessaci et al. developed a parallel bi-objective hybrid genetic algorithm (GA),

which uses a hybrid scheduling algorithm based on the multi-purpose parallel GA and energy-conscious scheduling heuristic (ECS). It considers the task completion time and energy consumed as dual objectives to be fulfilled. The scheduling process mainly consists of allocating n different tasks to "p" different processors, by minimizing the makespan and keeping the energy consumption level to low. The voltage scaling technique used is DVS [191].

A multi-task scheduling technique using GA was proposed by Wang et al. The technique exploits the vast processing capacity of the Google network and uses some training and interpreting techniques for network entities. It then constructs an appropriate energy function based on each entitys robustness. The technique tends to optimize the CPU utilization because energy consumption did by each task depend on server CPU utilization that is difficult to handle [192]. Another GA-based technique was designed by Nguyen et al. It proposes a genetic algorithm for power-aware scheduling (GAPA) for resource allocation, and solves the static VM allocation problem by finding an optimal VM allocation strategy [193].

A technique based on Low Carbon Virtual Private Cloud (LCVPC) has been proposed in [194]. This technique lowers the carbon footprint of remote data centers interconnected through personal wide-area networks or the Internet, but operating in multiple areas. An intelligent live VM migration within a WAN was used. The VPC manager first calculates the carbon footprint as well as the energy consumed by the entire network and then applies a modified GA. The technique also uses dynamic VM consolidation for optimizing carbon footprint levels.

Some of the bio-inspired computing techniques are based on ACO as it has been judged to be the best swarm intelligence-based algorithm until now. [195] proposed a technique that provides an energy-efficient resource allocation mechanism. The adaptive allocation mechanism of the technique allocates the resources required by various applications according to their QoS requirements. For reducing the energy consumed by the data center resources, it models ant agents as changing server loads.

The research work carried out in [196] presents a bio-inspired workload consolidation algorithm based on an ACO Meta-heuristic. The proposed algorithm tries to dynamically make decisions regarding the placement of workload according to current workload. It overcomes the workload consolidation problem of single resource applicability and models the difficulty in workload assignment as a multidimensional bin-packing problem.

In [197], the authors combine the benefits of both ACO and Cuckoo search. It is indeed a hybrid approach that reduces the energy consumption by scheduling the tasks in an energy efficient manner and utilizes the voltage scaling factor approach to accomplish energy reduction. The authors in [198] have used Particle Swarm-Optimized Tabu Search Mechanism (PSOTBM), where unused servers are turned off to reduce energy wastage. Initially, PSOTBM allocates computing, storage, and communication resources, and then applies a mathematical scheme for finding an optimal solution based on a PSO-improved algorithm. Furthermore, after making certain improvements in PSO, the authors propose an improved PSOTBM.

Bergmann et al. have proposed a Simplified Swarm Optimization (SSO) method, a variation of PSO, based on distinct standards for reducing energy consumption in

cloud computing systems. The method also uses a DVS technique through which it can optimize the energy consumed without performance degradation. It implements graph-based representations, DAGs, for storing tasks entering a system and detecting the critical paths for effectively implementing DVS. The primary focus of SSO lies in minimizing the makespan and power usage [199]. Table 2.6 comprehensively compares different bio-inspired computing based techniques.

Table 2.6: Comparison of Bio-inspired Computing based Techniques

Techniques	Basis	Benefits	Drawbacks	Scheduling Technique/ Algorithm Used
Kessaci et al. 2011 [191]	Based on multi objective parallel GA and ECS heuristics	Offers deadline based optimization	Focus on multi objectives increases the cost	Parallel bi-objective hybrid genetic algorithm
Wang et al. 2012a [192]	Based on Google's massive data processing network	Implements a load balancing scheme	Longer computational time and no QoS and cost reduction guarantee	Energy-efficient multitask scheduling algorithm
Nguyen et al. 2013 [193]	Operates in private cloud	Minimizes energy consumed by the system	Longer computational time	GAPA algorithm
Moghaddam et al. 2011 [194]	Intelligently and dynamically migrates VMs combined with a heuristic algorithm (a modified GA)	Proposes comprehensive cost function applicable to the entire system	Mainly focuses on minimizing the carbon footprint and ignores performance metrics	Modified genetic algorithm
Chimakurthi and Madhukumar S.D 2011 [195]	Allocates the tasks bound to CPU & memory dynamically	Obeys SLAs, and attains high performance such as a high throughput and response time	Lacks load prediction	Implements a power-efficient, agent-based solution
Feller et al. 2011 [196]	Based on ACO meta-heuristic	Requires fewer machines and attains better resource utilization level compared with other algorithms	• Extreme dependency on workload and resource details • Cannot handle the dynamic nature of tasks	ACO meta heuristic based algorithm, and Energy aware ACO based workload consolidation algorithm
Babukarthik et al. 2012 [197]	Based on ACO and cuckoo search, and also uses a voltage scaling factor	Efficiently reduces energy consumption	Difficult to implement	Hybrid scheduling based on ACO and cuckoo search
Wang et al. 2012b [198]	Particle Swarm optimized Tabu search Mechanism	• Maximizes resource utilization • Minimizes the task finish time • Enhances revenue acquisition	Should improve the pricing function to be more profitable	Resource consolidation algorithm for energy efficient VM consolidation

Bergmann et al. 2013 [199]	DVS and swarm intelligence	• Can plan jobs intended for other shared components	Not feasible for non-time-sensitive systems	Simplified swarm optimization (SSO) method

2.5.7 Miscellaneous Techniques

Certain techniques, which could not be grouped into any aforementioned categories, are listed in the miscellaneous category. The authors in [200] have proposed a technique called the meta-scheduling policies algorithm. The technique gives a precise model for efficient energy use based on several influencing factors, including energy-associated costs, rate at which carbon is emitted, loads of work available, and power effectiveness of CPU. It uses the best possible meta-scheduling policies to lower the rate of carbon emitted and enhances the net revenues of cloud providers with least relocation and infrastructure adjustments.

A Dynamic resource scheduling algorithm for isomorphic nodes designed by [201] is based on the energy optimization of CPU, main memory, and storage. It first establishes an energy model based on the energy consumption of CPU, memory, storage, and network. The algorithm consists of three stages: infrastructure preparation, job preprocessing, and job execution, where all the hardware's, its energy consumption, and adjustment methods are counted. The previous energy-saving measures of unused nodes are counted and they are hibernated to used nodes to count their energy-efficient templates during the job execution stage. After completing the three stages, actual resource allocation and scheduling process begin.

The author in [202] have proposed M/M/1 Queuing Theory Predicting method (MMQMPM), where Linear predicting method (LPM) and Flat Period Reservation reduced method (FPRRM) have been utilized for obtaining important information from the utilization log. The use of LPM and FPRRM helps MMQMPM to achieve a better response time during quick rising period and also reduces the number of reserved resources in a steady sequence.

The authors in [203] have proposed a data center wide energy efficient resource scheduling framework (DCEERS) that schedules heterogeneous resources for performing task allocations while offering data center-wide energy savings. It tends to solve Mixed Integer Linear Programming (MILP) formulation problem through Benders decomposition heuristic in linear time.

An Energy-aware Autonomic Resource Scheduling (EARTH) in Cloud Computing is proposed in [204]. It is an energy-aware framework based on fuzzy logic to carry out automatic scheduling of the data center resources while considering energy as a QoS parameter and maintains SLA. It handles heterogeneous workloads and optimizes QoS parameters such as energy and resource utilization.

Recently, the authors in [205] have proposed an energy-aware “Tree-to-Tree” task scheduling method for the complex data-intensive applications while lowering the SLA violations specifically in terms of deadline satisfaction. The proposed method

maximizes the energy efficiency; reduces the number of active machines within cloud system, optimizes resource utilizations and network bandwidth. It is vital in the current era of rising intricacy and complexity of big data applications. Table 2.7 compares the miscellaneous techniques using certain metrics.

Table 2.7: Comparison of Miscellaneous Techniques

Techniques	Basis	Benefits	Drawbacks	Scheduling Technique/ Algorithm Used
Garg et al. 2011 [200]	Dual-objective technique based on heuristics	Supports the minimization of power consumption by data centers and the optimization of energy costs	Lacks applicability in a virtualized cloud environment	Meta-scheduling policies based on heuristics implemented by a meta-scheduler
Luo et al. 2012 [201]	Based on hardware optimization	Offers efficient energy optimization of CPU, main memory, and storage	• Incompatible with certain open-source cloud projects • Does not offer cost optimization	Round-robin scheduling algorithm
Shi et al. 2011 [202]	Based on MMQMPM	Emphasizes on improving resource utilization level, and thus energy efficiency	Forecasts invalid performance values in case of flat service processing sequence	LTPM-combined algorithm and FPRRM in MQMPM
Shuja et al. 2014 [203]	Based on Benders decomposition heuristic	• Offers deadline and budget optimizations • Handles heterogeneity, scalability	Involves variable throughputs depending on the number of resources	Benders decomposition algorithm
Singh and Chana 2016 [204]	Fuzzy logic based energy aware resource scheduling framework	• Supports automaticity • Improves resource • Enhances QoS while satisfying SLA	• Compromises cost and fault tolerance • Budget and Deadline factors yet under scrutiny	Energy-aware Autonomic Resource Scheduling
Qing et al. 2016 [205]	Based on an offline task scheduling method	• Satisfies SLA • Reduces number of active machines • Improves performance and fulfills deadlines	Compromises budget	"Tree-to-Tree" task scheduling

2.6 Comparative Analysis of Existing Energy Efficiency Techniques in Cloud Computing

The comparative analysis of the existing energy efficiency techniques in cloud computing is important to clearly classify and understand them based on various parameters. This section comprehensively reviews the techniques for parametric categorization and metric categorizations.

2.6.1 Categorization Based Overlapping

Techniques falling into more than one category are depicted using an overlapping region diagram (Figure 2.3). The techniques of [147, 148] and [174] are based on virtualization and consolidation. [147, 148] used the VM consolidation method in their techniques. By contrast, [145], [181] and [182] used both virtualization and power-aware voltage scaling techniques to achieve energy efficiency in their techniques. The authors in Kim et al. 2011 [181] have used hardware-level virtualization; Chen [182] has used resource-level virtualization while the authors in [145] have used voltage scaling technique. In another technique, Rodero et al. [186] combined the virtualized cloud environment and thermal-aware management strategies while the technique presented in [188] is collectively a virtualized, VM-based consolidation and thermal-aware management scheme. Certain techniques have combined

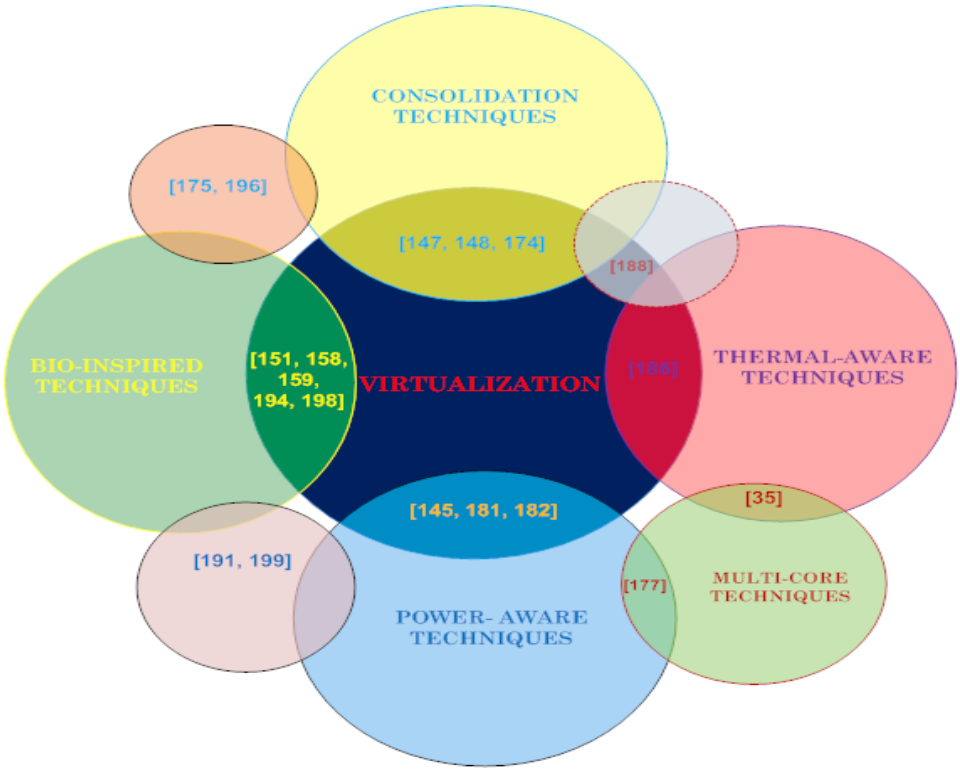

Figure 2.3: Overlapping of the Base Technologies Used by Different Techniques

both multi-core architectures and power management techniques. [177] developed a technique that implemented both the multi-core architecture and a power-aware DVFS scheme. Similarly, Coskun and Rosing [35] designed a technique based on the exploitation of multi-core architectures and thermal-aware management techniques. The techniques of [151, 158, 159, 191, 194, 198, 199] all are based on bio-inspired optimization methods. [191] and [199] also used power-aware management schemes

such as DVS, whereas [194] and [198] used the virtual cloud environment. In techniques [175] and [196], the authors exploited consolidation of workloads along with the utilization of ACO and MPSO bio-inspired methods respectively.

2.6.2 Metric based Overlapping and Classifications

Figure 2.4 represents the comparative analysis Venn diagram for the existing energy efficiency techniques. It compares the existing literature on the basis of the metrics of optimizations implemented in them. The Venn diagram compares the techniques based on energy, deadline, budget and performance metrics. In other words, it shows the similarity and dissimilarity factors between techniques based on above metrics.

Similar to Figure 2.4, Figure 2.5 depicts the major optimization parameters in the existing energy efficiency techniques in cloud computing. It considers deadline, QoS, budget, resource utilization, SLA fulfillment etc. as parameters of main consideration. Clearly, the energy efficiency forms the basic optimization objective of all the techniques (indicated in base circle). It can be inferred from the figure that the technique [84,155,160,171,172,177,182,188] and [195] primarily focus on energy-performance optimization but ignore deadline and budget factors. Although technique [171] offers energy-performance optimization but simultaneously it focuses on SLA fulfillment, QoS, and load balancing.

The techniques presented in [156], [197] and [203] stress more on energy-deadline optimization ignoring performance. Even [197] forestalls budget optimization and also compromises QoS. In common, these techniques are based on heterogeneous infrastructure. The technique presented in [203] is also based on energy efficiency on heterogeneous infrastructure and also offers deadline-and-budget satisfaction but it overlooks performance.

Certain techniques as in [90,148] and [171] exploit heterogeneity and energy efficiency only but avert budget optimization. However, other techniques proposed in [198] and [200] offer primarily budget-energy efficiency optimization as the primary objective overlooking deadline and performance metrics. Likewise [198] and [200] techniques, the techniques [145] and [203] also offer budget-energy efficiency but comparatively achieve deadline optimization as well. Also, the technique [205], even though offers energy, deadline and performance optimization but it misses budget metric. Thus, although the techniques surveyed have focused on minimizing energy consumption primarily while fulfilling deadline, budget and performance metrics but all these metrics are not catered collectively in any of the surveyed techniques.

2.6.3 Implementation Environment based Analysis

Table 2.8 describes the implementation environments for the different energy efficiency techniques in cloud computing and performance improvements made by the techniques. The performance improvements include the percentage of energy savings, percentile of SLA violations, response time, Energy Delay Product (EDP) that provides energy consumed per event, and number of VM migrations carried out.

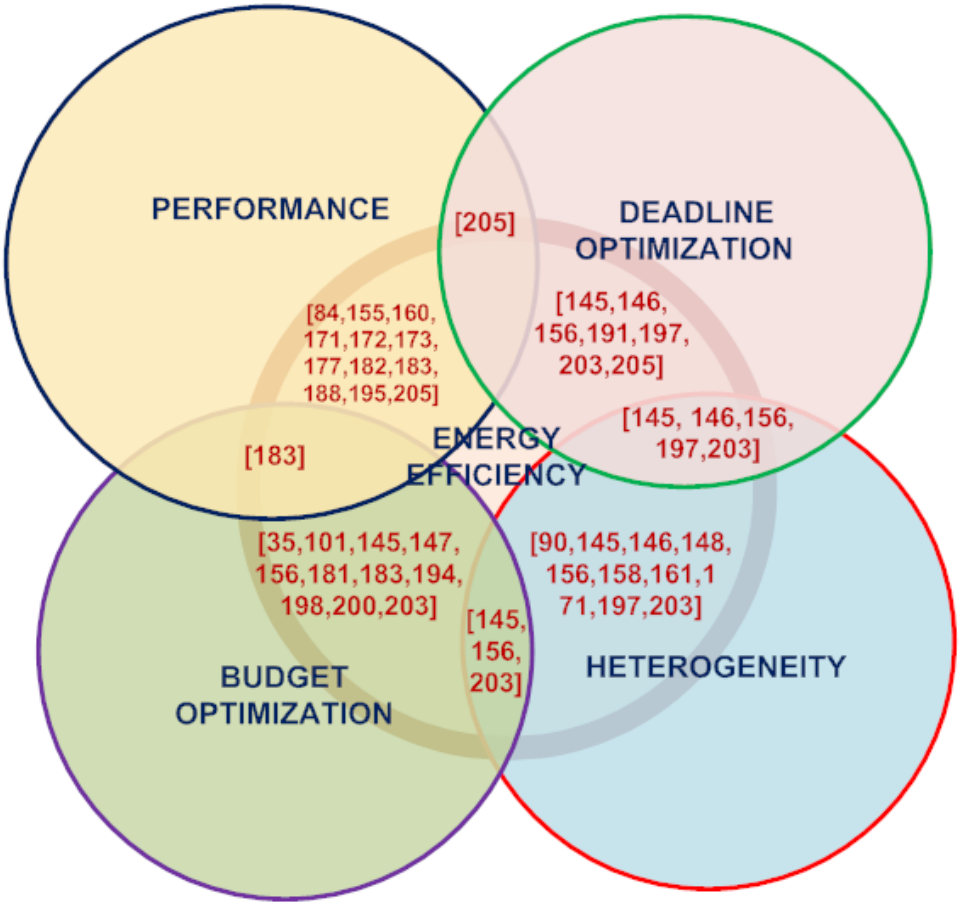

Figure 2.4: Comparative Analysis Venn Diagram

Also, some of the aforementioned authors have compared their techniques with the techniques of other authors. All such information is listed in Table 2.8.

Table 2.8: List of the Implementation Environment and Performance Improvement of Energy-Efficiency Techniques in Cloud

Techniques	Simulation/ Implementation Environment	Performance Improvement
Li et al. 2009 [141]	Implemented in iVIC system with Python, Xen as a hypervisor, Linux 2.6.18 as OS	• Saves 10% - 13% of energy • Uses lesser number of active nodes unlike first fit and best- fit algorithms

Rodero et al. 2010 [82]	A C++ event-driven simulator named kento-perf simulator	• 8% - 25% of energy savings
Ding et al. 2015 [145]	Four PC processors and four server processors in simulated Cloud environment	• Achieves above 20% energy reduction and 8% boost up in processing capacity
Du et al. 2014 [146]	CloudSim toolkit package with a PC consisting of 2.6GHz Pentium dual core processor and based on Windows XP platform	• Proposed algorithm is more energy efficient than GA and Hybrid PSO and Heterogeneous Earliest Finish Time (HEFT) with Average Performance Improvement compared to HPSO ranging from 33.23% to 55.36%
Beloglazov and Buyya 2010a; Beloglazov and Buyya 2012 [147, 148]	CloudSim toolkit 2.0	• Consumes approximately the same level of energy but ensures almost less than 1% SLA violations and lesser number of VM migrations compared with other algorithms such as DVFS and Non-power aware policy
Liao et al. 2012 [150]	The development environment consists of Ubuntu-11.10-server-amd 64 and qemu-kvm-0.14.1 as the hypervisor with 10 heterogeneous physical machines. OpenStack acts as the cloud platform	• Requires lesser number of physical machines to balance the workload as compared to the Round Robin (RR) scheduling scheme • Consumes less energy compared with RR scheduling algorithm and Random • Involves low SLA violations with a stable rate of increase in the SLA violations
Sharma and Gudetti 2016 [151]	Uses Java language for implementing proposed HGAPSO, The experiments have been conducted in both heterogeneous and homogeneous data center environments with three variable PMs, namely, ProLiantM110G5XEON3075, IBMX3250Xeonx3480 and IBM3550Xeonx5675 and 4 variable VMs	• Achieves energy consumption up to 9.8% less and improves resource utility by 13 percent higher in homogeneous and heterogeneous environments
Li et al. 2011 [152]	Setup consists of four PCs with four cores, 4 GB RAM, 300 GB HDD, and Ethernet. Eucalyptus is used to setup a private cloud environment	• Outperforms the greedy algorithm in terms of the average response time, but it is slightly behind the RR scheduling algorithm in the same case • Conserves more energy than the greedy algorithm, whereas RR conserves no energy • The approach is also more beneficial in load balancing than the greedy algorithm and RR scheduling algorithm

Deore et al. 2012 [153]	VirtualBox 3.1 Cloud Environment	• Involves lesser number of VMs compared with the bully approach, hybrid algorithm, and RR scheduling algorithm • Conserves approximately 21% of energy, more than that conserved by the bully, RR, and hybrid algorithms
Quan et al. 2012a; Quan et al. 2012c [154, 155]	It uses single, dual, quad, and six core servers	• Saves up to 31 • Carbon emission reduction up to 87%
Yue et al.2013 [156]	Implements Monte Carlo simulations for randomly generated large scale task graphs	• Experimentally, its globally computed energy costs improves by more than 23% while servicing 30-50 users and more than 16% while servicing 60-100 users
Beloglazov and Buyya 2010b; Beloglazov and Buyya 2012 [90, 148]	CloudSim toolkit	• When thresholds are between 30% and 70%: Minimum migration policy results in power cutback by 66%, 83%, and 23% compared with Dynamic-voltage frequency scaling, Non Power- Aware, and ST schemes respectively • When thresholds between 50%-90%: MM policy offers 87%, 74% and 43% energy savings
Kessaci et al. 2012 [157]	VMs are created using XML and are implemented in the OpenNebula cloud	• Typically, 1.3% higher VMs are scheduled by the Energy-aware multi-start local search algorithm for an OpenNebula-based Cloud (EMLS-ONC) as opposed to a preceding OpenNebula scheduler • Shows 15.6% more improvement than EMLS-ONC
Kansal and Chana 2016 [158]	CloudSim cloud simulator	• Upon comparison with two other techniques, FFO-EVMM achieves average energy consumption of about 44.39 % by reducing an average of 72.34 % of migrations and saving 34.36 % of hosts
Duan et al. 2017 [159]	CloudSim cloud simulator	• Attains 76.17% and 75.28% improved energy levels as compared to over First-Fit and Round-Robin

Quan et al. 2012b; Quan et al. 2012c [155, 160]	Consists of an environment composed of single, dual, quad and six core servers implemented on three types of resource centers, that is, old, normal, and modern data centers	• By using the traditional single algorithm, it consumes 46.58, 36.13, and 32.18 MWt of power, which is much less than that consumed by RR and greedy approach • By using the Cloud Global Optimization Algorithm, 44.76, 34.47, and 30.7 MWt of power is consumed in different simulation runs. These values are far less than those of RR and greedy approach
Knauth and Fetzer 2012 [161]	Implemented in Python	• OptSched lowers the CMU by up to 60.1% and 16.7% in contrast to RR and firstfit algorithms
Paya and Marinescu 2013 [163]	Implemented on clusters with variable sizes, namely, 20, 40, 60, 80, and 100 cluster nodes	• Approximately 70% of servers operate in the optimal region, and only 5% of them are in the two undesirable regions and 25% run in the two suboptimal regions in a cluster of 20100 servers
Xu et al. 2016 [164]	CloudSim Cloud Framework	• Achieves 24.45% energy savings when number of scientific workflows is 50s • Achieves 23.59% energy savings when number of scientific workflows is 100 • Achieves 24.51% energy savings when number of scientific workflows is 150 • Achieves 23.17% energy savings when number of scientific workflows is 200 • Achieves 20.28% energy savings when number of scientific workflows is 250 • Achieves 17.07% energy savings when number of scientific workflows is 300
Shi et al. 2016 [165]	Implemented on OnTaPRA scheduler in a data center based on FatTree architecture where a y-array FatTree architecture contains y pods each having y=2 racks with y=2 servers on each rack	• TaPRA and TaPRA-fast algorithm achieve 40%-430% lesser Job Completion Time (JCT) • OnTaPRA scheduler while running TaPRA/TaPRA-fast minimizes JCT on average by 60%-280% • TaPRA-fast performs 10 times more rapidly than TaPRA and comparatively realizes 5% performance degradation
Merkel and Bellosa 2006; Merkel and Bellosa 2008 [168, 169]	Linux implementation that runs on an Intel quad- core	• Energy delay product (energy consumed per event) is reduced by 0:918 parts at a fixed 2.4-GHz value

Coskun and Rosing 2008 [35]	Simulation on UltraSPARC T1	• 10% energy savings as compared to 20% achieved by DVFS
Hussin et al. 2011 [171]	Tests the proposed scheduling approach on real-workload traces, SDSC Blue, and HPC2N	• Achieves more performance in terms of task execution rate as compared to fit-selection, minmin, and random-selection scheduling approaches • Conserves more energy than Fit-selection and Min-Min strategy
Lee and Zomaya 2012 [101]	Simulation performed on 50 tasks	• The MaxUtil attains an average of 13% of energy savings while ECTC achieves an average of 18% energy savings, regardless of whether migration is performed
Kliazovich et al. 2013; Kocaoglu et al. 2012 [172, 173]	• GreenCloud simulator • DVFS and DPM scaling approaches • Energy-monitoring tools • Consists of 3-layered tree-shaped data center design containing 1536 servers organized onto thirty-two racks, with forty-eight servers enclosed within each rack	• DENs technique is compared with RR and Green scheduler • DENs methodology leaves lesser number of servers idle compared with the green scheduler • RR is the least energy efficient • DENs methodology consumes about 4% more energy, which is slightly more than that consumed by the green scheduler as it also adds a network awareness factor and uses additional computing and communicational resources
Salehi et al. 2012 [174]	Implemented on Haizea, an open source scheduling environment. Experiments conducted on Blue Horizon cluster in San-Diego Supercomputer Center (SDSC)	• PEMP consumes lesser energy as compared to SLA-based energy saver policies (SESP) and achieves 18% of energy conservation • Greedy Energy Saver Policy (GESP) is least power consuming as compared to both PEMP and SESP
Hongjian Li et al. 2016 [175]	CloudSim, framework in which a data center comprising of 250 heterogeneous physical nodes, half of which are HPProLiant ML110 G4 servers, and the others consist of HP ProLiant ML110 G5 servers is simulated	• Upon comparison with Modified Best Fit Decreasing (MBFD), MPSO consumes lesser energy and has lower number of active nodes
Laszewski et al. 2009 [177]	Implemented on OpenNebula and runs the nBench Linux Benchmark version 2.2.3. Simulation is performed using DVFS-SIM.	• Attains 34% improved total performance
Younge et al. 2010 [84]	Scheduling algorithm runs in the OpenNebula environment	• Conserves 12% of system's power under normal load situations

Berl et al. 2010, Xian et al. 2007 [75,180]	Sets variable voltage and frequency values keeping track of power consumed by Intel XScale	• Achieves 12%-30% energy savings during worst-case-based partitioning plus implementation pace set as dynamic slack reclamation (WP0) • Realizes 12%-30% energy savings during worst-case-based partitioning with probabilistic uni-processor speed settings (WP1) • Achieves savings of up to 3%-17% compared with WP0 and WP1
Kim et al. 2011 [181]	Simulates through the CloudSim toolkit and creates a data center with four systems with a 16-DVFS enabled processor	• The adaptive provisioning methods, that is, Adaptive-DVFS and -advanced-DVFS consume less electrical power and generate higher profits irrespective of workload
Chen 2011 [182]	Uses a power meter and utilizes the hardware provided by the Distributed ASCI Supercomputer 4 (DAS-4)	• The algorithms optimize the energy consumed but involve a number of VM migrations, which adds to both energy- and performance-related costs
Kusic et al. 2009 [183]	Creates a setup composed of heterogeneous Dell PowerEdge servers, VMWare's ESX Server 3.0 Enterprise Edition running Linux RedHat 2.4 kernel, and SUSE Enterprise Linux Server Edition 10 OS per virtual machine	• Compared with a system without dynamic control, the LLC-based technique saves up to 26% energy with 1.6% of total service requests SLA violations
Ahmad and Vijaykumar 2010 [68,184]	The proposed scheme has been tested in a forty by twelve inch data center consisting of 1120 blade servers arranged in 4 rows. Every single row has 7 40U racks. It uses AirPAK, a computational fluid dynamics simulator for experiencing cooling mock-up in a data center	PowerTrade has been compared with spatial subsetting and inverse temperature. The following observations have been made as follows: • PowerTrade and spatial subsetting acquire free power; however, both alleviate response period dilapidation • The electrical power savings for PowerTrade scheme vary with load levels: – For loads ranging from 30% to 70%: PowerTrade-d attains considerable power cutback up to 22%-36% – At 80% loading: PowerTrade scheme performs better than PowerTrade-d • PowerTradeSurgeGuard duo lower power usage by 30% compared with spatial subsetting • Both retain 1.7% time to respond unlike spatial subsetting. The response period degradation is affected by a factor of almost 2.8
Teng 2011; Younge et al. 2010; Tang et al. 2008 [15,84,185]	Implemented in OpenNebula project	• Achieves about 1230% cooling energy savings

Rodero et al. 2012 [186]	Two Dell servers running CentOS Linux OS and an Intel Xeon X3220 processor having patched 2.6.18 kernel including Xen 3.1	• Achieves 12% reduction in energy consumption and 18% reduction in the execution time • Offers more energy conservation and performance as compared to first-fit, random, temperature-aware and energy-aware Thermal thermal management VM management strategies
Pakbaznia et al. 2010 [187]	A small scale datacenter of 9.6m8.4m3.6m in size and 7U server blades with total number of 1000 servers	• Compared with two greedy heuristics, GREEDY and TA-GREEDY. The comparison results are as follows: • TA-GREEDY performs better than GREEDY • The comparison report of TA-GREEDY and TA-DRP is the same for fresh servers but TA-DRP excels over TA-GREEDY when the servers leave
Raj et al. 2016 [188]	Uses PlanetLab workloads running in CloudSim toolkit 3.0	• Obtains energy savings around 12
Kessaci et al. 2011 [191]	Real World Setup of Fast Fourier Transformation Task Graph	• Energy consumption reduced by 47.49% and makespan reduced by 12.05%
Wang et al. 2012a [192]	Experiments are performed in a data center consisting of 200 servers	• Improves the energy of servers outperforming Hadoop MapReduce scheduling methodology
Nguyen et al. 2013 [193]	Simulation environment consists of 211 VMs and 100 physical machines (hosts) and 2 servers, IBM server x3250, and PowerEdge R620 server. It uses CloudSim version 3.0	• GAPA can find a better VM allocation (lesser energy consumption) than the minimum increase of power consumption (best-fit decrease) heuristic
Moghaddam et al. 2011 [194]	Implemented on a flat VPC that enables flawless virtual machines movement over the Internet	• The proposed GA-based method can significantly cut the carbon footprint by using Cloud net • Attains 36.64% of reduction in the carbon footprint but at the cost of a 4.96% increase in energy consumption • When simulated and tested under low and high workloads, it shows improvements in cutting the carbon emission rates for variable loads
Chimakurthi and Madhukumar S.D 2011 [195]	CloudSim toolkit	• Yet to be implemented

Feller et al. 2011 [196]	Implemented in a Java-based simulation toolkit developed by the authors and operates on homogeneous hosts, each having 24 cores, 50 GB RAM, 1 TB storing capability, and 10GBit/sec net connectivity	• ACO Meta heuristic algorithm has been compared with the adaptive edition of first-fit decreasing approach • Based on the comparison, ACO-based approach utilizes less number of host machines and offers higher resource utilizations and energy profits • Normally, 4.7% of hosts, and 4.1% of energy is saved
Babukarthik et al. 2012 [197]	Simulation done using Matlab, IaaS created through Xen Cloud Platform, and PaaS through Microsoft Windows Azure	• Comparing the hybrid algorithm with ACO according to number of tasks and processors, former achieves considerably more energy efficiency and performance than ACO does
Wang et al. 2012b [198]	CloudSim toolkit	• The designed algorithm consumes lesser energy compared with Energy-efficient topology control algorithm (ECTC) and Random algorithms • Energy consumption has reduced by 67.5% on an average when compared with ECTC • Revenue enhanced by 8.14 times on average as compared to Random
Bergamann et al. 2013 [199]	Utilizes a graph generator for generating a large set of Directed Acyclic Graphs (DAG) files for the testing data	• SSO has been compared with PSO and both can accomplish up to 20% power reduction • To some extent, Particle swarm optimization performs better than SSO
Garg et al. 2011 [200]	Evaluation across 8 data centers located across the world	• Compared with GMCE (Greedy Minimum Carbon Emission) for minimizing carbon emission and with Greedy Maximum Profit (GMP) for maximizing profit • The comparison has been done with both withDVS and withoutDVS approaches • Offers 25% of energy savings and performance improvement up to 33% on an average when compared with GMCE-withoutDVS and GMP-withoutDVS approaches
Luo et al. 2012 [201]	Creates a real cloud environment through Eucalyptus, and data processing program used is Hadoop	• Realizes energy savings of up to 45%

Shi et al.2011 [202]	CloudSim Cloud simulator	• In case of implementing pure MQMPM, some delay in predicted sequence is observed, whereas with implementation of LTPM and MQMPM, the numbers of violations due to delayed utilization trends are reduced • In a continuous increasing period, the combined algorithm is better • During flat and smooth periods, the use of FPRRM helps to reduce unnecessary high resource reservations
Shuja et al. 2014 [203]	Uses GreenCloud packet-level simulator and IBM CPLEX 12.0	• More energy efficient than Greedy, GreenCloud scheduler, Topology-aware heuristic
Singh and Chana 2016 [204]	CloudSim Cloud simulator	• Achieves resource utility up to 12.04% and reduces the energy by up to 17.02%
Qing et al. 2016 [205]	Implemented on 10 servers including 8-cores based two machines with 16GB memory; 16-cores based 6 machines with 32 GB memory; and 32-cores based 2 machine with 64GB memory	• Caters to energy reduction as per the workloads such that: – Attains 37.5% power reduction at low workloads – 36.6% power reduction at medium workloads and – 24.5% power reduction at high workloads

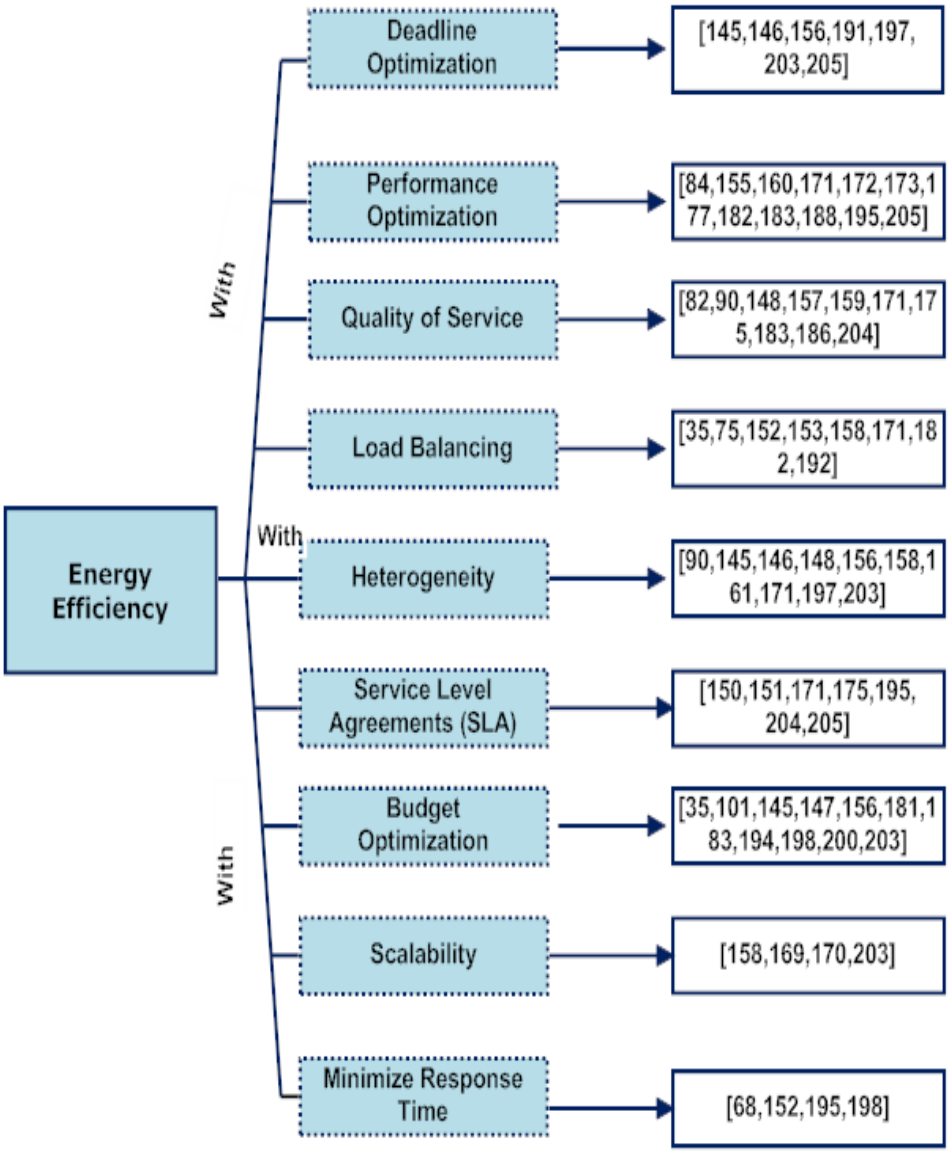

Figure 2.5: Analysis of the Existing Techniques Based on Their Optimization Metrics

2.7 Taxonomy of Energy Efficiency Techniques in Cloud Computing

The taxonomies related to the stated cloud based energy efficiency techniques have been designed in this section. These taxonomies provide a detailed and comprehensive look at the characteristics of different energy efficiency techniques. Such details can aid in making meaningful comparisons of different approaches.

The virtualization taxonomy is shown in Figure 2.6. The technique [[141], [165], [186] and [205] offer application level of virtualization where a single application is available to the users without having to completely install that application on the users local system. However, the technique [186] is primarily a thermal-aware energy management scheme that actually allocates the VMs based on the application characteristics accordingly. As shown in the Figure 2.6, the techniques [82, 147, 148, 154 and [160] utilize VM consolidation to realize energy efficiency. Apart from these, [174 and [175] also uses VM consolidation across multiple servers while [194] and [198] although being bio-inspired optimization techniques but employ VM consolidation method across multiple servers. The technique [188] employs VM consolidation but is primarily thermal-aware energy efficiency technique.

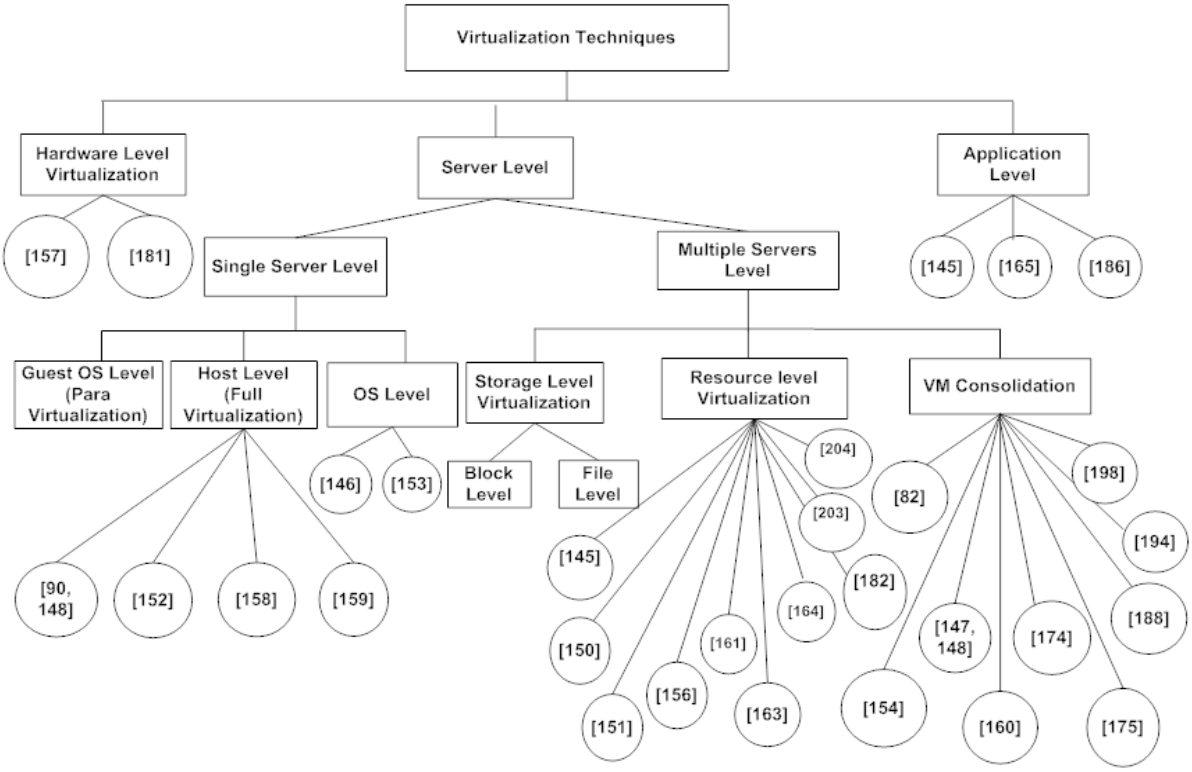

Figure 2.6: Taxonomy for Energy Efficiency Techniques Based on Virtualization

Resource level virtualization has been implemented by techniques [145], [150], [151], [156], [161], [163], [164], [203] and [204]. The technique [182] is a power-aware management scheme that is additionally based on resource level virtualized environment where resource pool is managed and then VM migrations are done.

The techniques, [90,148], [152], [158] and [159], involve switching off unused nodes to reduce the energy consumption and implement full virtualization methodology. Both [146] and [153] involve running of multiple virtual OS over a fully functioning host OS whereas [157] and [181] implement hardware virtualization which consists of running multiple guest OS directly on top of base hardware. The technique [181] also uses the power-aware DVFS scaling for the energy management.

The taxonomy based on multi-core architectures has been presented in Figure 2.7. The techniques that have exploited the capability of multiple cores along with their logic level of implementation have been shown. These include [35], [169] and [171]. However, the technique presented in [177] is largely a power-aware management technique but has been implemented on a multi-core cluster.

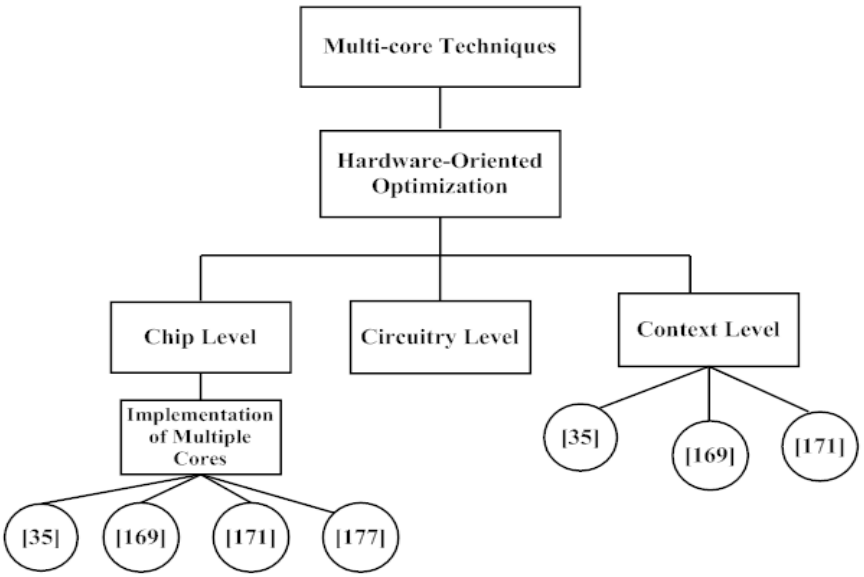

Figure 2.7: Multi-core Level Taxonomy

Figure 2.8 shows taxonomy of consolidation based techniques. The techniques [101] and [172,173] are based on task consolidation based methodology while [174] is an energy efficiency technique consolidating the processing capacity of VMs running on the multiple servers. Although, the technique, [196], is based on the bio-inspired energy optimization but it involves consolidation of the workloads. Similarly, the technique presented by [147,148], [175] and [188] consist of VM allocation and scheduling using the VM consolidation mechanism.

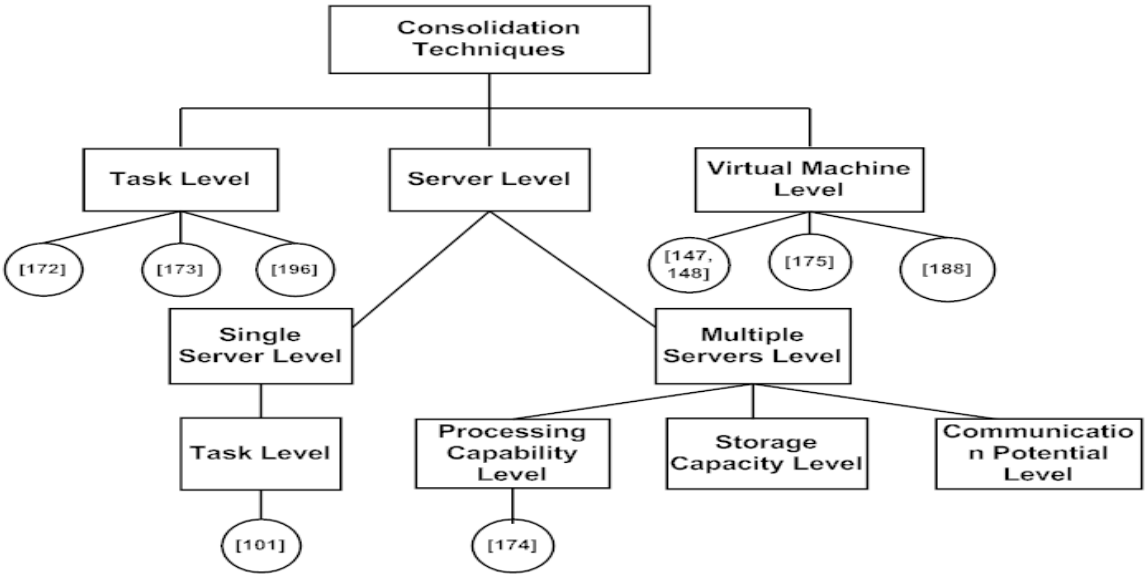

Figure 2.8: Taxonomy on Consolidation Based Techniques

The taxonomy of bio-inspired energy efficiency techniques is presented in the Figure 2.9. The techniques were categorized into those based on evolution theory (including Genetic Algorithms) or on the behavioral instincts of different species on earth. The technique [191] implements GAs for hardware-oriented optimizations while [192], [193] and [194] techniques utilize the capability of GAs for software energy optimization. Certain techniques implement and map the behavioral patterns of the bio-species and thus accomplish energy optimizations by mapping the specie behavior such as in ACO or swarm based algorithms. The technique [197] offers hardware-oriented energy optimization using ACO whilst [195] and [196] offer software-oriented optimization using ACO. The techniques [198] and [199] are swarm behaviour based optimization techniques that differ in terms of hardware-oriented or software-oriented optimization methodology.

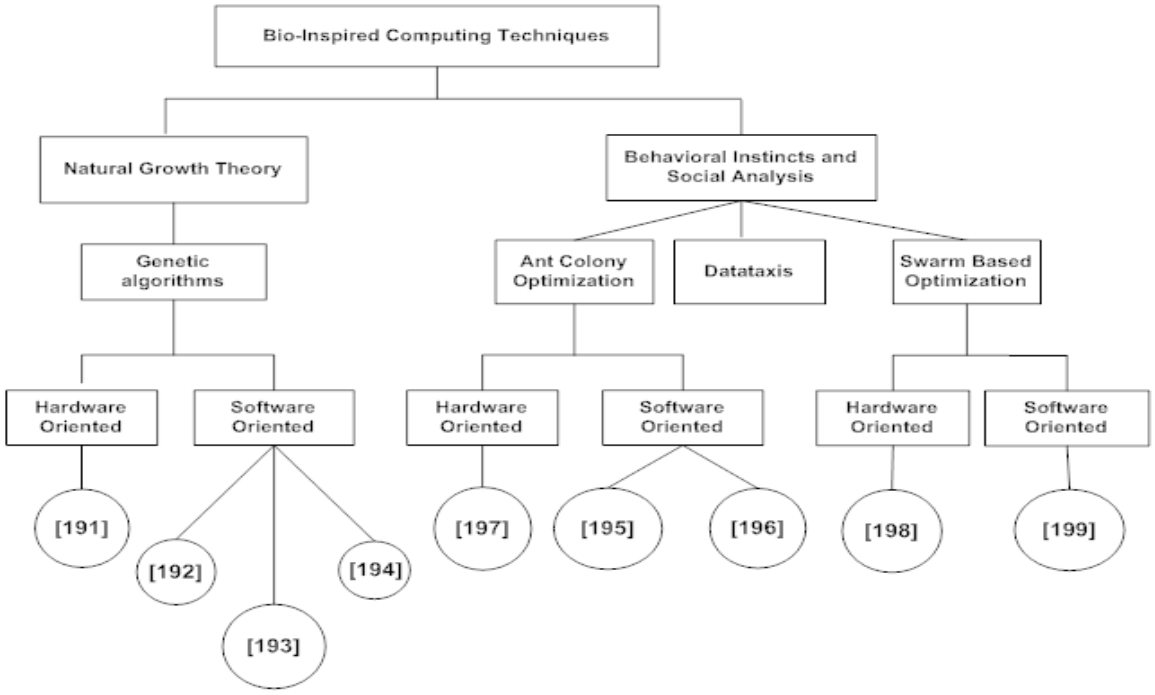

Figure 2.9: Bio-Inspired Techniques Taxonomy

2.8 Problem Statement and Research Objectives

As per the literature survey, one of the main issues in cloud computing is to make resource scheduling energy aware in order to reduce high carbon emissions to the environment and to reduce the cost incurred in cooling these systems. The scheduling of computational resources is a fundamental problem in order to optimize program execution and usage of necessary cloud infrastructure. Therefore, there is a need of energy-aware scheduling techniques that decrease the overall energy consumption and improve the performance of cloud computing by efficiently scheduling the user requests and thus achieving maximum resource utilization. The general trend today in computing is a shift towards multi-core architectures. Besides better performance, multi-cores have been observed to be low electrical power consumers and less heat generators. Thus, they have a strong environmental impact and it becomes highly important to explore their role to achieve energy efficiency especially in cloud computing.

The objectives of this research work are:

(i) To analyze the existing energy efficient resource scheduling techniques and the role of multi-core architectures in Cloud computing.

(ii) To propose, design and develop energy-efficient resource scheduling technique.

(iii) To optimize the proposed technique for performance and energy related costs.

(iv) To test and validate the proposed resource scheduling technique in Cloud environment.

2.9 Conclusion

This chapter discussed the research work accomplished in the area of energy efficiency in cloud computing primarily focusing on the software-oriented energy efficiency techniques. Based on the review, it can be concluded that the best software-dependant energy minimization can be accomplished through energy-aware scheduling of tasks to appropriate resources.

The next chapter presents the proposed Energy-aware scheduling model for cloud computing for energy efficient allocation and provisioning of cloud tasks.

Chapter 3

Proposed Green Cloud Scheduling Model

The previous chapter presented a comprehensive and comparative review of the research work conducted in the area of energy efficient scheduling in cloud computing along with the discussions and analysis of the significance of the need to design energy efficient techniques via cloud computing.

This chapter proposes an energy-aware scheduling model for cloud computing. The process of optimally allocating and scheduling the tasks on low energy consuming nodes in the cloud data centers contributes significantly towards accomplishing energy efficiency as observed in the literature survey. Additionally, the management of nodes and their resources as per the energy demands tends to minimize the energy consumption, related expenditures and forestall performance dilapidations. Moreover, as per the observations, high energy consumption within data centers can cause performance degradation thus hindering the processing time of tasks. The proposed model, Green Cloud Scheduling Model (GCSM) energy-efficiently allocates and schedules tasks on cloud nodes. GCSM principally tends to optimize the energy consumption of cloud nodes and thus is referred to as green model. It also attempts to fulfill the task deadlines, achieve maximal utilization of nodes and computing resources and prevents performance degradation.

Initially, Section 3.1 introduces the GCSM model and highlights its significance and operational characteristics. Section 3.2 discusses the architecture of GCSM and the entities involved in its functioning. Further on, Section 3.3 discusses the mathematical explication for the same followed by detailed working process of GCSM and corresponding flowchart. The Green Cloud Scheduler unit and its Green Cloud Scheduling (GCS) algorithm have been thoroughly discussed in Section 3.4. Later on, Section 3.5 presents the experimental analysis of the GCSM on major metrics with different approaches. Finally, Section 3.6 concludes the chapter.

3.1 Green Cloud Scheduling Model (GCSM)

Green Cloud Scheduling Model (GCSM) is an energy aware cloud-based model for performing task allocation and scheduling. GCSM facilitates the provisioning and scheduling of the cloud tasks to the nodes that are energy-aware or energy-conscious or low energy consuming nodes. The Scheduler Unit of GCSM provisions and schedules the tasks delimited to only energy-aware cloud nodes. It takes the energy-aware task allocation decisions dynamically and aims to prevent performance degradation and achieves desired QoS like deadline satisfaction. The dynamism involved in allocating and scheduling of the tasks infuses and supports node reliability by preventing their under-utilization and over-utilization [206]. Additionally, the run time allocation decisions made by the scheduler support higher throughput and avoid slow run of the systems. Generally, this can be inferred from the fact that dynamic allocation and scheduling decisions prompt improved system execution capabilities and reliability [207–209].

3.1.1 Significance of GCSM in Current Computing Systems

GCSM envisions not only enforcement of the minimization of energy consumption but also prevents energy wastage and supports efficient processing and proficient utility levels of the nodes. GCSM 's Scheduler Unit runs a Green Cloud Scheduling (GCS) algorithm that makes the task allocation and scheduling decisions while considering the energy-aware capability of the heterogeneous cloud nodes and also considers the nodes that can fulfill the task completion time limits as imposed by the users. In other words, the GCSM system not only achieves energy efficiency but also tends to offer desired QoS to the users in terms of task deadline satisfaction. The other significant contributions of the proposed GCSM model are:

(I) GCSM performs heterogeneous task allocations for the heterogeneous cloud nodes in an energy-aware manner without affecting the system performance and within the task deadlines imposed by the users.

(II) The reduction in energy consumption tends to prevent performance degradation else the rising energy wastage and higher carbon footprints cause a halt to the performance [50, 76]. This deduces from the fact that in case the equipment installed within a cloud data center gets over-heated, the performance degradation can occur and also yield incorrect calculations etc.

(III) GCS algorithm identifies the best energy-aware nodes for allocating the tasks followed by exploration and analysis of the identified energy-aware nodes for their deadline fulfillment capability. The deadlines are important to be satisfied in order to develop user trust on service delivery and enable the application speedup. [156, 210–212].

(IV) It identifies nodes that are under-utilized and prevents their unnecessary power withdrawal or idle power else they can cause energy wastage [111]. In other

words, it identifies the non-active nodes within the cloud system and sets them to sleep or switch off mode to prevent energy wastage done by them when sitting idle. It also restrictively monitors the nodes for possible high energy levels and prevents there over-utilization.

(V) GCSM implements GCS algorithm that is a heterogeneous computing technique, vital in the current era of distributed computing systems and with the growth of heterogeneous Clusters and Grid systems.

(VI) Overall, GCSM is beneficial for both the clients and the service providers. From the provider perspective, the reduced energy consumption facilitates energy efficiency; helps to lower the energy-related costs and prevents performance degradation. From the client perspective, the task executions performed by GCSM within the specified deadlines is the overall benefit.

(VII) Exclusively, GCSM is feasible as a greener solution facilitating faster processing and satisfying timing obligations. A typical application area includes banking and financial sector which is currently adopting greener solution and is highly time constrained sector.

3.1.2 Process of Energy Management and Deadline Satisfaction in GCSM

The process of provisioning and scheduling of tasks to the energy efficient nodes is highly crucial and complex job performed by GCSM. The initial job of identification of energy-aware nodes and further investigation of nodes for possible deadline fulfillment involves the following key working characteristics that indicate the GCSM process cycle:

- GCSM monitors every node in terms of the energy consumed by it to make appropriate energy-aware scheduling decisions running GCS algorithm by its Green Cloud Scheduler Unit.
- The GCSM's scheduler is an energy-aware scheduler and thus called as green cloud scheduler unit. It performs dynamic new task allocations and schedules tasks to energy-efficient nodes accomplishing desired QoS in terms of minimizing execution times and deadline parameters. Thus, GCSM offers faster task executions in addition to the energy awareness.
- GCS algorithm discovers the best energy-aware nodes for provisioning the tasks. After this, it explores and analyses the identified energy-efficient nodes for their deadline fulfillment capability.
- A power monitoring unit is used to monitor the energy consumed by each node in the cloud data center.

- The power consumed by each node is taken in terms of the power consumption of individual computing units available on a node. The computing units considered here correspond to the CPU and memory.
- A database is used to analyze the past energy efficient and deadline based task allocations. The information stored in the database helps to track previous task allocations and make the future energy-aware allocations.

3.2 Entities of GCSM

GCSM consists of number of components that aid it in performing the energy-conscious allocation and scheduling and corresponding functional perspectives. Table 3.1 lists the different terms used in the model. Figure 3.1 shows the proposed GCSM model. The GCSM is composed of the following components:

Table 3.1: Description of Various Terms

Terms Used	**Description**
Virtual Machine (VM)	VM is an emulation of a computer system
Server/Node/Host	A rack-mountable hardware or a computer system where VMs are running
Virtualized Environment	Nodes running variable number of VMs and comprising the cloud environment
Computing Units/ Resources	CPU and Memory units
Heterogeneous Tasks/Jobs	CPU-intensive/ Memory-intensive tasks/jobs

*Server, Node and Host have been used interchangeably
*Computing units and resources have been used interchangeably

- *Cloud Users:* The users submitting their tasks for execution to the GCSM system. Each user submitting the tasks has to specify the resources required and the deadline for task completion.
- *Cloud Data Center Infrastructure:* It comprises the actual cloud infrastructure involving heterogeneous computing systems, clusters and virtualized cloud servers.
- *Task Handler and Task Analyzer Unit:* The Task Handler and Task Analyzer unit handles and analyzes the tasks submitted by the users. While functioning as a task analyzer, it examines the resource requirement and deadline imposed by the user for each task and stores this information in its database. It tracks the existing tasks running in the system and stores their information in its database.

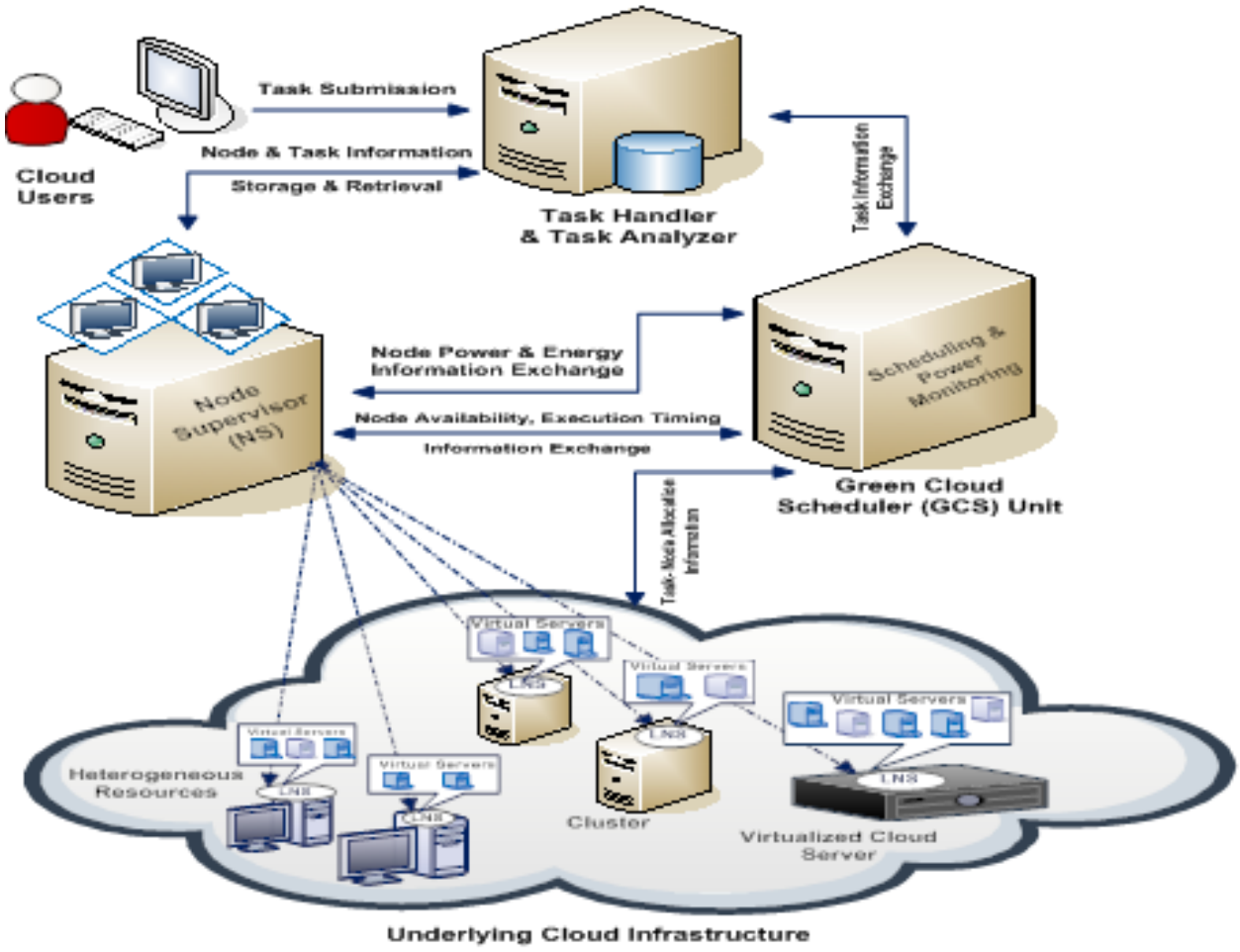

Figure 3.1: Proposed Green Cloud Scheduling Model (GCSM)

- *Node Supervisor Unit:* The nodes operating in the cloud infrastructure are monitored and supervised by the Node Supervisor (NS) unit that employs Local Node Supervisors (LNSs) present on each node. The NS unit keeps track of the status of the nodes (whether in active, energy saving mode or switch off mode) and the number of resources with each node based on the information provided by LNSs. The LNSs keep track of the node capabilities in terms of the resources available with them and number of VMs running on each node. This collected information regarding node utilization and VMs chosen to migrate is sent to the NS unit. In Figure 3.1 , the VMs that are being utilized are powered on and have been highlighted while the lightened VMs correspond to nodes either in sleep mode or in switch off mode.

- *Green Cloud Scheduler Unit:* Green cloud scheduler is the core scheduling compo-

nent of the proposed GCSM. It identifies less energy consuming nodes and schedules the tasks to them nodes upon analyzing their energy consumption values and deadline fulfillment capability. It consists of a power monitoring unit that monitors the power of each node and thus computes energy consumption on each node and forwards this information to the scheduler.

It runs GCS algorithm that performs the scheduling of the tasks on the nodes identified as the most energy-efficient and deadline fulfilling nodes using the information given by the NS unit and Power-monitoring unit. This helps to attain not only energy efficiency and faster task executions but also helps to improve the system performance.

3.3 Mathematical Formulation and Working of GCSM

The complexity of the proposed GCS algorithm lies in its job of finding the best nodes for allocating the tasks. This task of finding the most energy-efficient nodes for task allocation is simultaneously carried out considering performance optimization and deadline satisfaction factor to prevent negatively impacting the overall system performance and task execution timings [50, 76, 213]. Thereby, apart from scheduling the tasks to energy-aware nodes within the deadlines, GCSM achieves optimization in the performance as well. The proposed GCSM follows certain assumptions and postulations as listed below:

(i) GCSM consists of fully virtualized cloud environment where the tasks are allocated to energy-aware heterogeneous server nodes with variable number of VMs running on each node.

(ii) The tasks are autonomous units submitted by the cloud users along with the fixed respective deadlines. These tasks are initially handled by the task handler and analyzer unit for information extraction and then forwarded to green cloud scheduler for subsequent allocation to the relevant system nodes.

(iii) Tasks are infinitesimal, indivisible units of workload that cannot be pre-empted in between the running state.

(iv) GCSM tends to optimize the energy consumption done by the CPU and memory that have been observed to be important units impacting the overall energy consumption of the nodes. However, in GCSM, the energy consumed by the network components and storage units is considered as negligible.

(v) The electrical power consumed by the processor and memory is cumulative when considered collectively. Also for each computing unit, its additional power $Power(U_a, t)$ is either null (not active) or maximal (active). The assumption is based on the node level power consumption model given in [214].

(vi) Some idle power is being consumed by each node during its inactivity period. This inactivity period pertains to the instances when a node although being active, is sitting with a fixed screen image while no user interaction or application is being run.

(vii) The computed energy consumed by the nodes is compared with a node energy threshold. The task allocation is done only to the nodes with the energy consumption value lesser than the energy threshold value. A careful selection of this threshold value helps to allocate the tasks energy efficiently.

Analytically, the optimization algorithm tries to locate the best node with least Energy Function, (EF) that is computed based on the power consumption values measured by the Power-monitoring unit implementing a Joulemeter [215, 216]. GCSM aims to minimize EF along with satisfaction of task deadlines. The EF value for each node is computed by the Power-monitoring unit after measuring the power consumption $Power_i$ done by each node i. The EF_i is the energy consumed by a node given by the product of $Power_i$ during t time instant. Similarly, total EF value is computed to obtain the overall energy consumption done by the entire system at time instant t. The deadline satisfaction is judged by computing the ET_i value that corresponds to the total execution time taken by node i running j VMs for executing k number of tasks.

Mathematically, the problem formulation comprises of variable constituents. The cloud infrastructure is composed of mixed nodes such as cluster nodes and server nodes. It is composed of a set $\{sn_1, sn_2, sn_3, ..., sn_N\}$ of N system nodes such that, $SN = \{sn_i | \, 1 \leq i \leq N\}$ is the collection of N multicore nodes; a set $P = \{vm_1, vm_2, vm_3, ..., vm_M\}$ is the collection of M VMs subject to the constraints, $P = \{vm_j | \, 1 \leq j \leq M\}$; and a set $Task = \{task_1, task_2, task_3, .., task_L\}$ of tasks such that $Task = \{task_k | \, 1 \leq k \leq L\}$ is the collection of L tasks to be run on VMs operating on N nodes. The set $U = \{a_1, a_2, a_3, .., a_u\}$ forms the total number of computing units operating on each node.

On each node, the power consumed by a node can be divided into active power and idle power given by, $Power_i$ and $Power_{Idle}$ respectively [214]. $Power_{Idle}$ is the power consumed by a node in idle state, assumed to be constant when a task executes and power consumption of computing units, $Power(U_a, t), 1 \leq a \leq u$, at time instant t, if active [217]. GCSM computes EF_i for each node based on the measurement of power consumption $Power_i$ done at a give time t for i^{th} node. At a given time instant, $Power_i$ for the i^{th} node running M VMs is given as:

$$Power_i = \sum_{k=1}^{L} \sum_{j=1}^{M} \sum_{a=1}^{U} Power_{jk}(U_a, t) + Power_{Idle} \tag{3.1}$$

where M is the number of VMs running on i^{th} node and L is the number of tasks assigned to M VMs. The used power during activity period of a node is calculated

by obtaining the difference between $Power_i$ and $Power_{Idle}$ and is represented as,

$$Power_{used} = Power_i - Power_{Idle} \tag{3.2}$$

The proposed system takes into account the multiple cores in the nodes. There is a possibility that all the cores are not used when the node is in running state. Thus, $Power_i$ needs to be optimized with the available number of cores during the task execution. For this purpose, the computed $Power_{used}$ value is then multiplied with the number of cores used in the operation divided by the total number of cores in the node.

$$Power_{multicore} = Power_{used} \times \frac{Used_{cores}}{nmc} \tag{3.3}$$

where $used_{cores}$ is the number of cores used in the operation of the node in running state and nmc is the number of cores in the node. The actual power consumption of the node is thereby given by,

$$Power_i = Power_{multicore} + Power_{Idle} \tag{3.4}$$

The Energy Function (EF) for i^{th} node gives the energy consumed by a node. This is defined as the power consumed $Power_i$ by i^{th} node during t units of time.

$$EF_i = Power_i \times t \tag{3.5}$$

The total energy consumption by the system (EF) is the summation of the EF_i for all the nodes subject to integrating the energy consumed by the system during period T.

$$EF = \int_0^T \sum_{i=1}^{N} (EF_i) dt \tag{3.6}$$

In order to satisfy the deadline constraint, the total node execution time has to be computed. The Execution Time, (ET) for i^{th} node is equal to the time taken by the node to execute all the tasks running on it.

$$ET_i = \sum_{k=1}^{L} \sum_{j=1}^{M} ET_{ijk} \tag{3.7}$$

where ET_{ijk} is the time taken by j VMs to execute k tasks on i^{th} node. The main goal is to optimize the energy consumption and satisfaction of the deadline specified by the users. Thus, it is important to compute the fitness function for the same. It is given as:

$$fit_N = \theta(EF) + \delta(ET) \tag{3.8}$$

$$EF = min(EF(sn_i, task_k)) \;\; \forall 1 \leq i \leq N, 1 \leq k \leq L \tag{3.9}$$

$$ET = min(ET(sn_i, task_k)) \ \ \forall 1 \leq i \leq N, 1 \leq k \leq L \tag{3.10}$$

where $0 \leq \theta < 1$ are the weights to prioritize the components of fitness function. $EF(sn_i, task_k)$ is the energy consumed by task k on node i. The energy fitness function is subject to the constraints:

$$EF_i \leq EF_{th} \tag{3.11}$$

that is, the Energy Function (EF_i) of the i^{th} node should not exceed the threshold value Energy Function (EF_{th}) set for the node i. The EF_{th} value depends on the utility level of a node which is computed by obtaining the total number of instructions running on a node per unit of time. Usually, the utility level of node indirectly affects the energy consumed by a node and is asymptotically bounded to EF_{th} [218]. The nodes that are optimally utilized incur less energy as compared to nodes that are under-utilized or over-utilized. This is because the under-utilization of a node prevents it from performing optimally, incurring idle time whereas the node over-utilization levels cause it to function more, thereby, sometimes, degrading the nodes performance [101]. Thus, the optimal utilization of the nodes positively impacts the energy consumption levels. The value of EF_{th} must be less than the maximum utility level of a node.

Detailed working of the proposed GCSM is given below: GCSM is composed of heterogeneous computing environment with each node handling heterogeneous workloads, providing different computing capabilities, having multiple cores and different power consumption values depending on the computing units operating on each. It offers energy and deadline optimizations in cloud environment where time for executing the tasks and energy consumed varies from node to node. Consequently, the nodes with minimum energy consumption value and least task execution time will be in demand in the system. GCSM involves the following phases but depending on the utility, a task may or may not follow all the phases. Figure 3.2 depicts the flowchart of operation of GCSM.

I. **Task Submission and Analysis Phase:** Each cloud user submits some tasks to the cloud system through the Task Handler and Task Analyzer unit, specifying the resources required to run each task, the respective deadline and the type of task. The user specifies the number of processing units needed, the amount of memory required, task deadline and task type. All this information is stored in the database of Task Handler and Analyzer unit. The database holds the already running and existing tasks data, pertaining to their node and resource utilizations and energy consumption. The resource utilization data comprises of the resources being used by each task and the deadline specified by the user. Initially, when no task information is available, the required resources are allocated to the tasks. Only after the tasks get processed, their

complete information is stored in the database. This stored task information is used by the scheduler to make future scheduling decisions.

In future, if same task arrives, then Task Handler and Analyzer unit will check its database about this task and if available, it can use the stored data corresponding to it. In the later phase, the task is allocated to the nodes in accordance with the user specified constraints and thereafter the processed task information is stored in database else the present data is used for task allocation to the nodes. With the termination of the first task submission phase, the tasks are adjudged to be either CPU-bound or memory-bound.

Based on the check on the task data availability in its database, the Task Handler and Analyzer unit either self processes the task or sends it to the scheduler for allocation. In other words, it can easily classify the task into its type i.e., whether CPU- bound or memory- bound based on the task resource requirements, node and resource utilization and energy consumption data and send the same detail to scheduler or can directly send the task to the scheduler.

The NS unit tracks the node status information where the node status indicates the node state whether in running state or in energy saving state. For this purpose, it checks a *Sleep* variable associated with the nodes. The value of *Sleep* if 0 indicates the node is in sleep mode or energy saving mode while a value of 1 indicates the node is in running state. In order to bring a node back from sleep to running state, another Boolean variable *PowerOn* can be set to 1. The 0 value of *PowerOn* variable means the node is switched off. The NS unit is regularly updated about the utilization levels of each node by the LNSs. In addition, LNSs gather information regarding the total number of hosted VMs with every node, the number of active VMs, the resources currently active on every node and the number of available resources for future task allocations.

II. **Green Cloud Scheduler Phase:** All the information pertaining to the nodes and tasks as maintained by the NS unit and Task handler unit respectively is sent to the green cloud scheduler unit. Using the node information, the Power-monitoring unit measures the power and then computes energy consumption of the nodes. The scheduler unit after obtaining the node and task information performs energy-efficient scheduling of the tasks on most appropriate nodes without violating the task deadlines. It also scans the idle nodes and eliminates them. The scheduling decisions taken by scheduler unit depend on the comparative analysis of the nodes in terms of the energy consumed by each node compared with threshold value of energy consumption set for that node. All the nodes having energy consumption value lesser than the threshold are listed in a LessEnergyConsumeNode list while those nodes with greater energy consumption value than the threshold are listed into MoreEnergyConsumeNode list. Both the lists are sorted in the increasing order such that the first element in LessEnergyConsumeNode list is most energy-efficient node that can be chosen to allocate a task to it subject to the availability of resources and check on whether it can fulfill the task deadline.

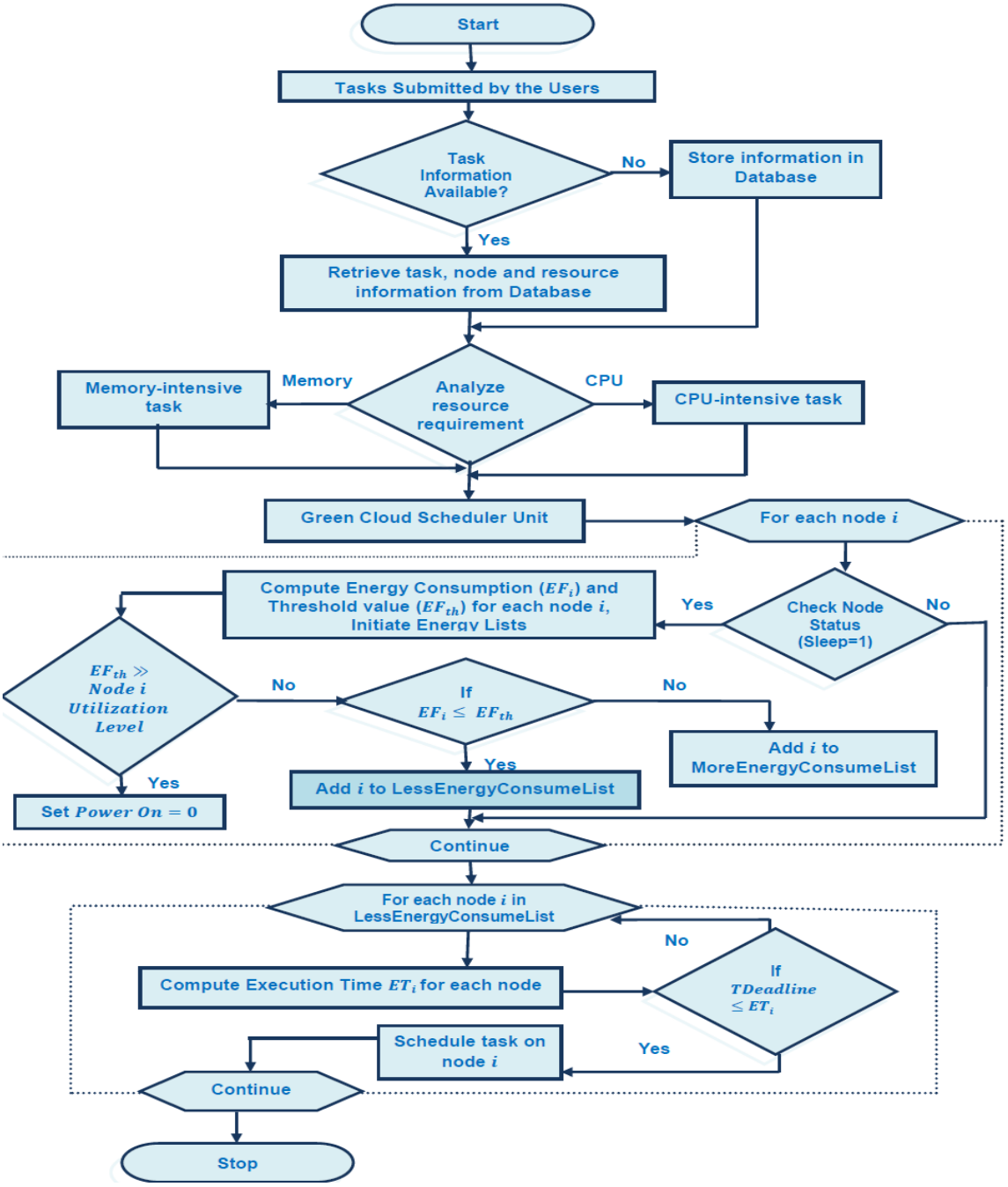

Figure 3.2: Flowchart depicting Operation of GCSM

In case no node in the LessEnergyConsumeNode list satisfies the deadline fulfillment constraint then a new node in energy-saving mode, is turned ON for processing that task. This process is repeated for every unassigned task entering the system. After the tasks have been scheduled, the node utilization values and energy consumption values are propagated to the database of the Task Handler and Analyzer unit for future task allocations. For the operation of GCSM, different system equations and fitness functions that decide the system behavior have been discussed already in the foregoing section.

3.4 Green Cloud Scheduling (GCS) Algorithm

This section presents the pseudo code of GCS algorithm for the energy-efficient scheduling technique used in the proposed GCSM. The input to GCS algorithm is a set of jobs and a set of nodes. Many parameters such as Set of Tasks (*Task*), Set of Nodes (*SN*), Old TaskInformation (*OT*),*Sleep,Power On*, Number of nodes (*N*), Task type (*TBound*), Task deadline (*TDeadline*), and Energy threshold have been initialized. First of all, the tasks will be analyzed and factorized according to the past data stored in Task handler.

The green cloud scheduler module optimally allocates the nodes to the tasks in energy-efficient way and also monitors the deadlines for each task completion. It ends the process when all the jobs have been allocated to the best nodes. The detailed GCS pseudo code is as follows:

Algorithm 1: GCS Algorithm
Input: Set of Tasks, Set of Nodes
Output: Scheduling of Task

```
Begin
    Task← set of tasks
    SN← set of nodes
    Sleep← boolean variable                          /*Has value 1 if node is in running state else 0 */
    Power On← boolean variable                       /*Sets a node to running state by setting Power On value to 1*/
    OT← Boolean variable                             /*If past node, resource and energy consumption values are stored in the database*/
    Input N← Number of Nodes                         /*Number of Nodes*/
    Input TBound                                     /*Type of Task*/
    Input TDeadline                                  /*Task Deadline*/
    Set EF_th                                        /*Energy Thresholds for each node*/
    UnassignedTask← R                                /*Set of unassigned tasks*/
    while UnassignedTask≠Φ do
     for (each r ∈ Task)
       if (OT=0) then                                /* No old task information is available in database*/
         GCS()                                       /*Call to GCS()*/
       end
       else if(TBound=CPUBound) then
           AllocateTaskTo_CPU()                      /*Task is CPU bound. Assign task to CPU */
       end
       else
           AllocateTaskTo_Memory()                   /*Task is memory bound. Assign task to memory*/
       end
    GCS()
    end
  end

GCS()
for i=1 to i≤ N do
   Compute EF_th for each node                       /*Identification of idle nodes*/
   if (EF_th ≫ Node Utilization Level)               / * If EF_th far greater the node utilization level, it means that node is running idle*/
     Set Power on=0                                  /*Eliminate idle nodes*/
   end
end
```

```
LessEnergyConsumeList= {0}                                  /*Create an empty LessEnergyConsumeList */
MoreEnergyConsumeList={0}                                   /*Create an empty MoreEnergyConsumeList */
begin
  for i=1 to i≤ N do                                                          /*Scan through every node*/
    if(Sleep=1) then                                                          /* Check on the Node state*/
      Compute Power_i and EF_i using eq. (3.4) and (3.5)        /*Compute node power and energy consumption*/
      if (EF_i ≤ EF_th)                    /*Compare node energy consumption with energy threshold value set for the node*/
        LessEnergyConsumeList.add(i)       /*Add node with lesser energy then threshold value to LessEnergyConsumeList*/
      else
        MoreEnergyConsumelist.add(i)       /*Add node with higher energy then threshold value to MoreEnergyConsumeList*/
      end
    end
  end

LessEnergyConsumeList.sort()                                                  /*Sort both the lists*/
MoreEnergyConsumelist.sort()
for i=1 to LessEnergyConsumeList.len() do                                     /*Deadline based task allocation*/
    Compute the execution time, ET_i using eq. (3.7)
    for k=1 to k≤ L do
      if (TDeadline_k≤ ET_i)
        Schedule the task on the node i
      end
    end
end
 if scheduling fails on all nodes in LessEnergyConsumeList then
        for i=1 to i≤ N do
          if (Sleep=0)
            Set Power on=1
            Schedule the task on node i
            Set Sleep=1
          end
        end
  end
end
```

The detailed working of the GCS technique has been already discussed above. The input for the algorithm includes the set of nodes (SN) and tasks ($Tasks$). The user specifies the type ($TBound$) and deadline ($TDeadline$) for each task. Firstly, for each unassigned task R in the set of $UnassignedTask$, the Boolean variable OT is checked. The OT value 0 indicates that no initial task utilization and energy consumption values for a node are stored in database, the task is sent to the scheduler unit for allocation. Contrarily, the value of 1 for OT indicates that the task information is available in Task handler database. In case the task information is available, then the tasks are checked for $TBound$ variable to extract the type of task and factorize them into either CPU bound or Memory bound. After the determination of task type, a call to the GCS is made.

When a call to the $GCS()$ function is made, the nodes are initially checked for their utility levels. The EF_{th} for the nodes is also computed that is asymptotically bounded to the node utility levels. The nodes having EF_{th} far higher than their

utilization levels are identified as idle nodes. The benefit of obtaining the node utilization levels to spot idle sitting cloud nodes and in consequence eliminate them to prevent their unnecessary power consumption by setting their *PowerOn* value to 0.

Further on, the energy optimization lists are initialized as empty lists. For all the nodes, the value of *Sleep* variable is checked. The Power-monitoring unit of scheduler computes the power and energy consumption of each node having *Sleep* value 1. GCS algorithm then compares these values with the energy threshold value set for the node. For all nodes with EF_i value lesser than EF_{th}, the nodes are added to a list, LessEnergyConsumeList (initially created as an empty list) while all other nodes with EF_i value more then EF_{th} are added to MoreEnergyConsumeList. Both the lists are sorted in subsequent steps. The first element of LessEnergyConsumeList is the most energy-efficient node while the last element in the MoreEnergyConsumeList is the least energy-efficient node. After getting the list of least energy consumer nodes (LessEnergyConsumeList), the next aim of the scheduler is to allocate the tasks to the nodes that can satisfy the deadline also. For this purpose, the Execution Time, ET, for each node in LessEnergyConsumeList is computed. This value is compared with the deadline for each task, that is, $TDeadline$. The node with ET_i value more then $TDeadline$ for that task is the best node for allocating the task.

In case all the nodes in the LessEnergyConsumeList fail to satisfy a task deadline constraint for, then the task is allocated to the node in the energy saving state. In other words, out of all the nodes available in the system, the node with *Sleep* value equal to 0 is considered for allocating the task.. Thus, the *PowerOn* value for that node is set to 1, that is, it is now set to running state and is therefore used for allocating the task for execution.

3.5 Experiments and Results

In this section, the performance evaluation of the proposed GCSM has been presented. In order to validate our proposed system, a prototype environment as close as possible to a cloud data centre has been simulated. The trials have been performed inside our cloud lab environment. A heterogeneous cloud environment consisting of different virtualized servers to exploit the power of cloud computing was created. Table 3.2 shows the configuration of the cloud environment set up for the evaluation of the proposed model. The initial 3 columns depict the configuration of cloud nodes and the columns 5^{th} and 6^{th} are the power consumption values on each node as measured by Joulemeter [215, 216]. Columns 7^{th} to 9^{th} are the computed values using eq. (3.4), (3.5) and (3.7) respectively.

The workload data was obtained from workload traces available at the Parallel Workload Archive [219]. The trace contains real time workload of various resources/supercomputers including CTC SP2, KTH SP2, LANL CM5, LANL Origin, NASA iPSC, SDSC Par96, SDSC Blue, SDSC SP2. The workloads pertaining to Los Alamos National Lab (LANL) Origin 2000 Cluster (Nirvana) log consisting of accounting information have been used. The trace consisted of user specified require-

ments, including the number of processors, memory requirement, task submission time, task end time. These parameters met the user specified parameters for GCSM. However, the traces in [219] do not have deadline information. Thus, deadlines have been synthetically assigned. For estimating the energy consumption, a time period $t$was defined and was set to on a single day execution. The system runs were carried out in duration of approx. 3 hours.

Table 3.3 shows the cloud data center parameters used. This includes the number of VMs, number of nodes, average power consumption of the data center, average energy consumption in the data center. The total power consumption of the system peaks at about 112 W during the experiment. The average power consumed by the computing units in idle state has been observed to be 43W with all components attached.

Table 3.2: Configuration of Cloud Environment

Processor Used	**CPU Frequency (GHz)**	**Total Cores**	**Cores used in operation**	**Power Consumption (Watt)**	$Power_{Idle}$ **(Watt)**	**Optimized** $Power_i$	EF_i **(Joules)**	ET_i **(min)**
Node 1	2.67	4	3	34.0	10.1	28.025	84.075	2.2
Node 2	3.07	2	2	101.4	43.1	101.40	304.01	1.44
Node 3	2.53	2	1	26.76	14.30	17.530	52.59	12.3
Node 4	2.01	4	4	42.31	10.0	42.310	126.93	3.60
Node 5	2.60	6	5	50.46	5.30	42.933	128.79	4.6
Node6	1.86	2	1	55.4	12.44	33.920	30.92	1.51
Node 7	1.90	12	10	64.10	15.56	56.01	168.03	2.12
Node 8	2.66	2	2	67.6	29.01	67.60	202.80	2.31
Node 9	2.40	4	3	78.32	22.34	64.325	192.96	1.12
Node 10	2.01	4	3	91.56	20.30	73.745	221.24	1.15
Node 11	2.5	4	2	93.25	15.11	54.18	164.4	5.30
Node 12	2.13	4	4	87.02	13.57	87.02	84.02	2.56
Node 13	2.2	4	2	70.87	18.76	44.815	134.45	4.32

Table 3.3: Prototype Cloud Data Center Parameters

Parameter	**Value**	**Comment**
No. of VMs	25-50	Enabling Virtualized Cloud
No. of Nodes	10-15	Nodes/Servers running VMs
Average Energy Consumed	392 Joules	Average energy consumed during completion of jobs
Average Computational Energy	40 Joules	Average energy spent for computing the placements of jobs
Average Power drawn	112 W	Power rate of nodes

Case 1: Scheduling Tasks Using First Come First Serve (FCFS) Basis The initial run of the cloud system has been done using FCFS order of task execution to estimate the energy consumption done by it without running the proposed GCS algorithm. The criteria for choosing FCFS is due to its non pre-emptive behavior for task allocation. The system has been tested for the energy consumption done by it and level of performance achieved when using FCFS based task execution order.

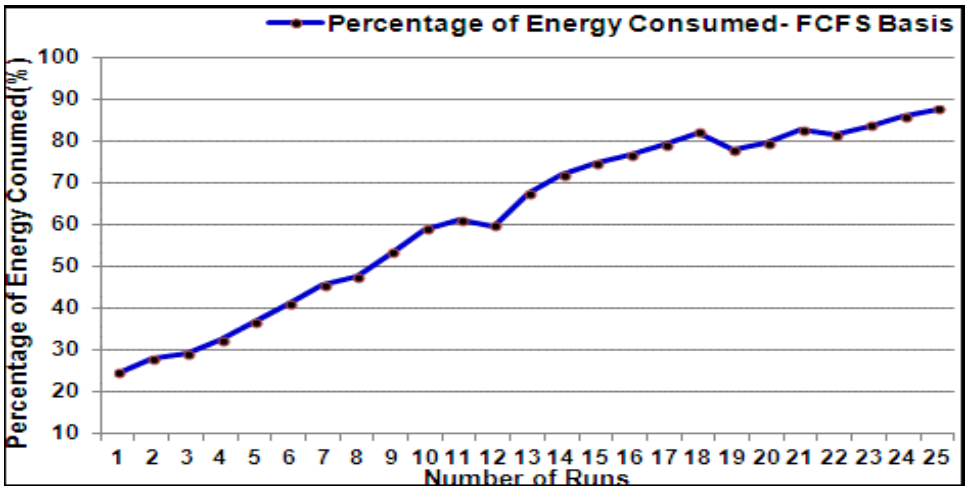

Figure 3.3: System Energy Consumption in Multiple Runs- FCFS Basis

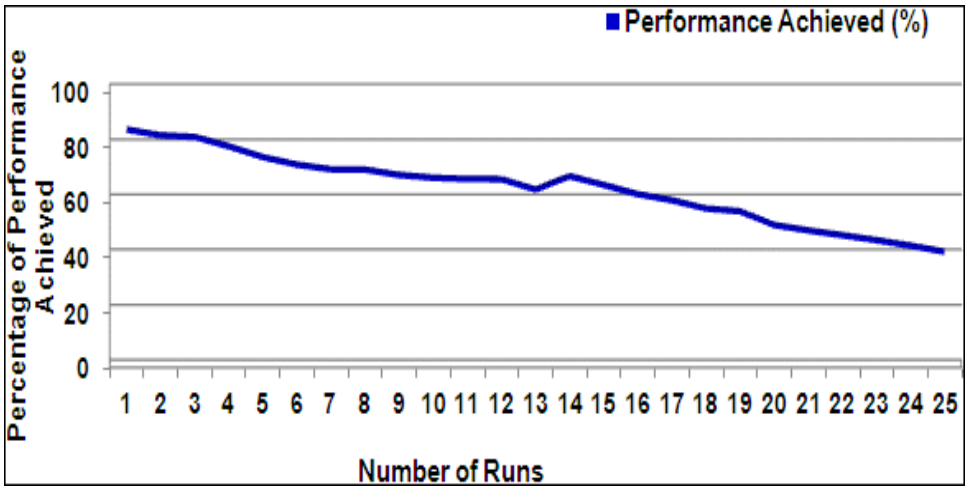

Figure 3.4: Percentage of Performance Achieved by the System- FCFS Basis

Based on the analysis, it has been observed that FCFS suffers from the growing energy levels and degrading performance levels with every next iteration run of the system (Figure 3.3 and Figure 3.4). This happens due to the non-parallel pattern followed for the task execution resulting in poor node utilization levels. The poor

utilization pertains to the low utility levels of resources on nodes indicating that most of the nodes were sitting idle drawing unnecessary power. The energy consumption thus rises on the server and keeps increasing thereby negatively affecting the performance in long term and in multiple runs.

Case 2: Scheduling of the Deadline-constrained Tasks as FCFS Basis

The experimental evaluation of the cloud system has also been done using FCFS technique when the tasks are deadline constrained. However, the attempt to allocate the tasks as per the energy optimization was not done in this scenario. Eventually, the tasks are scheduled as and when they enter the system as per the FCFS pattern followed for the task execution and consequently no stress could be given to the deadline fulfillment. In other words, most of the tasks could not be completed within the deadlines specified by the users. In order to track the status of the percentage of deadlines fulfilled by the system, a Deadline Fulfillment Factor is defined. The deadline fulfillment factor here indicates the number of task deadlines satisfied per system runs (as depicted in Figure 3.6).

It can be observed from Figure 3.6 that with the consecutive runs of the system, the deadline fulfillment factor drops down incessantly. This happens due to FCFS pattern of task execution that also involves non pre-emptive scheduling policy thereby resulting in delayed task execution irrespective of deadlines specified for the tasks.

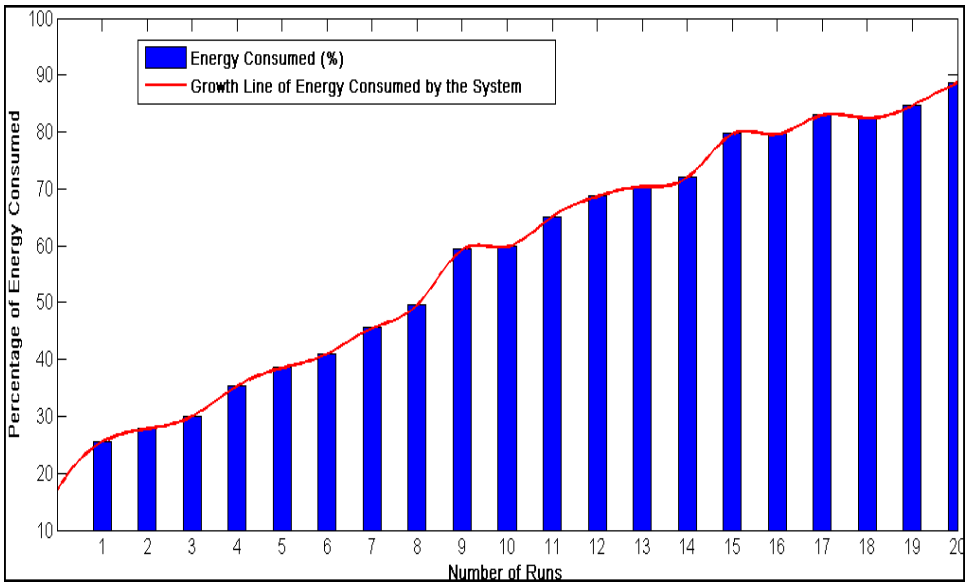

Figure 3.5: Energy Consumption Trend when using FCFS- Deadline Based Task Scheduling

In due course, the energy consumption in the system also continues to grow with each system run as depicted by the energy trend in Figure 3.5. The percentage of energy consumed by the system continues to rise. This increase in the energy consumption results from non-energy aware task allocation carried out in this scenario. The ascended energy consumption further results in overall performance degrada-

tion. The performance graph is given in Figure 3.7. It can be observed that initially the system performance is better, and then drops down little and gradually drops with intensive system energy rise.

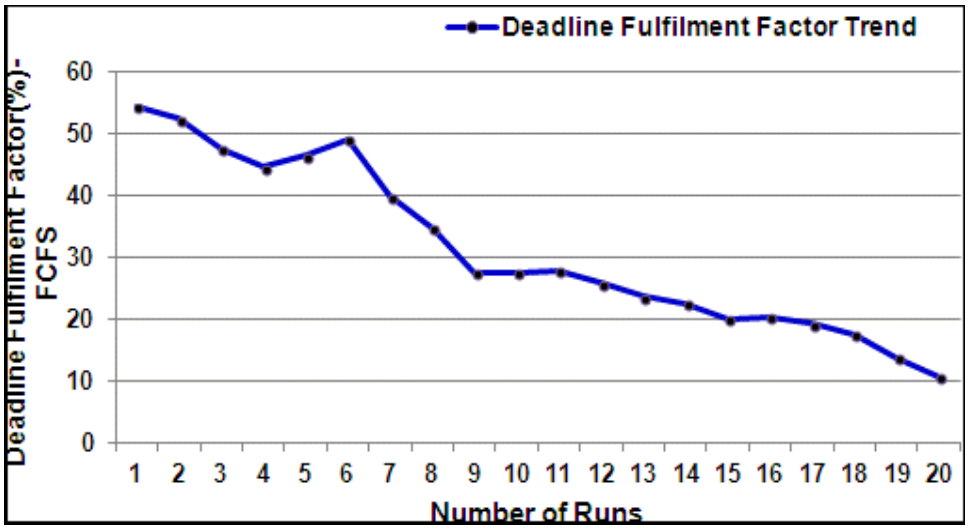

Figure 3.6: Deadline Fulfillment Factor Trend- FCFS when Deadline Constrained

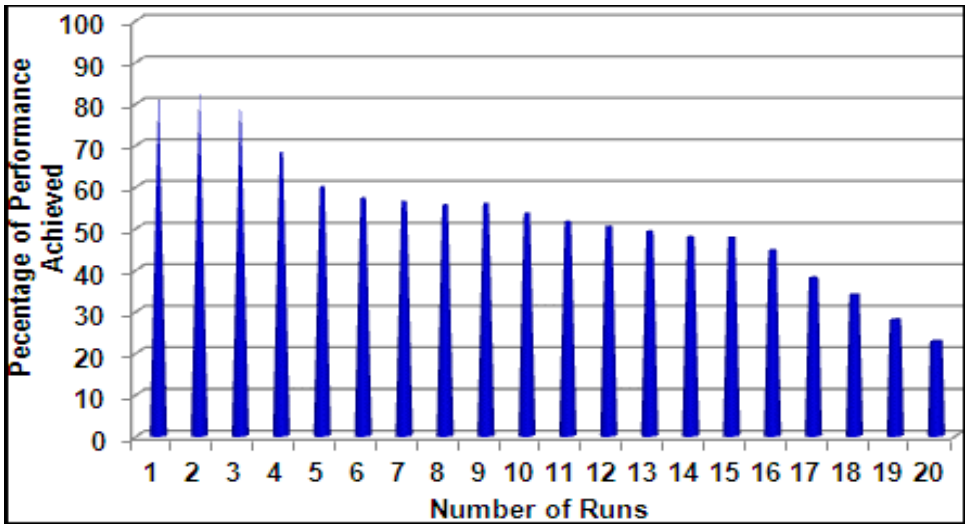

Figure 3.7: Percentage of Performance Achieved- FCFS Basis when Deadline Constrained

Case 3: Energy-Aware Scheduling of the Deadline-Constrained Tasks using GCSM

This case involves the implementation of the proposed GCSM and the results obtained thereof. The energy-aware allocation of deadline-bound tasks has been evaluated. Figure 3.8 depicts the comparative analysis of the power and energy consump-

tion done by the system nodes over a period of time. It is evident that overall the energy consumption done by the nodes remains minimal thereby achieving energy-efficient operation. Moreover, the execution time for the processing of the tasks remains low satisfying the overall response time of the system and this helps to enhance the system quality of service in terms of the earlier completion of the tasks in particular within the deadline imposed by the users.

Figure 3.9 shows the percentage of energy consumption of the cloud system. It can be analyzed that the overall energy consumption done by the system minimizes. The values pertain to the energy consumption done by the active VMs running on different cloud nodes. It depicts the total number of active VMs during different runs and the corresponding energy consumption done by the system.

Figure 3.10 depicts the percentage of deadlines completed as per the tasks, fulfilled by GCSM. The percentage is obtained by analyzing the number of deadlines fulfilled and is depicted using Deadline Fulfillment Factor. Also, a deadline fulfillment growth line is obtained that indicates the intensification and satisfaction line for the task deadlines. Approximately 82% of tasks complete within the deadlines specified by the users.

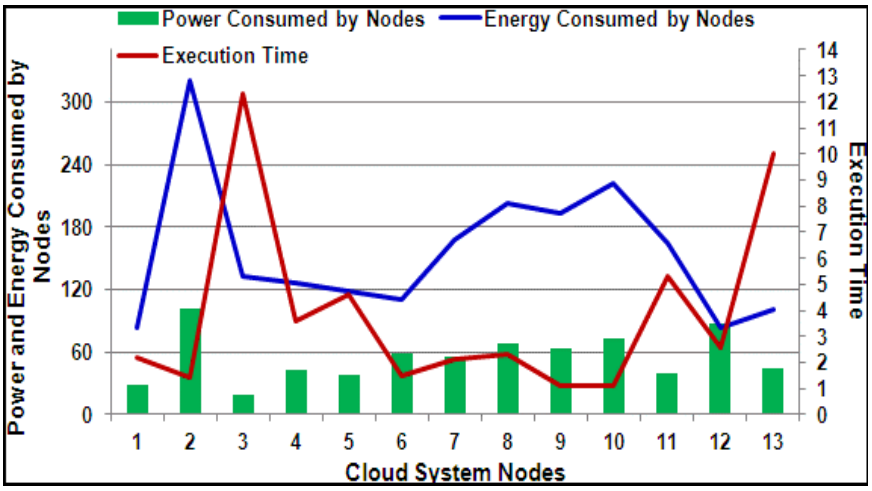

Figure 3.8: Power, Energy and Execution Time Monitoring Result of the Cloud System Nodes

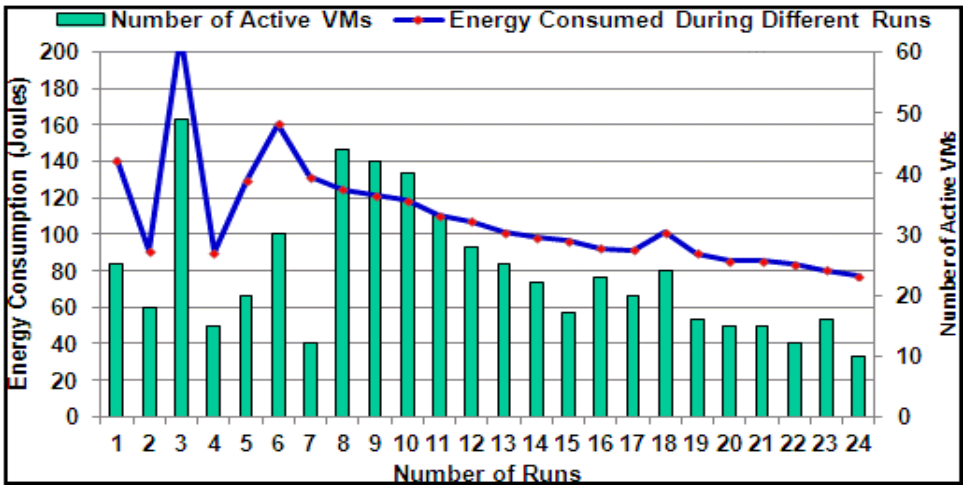

Figure 3.9: Overall System Energy Consumption during Different Runs of the Cloud Environment

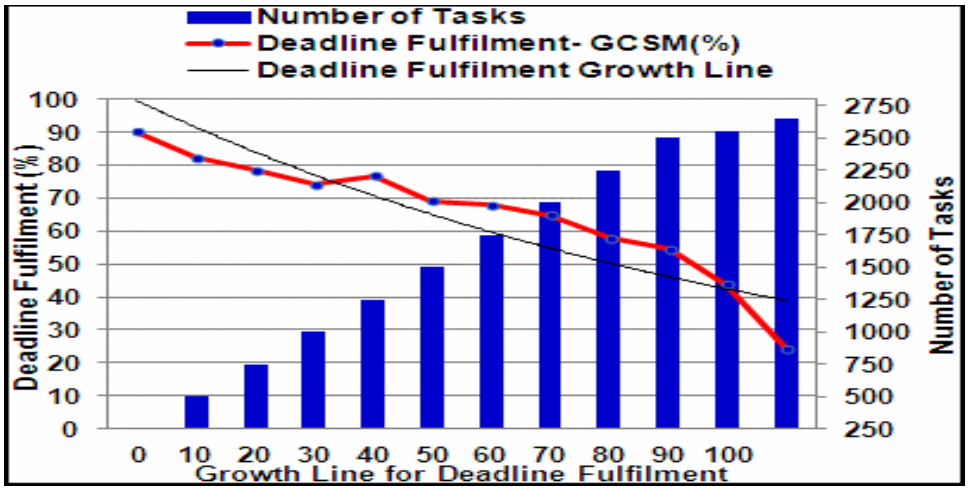

Figure 3.10: Deadline Satisfaction Percentage of GCSM

Case 4: Experimental Comparison of proposed GCSM with Priority Based Scheduling Technique and SLA based Resource constraint VM scheduling

The proposed GCSM is experimentally compared with another two techniques proposed in [150] and [171]. The experimental results are shown in the Figures 3.11, 3.12, and 3.13.

Figure 3.11 depicts levels of energy consumed by GCSM and other two techniques, namely, SLA-Based Resource Constraint VM Scheduling and Priority Based Scheduling respectively. It is inferred that GCSM attains lower energy consumption than the other two scheduling techniques and achieves lesser execution time.

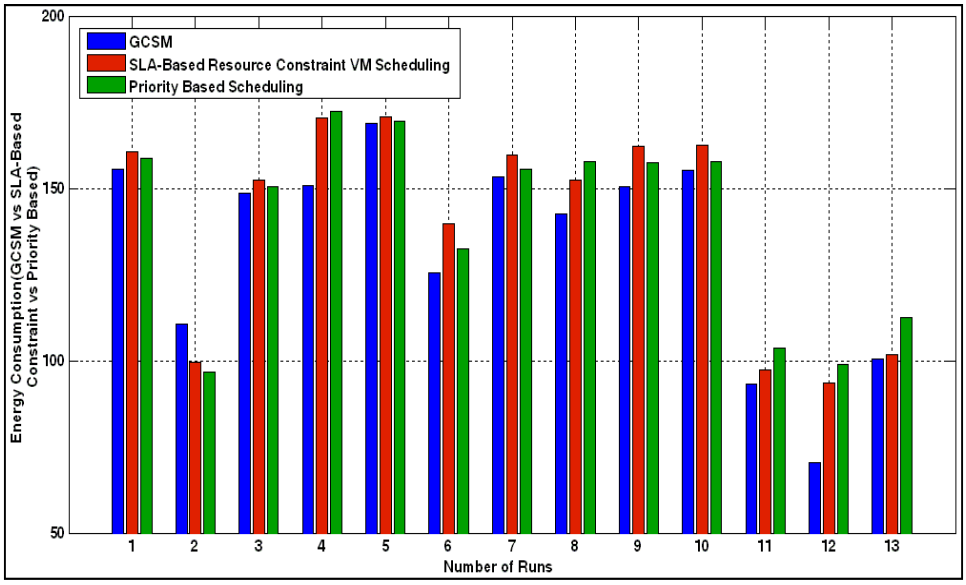

Figure 3.11: Energy Consumption Comparison (GCSM vs SLA-Based Resource Constraint VM Scheduling vs Priority Based Scheduling Scheme)

The graph in Figure 3.12 shows the percentage of energy savings obtained by all the three techniques. Overall, GCSM achieves 71% of energy savings compared to 67% and 65% of other techniques.

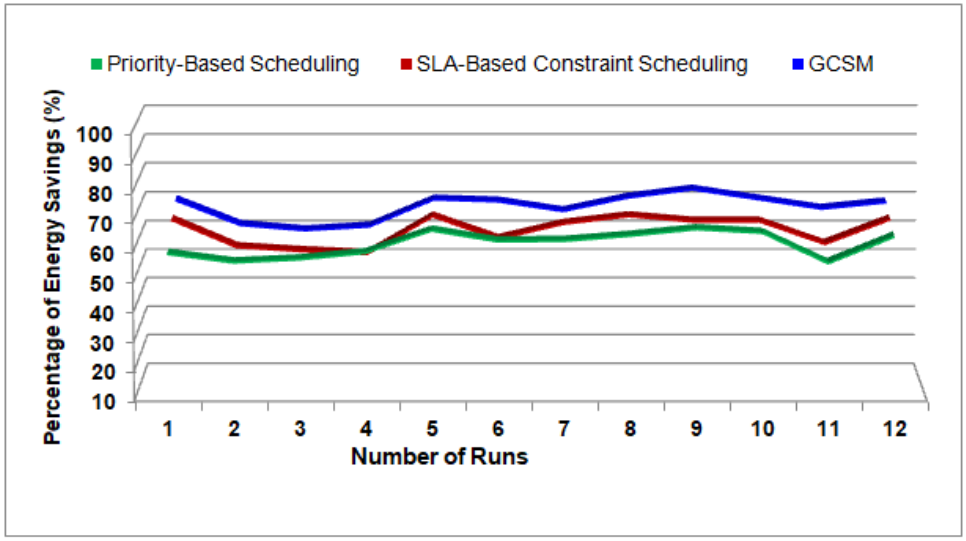

Figure 3.12: Percentage of Energy Savings Achieved (GCSM vs SLA-Based Resource Constraint VM Scheduling vs Priority Based Scheduling Scheme)

The energy-aware allocation of tasks performed by GCSM prevents performance degradation and improves the overall system utility levels by lowering the number

of idle nodes. Figure 3.13 shows the percentage of performance levels obtained by GCSM and other two techniques. All in all, GCSM offers optimal performance compared to other two techniques as its energy consumption levels are comparatively low.

Thus, the proposed GCSM system helps to achieve energy efficiency and accurate task allocations within the time bounds as per the user requirements.

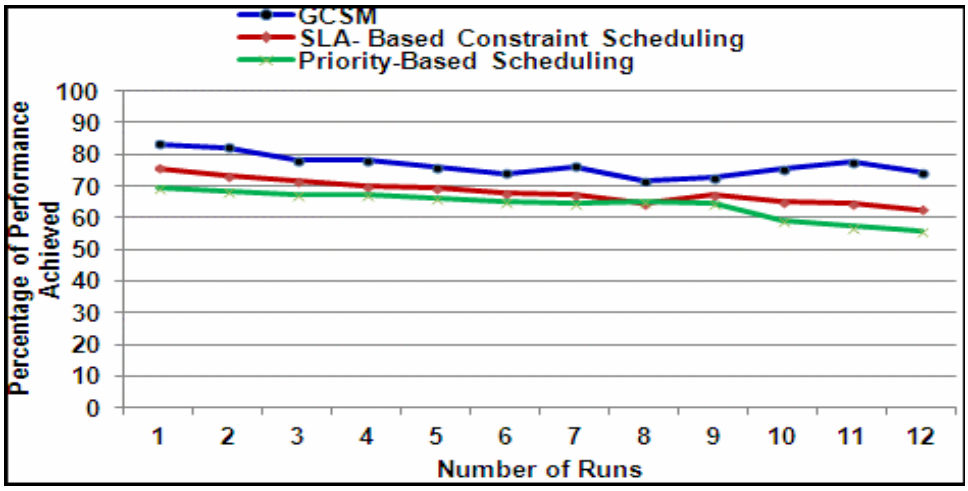

Figure 3.13: Percentage of Performance Achieved (GCSM vs SLA-Based Resource Constraint VM Scheduling vs Priority Based Scheduling Scheme)

3.6 Conclusion

This chapter discussed the proposed Green Cloud Scheduling Model (GCSM) with its detailed functional characteristics. The proposed GCSM implements a green cloud scheduler unit that makes task allocation and scheduling decisions using Green Cloud Scheduling (GCS) algorithm. GCS algorithm considers the energy-aware capability of the heterogeneous cloud nodes and also considers the nodes that can fulfill the task completion time limits as imposed by the users. In other words, GCSM not only achieves energy efficiency but also tends to offer desired QoS to the users in terms of task deadline metric satisfaction. GCSM performs energy-aware task allocation and scheduling in a faster manner.

Exclusively, GCSM is an energy and deadline optimization model feasible in situations where environmental sustainability, faster processing and constrained timing are desired. The model's elaborative performance evaluation has been conducted and discussed in detail. The experimental results predict that GCSM achieves energy savings up to 71% and almost 82% tasks are completed within the deadline constraints as imposed by users pertaining to each task.

The next chapter presents the extended GCSM model named as GreenSched model, the design based on the GCSM. Comparative to GCSM, GreenSched is an energy-aware model that facilitates deadline- and budget- constrained task allocation in cloud data centers.

Chapter 4

GreenSched: A Proposed Intelligent Energy Aware Scheduling Model

The previous chapter discussed the design of proposed GCSM that performs allocation and scheduling of tasks to energy-aware cloud nodes. GCSM tends to optimally provision and schedule the tasks entering the system while maintaining minimal energy consumption, maximizing utilization of resources, fulfilling task deadlines and averting degradation in the overall performance. GCSM is prominently feasible and useful in several applications considering energy and deadline optimization such as banking and financial applications and services.

In this chapter, GreenSched model, an extended model based on GCSM has been proposed. GreenSched model implements a machine learning technique that proactively allocates and schedules deadline-and-budget constrained tasks to identified energy-aware nodes. It is a Green scheduling model as it focuses on energy-efficient scheduling. It implements Forward-only Counter Propagation Network based intelligent scheduler unit where cloud nodes are initially adjudged for energy-awareness and then for deadline-and-budget fulfilment potential. Contrary to GCSM, GreenSched model is an intelligent machine learning model that apart from energy efficiency also assures high QoS, that is, gratifies both deadline and budget. In other words, it assures contentment of deadline and budget restrictions imposed by users particularly when energy, deadline and budget are closely inter-related parameters.

Section 4.1 presents an insight into GreenSched model with its significance and relativity to GCSM. Section 4.2 presents GreenSched's system and operational model followed by its power model in Section 4.3. In the Section 4.4, an in-depth exploration of GreenSched's scheduler unit, its scheduling scheme and flow diagrams are given. The experimental study and results for GreenSched scheduling technique and comparative analysis with some recent works are presented in Section 4.5. Section 4.6 summarizes the conclusion for this chapter.

4.1 GreenSched- An Introduction

GreenSched model is an intelligent, cloud-based, energy-aware scheduling model for deadline-and-budget constrained tasks. GreenSched model realizes energy efficiency while offering QoS by fulfilling the deadline and budget specifications imposed by the users. It operates in the cloud environment comprising of heterogeneous and multi-core nodes and schedules the tasks on them for processing while optimizing the energy consumption. Initially, it discovers energy efficient nodes within the cloud system, that is, the nodes which consume lesser energy compared to other nodes and then autonomously allocates the tasks to them depending upon the further verification of the node's capacity to fulfil task deadline period and budget specifications. In addition, GreenSched also prevents performance degradation by improvising the node utilization levels keeping the energy demand to minimal.

4.1.1 Exclusivity Characteristics of GreenSched Model

GreenSched intelligently and proactively allocates and schedules the deadline-and-budget constrained cloud tasks to determined energy conscious or energy aware nodes. Exclusively, it implements a Forward-only Counter Propagation Network (CPN) based intelligent scheduler unit, called as CPN-based Green Cloud Scheduler (CGCS). CGCS is a machine learning model that runs Forward-only CPN-based scheduling (FCS) technique to identify, classify and cluster the best nodes for the task allocation process. The best nodes are the one consuming least energy (or energy aware nodes in this case) and possessing capability to fulfil deadline and budget restrictions. Additionally, several other factors signify the uniqueness of GreenSched model derived from following characteristics:

i GreenSched provides autonomic and dynamic provisioning and scheduling of deadline-and-budget constrained tasks facilitated by its intelligent scheduling unit, CGCS.

ii The CGCS unit is called intelligent because it schedules the tasks to the priorly adjudged or predicted energy efficient resources according to the deadline and budget constraints imposed by the users thereby offering faster executions and economic advantage.

iii CGCS inherits from Forward-only CPN model which is well-known for its unfussiness, easiness and simplicity. CGCS possesses easy training characteristics and excellent statistical model depiction of the input environment [220, 221].

iv The major criteria for choosing CPN is its capability to learn fast as compared to other networks due to its efficient learning algorithm that helps to solve different forecasting (predicting the energy efficient nodes in this case) and judgemental (observing best nodes to fulfil deadline and budget apart from energy parameter in this case) problems [222–224].

v GreenSched optimizes the energy consumed by the memory and the processor units. This is because CPU and memory are primarily responsible for increased energy consumption within a node. GreenSched exploits the capacity of multiple cores in the processor to optimize the energy consumption levels.

vi In GreenSched, CPU and memory are considered as the critical parameters of interest as it is important to analyse and handle the impact of time in terms of the energy consumed by CPU and memory because the increase in energy affects the processing time of the processor and memory.

vii The time and cost for running the tasks and the energy consumed by each of these tasks varies from node to node. Consequently, the nodes with minimum energy consumption value, lesser cost and least task execution time are in demand in the system.

viii GreenSched offers budget optimization and as well as offers economic scheduling for the service providers since it prevents energy related expenditures by achieving energy efficiency and consequently averting hardware oriented cooling equipment costs and other incurrence costs [36, 65, 66, 69, 70, 225, 226].

4.1.2 Similarities and Dissimilarities between GCSM and GreenSched Model

GreenSched and GCSM are both energy aware allocation and scheduling models operating in cloud infrastructure while optimizing QoS in terms of variable metrics. Actually, GreenSched is an extended model based and similar to capabilities of GCSM. Likewise, GCSM, GreenSched also offers scheduling of deadline-constrained tasks to the energy-aware cloud nodes in a virtualized cloud environment while targeting time and space parameters. Both these models are complimentary for both the cloud vendors as well as the cloud users. This can be inferred from the fact that achievement of energy efficiency prevents performance dilapidation and enhances resource utility levels while energy efficiency combined with minimal budget and execution speeds yields benefits for the cloud users.

Contrary to GCSM, in addition to energy and deadline optimization, GreenSched offers budget optimization as well. It involves autonomic provisioning and scheduling while achieving higher resource utilization levels and without affecting system performance. Distinctively, the CGCS unit of GreenSched running FCS technique is unique feature of GreenSched. FCS is a machine learning algorithm that can predict the energy-aware capability of nodes priorly. FCS intelligently identifies the least energy consuming nodes out of all the nodes for task allocation and then proactively performs scheduling on them while optimizing task deadline and budget constraints. Initially all the nodes are clustered and classified as per their energy consumption levels upon comparison of their energy consumption values with a threshold value. The nodes in the less energy consuming cluster are then adjudged for deadline and budget fulfilment based on scrutiny of their processing speeds and execution costs.

4.2 Proposed GreenSched System Model

GreenSched enables the system managers to manage the resources and servers energy efficiently. By allocating the user requests to the energy efficient nodes combined with the attempts to minimize their execution timings on them, it tends to maximize the node utilization levels. This improvement in the utility levels helps to leverage more efficiency and prevents performance degradation that can otherwise occur with inefficient utilization [101]. Additionally, the pricing calculation mechanism used in GreenSched helps to yield economic benefits for running the tasks within the user-defined budget. For accomplishing its objectives, the FCS technique in CGCS makes dynamic scheduling decisions using two algorithms, "TaskClassification Algorithm" and "BestNode-Task Pair Algorithm". These algorithms basically classify and cluster the nodes into more or less energy consuming nodes based on analysis of their power consumption values, and then predict the forecast result for allocating the tasks.

4.2.1 GreenSched Model

The GreenSched model consists of variable components. Figure 4.1 represents the component and application structure map for GreenSched. It depicts the variable components in the GreenSched model and the prime applications or techniques running in them. The relationship and information exchange among the components and applications has been represented in the Figure 4.2. Each GreenSched component has a unique functionality as discussed below:

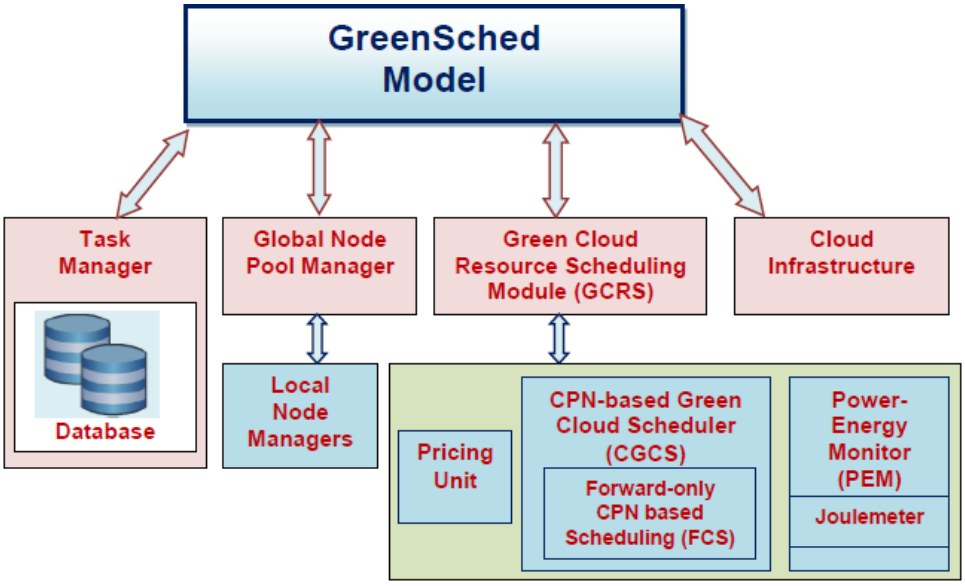

Figure 4.1: High-level Component and Application Structure Map of GreenSched

- Cloud Users: The cloud users submit their tasks for execution to the GreenSched. Each user submitting the tasks has to specify the task type including the resources required by it. They have to indicate the deadline (in seconds) for task completion and budget (in dollars) to complete the task within it.

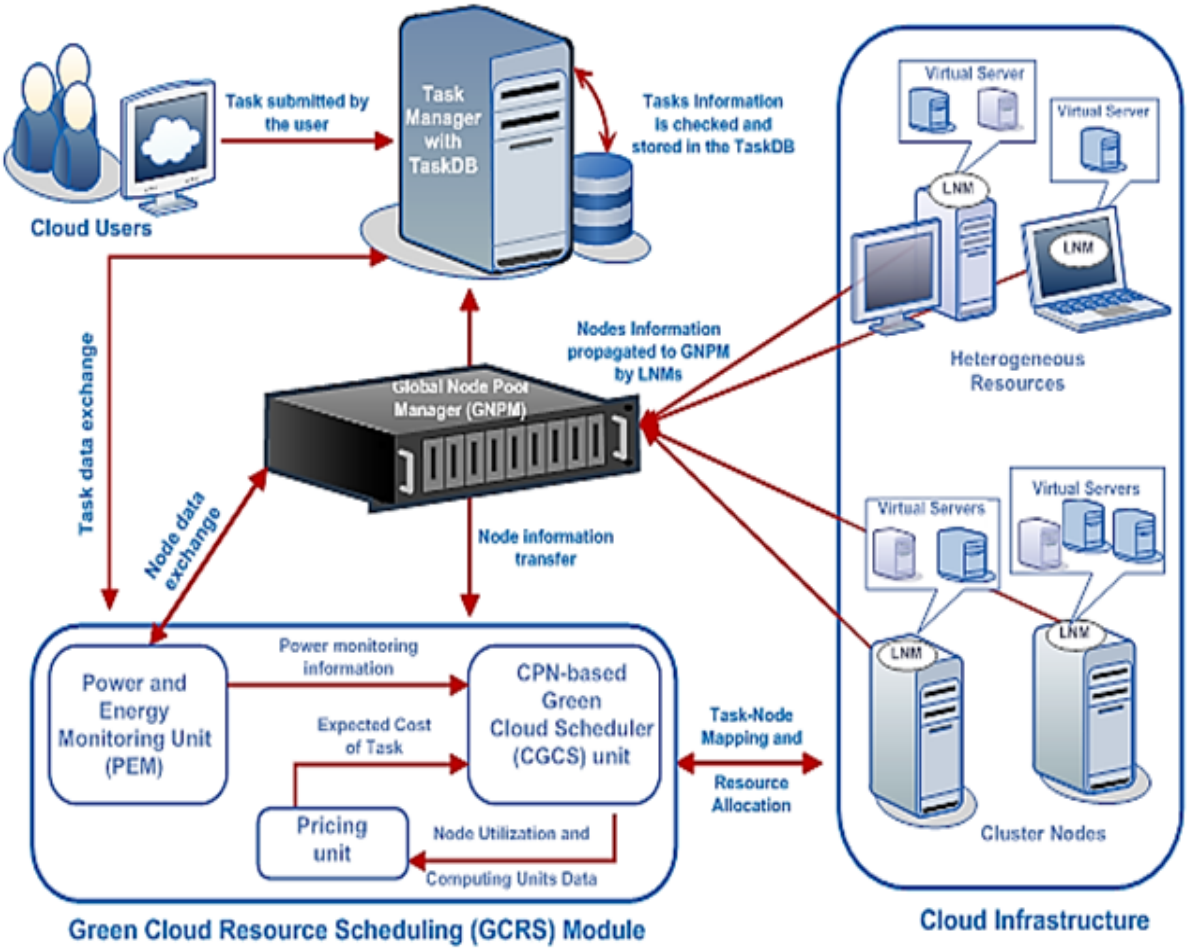

Figure 4.2: Proposed GreenSched Model With Flow Exchange

- Cloud Infrastructure: It is composed of the actual physical infrastructure with heterogeneous computing systems, clusters and virtual server operating in a cloud data center.
- Task Manager with TaskDB: It handles the tasks submitted by the users and analyses their resource requirements; deadline and budget information as provided by the user. It also examines the existing tasks running in the system and stores

their information in its database called as TaskDB. TaskDB is a repository to hold the information such as resources used, execution details including time and cost, resource utilization levels and energy consumption data of the tasks that are already executing in the system or have completed execution and have used the system resources. TaskDB is useful in analysing past task allocations done by GreenSched. This further aids in taking future energy-aware task allocations decisions.

- Global Node Pool Manager (GNPM): Every node related information is handled and maintained by the GNPM based on the data received from the Local Node Managers (LNMs) sitting on each node. The GNPM keeps track of the nodes status (whether in active or energy saving mode), the total number of resources with each node, the number of resources available for allocation and those already allocated to different tasks. Each node is also running variable number of VMs. The VMs that are being utilized are powered on and are visible as highlighted ones in the above Figure 4.2 while the VMs that are sitting idle are in sleep mode to save energy are in dimmed lighter colour.

- Local Node Manager (LNM): The LNMs sit on each node and monitor their host nodes. LNMs collect all the information about the computing capabilities of each node in terms of the resources available and number of VMs running on each node. They update GNPM with the entire node information about its resource utilization and number of VMs running on each etc.

- Green Cloud Resource Scheduling (GCRS) Module: This is the core module of the GreenSched. It schedules the tasks to the most energy efficient and deadline-and-budget satisfying nodes. It has 3 key sub- components:

  - Power and Energy Monitoring (PEM) unit: It computes the power consumption on each node and delivers the same to the scheduler, that is, CGCS.
  - Pricing unit: The pricing unit is responsible for computing the cost of running a task on any node. It helps in cost based optimizations and considers the resource and task characteristics to adjudge their capacity for the fulfillment of the budget restriction. When a new task arrives at the GCRS module, it computes the expected cost of task on each node using the information related to the number of computing units needed by it, the cost set for using each computing unit and the expected time the task will run on the node. It then transfers this information to the CGCS unit.
  - CPN-based Green Cloud Scheduler (CGCS) unit: CGCS is an intelligent scheduler that obtains the best node-task pair out of all the nodes in the cloud and schedules the task on that node. The best node is the one that can fulfill the minimum energy requirement and can accomplish the task within the specified deadline and budget constraint. For obtaining the best node, CGCS identifies the most energy aware nodes and then proactively predicts the best node out of all the other before allocating the task to it.

4.2.2 Operation of GreenSched Model

GreenSched works through different phases and modules that uphold the overall operation of the energy aware allocation and scheduling of tasks to the resources. GreenSched has the capability to process variable workloads involving multiple, heterogeneous tasks. Physically, GreenSched supports an underlying cloud infrastructure with varied nodes that are assumed to provide different computing capabilities; are built on multiple cores and possess variable power consumption values.

Whenever a new task enters the system, it is stored in the task manager. The task manager analyses it and stores the required information in its database. The task is then forwarded to the GCRS module. The GCRS module has all the node information obtained from the GNPM. Based on this node information, CGCS component of GCRS module discovers energy aware nodes and then evaluates their execution times. In the meantime, the pricing unit sends the cost information for running task on each of the computed energy efficient nodes to the CGCS unit. Finally, CGCS unit astutely scrutinizes the best node out of all the energy efficient nodes. The best node is the one that is not only most energy efficient but can fulfil the deadline and monetary constraint associated with a task.

It is assumed that although every task entering the GreenSched follows maximum possible phases given below but depending on the utility, a task may or may not follow all the phases. The utmost likely phases for energy efficient allocation and processing of a task in GreenSched include:

A. **Task Submission and Analysis Phase**
Each cloud users can submit their tasks to the GreenSched through the Task Manager unit specifying the resources required to run each task, the respective deadline and budget for the same and the type of task (whether CPU- intensive or Memory- intensive). The user will be entering the required number of CPUs, required amount of memory, task deadline, task budget, and the task type. With the termination of the first task submission phase, the tasks can be adjudged to be either CPU-bound or memory-bound depending on the type of resources demanded during the task submission.

Every task entering the system is analysed and checked by the task manager for possible entry into TaskDB. Initially, TaskDB holds only the task information submitted by the user that includes task resource requirements, task type, task deadline and budget. After a task gets processed, additional information related to the task, such as the energy consumed by the task, resources used, resource utilization levels of the task are also propagated to the TaskDB by the CGCS unit. In future, if any similar task arrives, then task manager will query the TaskDB and forward this already stored task information to CGCS. The stored task information aids CGCS in future scheduling decisions. In case, no task data is available in the database, a non-availability reply is sent back to the task manager.

Depending on the task information given by the TaskDB, the task manager will either self process the task and forward it to GCRS module along with the

task details or will send it to the CGCS directly for deciding, choosing the most appropriate node for allocation.

B. **Node Management Phase**
It is imperative to track the status and operation of the nodes in order to make effective scheduling choices. For scheduling any task on a node, it is essential to inquire whether a node is free or has the capacity to run new task energy efficiently. The node information is provided to CGCS by GNPM. GNPM is actually responsible for the node management. It has employed LNMs sitting on each node to trail the node operation. The LNMs maintain information regarding the operating environment on their host nodes such as the total number of hosted VMs on them, the number of VMs active, the resources being used by them and the number of available resources for future task allocations.

The LNMs keep GNPM regularly updated about all this information along with the resource utility level for each node and the status of a node. The node status indicates the node state whether it is in running state or in energy saving state or is sitting idle drawing power at leisure. For this purpose, GNPM needs to analyse *Sleep*, a Boolean variable associated with every node. The value of *Sleep* if 0 indicates that the node is in sleep mode or energy saving mode while a value of 1 indicates the node is in running state. In order to bring a node back to running state or to activate any new node, a Boolean variable for a node *PowerOn* can be set to 1. GNPM can also individually control the VMs operating on a node. It can either set a VM to a suspended state or an active state using the VM properties menu depending on the requirement to save energy or to prevent unnecessary power withdrawal by any VM while sitting idle.

C. **GCRS Module (Scheduling Phase)**
The GCRS module has 3 primary components: PEM unit, CGCS unit and the Pricing unit. The CGCS is the core and main working entity that takes all the scheduling decisions. It finds the best node-task combinations for scheduling of the tasks on the nodes that it proactively predicts as the most energy efficient and deadline-and-budget fulfilling nodes. For this purpose, it requires the variable input information from other components such as PEM unit, GNPM and the Task Manager.

The PEM unit computes the power consumption for each node and forwards it to the CGCS unit. All the node information and task information maintained by the GNPM and Task Manager units respectively are sent to the GCRS module that forwards them to the CGCS unit. The node information (node status, resources with every node etc.) and the task information (resources required, user- specified deadline, task budget, task type) are actually required by CGCS unit for scheduling the tasks on most appropriate and energy efficient nodes without violating the deadline and budget for each task. The detailed working of CGCS unit is discussed in section 4.4.

4.3 Problem Formulation and Mathematical Model

This section discusses the problem of focus for the GreenSched model and formulates the power model for the same.

4.3.1 Problem Formulation

The problem of finding the best node-task pair that can accomplish task execution in most energy efficient and deadline-and-budget satisfying manner is a highly complex job performed by GreenSched. GreenSched dynamically and intelligently schedules the tasks to the resources that can successfully achieve its primary goals. Furthermore, it takes the scheduling decisions without compromising the performance. Usually, fluctuations in the system performance levels are observed with the rise in the energy consumption levels. This can be inferred from the fact that the increase in the consumed energy causes degradation in the overall system performance especially with the constrained energy availability. Overall, energy management plays an important role in context to performance maintenance. Thereby, the allocation of the tasks to the energy efficient nodes helps to prevent performance degradation. Also, the task provisioning done as per the deadline and budget impositions helps to upgrade the performance in terms of QoS, response time and yields economic benefits.

In actual terms, the purpose of the 'BestNode-Task Pair algorithm 'of GreenSched is to select the finest node (with least energy consumption value) that can fulfil the user-defined deadline and budget constraints for each task as compared to all the other available nodes [227]. Analytically, it locates the best node with least Energy Consumption Function (ECF) that is computed based on the power consumption values measured by the Power and Energy Monitoring (PEM) unit implementing a Joulemeter [215, 216].

4.3.2 Comprehensive Power Model Description

Mathematically, GreenSched targets to minimize the Energy Consumption Function (ECF) of a node together with the satisfaction of task deadline and task budget. The ECF value for each node is calculated after measuring the power consumption done by each node measured by the PEM unit during t units of time. Analytically, the problem can be formulated comprising of the following elements:

- The data center components forming the cloud infrastructure consists of set of heterogeneous resources, cluster nodes and server nodes. A set $\{s_1, s_2, s_3, ..., s_N\}$ of nodes such that $S = \{s_i | 1 \leq i \leq N\}$ is the collection of N multicore nodes;
- a set $\{v_1, v_2, v_3, ..., v_n\}$ of VMs such that $P = \{v_j | 1 \leq j \leq M\}$ is the collection of M virtual machines (VMs);
- and a set $\{r_1, r_2, r_3, ..., r_L\}$ of tasks to be run in VMs such that $R = \{r_k | 1 \leq k \leq L\}$ is the collection of L tasks.

- CU= $\{a_1, a_2, a_3, ..., a_u\}$ be the number of computing units
- PC_i (stands for Power Consumption of node i) is the power consumed by each node consisting of one or multiple VMs at time instant t
- PC_{Idle} is the power consumed by each node in free time when sitting idle
- d is the number of cores used in the operation of a node
- cmp is the total number of cores in a node
- ECF_i (Energy Consumption Function of node i) is the energy consumed by a node given by the product of PC_i during t time instant
- ECF (stands for Energy Consumption Function) is the total energy consumed by the entire system
- ECF_{th} (stands for Energy Consumption Function Threshold) is the energy consumption function threshold computed as per the maximum energy consumed by a node
- ETF_i (stands for Execution Time Function for node i) is the total execution time taken by a node running j VMs for executing k number of tasks
- $ECT(k, i)$ (stands for Expected Cost of Task) is the expected cost of running a task k on node i
- $TET(k, i)$ (stands for Task Execution Time) is the predicted task execution time for running task k on node i
- NOI_k (stands for Number of Instructions) is the number of instructions in task k
- UR_i (stands for Utilization Rate) is the utilization rate of node i
- NCU_k (stands for Number of Computing Units) is the number of computing units needed by task k
- $UTDeadline_k$ (stands for User specified Task Deadline) is the user specified task deadline
- $UTBudget_k$ (stands for User specified Task Deadline) is the user specified task budget
- ECF_p (stands for Energy Consumption Function) is the energy consumption function for each node p in the winning cluster
- ETF_p is the execution time function for each node p in the winning cluster
- $ECT(k, p)$ is the expected cost of task k on node p in the winning cluster
- $TET(k, p)$ is the execution time of task k on node p in the winning cluster
- UR_p is the Utilization rate of node p in the winning cluster
- $TResourceReq$ indicates the resource requirement of a node
- $OldTaskInformation(OT)$ is a Boolean variable to check old task information availability in TaskDB
- $Sleep$ specifies a Boolean variable to check node status(active or in sleep state)
- $PowerOn$ is a Boolean variable to set a node to either active or non-active state

GreenSched is based on certain assumptions listed below:

- The system environment comprises of pool of server nodes with variable number of VMs running on each node. The number of VMs running on the nodes does not change during a time period T.

- The tasks are assumed to be independent and they arrive at the task manager

through different cloud users and are to be allocated on the server nodes operating in the system.

- Each task is subjected to a fixed deadline and budget that are decided and specified by the users.
- No pre-emption behavior can be adopted for the tasks already running in the system.
- Only CPU and memory are the major computing units consuming maximum power in each node. In other words, the energy efficiency optimization is restricted to the processor and memory power optimization pertaining to in fact handling time-space parameters as well.
- The powers of the computing units are cumulative when taken together. Also, for every computing unit CU_a, its consumed power $PC(CU_a, t)$ is either null (if not active) or maximal (in case active) [68].
- The power consumed by the network components and storage units is assumed to be negligible.
- Each node is consuming some power during the inactivity period. This idle power is consumed when a node is either active but only sitting with a fixed screen image and no user interaction otherwise an idle application.
- GreenSched prefers ECD (Energy, Cost and Deadline) based optimization policy. Overall, the selection of a node for allocation and scheduling is done as per its capability to first optimize the energy, then cost and then deadline.

On each node, the power consumed can be partitioned into idle power and active power. PC_{Idle} is the Power Consumption Idle, that is, the node power consumed in idle state, assumed to be constant when no task executes. $PC(CU_a, t)$, $1 \leq a \leq U$, is the power consumption of computing units at time instant t, if active. Thus, PC_{Idle} and $PC(CU_a, t)$ values are monitored to compute total PC_i (Power Consumption) value for the i^{th} node considering the multi-cores used by it during its operation. Finally, ECF_i (Energy Consumption Function for node i) is computed based on the measurement of PC_i done at a give time t for i^{th} node. At a given time instant, the power consumption (PC_i) for the i^{th} node running one or multiple VMs is given as:

$$PC_i = \sum_{k=1}^{L} \sum_{j=1}^{M} \sum_{a=1}^{U} PC_{ijk}(CU_a, t) + PC_{Idle} \tag{4.1}$$

The PC_i value is computed by adding together PC_{Idle} and summation of all the $PC_{ijk}(CU_a, t)$ values. The $PC_{ijk}(CU_a, t)$, here is the power consumption done by j^{th} VM running k^{th} task while using a^{th} computing unit at the time instant t. The

proposed system model takes into account the multiple cores in the nodes. There is a possibility that all the cores are not used when the node is in running state. Thus, the power consumption PC_i needs to be optimized with the available number of cores used during the task execution. The optimization is done as follows:

$$PC_i = \frac{(PC_i - PC_{Idle}) \times d}{cmp} + PC_{Idle} \tag{4.2}$$

where d is the number of cores used in the operation of the node i in the running state and cmp is the total number of cores in the node.

The Energy Consumption Function (ECF) for i^{th} node gives the energy consumed by a node. This is actually defined as the power consumed by i^{th} node during t units of time, denoted by ECF_i, and given by,

$$ECF_i = PC_i \times t \tag{4.3}$$

Thereby, the total energy consumption by the system (ECF) is the summation of all the ECF_i for all the nodes subject to integrating the energy consumed by the system for overall time period T.

$$ECF = \int_0^T \sum_{i=1}^{N} (ECF_i) dt \tag{4.4}$$

Now, in order to assure the fulfillment of deadline as specified by the users, the total node execution time has to be computed. The Execution Time Function (ETF) for i^{th} node is equal to the time taken by the node to execute all the tasks running on it, computed as:

$$ETF_i = \sum_{k=1}^{L} \sum_{j=1}^{M} ETF_{ijk} \tag{4.5}$$

where, ETF_{ijk} is the time taken by j VMs to execute k tasks on i^{th} node. The total Execution Time Function ETF for the system is given by,

$$ETF = \sum_{i=1}^{N} ETF_i \tag{4.6}$$

where ETF_i is the time taken for computation by each node i in the system.

As per the objectives, the task allocation also has to be restricted to only those nodes that can fulfil the budget constraint apart from energy and deadline as defined by the users. Thus Expected Cost of Task, that is, $ECT(k, i)$ value is computed for every task entering the system on each identified energy efficient node. The $ECT(k, i)$ value signifies the expected cost of running a task k on node i and is

computed as,

$$ECT(k,i) = TET(k,i) \times [\sum_{a=1}^{U} Cost(CU_a) \times NCU_k] \tag{4.7}$$

$$\forall\ k \in R, i \in S, a \in CU$$

where,

$$TET(k,i) = \frac{UR_i(t)}{NOI_k} \tag{4.8}$$

$$\forall\ k \in R, i \in S$$

Here, $TET(k,i)$ is the time taken for executing task k on node i and is computed using equation (4.8). The $TET(k,i)$ value is equal to the utilization rate of node i at time instant t with respect to the number of instructions in the task. $Cost(CU_a)$ is the cost of using a computing unit a. The value for using a computing unit per hour has been fixed. The summative value for cost of each computing unit, that is, $\sum_{a=1}^{U} Cost(CU_a)$ is multiplied with the number of computing units NCU_k used by k^{th} task.

The computed $ECT(k,i)$ value has to be normalized because the ranges of the values of the important parameters $ECT(k,i)$ and $TET(k,i)$ differ from each other. The normalization equation is as follow:

$$ECT(k,i) = \frac{ECT(k,i) - min_{\forall N \in S}\{ECT(k,N)\}}{max_{\forall i \in S}\{ECT(k,N)\} - min_{\forall i \in S}\{ECT(k,N)\}} \tag{4.9}$$

$$\forall\ k \in Task\ in\ R, i \in S$$

where, min{ECT(k,N)} is the minimum value of the Expected Cost of Task k values on N nodes and max{ECT(k,N)} is the maximum value of the Expected Cost of Task k values on N nodes.

The main goal is to optimize the energy consumption and to accomplish the processing of a task within the deadline and budget as specified by the users. Thus, it is important to compute the fitness function for the same. It is given as:

$$fit_N = \theta(ECF) + \delta(ECT) + \gamma(ETF) \tag{4.10}$$

$$ECF = min(ECF(s_i, r_k)) \quad \forall\, 1 \le i \le N, 1 \le k \le L \tag{4.11}$$

$$ECT(k,i) = min(ECT(r_k, s_i)) \tag{4.12}$$

$$\forall\, 1 \le i \le N, 1 \le k \le L, [Max(ECT(k,p) - UTBudget_k)]$$

$$ETF = min(ETF(r_k, s_i)) \tag{4.13}$$

$$\forall\, 1 \leq i \leq N, 1 \leq k \leq L, [Max(ETF_p - UTDeadline_k)]$$

where $0 \leq \theta < 1$, $0 \leq \delta < 1$ and $0 \leq \gamma < 1$ are the weights to prioritize the components of fitness function, comprising of, $ECF(s_i, r_k)$; $ECT(r_k, s_i)$; and $ETF(r_k, s_i)$. The equation (4.11) corresponds to considering the minimum value of ECF for a node i in set of nodes S; task k in set of unassigned tasks r.

The equation (4.12) specifies minimum value of $ECT(k, i)$ to be considered when a task k from set r is to be executed on node i. It is subjected to constraints that set of nodes ranging from $1 \leq i \leq N$, set of tasks ranging from $1 \leq k \leq L$ and maximum value for expected cost of task k must be greater than the budget specified by the user, that is, $[Max(ECT(k, p) - UTBudget_k)]$. Similarly, the equation (4.13) specifies that minimum value for ETF to be considered when a task k from set r is to be executed on node i. However, this is subjected to alike constraints as above with an addition $[Max(ETF_p - UTDeadline_k)]$, that is, maximum value for ETF greater than the user specified deadline.

The fitness function is subject to constraints:

$$ECF_i \leq ECF_{th} \tag{4.14}$$

that is, ECF_i of the i^{th} node should not exceed the threshold value Energy Consumption Function (ECF_{th}). This ECF_{th} value depends on and should be less than the maximum energy consumption value of a node out of all the nodes in the system. Let time be constant at time instant t, then the maximum energy consumed by a node depends on its utility. In other words, a node with lesser utility can be assumed to be sitting idle for most of the time and drawing unnecessary power, thereby consuming more energy wherein a node with more utility level tends to consume lesser energy [101]. The focus is to maximise the node utilization rate (UR_i) to minimize consumed energy.
Thus,

$$ECF_{th} = \phi_i \times min(UR_i) + y \tag{4.15}$$

In the above equation, $min(UR_i)$ has been considered keeping in objective that energy efficiency has to be maximized. Thus, the lower the value of UR_i of a node, the higher will be the energy consumption and consequently computed energy threshold value will be high. Here y is a constant value. Let ϕ_i be poissonly distributed over the standard variate of threshold, thereby the application of the L number of tasks assigned and running on M VMs, it can be computed as,

$$\phi_i = \frac{e^{-L}(L^M)}{M!} \tag{4.16}$$

$$\forall 1 \leq j \leq M, 1 \leq k \leq L$$

Therefore, ECF_{th} can be written as,

$$ECF_{th} = \frac{e^{-L}(L^M)}{M!} \times min(UR_i) + y \tag{4.17}$$

The computed ECF_{th} value obeys Poisson distribution as the number of tasks entering the system is not fixed. It is also state-independent and considers no task dependencies.

4.4 Proposed GreenSched Scheduler Model

CGCS, the scheduler unit of GreenSched, is based on a multilayer forward-only Counter Propagation network [228] with the combination of input, output and clustering layers. It has a three-layer neural network model that performs input-output mapping, producing an output vector i (in this case the best node with least energy consumption) in response to an input power vector PC_i (power consumption value for each node) on the basis of competitive learning [228].

4.4.1 Operation of CGCS

The three layers in the CGCS network model correspond to the power input layer; the energy optimized cluster layer (Kohonen layer) and the output layer (Grossberg layer) (as shown in Figure 4.3). The forward-only CGCS model runs an intelligent FCS scheduling technique where initially the power vector between the power input layer and energy optimized cluster layer are trained and then the time, cost vector weights between the energy optimized cluster layer and the output layer are trained. Section 4.3.2 lists the notations used in CGCS unit.

In context, the training process for the CGCS consists of 2 steps. Initially, an input power vector in the form of power consumption of each node, that is, PC_i is presented to the different Power Computation Agents (PCAs) sitting on each input node. These PCAs compute the energy consumption for each node, ECF_i value by using the equations (4.1), (4.2) and (4.3). This is followed by competition between the nodes where the winner nodes are selected based on comparison of the computed $ECF_i(i = 1, 2, ..., n)$ for each node with the computed threshold, ECF_{th} value.

The nodes with ECF_i value less than ECF_{th} are the less energy consuming nodes (or the winner nodes or energy aware) while nodes with ECF_i value greater than ECF_{th} are more energy consuming nodes (or non-energy aware nodes). This process of comparison is actually done to classify and cluster the nodes. As a result of the comparative analysis, an energy optimized cluster layer is formed that consists of two distinct clusters of nodes called as "lessEnergy/winning" cluster and "moreEnergy" cluster. The lessEnergy cluster includes the adjudged energy aware or winner nodes while moreEnergy cluster has non-energy aware nodes. Consequently, the winning cluster nodes thus qualify for energy aware task allocation process. This initial phase

of CGCS training process follows unsupervised learning process to cluster the input vector to separate distinct clusters.

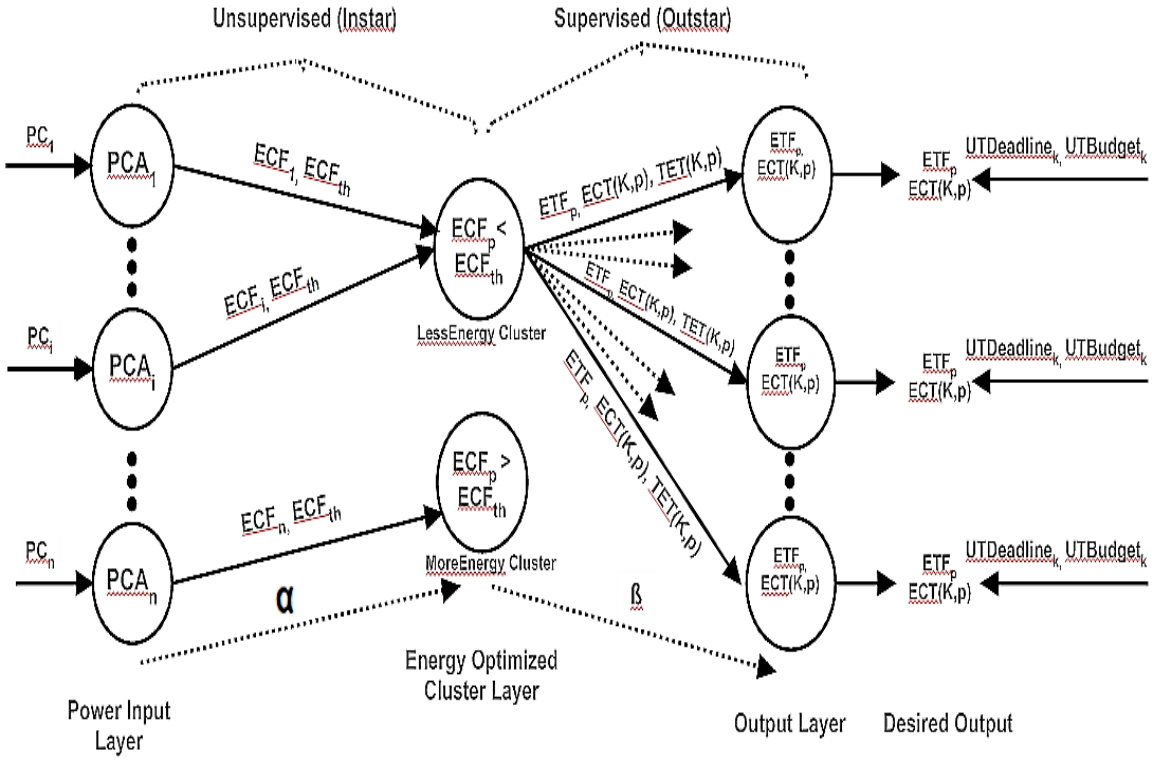

Figure 4.3: CPN-Based Green Cloud Scheduler Model

In the second step, once the competition of nodes is completed to obtain the cluster of winner nodes, only one cluster unit will be active (lessEnergy cluster comprising the winner nodes or the winning cluster here) in the energy optimized cluster layer and it sends signal to the output layer. At the output layer, the inputs are presented to the network, the user specified constraints on deadline and budget for a task are also presented simultaneously. Here also CGCS intelligently predicts and computes the best node out of all the winner nodes for task allocation using the competitive learning method.

At the next time instants, t', the PC_i and ECF_i values for the nodes are updated and the updating equations of input power and energy vector from the input PCA units to the cluster units is done using learning rule given below for $i = 1$ to n.

$$PC_i = \sum_{k=1}^{L} \sum_{j=1}^{M} \sum_{a=1}^{U} PC_{ijk}(CU_a, t') + PC_{Idle} \tag{4.18}$$

$$ECF_i = PC_i \times \alpha t' \tag{4.19}$$

Afterwards, for each node p in the winning cluster in the energy optimized layer, the power consumption values, PC_p and Energy Consumption Function values, ECF_p are updated automatically during the learning process in subsequent iterations and thus the members in the clusters are adjusted accordingly. Again the competition between nodes takes place nodes and winning cluster is formed where the nodes compete (winner take all) for attaining training precision by learning the input energy vector [150].

Now after the energy metric optimization, the focus shifts to obtain the node that can fulfil the deadline and budget constraints. For this purpose, the ETF_p for each node p of the winning cluster unit is computed. Also, the $ECT(k,p)$ is also computed by the pricing module and sent to CGCS to ascertain the expected cost of running a task k on each node p of the winning cluster. These values are updated at next time instant t' using equations (4.20-4.22),

$$ETF_p = \sum_{k=1}^{L} \sum_{j=1}^{M} ETF_{jk} \tag{4.20}$$

$$ECT(k,p) = TET(k,p) \times [\sum_{a=1}^{U} Cost(CU_a) \times NCU_k] \tag{4.21}$$

$$\forall\ k \in R, p \in S, a \in CU$$

$$TET(k,p) = \frac{UR_p(t')}{NOI_k} \tag{4.22}$$

$$\forall\ k \in R, p \in S$$

Here, $TET(k,p)$ is the time taken for executing task k on node p in the winning cluster and is computed using equation (4.22). The $TET(k,p)$ value is equal to the utilization rate of node p at time instant t' with respect to the number of instructions in the task. $Cost(CU_a)$ is the cost of using a computing unit a. The summative value for cost of each computing unit, that is, $\sum_{a=1}^{U} Cost(CU_a)$ is multiplied with the number of computing units NCU_k used by k^{th} task.

Finally, the computed ETF_p and $ECT(k,p)$ values are compared with the user specified $UTDeadline_k$ and $UTBudget_k$ for each task k entering the system. The difference between the $ECT(k,p)$ and $UTBudget_k$ is computed. The node with the maximum difference value can be considered to be the best node to fulfil the task energy efficiently and economically. Similarly, the difference between the ETF_p and

$UTDeadline_k$ is computed. The node with maximum difference value can be judged to be the best node that is energy efficient as well as can satisfy the deadline.

However, the priority is given to a node that can first optimize the energy, then cost and then deadline. In other words, CGCS unit prefers ECD (Energy, Cost and Deadline) based optimization policy. The later phase of Forward-only CGCS model uses supervised learning to obtain the variation between the CGCS model outputs and the corresponding desired target user constraints. Once the entire set of nodes in the cloud has been presented to CGCS, the learning rate α and β are reduced and the power consumption values are presented performing several iterations.

The node that has been judged to be the best node with optimal energy, satisfactory budget and deadline fulfilment is chosen for the current task allocation by the CGCS unit. In case, no node in the LessEnergy cluster can fulfil the deadline and budget constraint for a task then a fresh node with $Sleep$=0 is set to active state by setting its $Poweron$ to 1.

4.4.2 Proposed Algorithmic Formulation for FCS Technique of GreenSched

This section presents the algorithm for the FCS technique used in the proposed GreenSched. The notations for the algorithms are already discussed in Section 4.3.2. The first "TaskClassification Algorithm" corresponds to the first phase of GreenSched where tasks are analyzed and checked for possible entry into TaskDB. When the tasks are submitted by users, their deadline, budget and resource requirements values are analysed and stored in $UTDeadline_k$, $UTBudget_k$ and $TResourceReq$ respectively. Upon this, a TaskAnalysis() procedure begins that searches for past task information in TaskDB. It analyses OT, a Boolean variable for each task which if 0 means no old task information is available in TaskDB. Thereby, the tasks information are stored in TaskDB.

For a particular unallocated or unassigned task, r in the set R of unassigned tasks, the OT value if 0 leads to storage of task information in TaskDB and call to the next algorithm for deciding on the allocation of task to appropriate node. In case, the OT value for a task is 1, it means its prior task information is available in TaskDB, its resource requirements are retrieved from the database. Based upon the value of $TResourceReq$ of a task, it can be pre-processed to classify it into CPU-intensive or memory-intensive task and is then allocated to the CPU or memory accordingly. Once the successful task analysis has been done, the call to the next algorithm is made.

Algorithm 1: TaskClassification Algorithm

Input: Task and Task characteristics
Output: Task Classification as CPU- or Memory -Intensive

```
1.  Begin
2.  TaskSubmission()
3.  Obtain Task Deadline and Budget and store in UTDeadline_k and UTBudget_k respectively
4.  Obtain task required resources and store in TResourceReq        /*Resources required by Task*/

5.  TaskAnalysis()
6.  OT ← boolean variable   /*If past resource and energy consumption values are stored in TaskDB*/
7.  Update OT for each task in R                                   /*R is the set of unassigned tasks*/
8.  UnassignedTask ← R
9.  while UnassignedTask ≠ ∅ do
10.     for each task r ∈ R
11.     if (OT = 0) then                          /* No old task information is available in TaskDB*/
        Store task information in TaskDB
        Call BestNode-TaskPair()                                   /* Call to CGCS() Scheduler*/
12.  else if (TResourceReq=CPUBound) then
            AllocateTaskTo_CPU()                   /*Task is CPU bound. Assign task to CPU */
13.  end
14.  else
            AllocateTaskTo_Memory()                /*Task is memory bound. Assign task to memory*/
15.  end
16.  end
17.  Call BestNode-Task Pair()
18.  end
19.  end
```

The second algorithm, "BestNode-Task Pair Algorithm" corresponds to the third phase of operation of GreenSched (discussed in detail in Section 4.2.2 and in Section 4.4.1). It attempts to obtain the best node-task pair using the task and node data initially obtained by other modules of GreenSched and sent to the CGCS unit. Here, steps 1-7 correspond to first two phases of GreenSched as discussed in Section 4.2.2 where upon successful task analysis and node management, the tasks are forwarded to the GCRS. The GCRS module obtains pricing information for the task from its pricing unit and power values from its PEM unit and then forwards all this information to CGCS unit. The step 8-46 onwards correspond to CGCS unit as discussed in detail in Section 4.4.1.

Algorithm 2: BestNode-Task Pair Algorithm

Input: Set of Nodes, Task and Task Information, Node Characteristics
Output: Best Node-Task Pair for Scheduling

Task Analysis and Node Management Phase

1. Begin
2. Obtain tasks information from TaskDB /*Corresponds to the Task Analysis Phase*/
3. Obtain the nodes information from GNPM /*Corresponds to the Node Management Phase*/
4. Input the set of nodes ($S \leftarrow$ Set of Nodes)
5. $Sleep \leftarrow$ Boolean variable /*Value of Sleep if 1 means the node is in running state else 0 */
6. $Power\ On \leftarrow$ Boolean variable /*Sets a node to running state by setting Power On value to 1*/
7. Track $Sleep$ and $Power\ On$ value for each node

/*Variables to track the free nodes and set them to active state when needed*/

Scheduling Phase- CGCS unit

/*Call to the scheduler unit*/
8. CGCS()
9. Initialize PCA and learning rate /*Activate the Power input vector*/
10. while $(i < n)$ do /*Stopping condition for algorithm*/
11. Obtain PC_i of each node from PEM unit using Eq. (4.2) /*Compute Power consumed by each node*/
12. Compute ECF_i using dot product method using Eq. (4.3), ETF_i using Eq. (4.5) and $ECT(k,i)$ using Eq. (4.7) /*The energy consumed by a node, it's execution time*/ /*and expected cost of task on it are computed*/
13. Optimize the fitness function using Eq. (4.10) /*The optimization fitness function is computed*/
14. Calculate Energy Consumption Function Threshold, ECF_{th} /*Compute the threshold value*/

/*Comparative analysis of energy consumed by each node with the threshold value to obtain the less energ consuming nodes that are then added to the "LessEnergy" cluster or winning cluster C. This follows as: */
15. Compare the ECF_i and ECF_{th} to obtain the winning cluster C
16. if $(ECF_i \leq ECF_{th})$
17. Add node i to the LessEnergy or winning cluster C

/*As energy aware nodes belong to winning cluster C, thus, at next time instant, t', start scrutinizing thes nodes for deadline and budget optimization for task allocation as per energy, deadline and budget criteri This is done as: */
18. for each node p in the winning cluster C
19. Update PC_p at time instant $t' = (t+1)$ using Eq. (4.18)

/*Firstly, update the power, energy, execution time of nodes at next time instant t' as per the equations specified*/
20. Perform ECF_p value updation using Eq. (4.19) for node p
21. for each task k activations at the Output layer
22. Calculate the Execution Time Function, ETF_p using Eq. (4.20)
23. Compute $ECT(k,p)$ and $TET(k,p)$ using Eq. (4.21) and (4.22)
24. end for
25. end for
26. Reduce the learning rate α
27. end while

28. while $(i < n)$ do /*For each node i in the total number of nodes n*/
29. Set α small constant value
30. Repeat steps 31-46

Scheduling Phase- CGCS unit

31. Repeat steps 32-41
32. Set PC_i values to PCA vector. Set Output layer activations to output layer
33. Compute ECF_{th} /*Compute the threshold value*/
34. Find the nodes in the winning cluster C using Eq. (4.14)

/*Discover new energy aware nodes at next time instant and update the power and energy values*/

35. Update the PC_i and ECF_i from power input layer to Energy optimized cluster layer
36. for each p in winning cluster C and task k /*For each node in the lessEnergy Cluster and a task*/
37. Update the ECF_p, $ECT(k,p)$ and $TET(k,p)$ vectors from Energy optimized cluster layer to Output layer using Eq. (4.20), (4.21) and (4.22)
38. for each task k activations at the Output layer
39. Obtain $UTDeadline_k$ and $UTBudget_k$ as specified by user for task k
/*Compute the difference between $ECT(k,p)$ and $UTBudget_k$*/
40. Desired Budget Output= max $(ECT(k,p) - UTBudget_k)$ /*Desired Budget Output*/

/*Calculate the difference between ETF_p and $UTDeadline_k$ */

41. Desired Deadline Output= max $(ETF_p - UTDeadline_k)$ /* Desired Deadline Output */
42. Reduce learning rate β
43. Locate node p in the winning cluster and allocate the task k on that node p

/*p has been observed to be the energy efficient as well as can satisfy the budget and deadline*/

44. end for
45. end for
46. end while

Node Management Phase

/* In case no node can fulfil the energy, budget and deadline constraints, follow the steps below: */
/*that is, if scheduling fails on all nodes in LessEnergy winning cluster C*/

47. for (i=1 to $i \leq N$)
48. if ($Sleep$=0)
49. Set $Power\ On = 1$
50. Schedule the task k on node i

/*Now analyse the idle nodes in the system and turn them off*/
/*Analyse UR_i for node and if this value is far lesser then the energy threshold, this means the node is idle*/

51. if($Sleep$=1 && $Power\ On$=1)
52. if($ECF_{th} >> UR_i$)
53. Set $Power\ On$= 0
54. end if
55. end if
56. end for
57. end

4.4.3 Flowchart for CGCS Unit

The flowchart aids in clearly showing and understanding the training process of the forward-only CGCS model and illustrates the way in which different vectors are updated [228]. The training of forward-only CGCS is performed in two phases. The detailed working has already been discussed in the Section 4.4.1 above. Figure 4.4 below shows the flowchart for the training process of Forward-only CGCS. In general, Flowchart 1 (Figure 4.4) determines the winning cluster based on comparative

analysis and also updates the power and energy values for the nodes in the winning cluster.

Flowchart 2 (Figure 4.5) indicates the computation and update process carried out in the winning cluster at next time instant where initially, user specified constrains on deadline and budget are presented followed by the updation of power and energy values of the nodes in the winning cluster. The ETF_p and $ECT(k,p)$ values are computed for each task for further comparison with the user specified deadline ($UTDeadline$) and budget ($UTBudget$) values for each task in order to obtain the best node for allocating the tasks. The parameters used in the flowchart are given below.

α, β= Learning rate parameters where α=0.5 to 0.8, and β=0 to 1.

The typical values of learning rates may be α=0.6 and β=1

PCA= Power Computation Agents computing power and energy at the power input layer, that is, $PCA = (PC_1, ..., PC_i, ..., PC_n)$

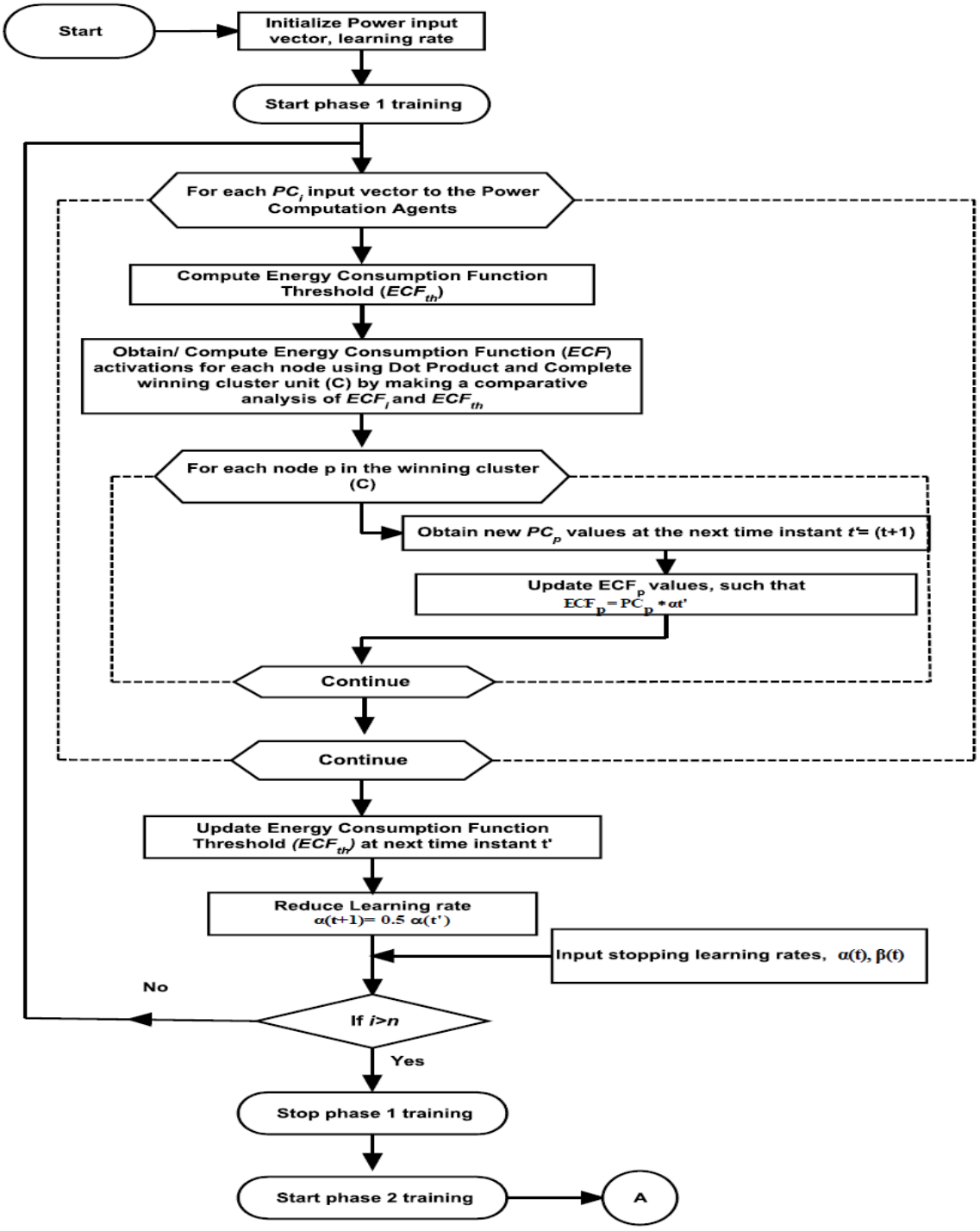

Figure 4.4: Flowchart 1 for Training of Forward-Only CGCS

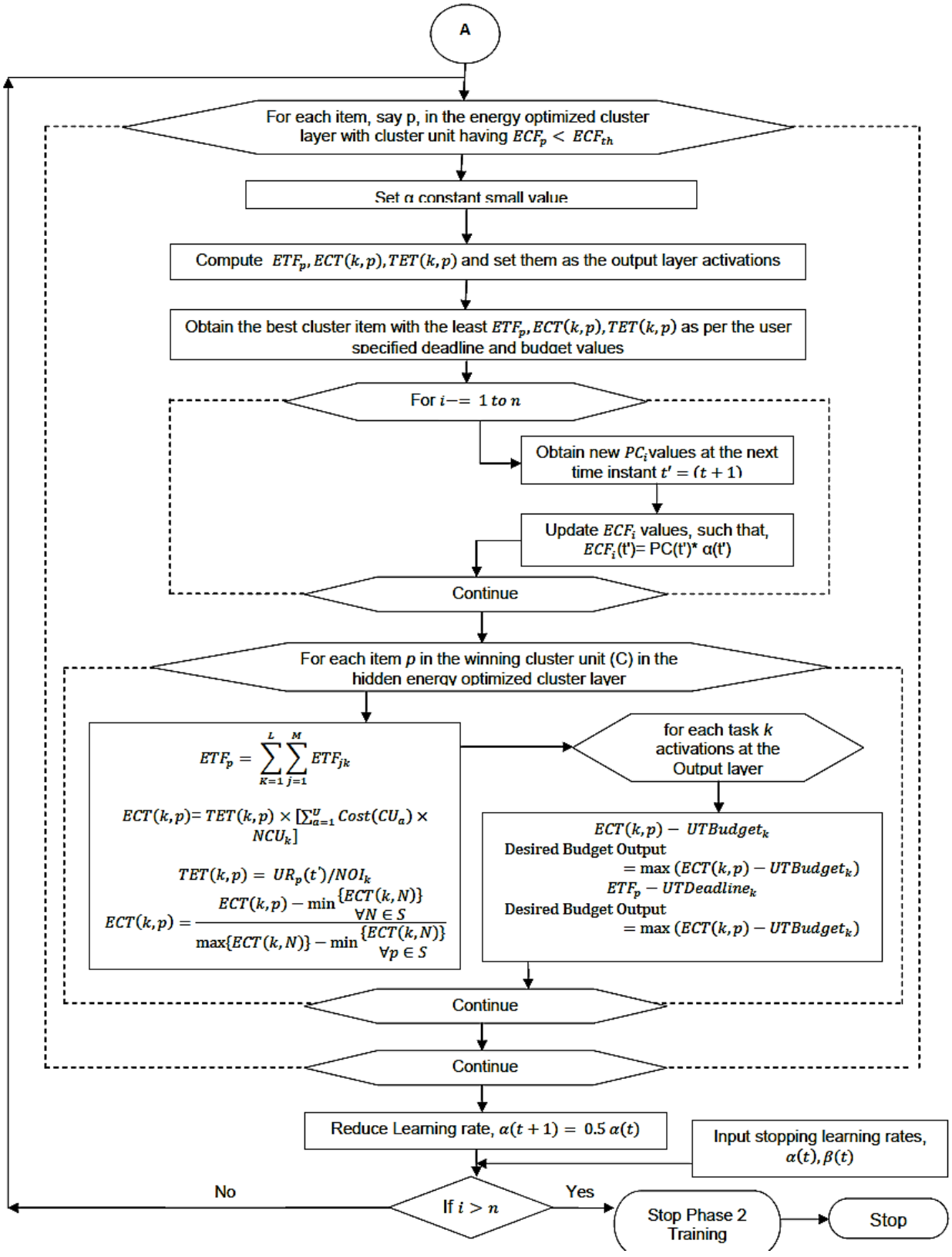

Figure 4.5: Flowchart 2 for Training of Forward-Only CGCS

4.5 Experiments and Results

The following section demonstrates the experimental analysis of the proposed FCS scheme running in GreenSched. In order to evaluate the proposed model, a simulated data center was created consisting of variable nodes, running multiple VMs using the customizable policies and methods available in the CloudSim [229] platform and settings such as VM characteristics. For obtaining a deeper understanding of the performance of the proposed model on a wider-scale before implementing in some actual environment, the simulation-based experiments have been conducted.

For testing purpose, the workloads available at Parallel Workload Archive (PWA) have been used [219]. A large number of workload traces were available consisting of real time workload traces. After a thorough investigation of the traces, LANL O2K log that contains accounting records produced by a software running on the cluster have been used. As per the requirements, variable operations can be performed on accounting data that can be both CPU- and memory- intensive. Additionally, the trace had user-specific characteristics similar to GreenSched such as the number of processors, memory required, task time etc.

The proposed model includes a Forward-only CPN scheduler CGCS, running FCS technique that proactively obtains the best node-task pair for task allocation as per the power consumption and energy values obtained from CloudSim. The CGCS unit has been designed using Kohonen Maps and Counter propagation Artificial Neural Networks (CPANN) Toolbox for $MATLAB^{TM}$ [230, 231] based on the support of $Matlab^{TM}$ platform. The identification of the best node by CGCS unit was obtained in the XML script and converted to JavaScript as an input back to the CloudSim.

Additionally, the comparative analysis of the proposed model has been done with PSOTBM mechanism running an ECC technique proposed in [198] and DCEERS proposed in [203]. The ECC technique offers energy efficient allocation of tasks with reduced finish times, enhanced revenue and improved resource utilization. Similarly, DCEERS is a Data Center-wide Energy-Efficient Resource Scheduling (DCEERS) that schedules the data center resources according to the current workload stressing upon energy efficiency, response time and scalability. The techniques have been chosen for relative examination of the proposed FCS technique based on related parameters of these techniques with FCS such as energy efficiency, task finish times and economic aspects.

4.5.1 Experimental Setup

A total 10-15 experimental runs were conducted on a simulated data center having 10-60 heterogeneous nodes with multiple cores running 20-250 VMs. Table 4.1 shows the configuration of the prototype cloud data center created in CloudSim for performing the experiment. For estimating the energy consumption, a time period t was defined and was set to on a single day execution with incremental instances. The inter-arrival timing of the tasks follows random uniform distribution between 10 and 100, and only processor and memory units are considered. Initially, the experimental

runs up to 15 were conducted. The total power consumption of the system peaks at about 112 W during the experiment measured with the Joulemeter [215] and finally computed by CloudSim. The average power consumed by the computing units in idle state has been observed to be 43.5 W with all components attached. The energy consumption of a node was computed using equation (4.3) while of the system using equation (4.4) respectively. For observing and evaluating the outcomes at the CGCS unit, the parameters set for GreenSched in Kohonen Maps and CPANN Toolbox are listed in Table 4.2.

Table 4.1: Configuration of the Simulated Cloud Data Center

S.No.	Parameter	Value
1.	Number of Nodes	10-60
2.	Number of VMs	20-250
3.	Number of Tasks	250-2500
4.	Bandwidth	10 Gbps
5.	Average Energy Consumed whilst task completion	832 Joules
6.	Average Energy spent whilst task allocations	40 Wmin
7.	Power rate of nodes	112 W

Table 4.2: Parameters for GreenSched in Kohonen Maps and CPANN Toolbox

S.No.	Parameter	Value
1.	Number of Epochs	15
2.	Number of Input Nodes	10-60
3.	Learning Rates	α= 0.5 to 0.8, β= 0 to 1

4.5.2 Results and Analysis

The figures below present the outcomes and analysis of the proposed FCS technique only as well as its comparative analysis with DCEERS and ECC technique.

- **Experimental Outcomes of running FCS Technique**

  Figure 4.6 shows the clustered nodes during initial run as per the assessment of energy consumed by each node (ECF - Energy Consumption Function). The node clustering has been obtained by comparing the ECF of each node with the energy threshold value. However, an upper and lower threshold boundaries were also defined to further refine the classification process of the nodes where the nodes with ECF way beyond the threshold are indicated with red circles around them.

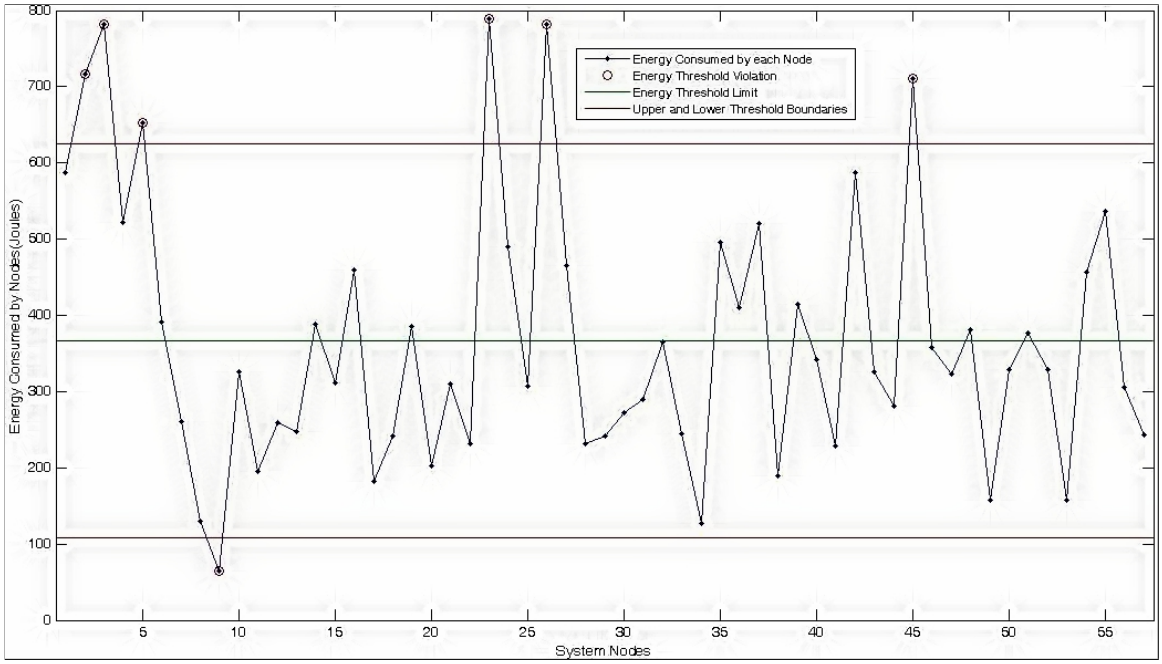

Figure 4.6: Node Classification Representation of Energy Consumed by GreenSched Nodes vs Energy Threshold

Figure 4.7 represents the percent-wise approximate estimation of the nodes that are energy aware that is, the nodes that have energy consumption lower than or closer to the threshold set for the energy violation. It can be observed that about 6.6 percent of nodes running in the system are sitting idle and drawing power. These nodes are contributing to the rise in the energy wastage in the system and thereby require elimination. On the other hand, about 80% nodes are energy conscious and can be employed for task allocation process. The simulation runs indicate that around 10 percent nodes are experiencing high energy consumption beyond the threshold and are therefore unsuitable for allocation of more tasks.

Figure 4.8 below shows the percentage of energy consumption done by the FCS technique during multiple experimental runs conducted at variable time instants. It can be observed that as the system runs proceed, the overall energy consumed in the system reduces. There is a drop in the percentage of energy consumed by the nodes. This can be attributed to the increase in the improvement of the node utility levels that occur with every consecutive system runs. The improvement in the utilization levels of the system nodes occurs due to energy management and energy-aware scheduling decisions being made by FCS technique at the successive simulation runs. Also, certain nodes that are sitting idle are switched off to prevent them from consuming power unnecessarily.

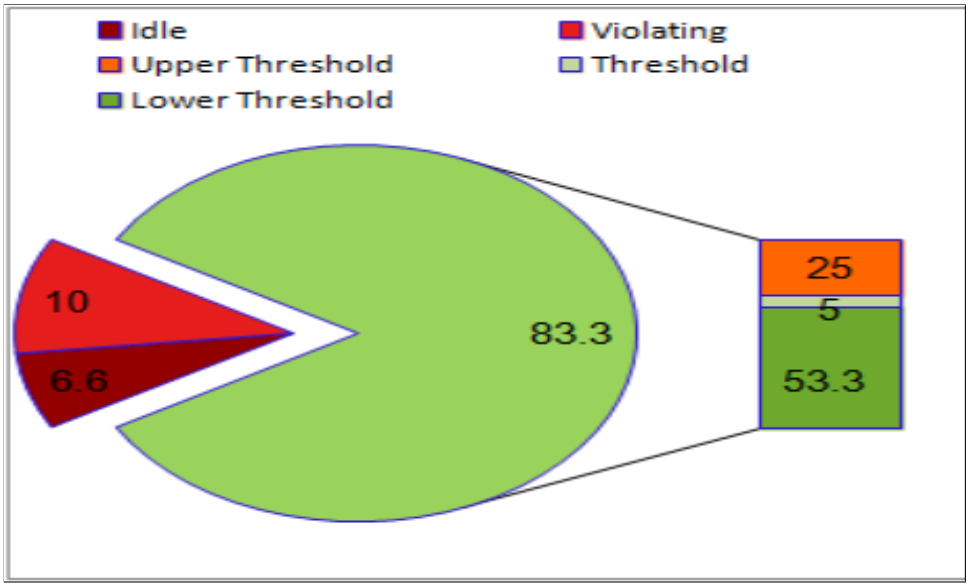

Figure 4.7: Energy Monitoring Result of the Cloud System Nodes

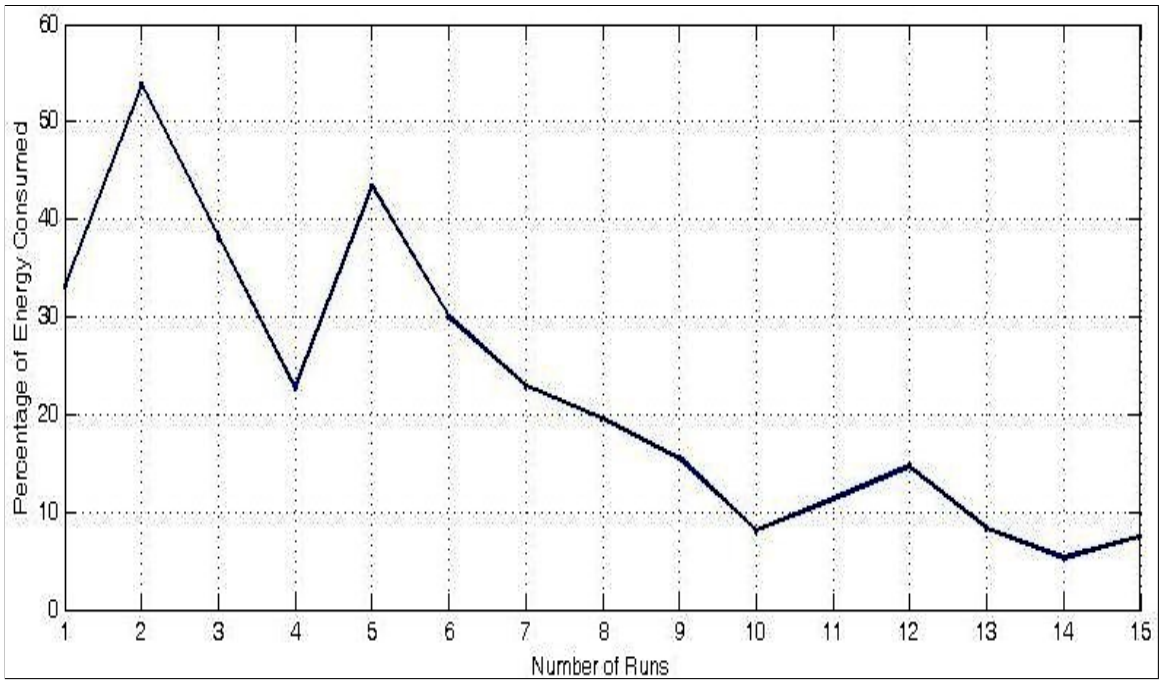

Figure 4.8: Percentage of Energy Consumed by FCS Technique during Multiple Experimental Runs

Similar to Figure 4.8, the graphical representation given in Figure 4.9 depicts

the percentage of energy savings made by the proposed FCS technique over time periods varying from one to ten hours or more. It can be clearly inferred from the trends visible in the graph that over the period of time, the energy savings are increasing. Apparently, the reduction in the energy consumption and enhanced energy savings result from improved performance, task executions within deadlines constraints. Overall, the proposed FCS technique attains energy savings up to 64.21 percent on an average.

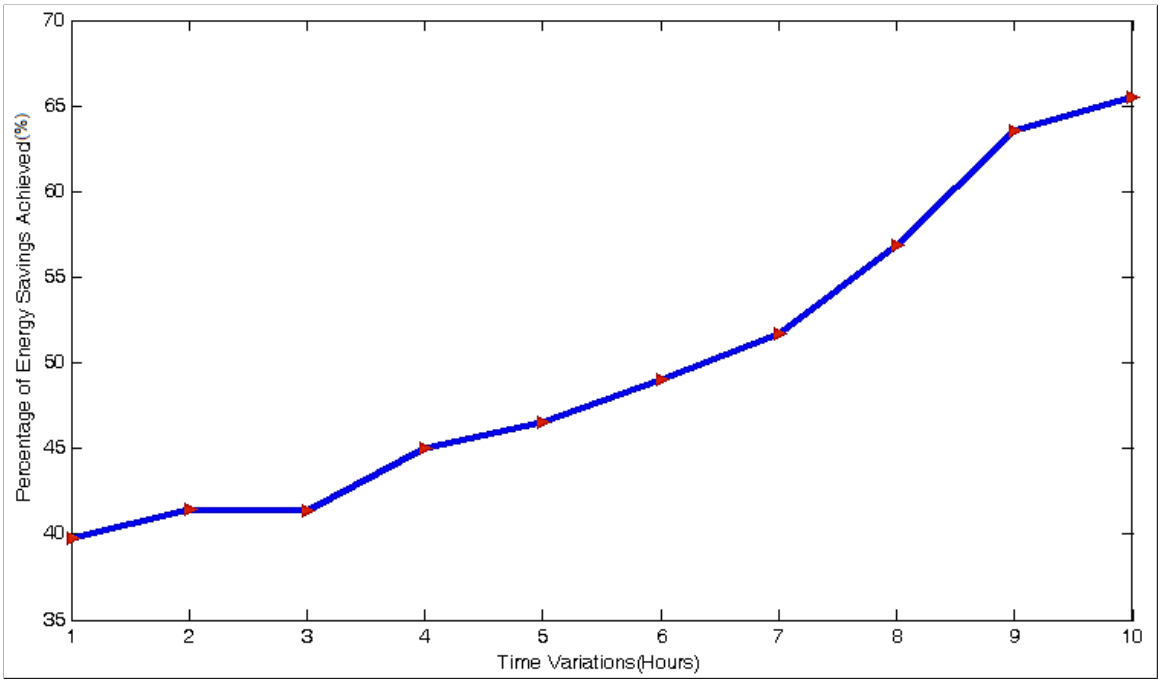

Figure 4.9: Percentage of Energy Savings by FCS Technique Realised Over a Period of Time

Based on the analysis of Figure 4.10, it is observed that FCS achieves optimal performance levels with every next iteration run of the system . This is attributed to the improved resource utilization levels and declining energy consumption done by the system upon running FCS technique. The improvement in the resource utilization pertains to the improved levels of node utility as a result of deduced energy consumption by FCS technique including the removal of the idle nodes. Consequently, the continuous drop in the energy consumption positively affects the overall performance in long term and in multiple runs.

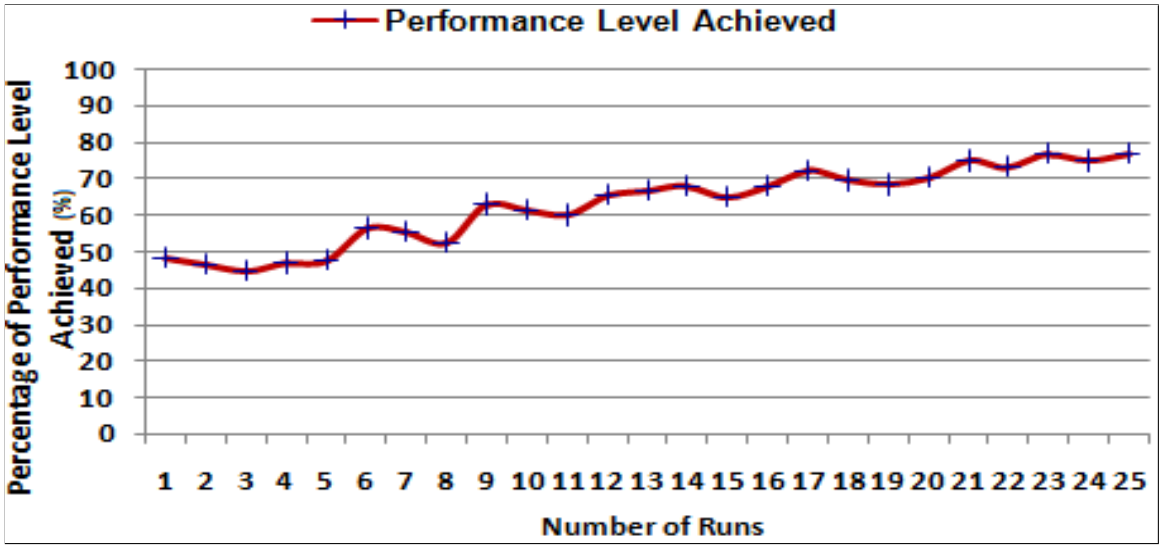

Figure 4.10: Percentage of Performance Achieved by FCS Technique

A concept termed as Energy proportional computing was proposed in [232] which analysed and stated that the computational power is proportionally related to the node utilization levels. It assumed that a linearly proportional relationship between power and utilization would cause the energy efficiency to be high throughout the activity period.

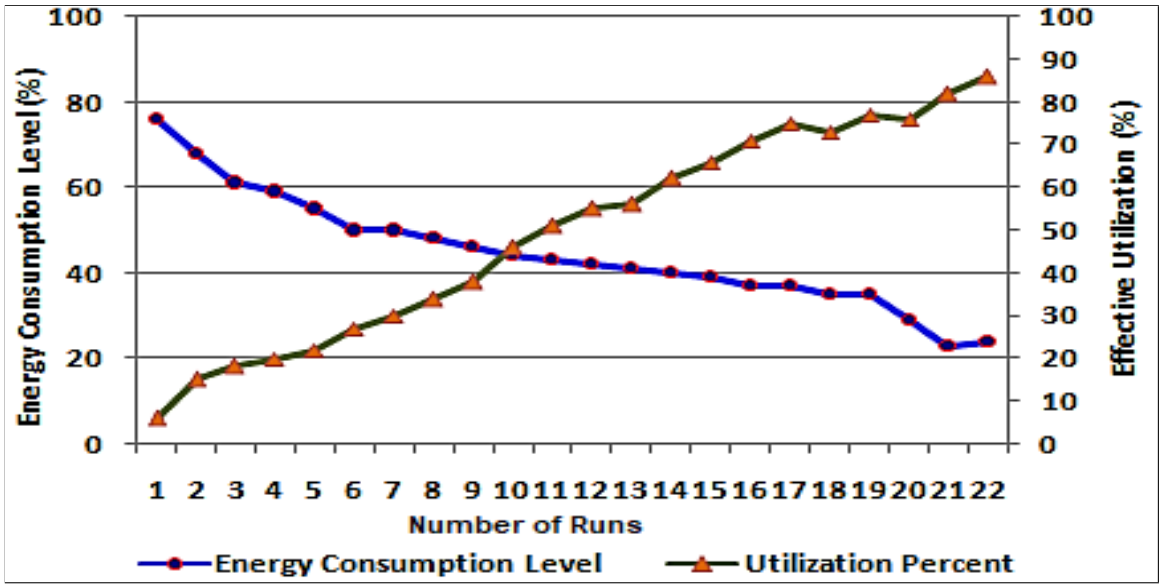

Figure 4.11: Effective Utilization Scenario in FCS

For this, an Effective Utilization Index has been defined. The Effective Utilization Index, here, is actually based on the analysis of the nodes in the system.

The lesser the number of idle nodes in the system and higher the optimally utilized nodes, the higher will be the energy efficiency. As discussed earlier, the decrease in the number of idle nodes in the system reduces the overall energy demand and energy consumption. The proposed FCS technique detects the idle nodes and switches them off. Also, it achieves higher utilization levels of the nodes. Therefore, it offers energy effcient operation. Figure 4.11 depicts such a typical scenario of relationship between energy conusmption level to the utilization levels. Clearly, as the overall utilization grows in the system, the overall energy consumption levels drop down.

- **Effective Utilization Scenario in FCS**

The Figures 4.12- 4.17 depict the comparative representation of the FCS technique, DCEERS and ECC technique. The techniques have been relatively compared on the basis of certain aspects including the energy consumption rates, performance levels, execution times and deadline fulfilment factors and cost facet.

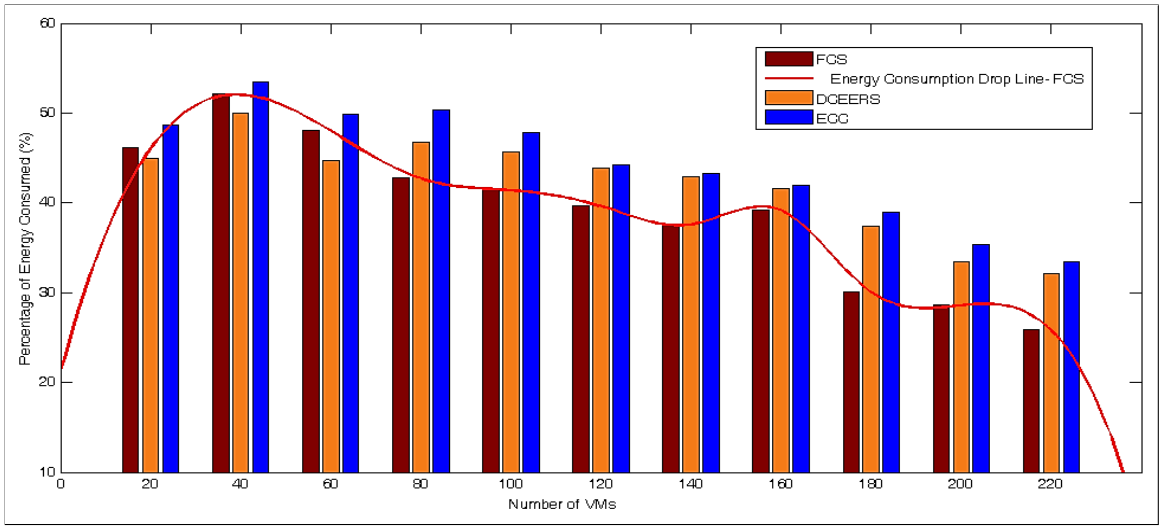

Figure 4.12: Percentage of Energy Consumed (FCS Vs DCEERS Vs ECC)

Figure 4.12 shows the results on energy consumption levels obtained on running FCS, DCEERS and ECC techniques. It can be observed that the percentage of energy consumed by FCS technique is very close to that of DCEERS and ECC. The bar lines depict the percentage of energy consumption made by the FCS, DCEERS and ECC techniques. A drop line in the red colour indicates the consecutive drift in the percentage of energy consumption done by FCS as per increased number of VMs. In other words, with the hike in the processing units (in the form of multiple illusive servers, that is, the VMs) running on the nodes, the energy consumption decreases. This is because of the increased computational power and distribution of the work among these VMs enabling faster executions and lesser

response time. The graph in Figure 4.13 shows the percentage of energy savings obtained by all the three techniques. Overall, FCS achieves 64.21% of energy savings compared to 62.45% and 57.75% of DCEERS and ECC techniques respectively.

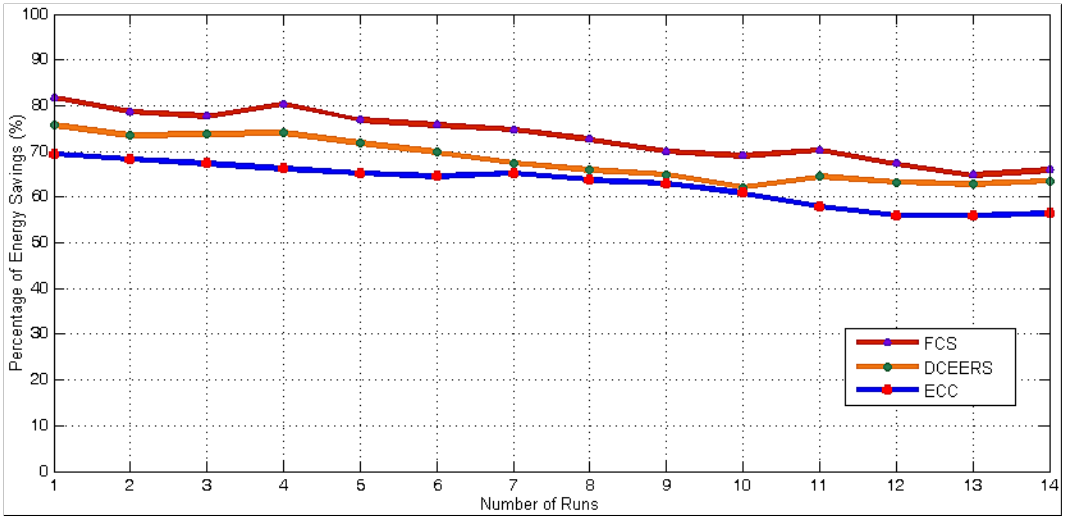

Figure 4.13: Percentage of Energy Savings (FCS Vs DCEERS Vs ECC)

Also, the performance improvement was higher in case of FCS then that of DCEERS and ECC. This is because the FCS technique offers much faster task executions within the specified deadlines as imposed by the users. Figure 4.14 represents the performance improvement curves for the FCS, DCEERS and ECC techniques. Clearly, the percentage of improvement in the performance levels is more in the case of FCS than both other techniques.

Figure 4.15 illustrates the comparison of the Total ETF of FCS, DCEERS and ECC techniques. The Total ETF, which is the total execution time of the system to get the final output, is measured in minutes (mins) using Equation (4.6). It can be noted from the graph that the overall ETF for the GreenSched running FCS technique is less than DCEERS and ECC. This is due to the dynamic task executions on the system nodes as per their energy levels whilst keeping up with the overall performance of the system. Largely, the overall tasks finish times are satisfied within the deadlines imposed by the users. To evaluate this, a deadline fulfilment factor has been defined that signifies the number of tasks processed or completed within the finish times as specified by the users already. It can be observed that deadline fulfilment factor is stable throughout the execution process.

Also, it is evident that as number of nodes rise, the ETF value for each technique also rises. This is because the increase in the number of nodes operating in the system tends to increase the overall system execution time resulting in higher ETF. However, FCS technique involves lesser ETF than DCEERS and ECC technique.

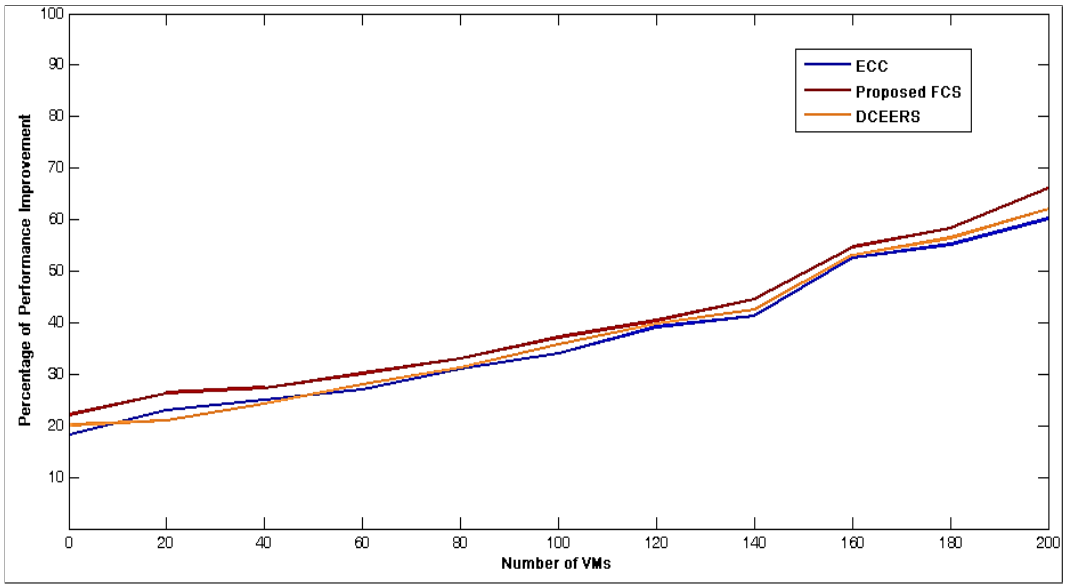

Figure 4.14: Performance Improvement Comparison (FCS Vs DCCERS Vs ECC)

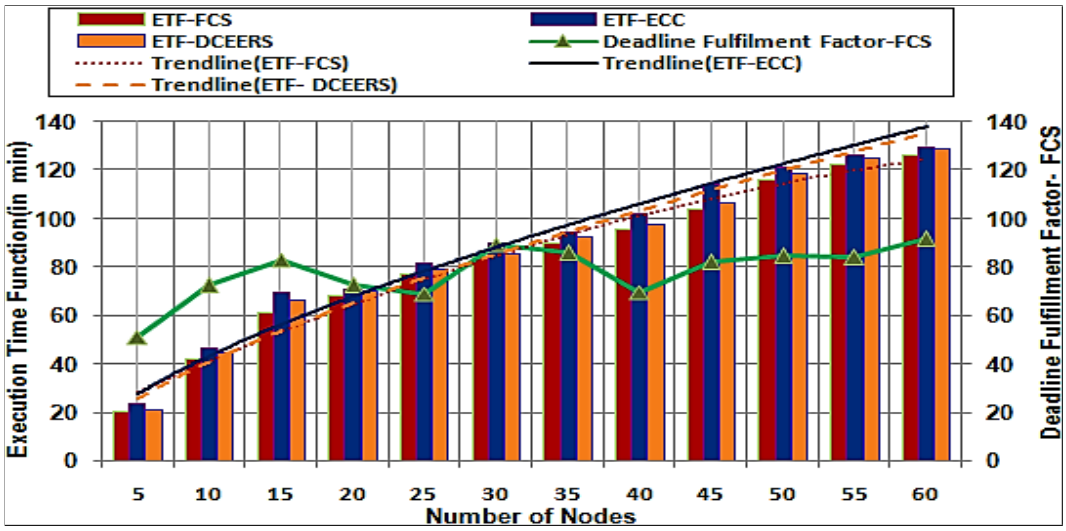

Figure 4.15: FCS Vs DCCERS Vs ECC in terms of Execution Time and Deadline Fulfilment Factor

Figure 4.16 presents the analysis of the impact of number of tasks running in the system on the budget fulfilment factor. A budget fulfilment factor has been statistically obtained value that represents a coefficient of variation of the budget satisfaction with the variation of the number of tasks running in the system. It can be analysed that with the rise in the number of tasks, the variation in the budget fulfilment factor curve reduces. It can also be analysed that the budget fulfilment

factor curve for FCS remains lower than that of DCEERS and ECC technique. This indicates that the proposed FCS technique offers much more stability in the budget satisfaction than other two techniques.

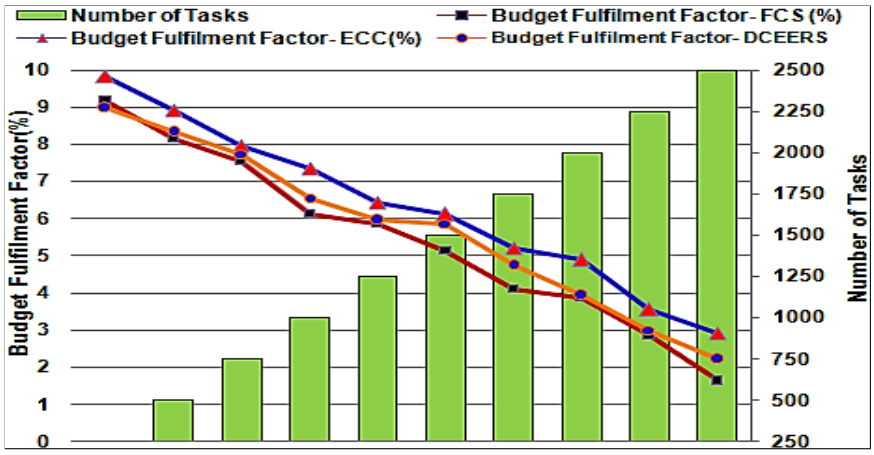

Figure 4.16: FCS Vs DCCERS Vs ECC in terms of Budget Fulfilment Factor

Additionally, as the number of system runs proceed, the cost of running the system decreases in all the three techniques. This has been represented in Figure 4.17. The decrease in the cost of operation in case of FCS is more. This can be attributed to the reduced time of execution of tasks due to improved performance and lesser overall system energy demand with each experimental run. The reduction in the execution time of a task pertains to lesser running time for a task on a node. This reduces a nodes running time and thereby the cost of operation of system nodes. Similarly, ECC has an improved cost of operation as it is primarily revenue enhancing technique and focuses on the same. Relatively, DCEERS has higher cost of operation than FCS and ECC as it principally focuses on energy reduction.

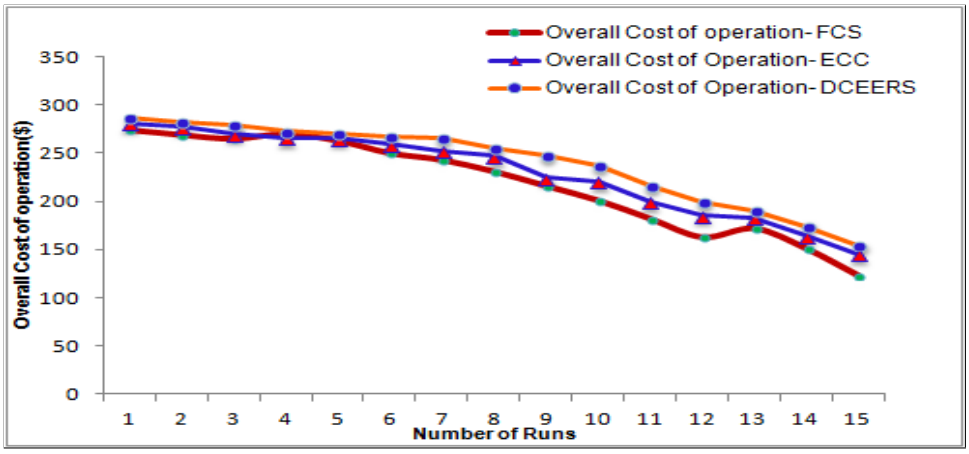

Figure 4.17: Overall Cost of Operation of FCS with each Experimental Run

- **Experimental Analysis: GreenSched vs GCSM**
  The proposed GreenSched model in this paper has also been experimentally compared for energy, performance and deadline metrics with GCSM. Experimentally, GreenSched has been tested in simulated data center using CloudSim toolkit [229] whereas GCSM has been evaluated in Cloud testbed set up in our research lab.

Figures 4.18 and 4.19 demonstrate the comparative analysis of FCS technique of GreenSched with GCSM. The graph in Figure 4.18 shows the percentage of energy savings obtained by FCS and GCSM techniques. Overall, FCS attains 67% energy savings as compared to GCSM with 65% energy savings. The slight improved energy savings in FCS can be attributed to the better predictive and autonomic task provisioning performed by CPN-based scheduling unit. It is also evident that consistency in energy efficiency has been maintained by both the algorithms.

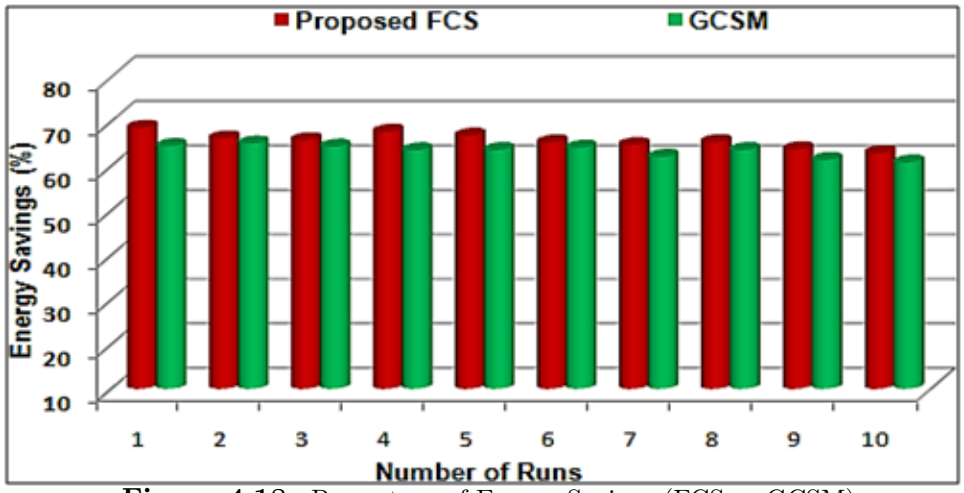

Figure 4.18: Percentage of Energy Savings (FCS vs GCSM)

Figure 4.19 demonstrates the comparative analysis of FCS and GCSM in terms of performance levels achieved by them. Overall, FCS attains higher performance levels as compared to GCSM. FCS technique of GreenSched offers improved resource utilization levels owing to declining energy consumption. The improvement patterns in the resource utility are due improvement in node utilizations and drop in the inactive nodes in the system. This results in higher performance in long term and in multiple runs.

Figure 4.20 depicts the deadline fulfilment factor fulfilled by FCS and GCSM. It can analysed that FCS has more deadline fulfilment factor than GCSM. Owing to faster scheduling unit based on counter propagation network, FCS offers faster task allocations and hence more deadline achievements.

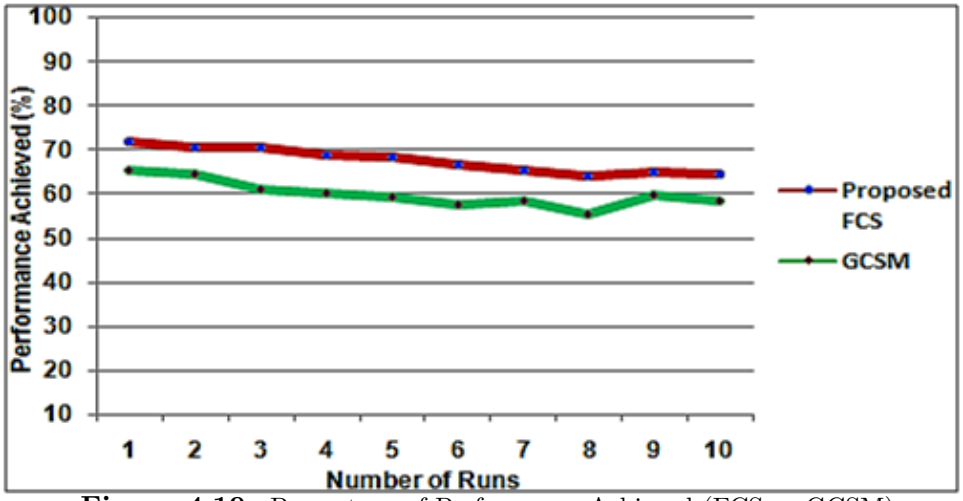

Figure 4.19: Percentage of Performance Achieved (FCS vs GCSM)

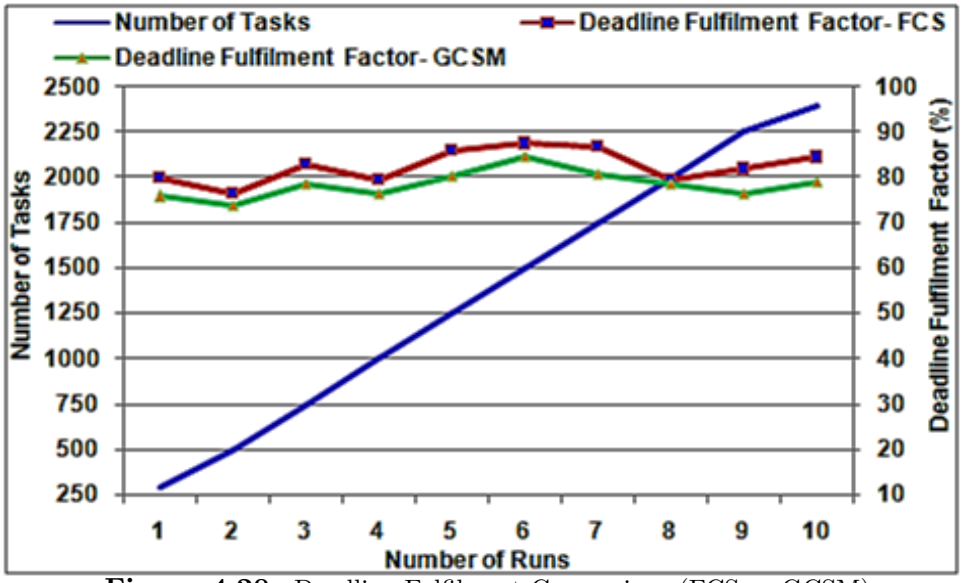

Figure 4.20: Deadline Fulfilment Comparison (FCS vs GCSM)

It can be observed that although both GreenSched and GCSM offer optimal energy efficiency. However, GreenSched offers energy conscious allocation of tasks for both deadline and budget bound cloud tasks. Conclusively, the proposed FCS technique offers energy reduction with an enhanced system performance satisfying the deadline and budget impositions as defined by the users pertaining to each task.

4.6 Conclusion

This chapter discussed an energy-aware GreenSched model for task allocation and resource provisioning that offers allocation of the tasks to least energy consuming cloud nodes in accordance to the deadline-and-budget constraints as imposed by the users. A detailed discussion on the operation, scheduling technique along with its algorithm and flowchart has been given. The chapter depicted the experimental evaluation of the proposed GreenSched model using CloudSim simulator. The outcomes indicate that the proposed FCS technique in GreenSched model results in overall improvement in the performance of the system combined with the core objective of reduced energy consumption and realises low execution times and economic benefits. The FCS technique achieves an overall energy savings up to 64.21% as compared to 62.45% and 57.75% of DCEERS and ECC respectively. The percentage of performance improvement for FCS technique is almost 69%, higher than that of DCEERS and ECC. Also, a comparative experimental analysis of GreenSched and GCSM models has been presented. Owing to better predictive and autonomic task provisioning, FCS attains slight improvement of 67% in the energy savings as compared to GCSM with 65% energy savings.

The next chapter focuses on the evaluation and verification of the proposed GCSM model at the Corporation Bank, Bakkarwala branch, New Delhi.

Chapter 5

Case Study: Empirical Evaluation of GCSM at Corporation Bank

The previous chapter discussed the design of proposed GreenSched model that implements an intelligent Forward-only Counter Propagation Network based scheduler unit. GreenSched allocates and schedules tasks to energy-conscious nodes while optimizing node utilization and thereby performance. Primarily, apart from energy efficiency, GreenSched tends to fulfill task deadlines and budget restrictions imposed by the users. Contrarily, the GCSM model proposed in chapter 3 also offers energy efficiency while focusing on deadline, performance metrics. GCSM is a viable model for different applications such as banking, ICT sector etc.

This chapter targets the evaluation of proposed GCS scheduling technique of GCSM in the banking sector. The GCS technique has been verified and validated through the case study conducted at Corporation Bank, Bakkarwala branch, New Delhi. Corresponding test cases have been designed for evaluating GCS at the bank branch and cloud-based Finacle scheduler implemented in the bank. The test cases involve analysis of energy consumption levels, resource utilizations (both CPU and memory) and performance achieved upon GCS and Finacle in-built scheduler implementation. A comparative analysis between both GCS and Finacle's scheduling procedures has also been conducted.

Section 5.1 provides a preliminary study of adoption of cloud computing in the banking sector followed by analysis of need for greener banks in the Section 5.2. Section 5.3 investigates the infrastructural and application configuration of Bakkarwala branch of Corporation bank. The existing cloud of the bank enabled through Finacle is explored in section 5.4. Section 5.5 presents the evaluation test bed designed to conduct the experimental work. The experimental case study and results for GCS scheduling technique and comparative analysis with some recent works are presented in section 5.6. Section 5.7 summarizes the conclusion for the chapter.

5.1 Cloud Computing and Banking- An Introduction

Cloud computing is a repository of pervasive, ubiquitous, flawless, extremely flexible services and facilities that are available persistently and distributedly across multiple domains. It facilitates agile deployment of IT infrastructure unlike contemporary, fixed infrastructural legacy systems and assets being used in multiple sectors [25]. Additionally, owing to the rich and extensive delivery patterns implemented in cloud computing, several business prodigious like Amazon, Infosys, Wipro are no longer just offering computing, storage, networking, databases etc. as facilities but are currently engaged in the task of offering cloud-based business solutions [233]. The easier and flexible mapping of the existing applications towards cloud based applications has significantly supported and prompted the businesses to inflate and tender highly profitable, productive, secure application deliverability using cloud [33, 34].

The Banking, Financial Services and Insurance (BFSI) sector in particular was initially slow in adopting cloud services than other business sectors due to security and compatibility issues. Such issues kept them hamstrung on transforming from mainframe and legacy systems towards novel, cloud-centric infrastructural facilities. But, over the period of time, BFSI sector has also caught-up and has adopted cloud based core banking solutions, services and infrastructure etc. The benefits associated with this adoption include [234, 235]:

- Economic Factors: The higher amount of capital investment is reduced and transformed into small configuration and operational costs. Such costs primarily include the incurrence of new hardware and software etc. Additionally, cloud provides scalability in storage, infrastructure facilities at lower costs.

- Security and Disaster Management: Cloud computing offers security-rich platform and offers back-up and recovery facilities, thus attracting the banking firms for leveraging it.

- Business Agility and Continuity: The deployment of cloud computing and creation of virtualized instances enhances agile development. It facilitates centralized management of the bank computing systems enabling flexible operation without compromising control. Moreover, the maintenance concerns can be handled by the cloud providers instead of the financial firms themselves.

- Integrity of Operation: Cloud computing makes it easier for the banking platforms to integrate customer data for faster accessibility and uniformity of operation. Also, the extension and incorporation of third-party services can be easily integrated into the existing platform.

- Incorporation of Green Banking: The amendment of banking services to cloud based services minimizes the carbon footprint and energy consumed within the banks, thus transforming banking into green banking. Also, the service delivery

enhanced through cloud minimizes idle time and facilitates efficient utilization of bank's computing systems.

The recent examples of big financial firms and banking names who are leveraging the capability of cloud for their operations as listed in the Table 5.1 [236]. Similarly, Table 5.2 lists a large number of banking software that have been incorporated within the banks for leveraging cloud benefits and adding uniformity.

Table 5.1: Big Banking Giants and their Cloud Providers

S.No.	Banking House	Cloud Service Vendor
1.	Commonwealth Bank of Australia, Australia [237, 238]	Salesforce Cloud Solutions
2.	Bankinter, Spain [239]	Amazon Web Services (AWS)
3.	DBS Bank, Singapore	Amazon Web Services (AWS)
4.	Capital One, Virginia (USA)	Cloud Technology Partners, USA
5.	Deutsche Bank AG, Frankfurt, Germany [240]	Hewlett Packard
6.	NedBank, South Africa [241]	IBM Cloud
7.	Mizuho Bank, Japan [242]	IBM Cloud
8.	UnionBank, Philippines [243]	IBM Bluemix cloud platform

Table 5.2: Globally Leading Banking Software and their Banking Clients [244]

S.No.	Software	Designed By	Clients
1.	Finacle [245]	Infosys	• Corporation Bank, India • Axis Bank, India • Emirates NBD, UAE • Standard Bank, South Africa • ING Bank in Belgium • Discover Financial Services, USA
2.	Flexicube [246]	Oracle Corporation	• Canara Bank, India • Yes Bank, India • Challenger bank Hampden, UK • United Commercial Bank Limited, Bangladesh
3.	BaNCS [247]	Tata Consultancy Services	• State Bank of India & Associates • Allahabad Bank, India • Banco Pichincha, Ecuador • Mercantile Bank • Cathay United Bank, Taiwan
4.	Xero Accounting Software [248, 249]	Xero	• Bank of Queensland, Australia • National Australia Bank, Australia • TSB Bank, UK
5.	TIBCO [250–253]	TIBCO Software Inc.	• Bank of Montreal, Canada • First Citizens Bank, USA • KuveytTurk Bank, Turkey

5.2 Augmentation and Initiatives for Greener Solutions in Banking Sector

The disastrous effect of environmental degradation and hampered sustainability have not only impacted the ICT sector but have considerably influenced BFSI industry. The BFSI sector observed high power and consequent energy demands and related expenditures due to large number of computing devices and equipments vested in the banks and other financial firms. The banks contribute towards carbon emissions and the use of scarce resource such as paper, electricity, cooling equipments etc adds to the energy crisis [254]. Furthermore, the current onset for environmental friendly technologies and services governed under strict regulatory and compliance require-

ments laid by the government agencies. The growing concern among customers and their interest in any company's energy policies, green products and services offered by it have all collectively motivated overall BFSI sector to incorporate greener solutions within them. These all make it imperative for the banking sector to understand and realize greener practices within themselves.

Accordingly, BFSI sector has begun creating greener banking facilities and solutions. Several initiatives are being taken out of which, the Equator Principles (EPs) and the United Nations Environment Program Finance Initiative (UNEP FI) are the leading initiatives. The UNEP FI focuses on encouragement of greener technologies and principles across all the operational levels in financial firms. The EPs include guidelines for the financial corporations to predict and handle social and environmental risks and constraints in their transactions and projects. The Reserve Bank of India (RBI) has also asked banks across India to responsibly act, adopt greener technologies and contribute towards enshrining environmental sustainability [255]. As a result, the banks (not only in India but also abroad) part from following pacts and policies are greening their processes, service delivery patterns, products. Even the banking infrastructure is being made greener.

Recently, a vital step towards empowerment of green banking, the banks have turned towards adoption of cloud-based computing, storage, infrastructure services thereby turning them to be environment friendly and energy aware. The use of virtualized desktops, printer consolidations, mail server consolidations are all making the realization of green banking through cloud possible. As another notable measure, a large number of BFSI firms are running software that are cloud based and energy conscious such as Finacle etc. [246, 256].

5.3 Prior Analysis and Exploration of a Banking System for Testing

The trend of implementing green solutions and services within the BFSI sector and adoption of cloud computing have both motivated our research that is primarily based on both of these principles. Consequently, the proposed GCSM model has been tested and validated in the real banking scenario. For this purpose, the evaluation has been conducted at a data center of one of a well run comity of public-sector banking firm in the country, Corporation Bank. This section presents a case study of the implementation and testing of GCSM at Bakkarwala branch, located in New Delhi, India.

Figure 5.1: Corporation Bank Data Center

However, prior to implementation of GCSM at Corporation bank, Bakkarwala branch, it became imperative to initially understand the bank facility, its working culture, infrastructural and application capabilities of the branch and interconnection network within the bank and supporting backbone network.

5.3.1 About Corporation Bank

Corporation Bank is a government undertaking banking company which is headquartered at Mangalore, India. The bank has large number of branches throughout the country. Presently, it operates with a network of 2501 fully automated branches; 4724 branchless banking units; and more than three thousand ATMs across the country. The bank also has representative offices in Dubai and Hong Kong. Figure 5.1 represents the data center server racks at Mangalore branch of the bank.

At operational levels, the corporation bank implements Core Banking System (CBS) where a primary server that is core admin server is hosted at its Mangalore headquarters with connectivity network established as a backbone and connected to all the major branches across the country (Figure 5.3). Within each branch, there is network of systems hosted that are interconnected to each other and to the primary server. In this research work, Corporation bank, Bakkarwala branch has been chosen for testing purposes. Figure 5.2 shows the internal working scenario within the Bakkarwala branch of the bank.

Figure 5.2: Internal View of the Bakkarwala Branch, New Delhi

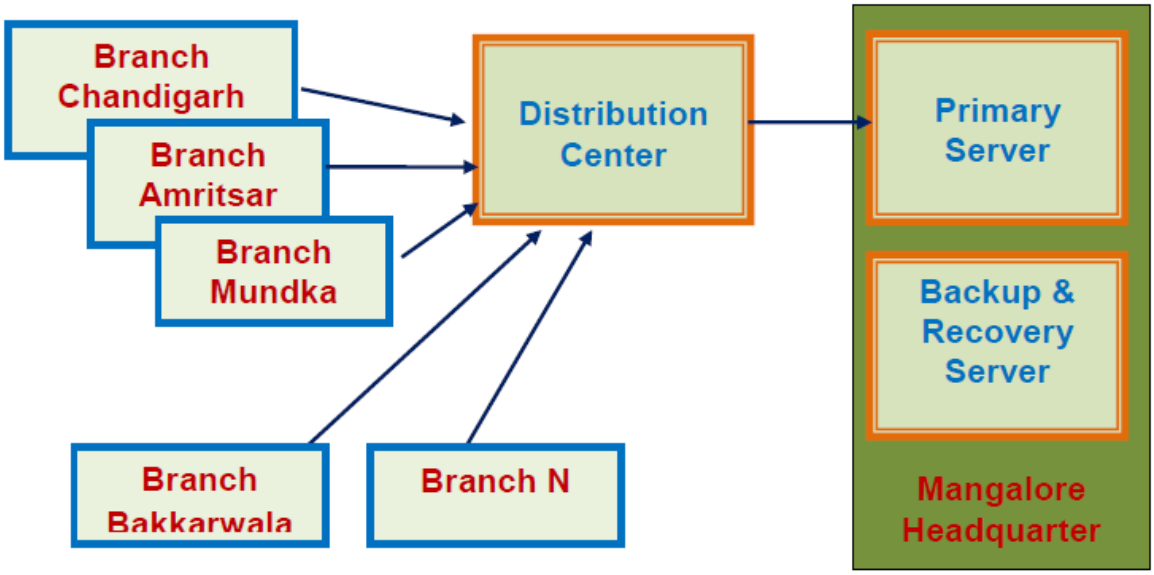

Figure 5.3: Across Country Connection Management in Corporation Bank

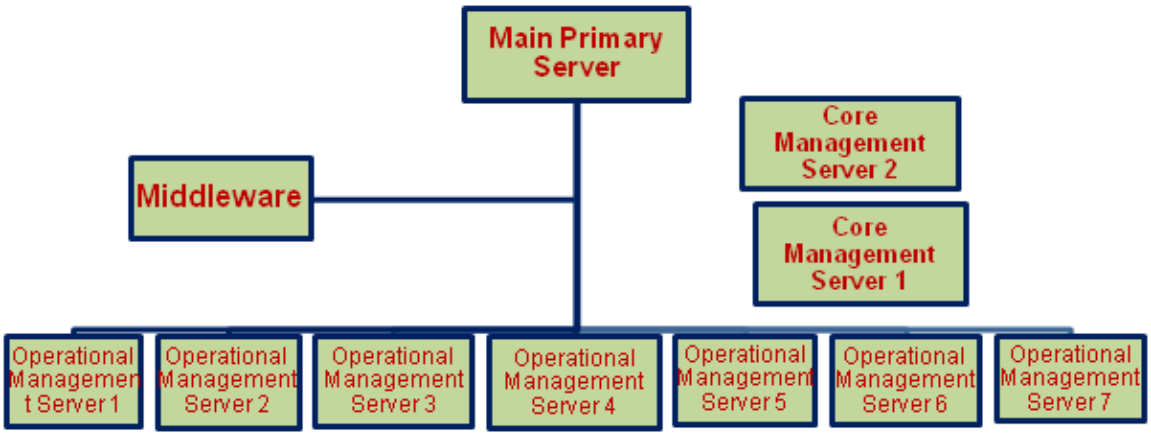

Figure 5.4: Server Hierarchy at Corporation Bank, Bakkarwala Branch, New Delhi

5.3.2 Corporation Bank, Bakkarwala Branch

Infrastructure and Software Details Corporation Bank, Bakkarwala branch hosts 10 servers at the branch out of which 7 are Operational Management Servers (OMSs) that maintain data pertaining to customer details and transactions and handle operations related to it. These servers are numbered from OMS1, OMS2 and so on up to OMS7. Two servers are Core Management Servers (CMS1 and CMS2) that are the primary handling servers within the branch. A middleware is used that integrates primary, operational and core management servers. The servers communicate with each other using middleware API calls.

The primary server hosted at the Mangalore branch controls the entire branches data, hosted on Oracle SuperCluster server series and is based on Solaris OS and stores entire country's branches data. Figure 5.4 depicts a representation for the interconnection network within the branch and to the main controller server or primary server. As depicted, the middleware acts as an bridge between primary server OS, database, control and applications with those hosted at the Bakkarwala branch servers.

The interconnections within the branch from the Access layer to distribution layers are managed through 64K/128K leased lines as primary and ISDN line being secondary. The interconnectivity to the core center and recovery sites is maintained via MPLS/2 Mbps links. Table 5.3 gives details for the Bakkarwala branch infrastructural configuration and Table 5.4 lists each server configuration with its hardware configuration.

Table 5.3: Bakkarwala Branch Infrastructural Configuration

Server	Number	Comment
Core Management Servers	2	To handle interconnections, security and other branch transactions
Middleware	1	To provide interface between applications and server
Operational Management Servers	7	To maintain and handle customer transactions, cash flow, non-cash flow etc.

Table 5.4: Server Operational Descriptions

Server Detail	Type	Application Software
Primary Server	• Hardware • Operating System • Database • Memory • No. of VMs	• Oracle SuperCluster T4-4 and SPARC T4-4 server • Oracle Solaris • Oracle 9 • 6 Oracle Exadata Storage Servers • About 128 VMs
Core Management Servers (CMSs)	• Hardware • Operating System • Database • CBS Software • We Browser • Secondary Memory • Primary Memory-RAM • No. of Cores • No of VMs	• Wipro Desktops Computer • Window 7 Professional • Oracle 9 • Finacle 10.2 • Internet Explorer • 1 TeraByte • 4 GB • 3 • 5-10
Middleware	Application	Red Hat JBoss Enterprise Web Platform
Operational Management Servers (OMSs)	• Hardware • Operating System • Database • CBS Software • We Browser • Secondary Memory • Primary Memory-RAM • No. of Cores • No of VMs	• Wipro Desktops Computer • Windows 7 Professional • Oracle 9 • Finacle 10.2 • Internet Explorer • 500 GB • 2 GB • 3 • 4-7

5.3.3 Banking Transactions and Workloads for Evaluation

CBS system across Corporation bank spans heterogeneous platforms, devices and applications with wide range of resources. Driven under assorted but unified environment, bank's CBS handles diverse multitier workloads hosted on its heterogeneous infrastructure. On higher level, the workloads include several front-end workloads and back-end workloads. The front-end workloads include access channel integration services and other enterprise integration services while back-end workloads include management of servers and databases, security concerns, content management etc. On a higher level, the banking workloads can be categorized into several modules such as:

- Core modules that include tasks pertaining to Management Information System (MIS), General Ledger (GL), customer information system, report generation etc.
- Common In-House Facilities pertaining to inventory management, cash-flow, collateral management activities (such as loan operations), asset management etc.
- User account management activities such as saving account, current account detailing and associated operations, Recurring deposit accounts, and certifications relating to these etc.

Whenever there are transactions related to core banking such as creation, updations, deletions or merging of user accounts etc., they are forwarded and handled through Customer Relationship Management (CRM) system hosted on the servers. These basically are a part of real-time frontend applications and CRM applications. All these transactions and operations form the higher level of workloads while the management and maintenance of these operations at the back-end involve large number of transactions such as continuous database accesses, transaction recordings, audit records, interface activities, security management activities etc.

At the Bakkarwala branch, the OMSs handle different workloads. The banking workloads include transactions related to core banking, retail banking, corporate banking, cash flow, Loan Issuances, asset management, inventory control, foreign exchange, CRM, non-cash flow transactions such as inter-bank transfers and intra-bank transfers including National Electronics Funds Transfer (NEFT) and Real-Time Gross Settlement (RTGS). The CMSs handle and control the OMSs and handle other highly secure core banking transactions. However, the transaction handling and management process follows batch processing as the real-time and live processing within bank is done in batches. The Online Transaction Processing (OLTP) is done on FCFS basis. These workloads require large number of processing and memory-intensive accessibility. For the evaluation purpose of GCSM, the transaction mix comprising of retail and corporate core banking and internet banking transactions including cash flow transactions, balance inquiries, statement requests, transfers and deposits requests were considered.

Recently, Corporation bank has adopted a financial banking package called "Finacle" for their unified workloads and transaction handling CBS implementation across all its branches in the country. Thus, it became extremely important to understand

the working and architecture of Finacle priorly implementing GCSM at the Bakkarwala bank as Finacle forms the baseline application handling all the transactions and workloads pertaining to them.

5.4 Finacle- An Overview

Finacle serves as the base model of banking in Corporation bank. It controls the entire banking services including core, retail, corporate, general ledgers, online, mobile etc. of the bank. paperless transactions

5.4.1 Introduction to Finacle

Finacle is a highly successful and predominant CBS developed by Infosys company and offers rich platform for comprehensive, supple, unified core banking opportunities. It empowers business functionality and facilitates flexible, dynamic configuration of products and processes within the existing banking systems. It is advantageous in terms of its ability to facilitate convenient banking from anywhere. Being based on C/C++ and Java programming languages, it instigates platform independence. It spans multi-lingual capabilities and ensures multi-level security whether at operational, database or application level. It offers large number of benefits for BFSI sector such as:

- Business Benefits: Finacle CBS provides an infinite palette of incredible features for designing and developing new products and services for various market segments. It embroiders the creation of various features at affordable pricing and easily customizable offerings. Figure 5.5 shows affluent solutions and services offered by Finacle [245].
- Enhanced Efficiency and Productivity: Finacle CBS incorporates business automation and orchestration of process, averting manual task handling and minimizing processing time. The error-free and redundant results enhance the productivity of the branch while limiting the burden. In fact, closely inter-related to the above productivity parameters, turnaround time reduces and customer response time improves.
- Improvised Customer-Bank Relationship: The enterprise customer management and CRM provisions in Finacle provide a unified environment of the customers thus strengthening relationships among banks and customers.
- Adaptability and Agility: Finacle allows IT teams within banks to easily make changes without tendering or altering existing base code thereby reducing operational risks and vendor dependencies.
- Standardized banking platform: Finacle CBS enables hassle-free delivery and uniform banking experience to the customers.
- Competitive Advantage: The robust and ample core banking facility of Finacle enables banks to expand and extend their services and product offerings thus augmenting competitive advantage to them.

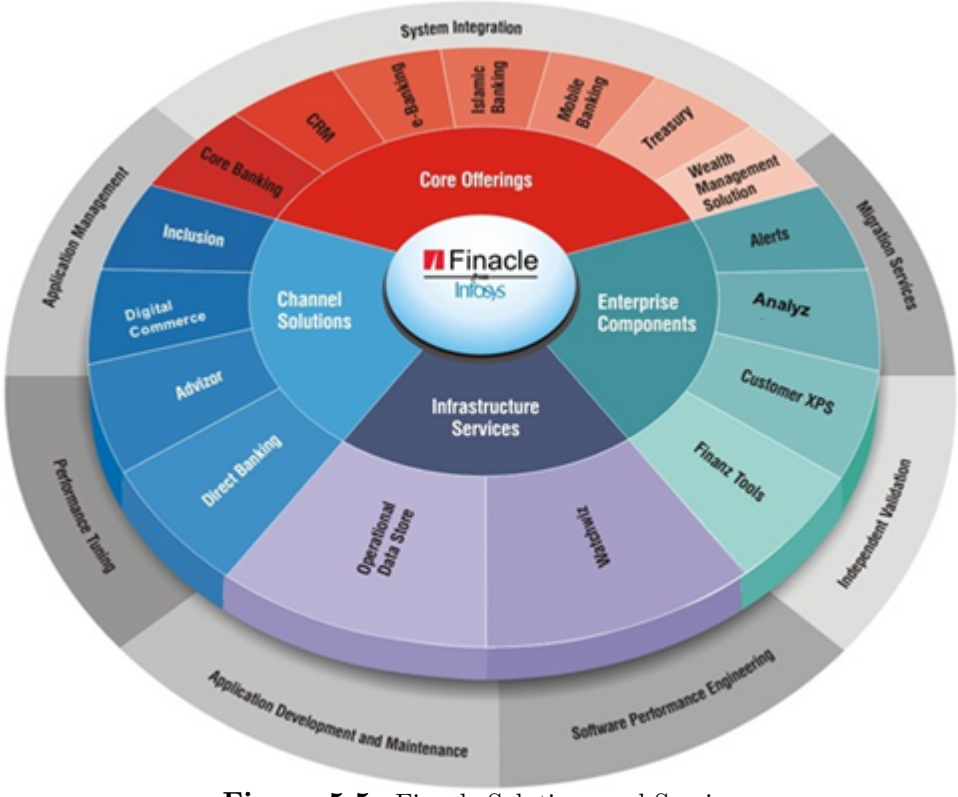

Figure 5.5: Finacle Solutions and Services

- Improvised monitoring and organization: The central control and maintenance of master data pertaining to customers and products improvises and eases organization and monitoring. This simplifies updations and report generation processes.
- High level security: Finacle provides highly secure management by using built-in checks at database, application and user- levels. The role-based accessibility and authentication procedures in Finacle prompt security at management level too.
- Spans multiple channels and supports multi-country technological aspects: The integration of financial and accounting systems with multi-currency support enables banks to span multiple channels and profit centers.

5.4.2 Functional Architecture of Finacle

Finacle has a component-based structure where integration of new modules and upgradtion of existing modules and components can be easily done. Additionally, the structure integration and compliance to different applications poses minimal alterations and costs.

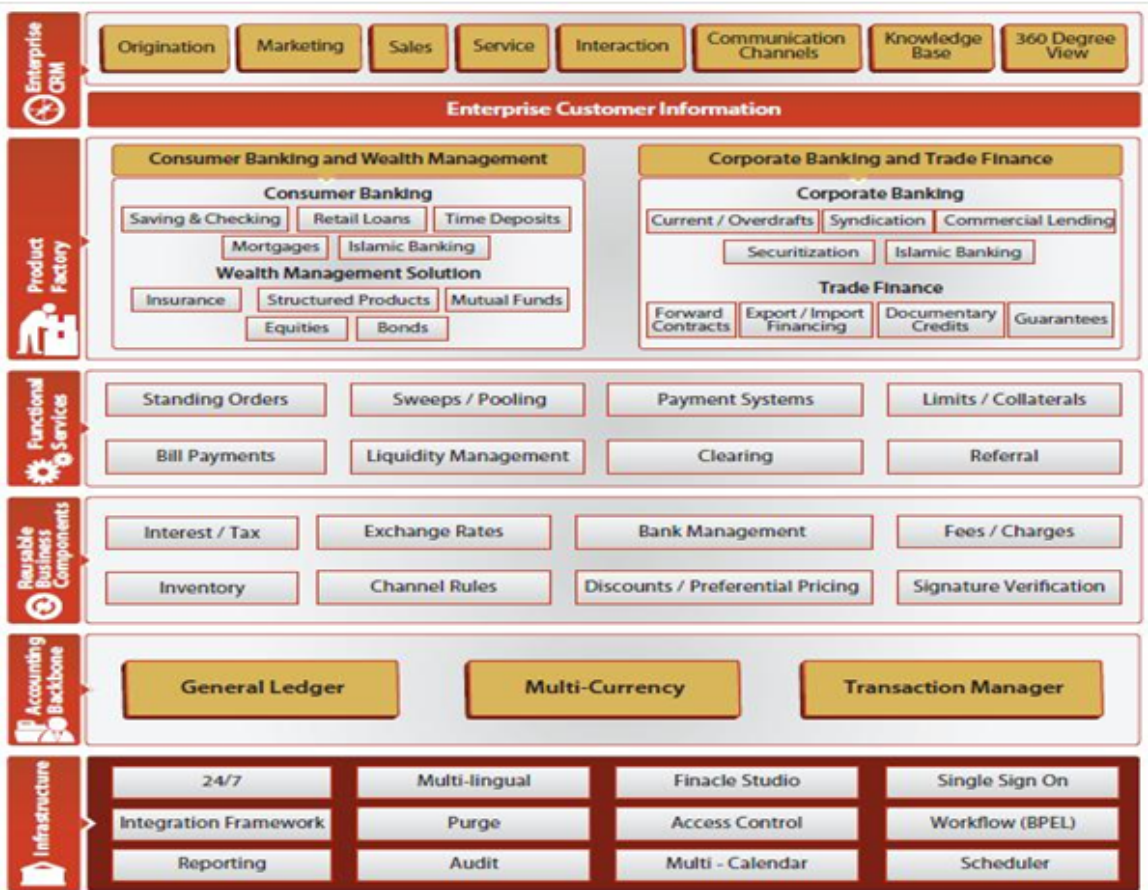

Figure 5.6: Functional Architecture of Finacle

Finacle follows Service-Oriented Architecture (SOA) and is a solution devised on a private cloud that has been tailored as core, mobile and internet banking enabler [257]. The SOA-based architectural model minimizes operational issues such as alterations, incorporations and adaptableness. Figure 5.6 presents the functional architecture of Finacle.

As shown in the architecture, Finacle has modular architecture where Enterprise CRM, Consumer banking, Corporate banking, Infrastructure, Wealth management, Trade Finance, Payments, Ledger, Liquidity management etc. are the key modules operating independently. These modules perform variable functions and consist of several menus and sub-menus. The Finacle also has a scheduler unit at the infrastructural level. The Finacle scheduler is a batch scheduling unit that performs the scheduling of the transactions in batch mode. Finacle offers greener solutions because it promotes paperless banking, induces minimal lines of code and increases the number of transactions processed per unit of energy consumed.

5.5 Setting Up of Cloud Test Bed for Evaluation

The tools used for test bed comprised of Eclipse Mars 4.5.0 [258], GCS Algorithm of GCSM, Finacle, Joulemeter [205] and Oracle. Eclipse Mars is an IDE for development of Java EE applications. GCS algorithm of GCSM has been written in Eclipse IDE. Joulemeter is used to measure the electrical power consumed by all the nodes in Bakkarwala branch. This electrical power measurement information is forwarded to GCSM that will compute the energy of each node and schedule the transaction to a node as per the energy values. Finacle application acts an interface for workload processing. The Eclipse web tools platform and Finacle integrator services act as integrator services between Eclipse and Finacle. These services help to integrate and extend the scripts written in Eclipse with Finacle. The integration of multiple environments used to conduct experiments is shown in Figure 5.7.

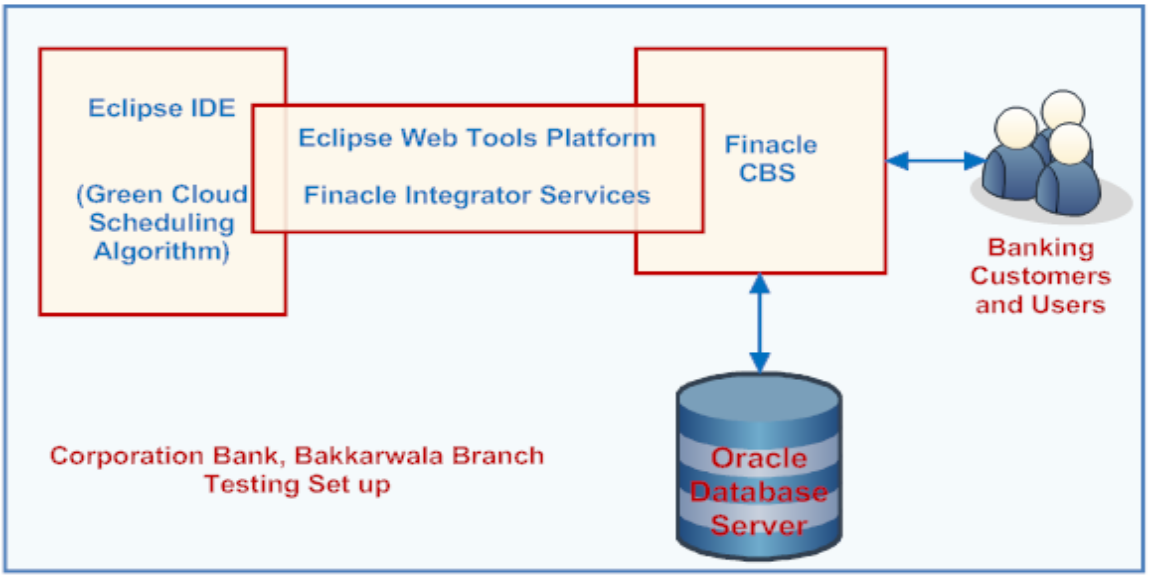

Figure 5.7: Evaluation Testbed established at Bakkarwala Branch

The solution suite consisted of multiple operational layers. Figure 5.8 shows the layered Corporation Bank, Bakkarwala branch for evaluating GCSM. The layers include:

i Physical Infrastructure and Network Interconnection Layer: This layer comprises of Finacle's installation and configuration and its underlying infrastructure and application servers. Primarily, Finacle's infrastructure is managed by the Finacle Infrastructure Management Services (IMS) that constantly monitor it..At Bakkarwala branch, the application servers include (OMS 1 to OMS 7; CMS 1 and CMS 2) and storage. This layer also comprises of primary web

server and back end database. The GCSM scheduling unit, that is, Green Cloud Scheduler also form part of this layer.

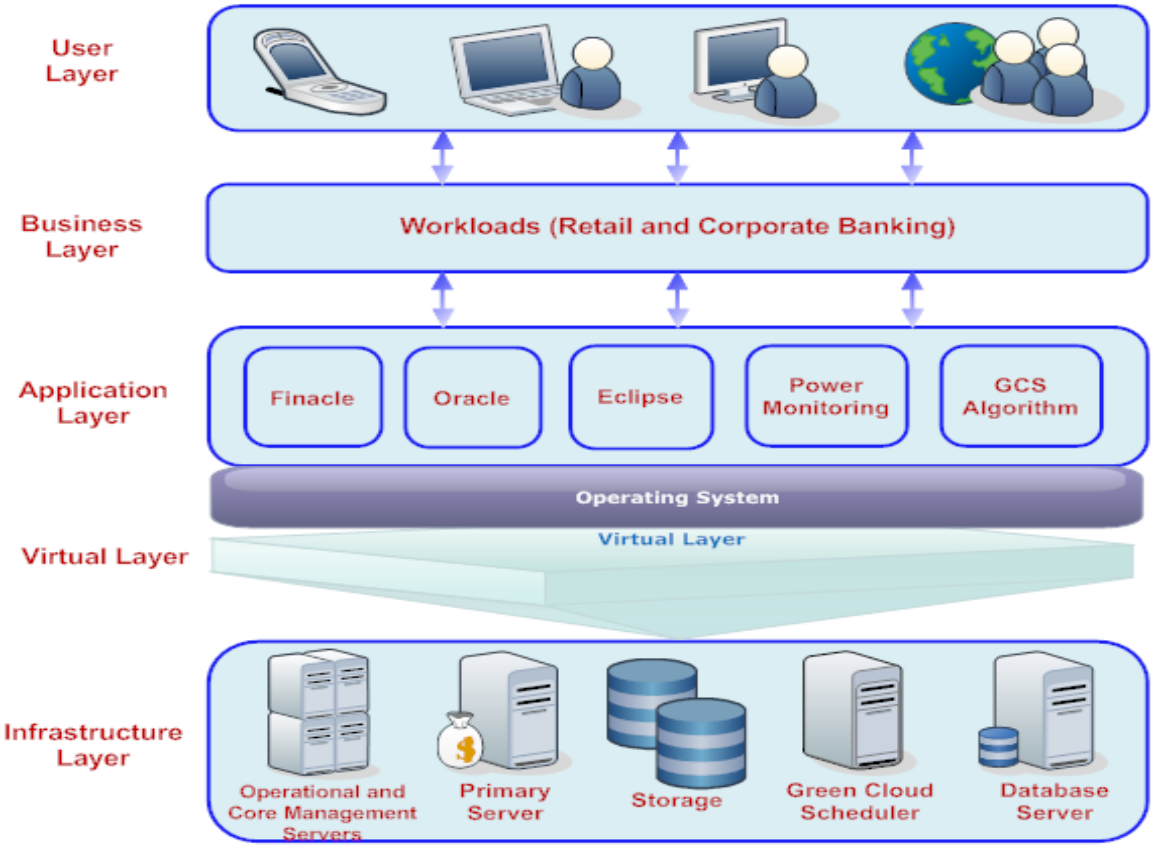

Figure 5.8: Layered Suite for Evaluation of GCSM at Bakkarwala Branch

ii Virtual Layer: This layer corresponds to the VMs running on the top of the physical infrastructure.

iii Application Layer: The application layer hosts the Finacle software, Oracle, Power monitoring application, OSs, Eclipse, middleware and GCS algorithm.

iv Business Layer: The business layer consists of the banking workloads for processing. In case of evaluation and experimentation of GCS, the analysis has been restricted to retail and corporate internet banking workloads and transactions.

v User Layer: The user layer corresponds to the interfacing equipments used by the customers and users such as desktops, laptops, smart devices, ATMs etc.

After a workload comprising of transactions is submitted by the user, the transactions are assigned job Id (or transaction Id) and branch Id by the primary server. The transactions are handled by the Finacle modules in batch processing mode. At the branch, each transaction information is stored in the database of GCSM. GCSM also maintains information of all the nodes (OMSs and CMSs) running Finacle at the branch. It obtains electrical power measurements for each node that are running power monitoring application on them and further computes energy for the nodes. It also stores all this computed information in its database. Based on the computed energy values, the transactions are allocated to nodes having least energy consumption value for processing. This process continues for each transaction. Further transaction processing is handled by Finacle and its modules while it accesses the Oracle database server.

5.6 Experimental Analysis and Results

The experimental analysis at Bakkarwala branch involved running of Finacle with its default in-built BJS scheduler followed by experimental run of GCS technique of GCSM model. A comparative experimental analysis between Finacle in-built BJS, GCS algorithm and an autonomic energy-aware resource scheduling technique proposed in [204] has also been conducted.

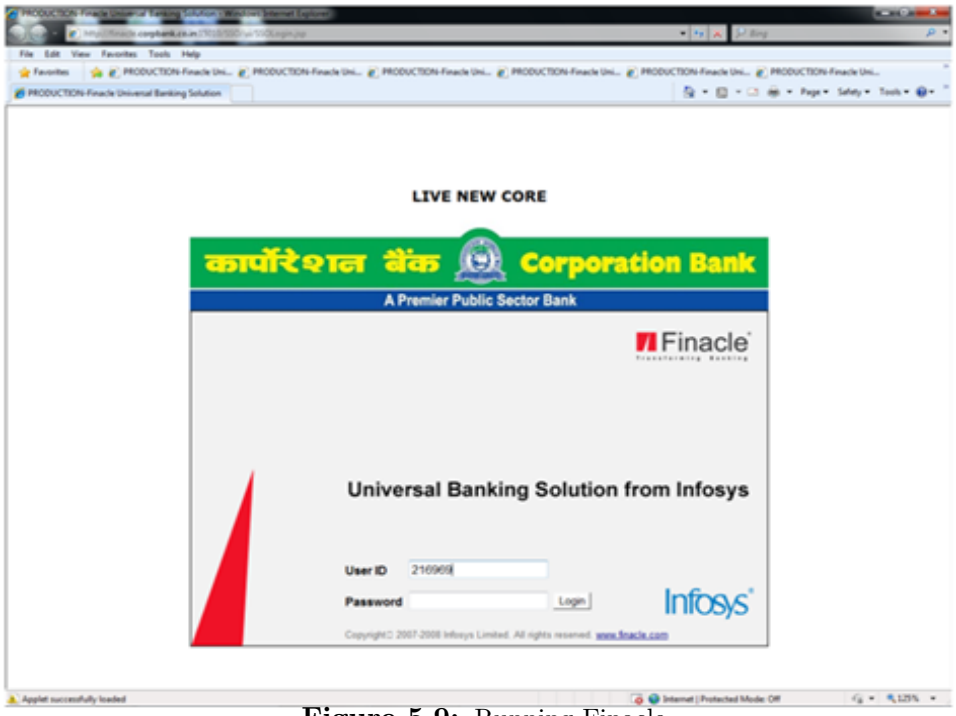

Figure 5.9: Running Finacle

Case 1: Experimentation of In-built Batch Job Scheduler (BJS) of

Finacle

The experimental analysis of In-built Finacle scheduler begins after running Finacle to handle the transaction on the branch nodes. Figure 5.9 depicts the initial running screen for Finacle with the employee ID details. Each branch employee is given unique Id and password for accessing Finacle. The branch Id for the analysis used in this research work is 216969.

Finacle has an In-built Batch Job Scheduler (BJS) that handles automation, invocation and processing of batch jobs or transactions at fixed schedules. For evaluation on Finacle, each transaction has been assigned a unique identification number called as "Tran Id" that identifies it. The transactions are also assigned a branch Id to identify the branch where it has been submitted for processing. The inbuilt scheduling algorithm for Finacle is written in Finacle scripting studio [256]. Figure 5.10 represents the running of Finacle in-built scheduler configuration file. Figure 5.11 represents the procedure used for generating transaction id and branch id. Figure 5.12 presents the background screen for transaction being handled in the Finacle scripting studio.

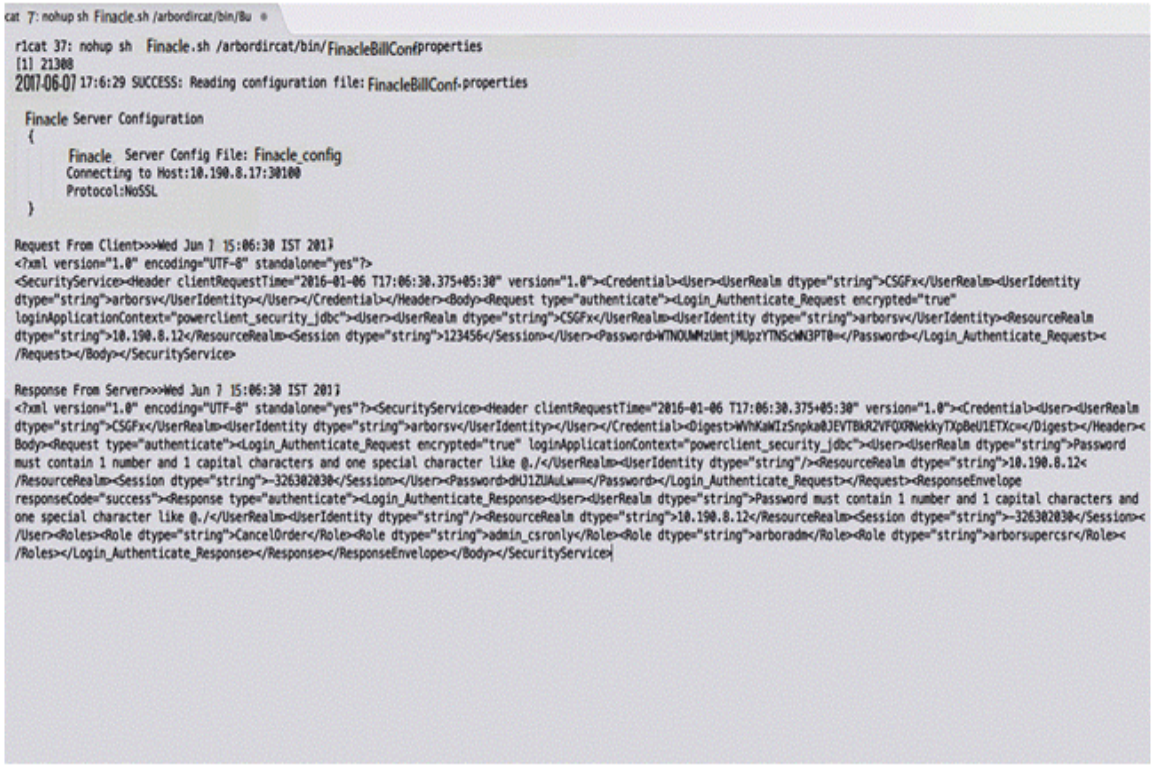

Figure 5.10: Initiation of In-built BJS of Finacle

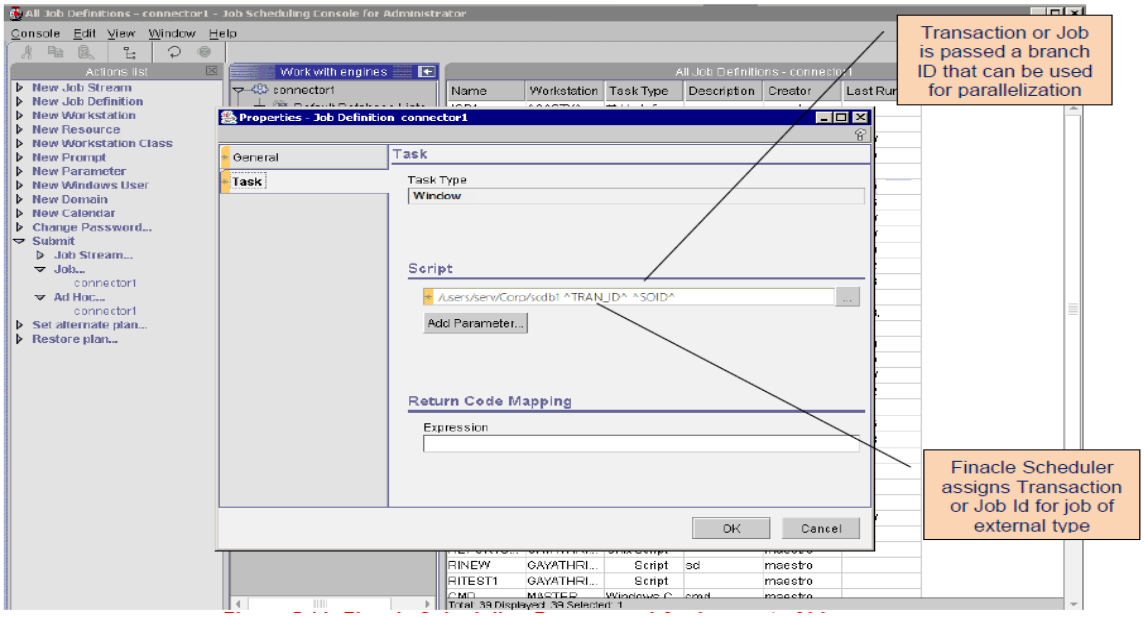

Figure 5.11: Finacle Scheduling Process and Assignment of Ids

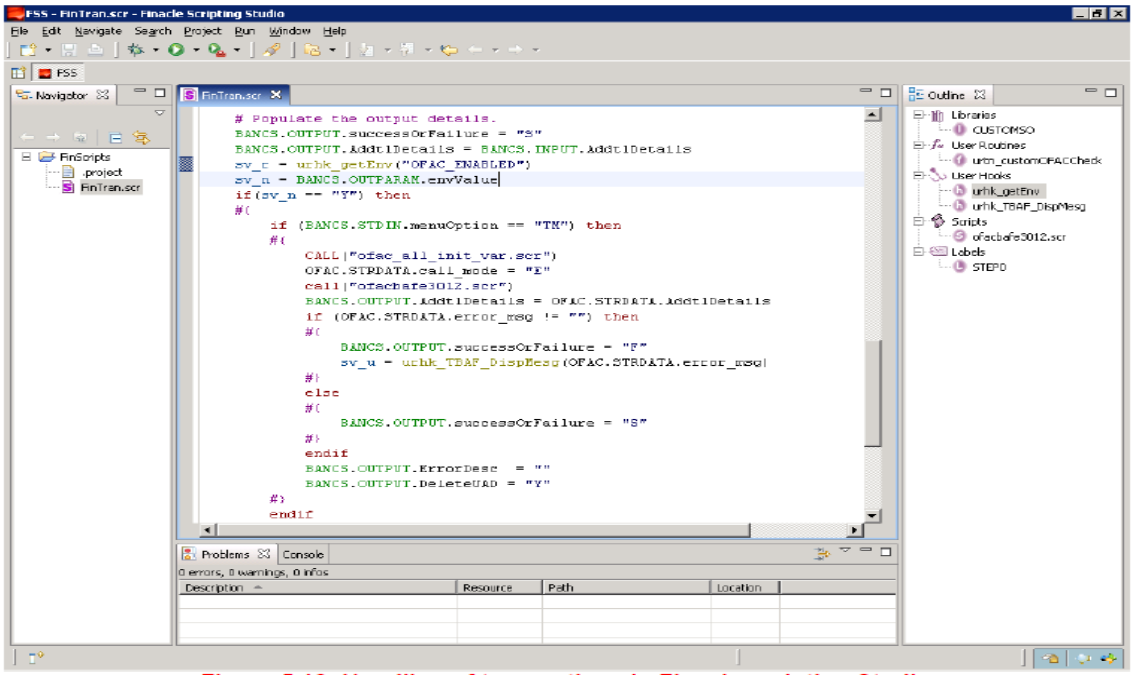

Figure 5.12: Handling of Transactions in Finacle scripting Studio

The results upon running the Finacle In-built BJS scheduler have been obtained

when the number of logged in concurrent users is 2506. The experiment has been restricted to the period of 4 peak hours when the workload mix of different transactions including account creation, cash flow and ledger access transactions takes place. It is important to note that even a single account creation, search, deletion command or transaction in Finacle involves about hundreds of sub commands or sub-transactions such as database access transactions, storage transactions etc (Figure 5.13). On a whole, it has been estimated that about 356 hundred number of transactions have been processed during the 4 hour period.

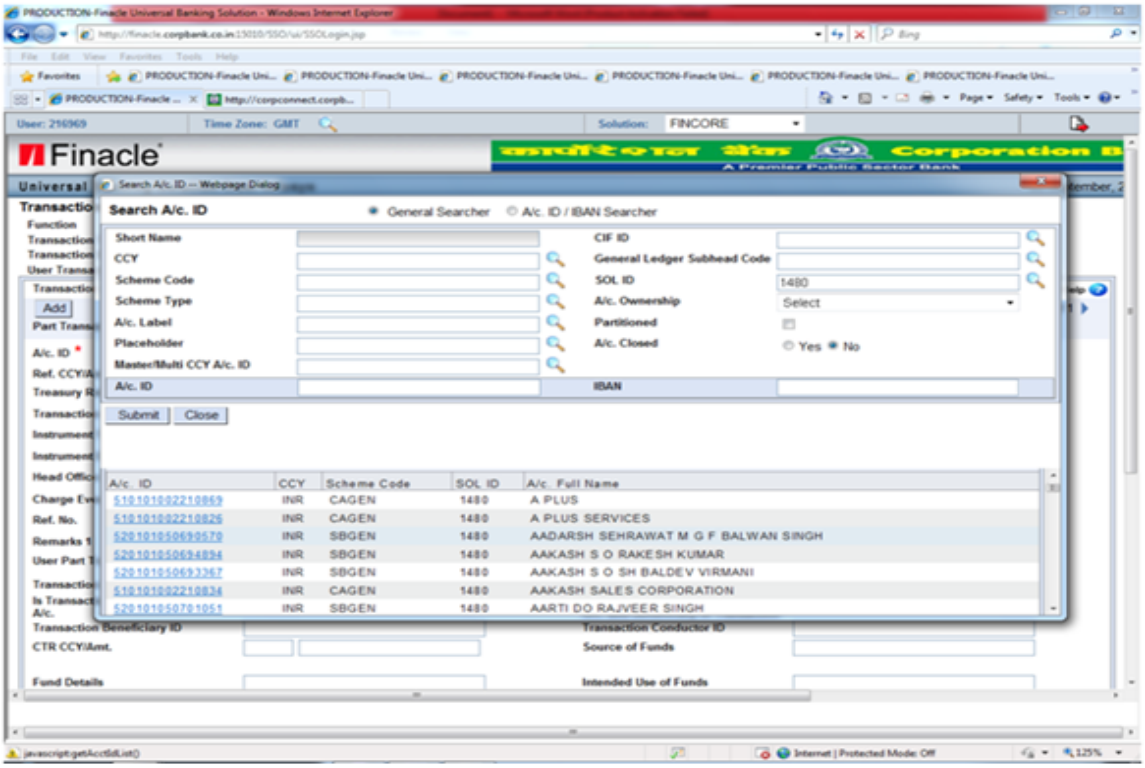

Figure 5.13: Account ID Search Command Prompting Additional Accesses to Records in the Database and Memory

Figure 5.14 depicts the energy consumed when running workloads under Finacle and its in-built BJS. It has been observed that the energy consumption increases arbitrarily. This perhaps is result of the fact that for every single workload, a large number of transactions are executed. For example, even the creation, updation, search or deletion of a user account involves about 20-300 transactions that require access to processor and memory regularly or alternatively. These fluctuation and alternate accesses cause increase in the overall system energy consumption. Although, as per [232], there exists a proportional relationship between workload and energy efficiency but depending upon the kind of workloads, this relationship cannot be generalized [259]. In this case of banking workloads, energy rises with the rise in

the workloads intensity especially when the workloads are processed in batches. The change in energy levels has been attributed to the variety of workloads mix processed by the nodes. Figure 5.15 depicts the variable transactions or jobs handled by OMS4.

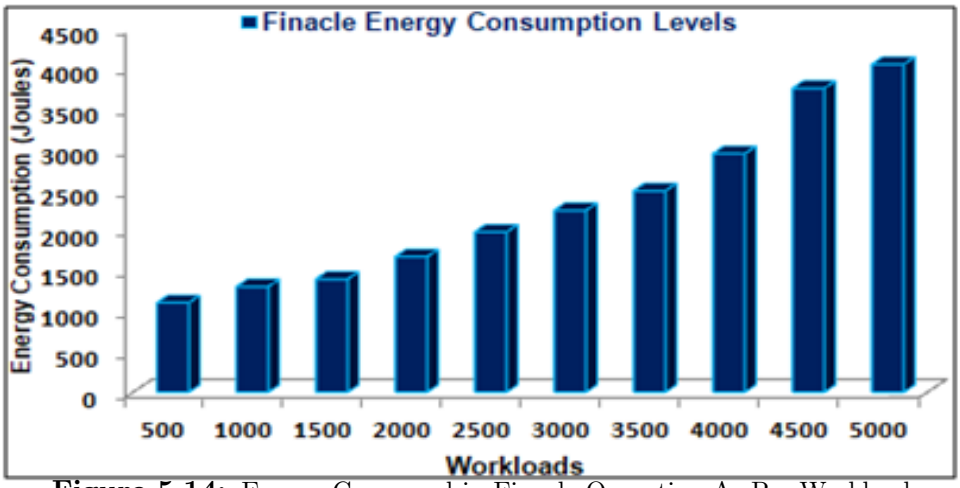

Figure 5.14: Energy Consumed in Finacle Operation As Per Workloads

```
CAT (telnet)          MW (telnet)          telnet
System: OMS4
Energy averages- 0.54, 0.36, 0.13
3352 Jobs- 2245 Running
Resource Utilization
CPU   ENERGY USER    NICE     SYS    IDLE   BLOCK   SWAIT    INTR    SSYS
 0     0.50  30.0%   0.0%    2.8%  67.2%    0.0%    0.0%    0.0%    0.0%
 2     0.53  24.7%   0.0%    3.0%  72.3%    0.0%    0.0%    0.0%    0.0%
 3     0.49  23.6%   0.0%    3.0%  73.4%    0.0%    0.0%    0.0%    0.0%
 4     0.53  29.4%   0.0%    4.2%  66.4%    0.0%    0.0%    0.0%    0.0%
 5     0.47  34.1%   0.0%    2.0%  63.9%    0.0%    0.0%    0.0%    0.0%
 6     0.31  24.9%   0.0%    2.0%  73.1%    0.0%    0.0%    0.0%    0.0%
---    ----  -----   -----   -----  -----   -----   -----   -----   -----
avg    0.47  27.8%   0.0%    2.8%  69.3%    0.0%    0.0%    0.0%    0.0%

Memory: 25142284K (7400444K) real, 26535348K (8393020K) virtual, 34156848K free

CPU TTY   JID USERNAME  PRI NI    SIZE     RES STATE     TIME %WCPU  %CPU COMMAND
 6   ?  1289 oracle    199 20  4737M 10028K run     701:41 20.70 20.66 oracleN02PCS12
 0   ? 13281 oracle    154 20  4743M  8744K sleep     0:42  6.27  6.26 oracleN02PCS12
 2   ?    66 root      152 20 79200K 70400K run    1515:38  6.04  6.03 vxfsd
 3   ? 17204 arbor     152 22  1376M  1142M run       0:07  4.25  3.99 java
 4   ? 14445 oracle    154 20  4800M 73444K sleep     0:08  2.61  2.60 oracleN02PCS12
 2   ?  7092 oracle    154 20 37600K  3368K sleep   299:29  1.33  1.33 tnslsnr
 5   ? 26617 oracle    154 20  4760M 35084K sleep   229:10  1.20  1.20 oracleN02PCS12
 6   ?  8330 root      154 20  7896K  1704K sleep     5:25  0.96  0.96 disk_em
 5   ?  8250 root      152 20 61088K  5604K run      89:59  0.92  0.92 WbemWrapperMonitor
 2   ?    40 root      191 20  1152K  1024K run     221:05  0.89  0.89 pagezerod
 9   ? 23131 oracle    154 20  4746M 19900K sleep    50:40  0.85  0.84 oracleN02PCS12
 0   ?  4071 oracle    154 20 21626M  8680K sleep     0:36  2.84  2.83 oracleN02PCS12
 3   ? 17057 oracle    148 20 22206M 14204K sleep    19:19  2.47  2.46 oracleN02PCS12
 0   ?  5603 oracle    154 20 21638M 20044K sleep    27:06  2.17  2.16 oracleN02PCS12
 4   ?  5605 oracle    154 20 21638M 20300K sleep    25:17  1.92  1.91 oracleN02PCS12
 6   ?  5024 root      -16 20 56472K 33316K run    21210:50  0.95  0.95 midaemon
 4   ?  5788 oracle    154 20 21625M  8168K sleep     0:01  1.11  0.95 oracleN02PCS12
 4   ?  5609 oracle    154 20 21638M 20028K sleep    23:37  0.92  0.92 oracleN02PCS12
```

Figure 5.15: Utilization and Energy Values for OMS4

Figure 5.16 represents the energy consumption done at all the nodes in the Bakkarwala branch (given in Table 5.3). The energy values have been obtained

from an average of total energy consumed by the nodes during the duration of 4 hours of operation. Clearly, the energy consumed by CMS systems is much more than the OMSs. This is because the CMS are the core servers that handled additional workloads pertaining to accessibility, interface services, audit reporting and security services etc.

During the testing period, the resource utilization has been observed to be within the acceptable levels and energy consumed changing considerably. Figure 5.17 depicts the utility percentage of primal system resources mostly responsible for high energy incurrence such as CPU and memory. It has been observed that the CPU utilization of OMSs is 60 percent while memory utilization falls around 49 percent. Contrarily, the CPU utilization of CMSs is 66 percent with 57 percent memory utilization. The higher utility level of CMSs has been attributed to the elevated memory, storage and processor accesses performed by them in contrast to OMSs.

These results have been obtained to analyze the impact of batch processing done by Finacle In-built BJS on the overall energy consumed by the system and by the nodes at the branch. The scheduler negatively impacted the level of energy consumption owing to the processing of transactions or jobs in batches. For OLTP transactions, Finacle follows FCFS scheduling that also causes increase in the overall energy consumed by the systems.

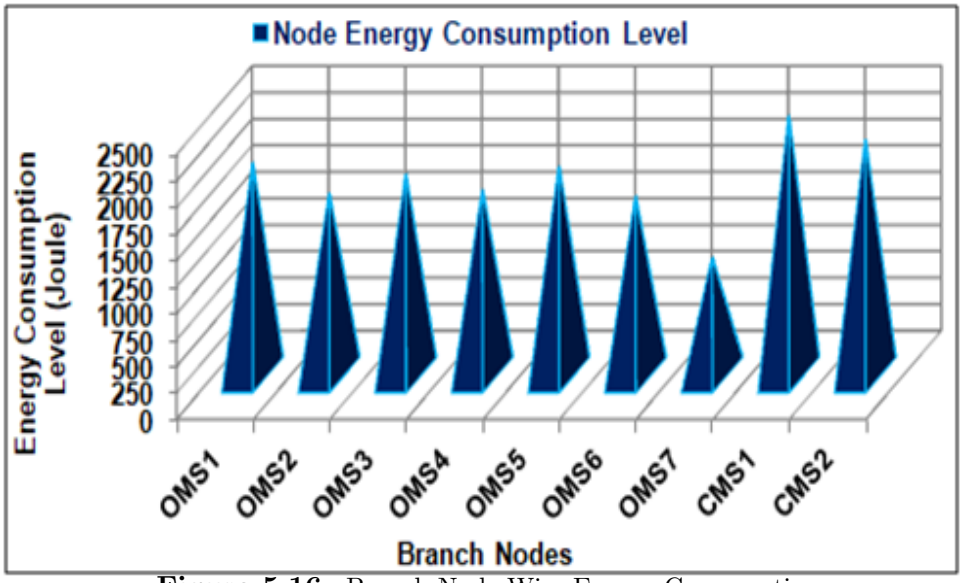

Figure 5.16: Branch Node-Wise Energy Consumption

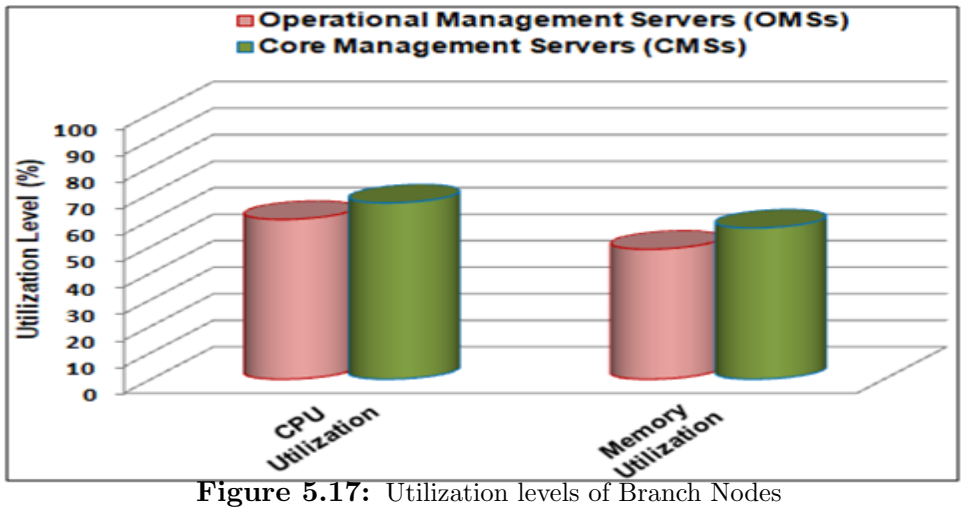

Figure 5.17: Utilization levels of Branch Nodes

Figure 5.18 represents the impact of batch processing on the performance. Although, Finacle has been designed to achieve high performance in terms of faster processing and throughput but due to high energy consumption and non-parallel batch executions, the depreciation in the performance takes place over the period of time. Also, the testing results have been obtained during peak hour period, when the transaction rate is high and significant processor and memory accesses have been performed along with the database accesses and network connectivity. This has resulted in affecting the amount the energy consumed and wasted in the processing activity thus hindering the performance in the long run.

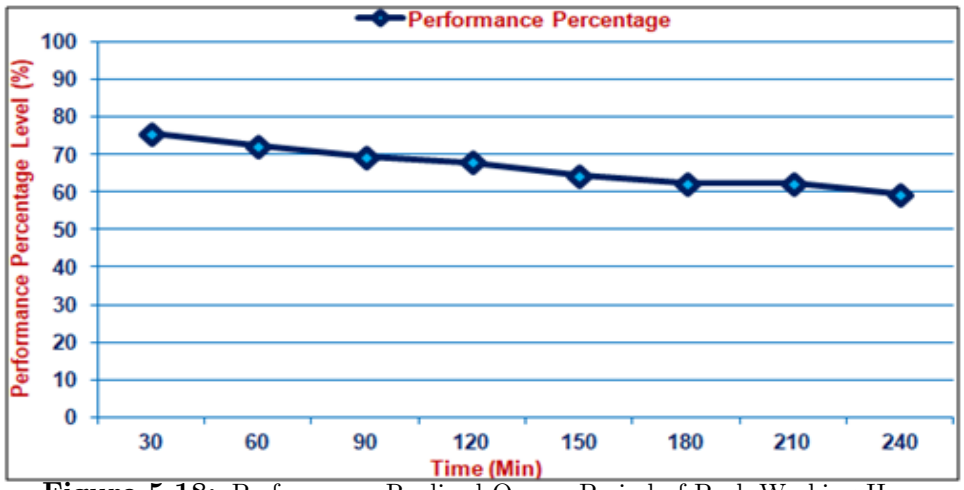

Figure 5.18: Performance Realized Over a Period of Peak Working Hours

Case 2: Experimental Results of Green Cloud Scheduling (GCS) Algorithm of GCSM

The experimental results upon running GCS algorithm of GCSM have been obtained when the Finacle BJS is bypassed with the scripting code of GCS. GCS keeps track of the branch nodes and their power consumption values. It computes the energy consumed by them and schedules the incoming transactions through Finacle interface on the nodes that consume permissible levels of energy. This energy based transaction handling tends to control the energy consumed during the operation.

Figure 5.19 depicts the energy consumption levels noted while running GCS at variable levels of workloads. It can be noticed that although initially the energy consumed is higher but later on it converges to limited levels even though the workloads rise. The initial up burst in the energy levels is due to the batch scheduling done by Finacle that protracts lowly utilized nodes or idle nodes. However, over the period of time , GCS algorithm identifies energy-conscious nodes and allocates transactions to them. This stabilizes the energy consumption. Additionally, GCS tracks the under-utilized nodes and turns them off to prevent further energy wastage.

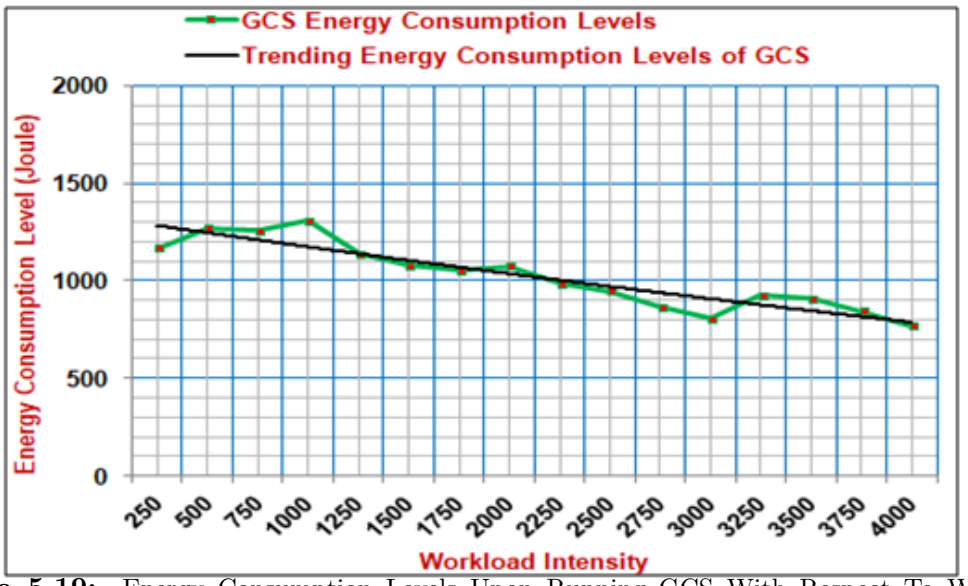

Figure 5.19: Energy Consumption Levels Upon Running GCS With Respect To Workload Intensity

The controlled energy aware handling also averts the performance degradation and improves the utilization of nodes. Figure 5.20 shows the performance percentage obtained by GCS. Clearly, although the intensity of workloads increases but because the energy consumption has been kept under guard by GCS, the performance is not affected. Thus GCS while lowering energy consumption, protects node utilization levels and thus prevents associated performance bottlenecks.

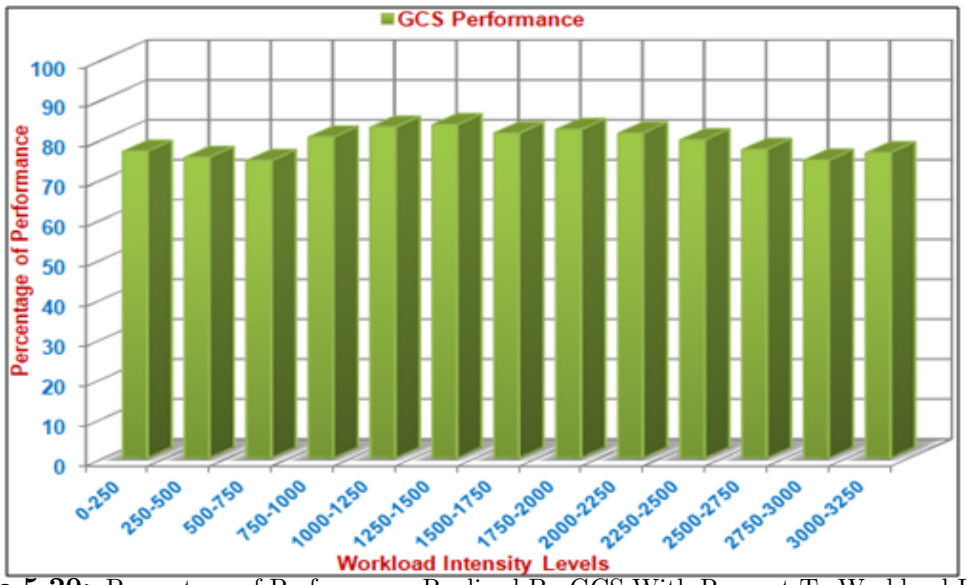

Figure 5.20: Percentage of Performance Realized By GCS With Respect To Workload Intensity

Similarly, Figure 5.21 depicts the percentage of performance realized by GCS at different time intervals. The performance-time analysis has been conducted after every 30 minutes interval. A short interval has been used in order to continuously analyze any performance fluctuations if occurring while large number of transactions have been processed .In this case, due to ability of GCS to continuously monitor the nodes for energy and performance, the performance has been stabilized.

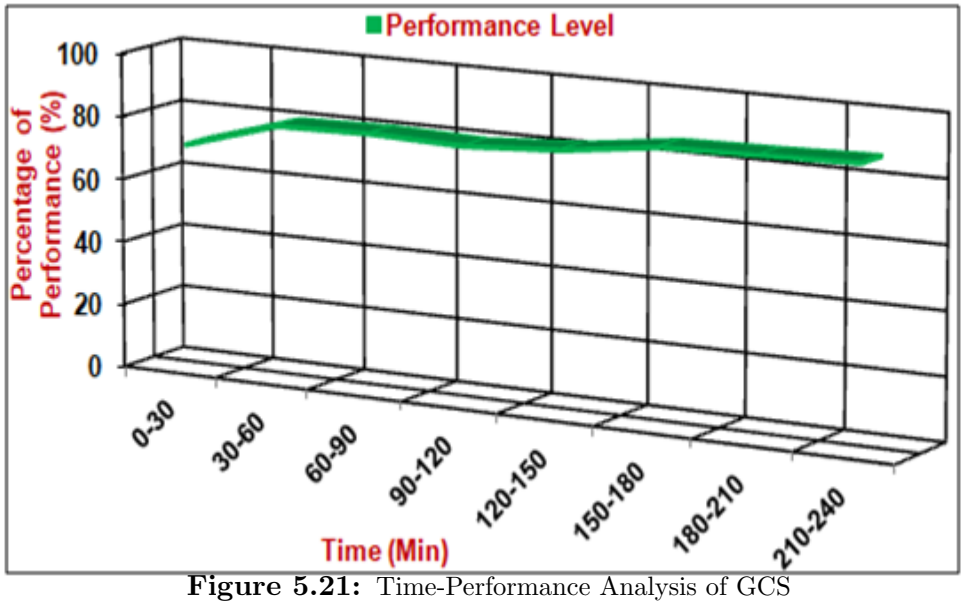

Figure 5.21: Time-Performance Analysis of GCS

Figure 5.22 shows the utilization of computing resources (CPU and memory) of the branch nodes when GCS algorithm is implemented. It can noticed that the

utilization levels are better than those realized while implementing Finacle in-built BJS scheduler. It has been observed that the overall 80 percent of CPU utilization has been achieved while accomplishing 67 percent utility of memory across the branch nodes.

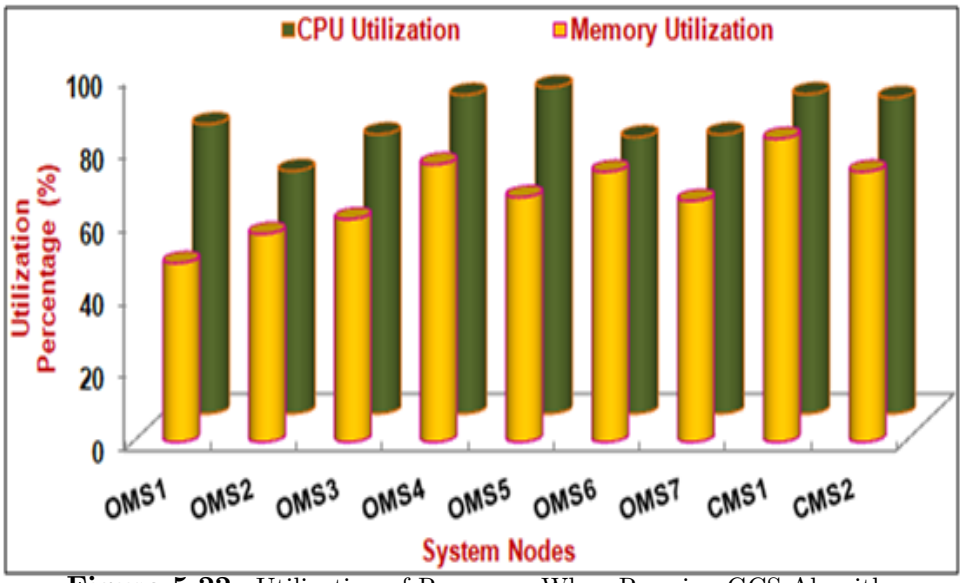

Figure 5.22: Utilization of Resources When Running GCS Algorithm

Case 3: Comparative Analysis of GCS Algorithm of GCSM with the Finacle In-built BJS and EARTH

The proposed GCS algorithm of GCSM model has also experimentally compared for energy, performance and deadline metrics with Finacle BJS and EARTH. The results obtained are represented and discussed below. Figure 5.23 shows the percentage energy savings realized by the three techniques measured at variable time intervals and indicated by the number of system runs. It can be attributed to the consistent energy consumption tracking and handling done by GCS that the energy savings are achieved. The energy efficiency obtained by GCS is stable and higher as compared to the two other techniques.

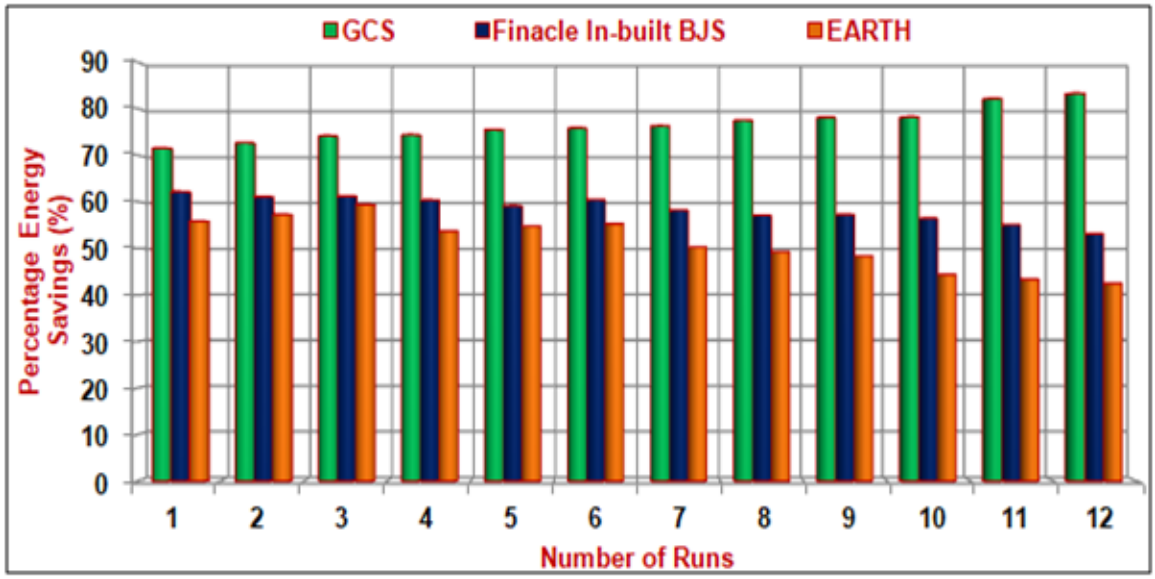

Figure 5.23: Percentage Energy Saving Achieved by GCS, Finacle BJS and EARTH

Figure 5.24 depicts the performance realized by the three techniques at variable time intervals spanning 4 hours analysis duration. It has been observed that the performance enhancement and stability is much higher when GCS operates as compared to Finacle BJS and EARTH. In fact performance dilapidation takes place at persistent level in case of EARTH technique. As shown in Figure 5.25, overall GCS achieves 81 percent performance as compared to 67 percent and 60 percent performance levels achieved by Finacle In-built BJS and EARTH respectively.

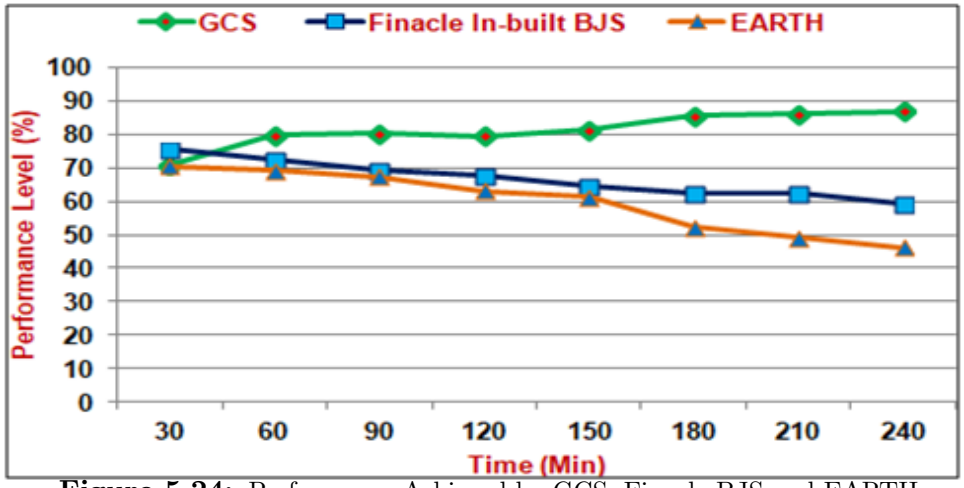

Figure 5.24: Performance Achieved by GCS, Finacle BJS and EARTH

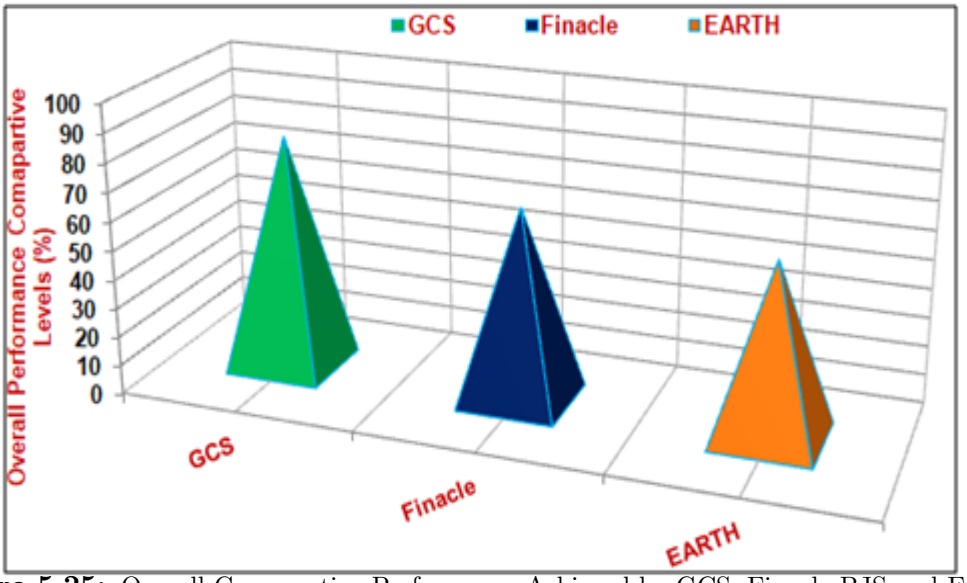

Figure 5.25: Overall Comparative Performance Achieved by GCS, Finacle BJS and EARTH

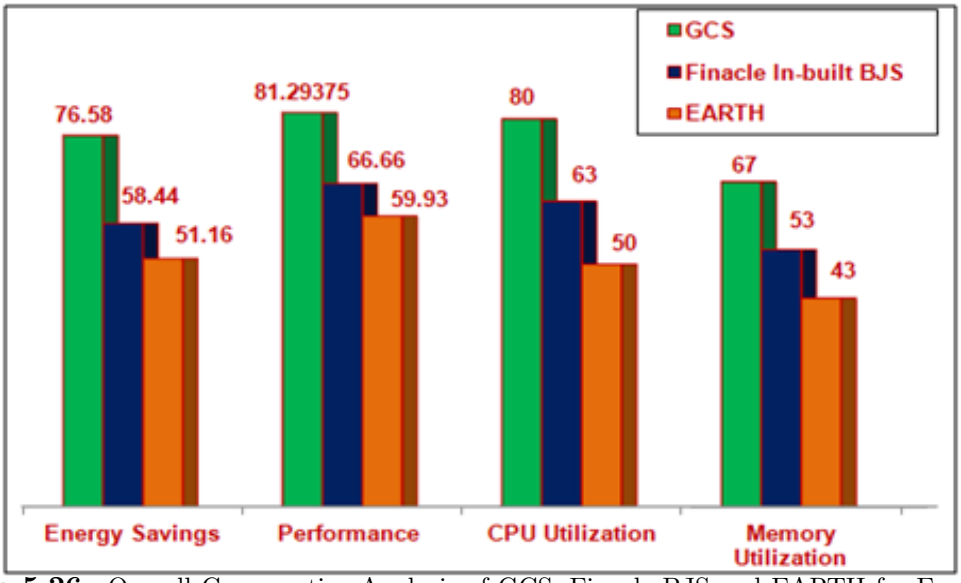

Figure 5.26: Overall Comparative Analysis of GCS, Finacle BJS and EARTH for Energy Efficiency, Performance, Resource (CPU Memory) Utilization

Discussion

This section discusses the operational efficiency and improvements if any made by the three techniques, that is, GCS, Finacle In-built BJS and EARTH. Figure 5.26 depicts the individual and overall energy efficiency, performance, utilization of resources (CPU and memory) accomplished upon implementing the three techniques.

It has been observed that overall GCS achieves 77% energy savings while utilizing 80% CPU and 67% memory capabilities along with 81% performance attainment. Contrarily, Finacle accomplishes 58.4% energy efficiency with 63% CPU and 53% memory utilization levels. An energy saving up to 51.16% is realized by EARTH with 50% CPU and 43% memory utility levels and 59.93% performance.

Clearly, GCS achieves higher energy savings, performance, utilization levels as compared to other two techniques. Figure 5.27 depicts the improvement graph of GCS over other two techniques.

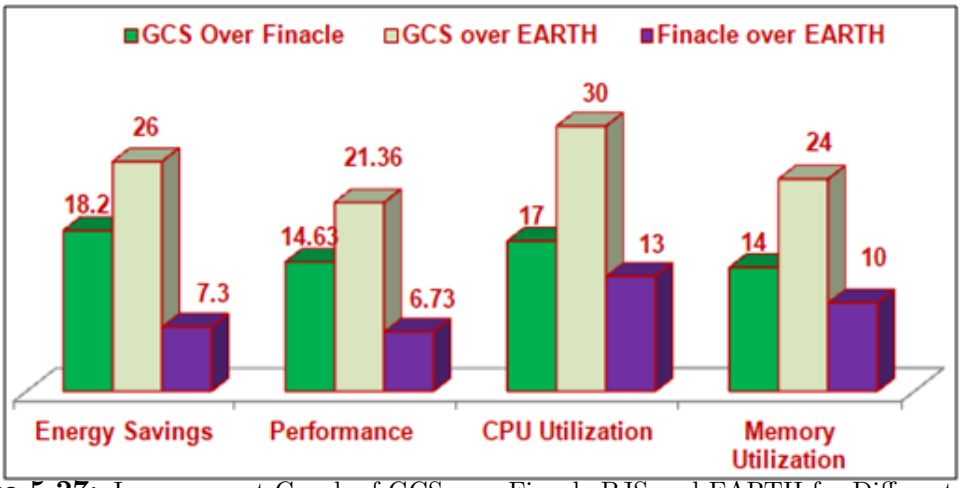

Figure 5.27: Improvement Graph of GCS over Finacle BJS and EARTH for Different Metrics

Thus, the implementation of GCS technique of GCSM yields more energy efficiency, performance and utilization as compared to other two techniques. Also, it emerges as a solution for promoting and enhancing greener banking efforts in the BFSI sector.

5.7 Conclusion

This chapter presented the verification details, experimental setup and testing results of the proposed GCS technique of GCSM at Corporation Bank, Bakkarwala branch, New Delhi. The experimental results demonstrate the efficacy of GCS by saving an average of 76.58% energy with 80% CPU and 67% memory utility levels. GCS achieves 81.3% performance while minimizing the energy and thus achieving green banking operation at the bank.

The next chapter concludes the thesis by highlighting the main contributions of this research work and gives the future research directions.

Chapter 6

Conclusions and Future Directions

Energy efficiency and cloud computing have emerged as the latest buzz in the ICT industry today. The growing trend of cloud computing and cloud data centers has led to the increase in the energy demand. Eventually, energy aware data centers have become a prime design restraint for the IT sector. The energy consciousness within the data centers ranges from incurring greener hardware equipments, air-conditioning systems to the implementation of energy-aware software, applications and mechanisms.

This thesis intends to incur energy efficiency in the cloud environments while optimizing QoS metrics such as deadline and cost and dealing with low resource utilization issues pertaining to performance. A Green Cloud Scheduling Model (GCSM) has been proposed to proffer energy-conscious allocation and scheduling of tasks in cloud computing. An intelligent, energy-aware, extended GCSM model, GreenSched has been proposed that possesses machine learning capabilities to allocate and schedule deadline-and-budget constrained cloud tasks. GreenSched involves a Forward-only Counter Propagation Network based scheduler unit called as, CPN-based Green Cloud Scheduler (CGCS). Both GCSM and GreenSched models primarily focus on energy optimization while optimizing slightly variable set of QoS metrics. They differ in their mechanism and functional aspects.

In this final chapter, the concluding remarks on this research work have been given. It describes the advancement made towards the objectives of the research, in terms of developing novel energy-aware resource scheduling techniques to minimize the energy consumption and achieve energy efficiency. The discusses the variable metrics and concerns arising from higher energy consumption levels. The chapter details the outcome of each chapter and later highlights the contributions. Furthermore, it discusses the directions for future research.

6.1 Conclusions

The aim of this research work has been to design and develop Energy-aware resource scheduling technique for cloud computing which has been addressed by Green Cloud Scheduling Model (GCSM) and GreenSched model. As investigated, the efficient allocation and scheduling averts high energy consumption in the systems, improves the utilization of nodes and its resources. Consequently, the minimization in the energy consumption prevents performance bottlenecks and helps to gratify the users with acceptable service quality in terms of deadline and budget fulfillment. Hence, the proposed GCSM and GreenSched models are based on these aspects. The primary aspect on energy optimization is focused in both Chapter 3 and Chapter 4. The energy-related performance and deadline are the optimization metrics handled in GCSM model proposed in Chapter 3 while deadline, budget and performance metrics are handled in GreenSched model given in Chapter 4.

It has been demonstrated as part of the literature review conducted in Chapter 2 that the trend of setting up huge data centers and services have added to the energy demands. The initial attempts of curbing energy issues included infrastructural and location alterations, incurrence of green hardware equipments. Predominantly, in spite of these initiatives, the ICT energy consumption levels remained undesirably high and persistent and prompted the ICT experts to find other alternative measures to curb the energy crisis. Eventually, the experts observed that the energy consumed does not depend only on the efficiency of the physical resources but also on the effective resource management strategies. Thereby, the software-based measures such as resource management and scheduling schemes gained prominence for realizing energy efficiency. A large number of resource scheduling techniques based on variable metrics and mechanism had been proposed. A comprehensive and comparative survey of the existing techniques given in the chapter projects their limitations and the need to develop an energy-aware resource scheduling technique without hindering performance and delivering high QoS in terms of deadline-and-budget optimizations.

Chapter 3 has dealt with the prime objective of energy management and proffers energy-aware allocation and scheduling technique for cloud computing. It proposed a Green Cloud Scheduling Model (GCSM) that with the support of its Green Cloud Scheduler unit facilitates the provisioning and scheduling of the cloud tasks to the nodes that are energy-conscious. The significance, entities, architectural details, mathematical explication and flowchart of GCSM have been discussed in detail. The Green Cloud Scheduling technique implemented by the GCSM's scheduler unit has been discussed with its operational details and algorithm description.

The GCSM model tends to manage the cloud resources energy efficiently and enhances the utilization of nodes and their resources. To minimize the energy consumption and prevent performance degradation, GCSM identifies the best task-node pair and also consolidates the task entering the system into CPU-intensive or memory-intensive depending upon their resource requirements. The best node is the one that consumes minimal energy out of all other nodes in the system and thus becomes the prime contender for allocating the tasks to it. In addition to energy constraint, GCSM handles deadline fulfillment metric and thus the nodes identified to be min-

imal energy consuming nodes out of all other nodes are adjudged for their deadline fulfillment capability. The execution time for the all the less energy consuming nodes are computed and there after compared with the task deadlines. The nodes having lesser execution time then the deadline specified for the task is the winner node. The task is allocated and scheduled for execution on the winner node. The verification and validation of proposed GCSM has been done in prototype cloud environment. The results included in this chapter, demonstrate the effectiveness of GCSM model in minimizing the energy consumption, incurring higher performance and energy savings. The reduction in the idle nodes enhances the overall system performance and thus lowers the energy demands as well. GCSM also fulfills the deadline constraints for the tasks imposed by the users. However, the energy-related cost factor gains highlight and thus requires to be handled. The task allocation and scheduling carried out energy efficiently can be can be accomplished while maintaining minimal budget.

The minimization of energy through energy-conscious task allocation and scheduling on the cloud nodes and resources requires prior prediction models for the identification of nodes that comply with the minimum energy restrictions. Thus, machine learning techniques can be implemented to converge the energy consumption issue involving initial identification and prediction of nodes that consume lesser energy as compared to other nodes. A GreenSched model, an extended model based on GCSM has been proposed. GreenSched applies an intelligent machine learning technique for proactive allocation and scheduling of deadline-and-budget constrained cloud tasks to predicted energy-aware nodes. The machine learning prediction mechanism used in GreenSched is based on Forward-only Counter Propagation Network (FCPN). GreenSched's scheduler unit applies FCPN and is called as Counter-propagation based Green Cloud Scheduler (CGCS). The intelligent CGCS unit runs an Forward-only CPN Cloud Scheduling (FCS) technique that adjudges the energy consciousness of the nodes and verifies the deadline-and-budget fulfillment capability of the nodes. GreenSched model has been tested in CloudSim simulator. The experimental results demonstrate that GreenSched attains higher energy efficiency and performance.

The proposed GCSM model has been empirically evaluated by implementing it in a real-world banking industry. It has been implemented at Corporation bank, Bakkarwala branch, New Delhi. Chapter 5 provides the operational details and shows the experimental results obtained for validating Green Cloud Scheduling (GCS) of GCSM. The impact of GCS technique of GCSM has been evaluated at the bank branch and the results obtained have been discussed from variable perspectives such as effect on energy consumption, performance levels, utilization levels etc. The precise contributions of this thesis are listed below:

- A comprehensive, detailed and comparative literature survey of the existing energy efficiency techniques in the cloud computing including taxonomies for refinement and classification of the techniques.

- An energy-conscious provisioning and scheduling tasks on nodes model for optimally allocating tasks as per the energy and deadline-constraints while controlling and minimizing overall energy consumption and enhancing performance and QoS in the ICT data centers.

- An Intelligent, machine learning based model, termed as, GreenSched that performs energy aware allocation and scheduling of deadline-and-budget constrained cloud tasks using its scheduler unit that implements Forward-only Counter propagation network for energy-aware node predictions.
- GreenSched offers energy efficiency and higher node and its resource utilizations in heterogeneous data center promoting heterogeneity which is currently viable and demanding in the intensely dominating era of distributed computing systems.
- The proposed models, GCSM and GreenSched benefit both the users and service providers in terms of minimization in energy consumption, prevention of performance degradation, improvement in utilization levels and satisfaction of deadlines and budget restrictions.
- The verification and validation of the effectiveness of proposed GCSM and GreenSched models in real-time and simulated environments.
- A case study at Corporation bank, Bakkarwala branch, New Delhi to evaluate and analyze the performance and efficiency in terms of energy of the proposed GCSM model.
- Both the models minimize energy consumption, thereby indirectly lowering carbon emissions and energy-related expenditures of the data centers, achieving the goal of green computing.

6.2 Future Research Directions

The proposed Green Cloud Scheduling model and GreenSched model can be enhanced for future research. Some of the research directions include:

- GCSM and GreenSched models can be examined and extended for possible improvements in other QoS parameters like fault tolerance and reliability. The synchronization of proposed models for SLA-management in terms of missed deadline situations and associated energy expenditures in precise manner. GCSM can be deigned to offer cost based optimization as well to overcome the ever growing energy-related expenditures.
- GCSM and GreenSched focus on the energy consumption done by processor and memory units, which actually are prime accountable units causing high energy demands. However, the consideration for the energy consumed by the other computing system resources such as, networking components and input-output devices is an important future aspect for inclusion.
- The concerns pertaining to security, privacy and portability form an important part of the distributed computing environment, thus both the models can be augmented to include these attributes too.

- GCSM has been successfully evaluated and tested at Corporation Bank, Bakkarwala branch, New Delhi. However, the evaluation of GreenSched model at the branch is under-consideration and is an important future task.

- The reduction in the energy consumption lowers the carbon emissions and instigates green computing but the direct measurements for analyzing extent of reductions has not been done in the current work. Thus, direct readings can yield more accurate measurements of cloud energy savings and thus can be done in the future.

- The intensification of extensively distributed services and applications has yielded huge data storage and handling concerns, thus evolving Big Data. It is imperative to explore and adjudge the possible energy-related concerns that can arise with Big Data. The integration of Big Data and cloud computing is the current underway trend. Thus, energy efficiency through cloud computing can be collaborated with Big Data to avert energy issues. Thus, more energy-efficient algorithms can be designed.

Bibliography

[1] T. Ercan, "Effective use of cloud computing in educational institutions," *Procedia-Social and Behavioral Sciences*, vol. 2, no. 2, pp. 938–942, 2010.

[2] L. Mei, W. K. Chan, and T. Tse, "A tale of clouds: Paradigm comparisons and some thoughts on research issues," in *Asia-Pacific Services Computing Conference, 2008. APSCC'08. IEEE*, pp. 464–469, Ieee, 2008.

[3] J. Baliga, R. W. Ayre, K. Hinton, and R. S. Tucker, "Green cloud computing: Balancing energy in processing, storage, and transport," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 149–167, 2011.

[4] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[5] R. Buyya, J. Broberg, and A. M. Goscinski, *Cloud computing: Principles and paradigms*, vol. 87. John Wiley & Sons, 2010.

[6] I. Sriram and A. Khajeh-Hosseini, "Research agenda in cloud technologies," *arXiv preprint arXiv:1001.3259*, 2010.

[7] P. Mell, T. Grance, *et al.*, "The nist definition of cloud computing," 2011.

[8] G. Pallis, "Cloud computing: the new frontier of internet computing," *IEEE internet computing*, vol. 14, no. 5, pp. 70–73, 2010.

[9] A. Singh and M. Hemalatha, "Cloud computing for academic environment," 2012.

[10] R. Buyya, C. S. Yeo, and S. Venugopal, "Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities," in *High*

Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on, pp. 5–13, Ieee, 2008.

[11] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, *et al.*, "Above the clouds: A berkeley view of cloud computing," tech. rep., 2009.

[12] "Cloud computing." Available on: www.wikipedia.com, Last accessed=October, 2017.

[13] S. S. Manvi and G. K. Shyam, "Resource management for infrastructure as a service (iaas) in cloud computing: A survey," *Journal of Network and Computer Applications*, vol. 41, pp. 424–440, 2014.

[14] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of internet services and applications*, vol. 1, no. 1, pp. 7–18, 2010.

[15] T. Fei, *Management of Data and Scheduling of Tasks on Architecture Distributees*. PhD thesis, E cole Centrale: A University Institution, Paris, 2011.

[16] "About google." Available on: www.google.com/, Last accessed=October, 2017.

[17] "About microsoft." Available on: www.microsoft.com/, Last accessed=August, 2017.

[18] "About amazon ec2." Available on: http://aws.amazon.com/ec2/, Last accessed=June, 2015.

[19] "About kubernetes." Available on: https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/, Last accessed=November, 2017.

[20] L. Wang, G. Von Laszewski, A. Younge, X. He, M. Kunze, J. Tao, and C. Fu, "Cloud computing: a perspective study," *New Generation Computing*, vol. 28, no. 2, pp. 137–146, 2010.

[21] D. Hilley, "Cloud computing: A taxonomy of platform and infrastructure-level offerings," tech. rep., Georgia Institute of Technology, 2009.

[22] B. P. Rimal, E. Choi, and I. Lumb, "A taxonomy and survey of cloud computing systems," in *INC, IMS and IDC, 2009. NCM'09. Fifth International Joint Conference on*, pp. 44–51, Ieee, 2009.

[23] F. Stroud, "Metal-as-a-service(maas)."

[24] 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.

[25] R. T. Fernandes, "Green-cloud: Economics-inspired scheduling, energy and resource management in cloud infrastructures," https://www.inesc-id.pt/publications/10630/pdf/.

[26] 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.

[27] "Amazon web services." "Available on: www.aws.amazon.com/, Last accessed=August, 2016".

[28] "Google app engine." "Available on: www.cloud.google.com/appengine/, Last accessed=June, 2016".

[29] "Microsoft azure." "Available on: www.azure.microsoft.com/en-in/, Last accessed=July, 2015".

[30] "Salesforce.com." "Available on: www.salesforce.com/in/, Last accessed=November, 2017".

[31] "Heroku." "Available on: www.heroku.com/, Last accessed=November, 2017".

[32] "Rightscale." "Available on: www.rightscale.com/, Last accessed=November, 2015".

[33] 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.

[34] J. Thaman and M. Singh, "Green cloud environment by using robust planning algorithm," *Egyptian Informatics Journal*, vol. 18, no. 3, pp. 205–214, 2017.

[35] A. K. Coskun and T. S. Rosing, "Improving energy efficiency and reliability through workload scheduling in high-performance multicore processors,"

[36] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," *ACM SIGCOMM computer communication review*, vol. 39, no. 1, pp. 68–73, 2008.

[37] H. Yuan, C.-C. J. Kuo, and I. Ahmad, "Energy efficiency in data centers and cloud-based multimedia services: An overview and future directions," in *Green Computing Conference, 2010 International*, pp. 375–382, IEEE, 2010.

[38] Y. Peng, D.-K. Kang, F. Al-Hazemi, and C.-H. Youn, "Energy and qos aware resource allocation for heterogeneous sustainable cloud datacenters," *Optical Switching and Networking*, vol. 23, pp. 225–240, 2017.

[39] M. Żotkiewicz, M. Guzek, D. Kliazovich, and P. Bouvry, "Minimum dependencies energy-efficient scheduling in data centers," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 12, pp. 3561–3574, 2016.

[40] J. Koomey, "Growth in data center electricity use 2005 to 2010," tech. rep., A Report by Analytical Press, Completed at the Request of The New York Times, 2011.

[41] M. Blazek, H. Chong, W. Loh, and J. G. Koomey, "Data centers revisited: Assessment of the energy impact of retrofits and technology trends in a high density computing facility," *Journal of Infrastructure Systems*, vol. 10, no. 3, pp. 98–104, 2004. http://dx.doi.org/10.1061/(ASCE)1076-0342.

[42] K. J. Christensen, C. Gunaratne, B. Nordman, and A. D. George, "The next frontier for communications networks: power management," *Computer Communications*, vol. 27, no. 18, pp. 1758–1770, 2004.

[43] A. Ogasawara, "Energy issues confronting the information and communications sector-need to reduce the power consumed by the communications infrastructure," tech. rep., NISTEP Science & Technology Foresight Center, 2006.

[44] P. Huber and M. P. Mills, "Dig more coalthe pcs are coming," pp. 70–72, 1999. http://www.forbes.com/forbes/1999/0531/6311070a.html.

[45] K. Kawamoto, J. G. Koomey, B. Nordman, R. E. Brown, M. A. Piette, M. Ting, and A. K. Meier, "Electricity used by office equipment and network equipment in the us: Detailed report and appendices," 2000.

[46] J. Mitchell-Jackson, J. G. Koomey, M. Blazek, and B. Nordman, "National and regional implications of internet data center growth in the us," *Resources, conservation and recycling*, vol. 36, no. 3, pp. 175–185, 2002.

[47] M. Vasudevan, Y.-C. Tian, M. Tang, and E. Kozan, "Profile-based application assignment for greener and more energy-efficient data centers," *Future generation computer systems*, vol. 67, pp. 94–108, 2017.

[48] S. NEW, "Green computing: Higher energy efficiency from silicon to the cloud,"

[49] M. P. Mills, "The cloud begins with coal," 2013.

[50] 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.

[51] Envantage, "World carbon dioxide emissions by country," 2013.

[52] J. Malmodin, Å. Moberg, D. Lundén, G. Finnveden, and N. Lövehagen, "Greenhouse gas emissions and operational electricity use in the ict and entertainment & media sectors," *Journal of Industrial Ecology*, vol. 14, no. 5, pp. 770–790, 2010.

[53] A. Ashraf, B. Byholm, and I. Porres, "Distributed virtual machine consolidation: A systematic mapping study," tech. rep., Turku Centre for Computer Science Technical Report No 1171, 2016.

[54] "Trends in global emissions." Available on: https://www.epa.gov/ghgemissions/global-greenhouse-gas-emissions-data.

[55] "Ict helping tackle climate change could help cut global emissions 20% by 2030," 2016.

[56] "Carbon footprints, quantify and report your carbon footprint." Available on: http://www.envantage.co.uk/carbon-management/carbon-footprinting.htm.

[57] "Global carbon atlas- emissions." Available on: http://www.globalcarbonatlas.org/?q=en/emissions.

[58] "Carbon descent: Delivering a sustainable future." Carbon descent: Delivering a sustainable future.

[59] "Global carbon atlas- CO_2 emissions." Available on: http://www.globalcarbonatlas.org/en/CO_2-emissions.

[60] "The largest emitters of carbon dioxide worldwide by country 2015," tech. rep., International Energy Outlook 2016.

[61] "Renewables can reduce CO_2 emissions by 70% by 2050," tech. rep., Climate Action Group Report, March, 2017. http://www.climateactionprogramme.org/news /renewables_can_reduce_co2_emission_by_70_by_2050.

[62] "Bp statistical review of world energy," tech. rep., BP, June 2016. http://www.bp.com/content/dam/bp/pdf/energy-economics/statistical-review-2016/bp-statistical-review-of-world-energy-2016-full-report.pdf.

[63] A. Qouneh, C. Li, and T. Li, "A quantitative analysis of cooling power in container-based data centers," in *Workload Characterization (IISWC), 2011 IEEE International Symposium on*, pp. 61–71, IEEE, 2011.

[64] M. K. Patterson, D. Costello, P. Grimm, and M. Loeffler, "Data center tco; a comparison of high-density and low-density spaces,"

[65] T. Mastelic, A. Oleksiak, H. Claussen, I. Brandic, J.-M. Pierson, and A. V. Vasilakos, "Cloud computing: Survey on energy efficiency," *Acm computing surveys (csur)*, vol. 47, no. 2, p. 33, 2015.

[66] "Gartner says data center power, cooling and space issues are set to increase rapidly as a result of new high-density infrastructure deployments," tech. rep., Tudor, Laurence and Petty, Christy, 2010. http://www.gartner.com/newsroom/id/1368614.

[67] J. Loper and S. Parr, "Energy efficiency in data centers: A new policy frontier," *Environmental Quality Management*, vol. 16, no. 4, pp. 83–97, 2007.

[68] F. Ahmad and T. Vijaykumar, "Joint optimization of idle and cooling power in data centers while maintaining response time," in *ACM Sigplan Notices*, vol. 45, pp. 243–256, ACM, 2010.

[69] N. B. Kamran, "Qos-aware vm placement and migration for hybrid cloud infrastructure," *Journal of Supercomputing*, pp. 86–96, 2017.

[70] S. Albers, "Energy-efficient algorithms," *Communications of the ACM*, vol. 53, no. 5, pp. 86–96, 2010.

[71] A. Weiss, "Computing in the clouds," *networker*, vol. 11, no. 4, pp. 16–25, 2007.

[72] "The outlook for energy: A view to 2040," tech. rep., Exxon Mobile, 2013. http://www.exxonmobil.com/Corporate/Files/news_pub_2013eo_us.pdf.

[73] Envantage, "New report reveals warehouses overspend on energy by 190m," tech. rep., Data by World Resources Institute, 2013. http://www.enmat.com/newreport-reveals-warehouses-overspend-on-energy-by-190m.

[74] A. Plepys, "The grey side of ict," *Environmental Impact Assessment Review*, vol. 22, no. 5, pp. 509–523, 2002.

[75] A. Berl, E. Gelenbe, M. Di Girolamo, G. Giuliani, H. De Meer, M. Q. Dang, and K. Pentikousis, "Energy-efficient cloud computing," *The computer journal*, vol. 53, no. 7, pp. 1045–1051, 2010.

[76] 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*, vol. 82, pp. 47–111, Elsevier, 2011.

[77] E. Hille, "Top 10 apps for cloud, private cloud implementation issues," tech. rep., Cloudcommons 2012, 2010. http://cloudcomputing.syscon.com/node/1653265.

[78] T. Dillon, C. Wu, and E. Chang, "Cloud computing: issues and challenges," in *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pp. 27–33, Ieee, 2010.

[79] S. Marston, Z. Li, S. Bandyopadhyay, J. Zhang, and A. Ghalsasi, "Cloud computingthe business perspective," *Decision support systems*, vol. 51, no. 1, pp. 176–189, 2011.

[80] F. Owusu and C. Pattinson, "The current state of understanding of the energy efficiency of cloud computing," in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pp. 1948–1953, IEEE, 2012.

[81] "Intel hopes poem will encourage office staff to save energy," tech. rep., Sayer, Peter, 2011. Available on: http://www.pcworld.com/article/234916/article.html.

[82] I. Rodero, J. Jaramillo, A. Quiroz, M. Parashar, F. Guim, and S. Poole, "Energy-efficient application-aware online provisioning for virtualized clouds and data centers," in *Green Computing Conference, 2010 International*, pp. 31–45, IEEE, 2010.

[83] J. Bruschi, P. Rumsey, R. Anliker, L. Chu, and S. Gregson, "Best practices guide for energy efficient data center design," tech. rep., Federal Energy Management Program Report, U.S Department of Energy, March 2011. Available on: www1.eere.energy.gov/femp/pdfs/eedatacenterbestpractices.pdf.

[84] A. J. Younge, G. Von Laszewski, L. Wang, S. Lopez-Alarcon, and W. Carithers, "Efficient resource management for cloud computing environments," in *Green Computing Conference, 2010 International*, pp. 357–364, IEEE, 2010.

[85] "Virtualization." Available on: https://en.wikipedia.org/wiki/Virtualization, Last accessed=June, 2013.

[86] S. Agarwal, S. Yadav, and A. K. Yadav, "An efficient architecture and algorithm for resource provisioning in fog computing," *International Journal of Information Engineering and Electronic Business*, vol. 8, no. 1, p. 48, 2016.

[87] "Cloud computing as an energy saving tool," tech. rep., Miller, Rich, 2010. http://www.datacenterknowledge.com/archives/2010/05/17/cloud-computing-as-an-energy-saving-tool/.

[88] J. Sahoo, S. Mohapatra, and R. Lath, "Virtualization: A survey on concepts, taxonomy and associated security issues," in *Computer and Network Technology (ICCNT), 2010 Second International Conference on*, pp. 222–226, IEEE, 2010.

[89] T. Brunzel and D. Di Giacomo, "Cloud computing evaluation: how it differs to traditional it outsourcing," 2010.

[90] A. Beloglazov and R. Buyya, "Energy efficient allocation of virtual machines in cloud data centers," in *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*, pp. 577–578, IEEE, 2010.

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

[92] N. J. Kansal and I. Chana, "Existing load balancing techniques in cloud computing: A systematic review," *Journal of Information Systems and Communication*, vol. 3, no. 1, p. 87, 2012.

[93] R. Wang, W. Le, and X. Zhang, "Design and implementation of an efficient load-balancing method for virtual machine cluster based on cloud service," in *4th International Conference on Wireless, Mobile and Multimedia Networks (ICWMMN 2011)*, pp. 321–324, IEEE, 2011.

[94] B. Varghese and R. Buyya, "Next generation cloud computing: New trends and research directions," *Future Generation Computer Systems*, vol. 79, pp. 849–861, 2018.

[95] K. Foundation, "Scalable, energy-efficient data centres and clouds," tech. rep., 2012. Available on: http://iee.ucsb.edu/files/Institute%20for%20Energy%20Efficiency%20Data%20Center%20Re

[96] J. E. Pecero, H. J. F. Huacuja, P. Bouvry, A. A. S. Pineda, M. C. L. Locés, and J. J. G. Barbosa, "On the energy optimization for precedence constrained applications using local search algorithms," in *High Performance Computing and Simulation (HPCS), 2012 International Conference on*, pp. 133–139, IEEE, 2012.

[97] L. Lefèvre and A.-C. Orgerie, "Designing and evaluating an energy efficient cloud," *The Journal of Supercomputing*, vol. 51, no. 3, pp. 352–373, 2010.

[98] A.-C. Orgerie, L. Lefevre, and J.-P. Gelas, "Demystifying energy consumption in grids and clouds," in *Green Computing Conference, 2010 International*, pp. 335–342, IEEE, 2010.

[99] M. Ahmed, A. Chowdhury, M. Ahmed, M. M. H. Rafee, *et al.*, "An advanced survey on cloud computing and state-of-the-art research issues," *International Journal of Computer Science Issues*, vol. 0, no. 1, 2012.

[100] M. Marzolla, O. Babaoglu, and F. Panzieri, "Server consolidation in clouds through gossiping," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a*, pp. 1–6, IEEE, 2011.

[101] Y. C. Lee and A. Y. Zomaya, "Energy efficient utilization of resources in cloud computing systems," *The Journal of Supercomputing*, vol. 60, no. 2, pp. 268–280, 2012.

[102] S. Srikantaiah, A. Kansal, and F. Zhao, "Energy aware consolidation for cloud computing," in *Proceedings of USENIX workshop on power aware computing and systems in conjunction with OSDI*, pp. 1–5, 2008.

[103] A. Verma, G. Dasgupta, T. K. Nayak, P. De, and R. Kothari, "Server workload analysis for power minimization using consolidation," in *Proceedings of the 2009 conference on USENIX Annual technical conference*, pp. 28–28, USENIX Association, 2009.

[104] B. Schauer, "Multicore processors–a necessity," 2008.

[105] S. Maiti and S. Pasricha, "Delca: Dvfs efficient low cost multicore architecture," in *Proceedings of the on Great Lakes Symposium on VLSI 2017*, GLSVLSI '17, pp. 107–112, ACM, 2017. 10.1145/3060403.3060422.

[106] L. Chai, Q. Gao, and D. K. Panda, "Understanding the impact of multi-core architecture in cluster computing: A case study with intel dual-core system," in *Cluster Computing and the Grid, 2007. CCGRID 2007. Seventh IEEE International Symposium on*, pp. 471–478, IEEE, 2007. http://dx.doi.org/10.1109/CCGRID.2007.119.

[107] G. Blake, R. G. Dreslinski, and T. Mudge, "A survey of multicore processors," *IEEE Signal Processing Magazine*, vol. 26, no. 6, 2009.

[108] J. Fruehe, "Multicore processor technology," *Reprinted from Dell Power Solutions www. dell. com/powersolutions (Obtained from the Internet on Mar. 23, 2012)*, pp. 67–72, 2005.

[109] S.-g. Kim, H. Eom, and H. Y. Yeom, "Virtual machine scheduling for multicores considering effects of shared on-chip last level cache interference," in *Green Computing Conference (IGCC), 2012 International*, pp. 1–6, IEEE, 2012.

[110] 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.

[111] C. Ghribi, *Energy efficient resource allocation in cloud computing environments*. PhD thesis, Institut National des Télécommunications, 2014.

[112] J. Koomey, C. Calwell, S. Laitner, J. Thornton, R. E. Brown, J. Eto, C. Webber, and C. Cullicott, "Sorry, wrong number: The use and misuse of numerical facts in analysis and media reporting of energy issues," pp. 119–158, 2002.

[113] K. Roth, F. Goldstein, and J. Kleinman, "Energy consumption by office and telecommunications equipment in commercial buildings volume i: energy consumption baseline," 2002.

[114] T. C. Group, "Smart 2020: Enabling the low power economy in the information age," tech. rep., Report of the Institute for Energy Efficiency.

[115] J. Ni and X. Bai, "A review of air conditioning energy performance in data centers," *Renewable and sustainable energy reviews*, vol. 67, pp. 625–640, 2017.

[116] S. Zeadally, S. U. Khan, and N. Chilamkurti, "Energy-efficient networking: past, present, and future," *The Journal of Supercomputing*, vol. 62, no. 3, pp. 1093–1118, 2012.

[117] S. M. Parikh, N. M. Patel, and H. B. Prajapati, "Resource management in cloud computing: Classification and taxonomy," *arXiv preprint arXiv:1703.00374*, 2017.

[118] "Delivering a sustainable future 2013," tech. rep., Carbon descent. Available on: http://www.carbondescent.org.uk/, Last accessed= June, 2015.

[119] J. D. Moore, J. S. Chase, P. Ranganathan, and R. K. Sharma, "Making scheduling" cool": Temperature-aware workload placement in data centers," pp. 61–75, 2005.

[120] "Enclosure strategies for efficiencydata centre efficiency practice starts with your racks," tech. rep., 42U Data, 2008. Available on: http://www.slideshare.net/42u/data-center-server-rack-strategiesn.

[121] K. K. Nguyen, M. Cheriet, M. Lemay, B. S. Arnaud, V. Reijs, A. Mackarel, P. Minoves, A. Pastrama, and W. Van Heddeghem, "Renewable energy provisioning for ict services in a future internet," in *The Future Internet Assembly*, pp. 419–429, Springer, 2011.

[122] J. Hamilton, "Cooperative expendable micro-slice servers (cems): low cost, low power servers for internet-scale services," Citeseer.

[123] D. G. Andersen, J. Franklin, M. Kaminsky, A. Phanishayee, L. Tan, and V. Vasudevan, "Fawn: A fast array of wimpy nodes," *Communications of the ACM*, vol. 54, no. 7, 2011. http://dx.doi.org/10.1145/1965724.1965747.

[124] "Fawn project." Available on: www.cs.cmu.edu/fawnproj/, Last accessed= June, 2017.

[125] D. Schall and T. Härder, "Towards an energy-proportional storage system using a cluster of wimpy nodes.," pp. 311–325, Citeseer, 2013.

[126] V. Vasudevan, J. Franklin, D. G. Andersen, A. Phanishayee, L. Tan, M. Kaminsky, and I. Moraru, "Fawndamentally power-efficient clusters.," pp. 22–22, 2009. https://www.cs.cmu.edu/dga/papers/fawn-hotos2009.pdf.

[127] A. S. Szalay, G. C. Bell, H. H. Huang, A. Terzis, and A. White, "Low-power amdahl-balanced blades for data intensive computing," *ACM SIGOPS Operating Systems Review*, vol. 44, no. 1, pp. 71–75, 2010. http://dx.doi.org/10.1145/1740390.1740407.

[128] K. Lim, P. Ranganathan, J. Chang, C. Patel, T. Mudge, and S. Reinhardt, "Understanding and designing new server architectures for emerging warehouse-computing environments," in *ACM SIGARCH Computer Architecture News*, vol. 36, pp. 315–326, IEEE Computer Society, 2008. http://dx.doi.org/10.1145/1394608.1382148.

[129] J. A. Laitner *et al.*, "Information technology and us energy consumption: energy hog, productivity tool, or both?," *Journal of Industrial Ecology*, vol. 6, no. 2, pp. 13–24, 2002.

[130] V. Tiwari, P. Ashar, and S. Malik, "Technology mapping for low power," in *Design Automation, 1993. 30th Conference on*, pp. 74–79, IEEE, 1993. http://dx.doi.org/10.1109/DAC.1993.203922.

[131] "Resource throttling." Available on: www.wikipedia.org/wiki/ Throttling_process_ (computing), Last accessed= May, 2014.

[132] M. Stillwell, D. Schanzenbach, F. Vivien, and H. Casanova, "Resource allocation using virtual clusters," in *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, pp. 260–267, IEEE Computer Society, 2009. http://dx.doi.org/10.1109/CCGRID.2009.23.

[133] G. L. Valentini, W. Lassonde, S. U. Khan, N. Min-Allah, S. A. Madani, J. Li, L. Zhang, L. Wang, N. Ghani, J. Kolodziej, L. Hongxiang, A. Y. Zomaya, C.-Z. Xu, P. Balaji, A. Vishnu, F. Pinel, J. E. Pecero, D. Kliazovich, and P. Bouvry, "An overview of energy efficiency techniques in cluster computing systems," *Cluster Computing*, vol. 16, no. 1, pp. 3–15, 2013.

[134] S. H. H. Madni, M. S. A. Latiff, Y. Coulibaly, *et al.*, "Recent advancements in resource allocation techniques for cloud computing environment: a systematic review," *Cluster Computing*, vol. 20, no. 3, pp. 2489–2533, 2017.

[135] S.-Y. Jing, S. Ali, K. She, and Y. Zhong, "State-of-the-art research study for green cloud computing," *The Journal of Supercomputing*, vol. 65, no. 1, pp. 445–468, 2013.

[136] A.-C. Orgerie, M. D. d. Assuncao, and L. Lefevre, "A survey on techniques for improving the energy efficiency of large-scale distributed systems," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, p. 47, 2014.

[137] S. Zhuravlev, J. C. Saez, S. Blagodurov, A. Fedorova, and M. Prieto, "Survey of scheduling techniques for addressing shared resources in multicore processors," *ACM Computing Surveys (CSUR)*, vol. 45, no. 1, p. 4, 2012. http://dx.doi.org/10.1145/2379776.2379780.

[138] "Ibm about." Available on: www-935.ibm.com/services/us/, Last accessed= June, 2017.

[139] J. Mei, K. Li, and K. Li, "Customer-satisfaction-aware optimal multiserver configuration for profit maximization in cloud computing," *IEEE Transactions on Sustainable Computing*, vol. 2, no. 1, pp. 17–29, 2017.

[140] M. A Vouk, "Cloud computing–issues, research and implementations," *Journal of computing and information technology*, vol. 16, no. 4, pp. 235–246, 2008. http://dx.doi.org/ 10.1109/ITI.2008.4588381.

[141] B. Li, J. Li, J. Huai, T. Wo, Q. Li, and L. Zhong, "Enacloud: An energy-saving application live placement approach for cloud computing environments," in *Cloud Computing, 2009. CLOUD'09. IEEE International Conference on*, pp. 17–24, IEEE, 2009. http://dx.doi.org/10.1109/CLOUD.2009.72.

[142] Y. Ho, P. Liu, and J.-J. Wu, "Server consolidation algorithms with bounded migration cost and performance guarantees in cloud

computing," in *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*, pp. 154–161, IEEE, 2011. http://doi.ieeecomputersociety.org/10.1109/UCC.2011.30.

[143] Y. Zhang and N. Ansari, "On architecture design, congestion notification, tcp incast and power consumption in data centers," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 1, pp. 39–64. http://dx.doi.org/10.1109/SURV.2011.122211.00017.

[144] J. Xu, J. Tang, K. Kwiat, W. Zhang, and G. Xue, "Survivable virtual infrastructure mapping in virtualized data centers," in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pp. 196–203, IEEE, 2012. http://doi.ieeecomputersociety.org/10.1109/CLOUD.2012.100.

[145] 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.

[146] G. Du, H. He, and Q. Meng, "Energy-efficient scheduling for tasks with deadline in virtualized environments," *Mathematical Problems in Engineering*, vol. 2014, 2014. http://dx.doi.org/10.1155/2014/496843.

[147] A. Beloglazov and R. Buyya, "Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers.," in *Middleware for Grids, Clouds and e-Science, 8th International Workshop on*, 2010. http://dx.doi.org/10.1145/1890799.1890803.

[148] 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 and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397–1420, 2012. http://dx.doi.org/10.1002/cpe.1867.

[149] M. Amoretti, F. Zanichelli, and G. Conte, "Efficient autonomic cloud computing using online discrete event simulation," *Journal of Parallel and Distributed Computing*, vol. 73, no. 6, pp. 767–776, 2013. http://dx.doi.org/10.1016/j.jpdc.2013.02.008.

[150] J.-S. Liao, C.-C. Chang, Y.-L. Hsu, X.-W. Zhang, K.-C. Lai, and C.-H. Hsu, "Energy-efficient resource provisioning with sla consideration on cloud computing," in *Parallel Processing Workshops (ICPPW),*

2012 41st International Conference on, pp. 206–211, IEEE, 2012. http://dx.doi.org/10.1109/ICPPW.2012.31.

[151] N. Sharma and R. M. Guddeti, "Multi-objective energy efficient virtual machines allocation at the cloud data center," *IEEE Transactions on Services Computing*, 2016. 10.1109/TSC.2016.2596289.

[152] J. Li, J. Peng, Z. Lei, and W. Zhang, "An energy-efficient scheduling approach based on private clouds," *Journal of Information &computational Science*, vol. 8, no. 4, pp. 716–724, 2011.

[153] S. Deore, A. N. Patil, and R. Bhargava, "Energy-efficient scheduling scheme for virtual machines in cloud computing," *International Journal of Computer Applications*, vol. 56, no. 10, 2012. http://dx.doi.org/10.5120/8926-299.

[154] D. M. Quan, A. Somov, and C. Dupont, "Energy usage and carbon emission optimization mechanism for federated data centers," in *International Workshop on Energy Efficient Data Centers*, pp. 129–140, 2012.

[155] D. M. Quan, F. Mezza, D. Sannenli, and R. Giafreda, "T-alloc: A practical energy efficient resource allocation algorithm for traditional data centers," *Future Generation Computer Systems*, vol. 28, no. 5, pp. 791–800, 2012. http://dx.doi.org/10.1016/j.future.2011.04.020.

[156] Y. Gao, Y. Wang, S. K. Gupta, and M. Pedram, "An energy and deadline aware resource provisioning, scheduling and optimization framework for cloud systems," in *Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2013 International Conference on*, pp. 1–10, IEEE, 2013.

[157] Y. Kessaci, N. Melab, and E.-G. Talbi, "An energy-aware multi-start local search heuristic for scheduling vms on the opennebula cloud distribution," in *High Performance Computing and Simulation (HPCS), International Conference on*, pp. 112–118, IEEE, 2012. http://dx.doi.org/10.1109/HPCSim.2012.6266899.

[158] 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.

[159] H. Duan, C. Chen, G. Min, and Y. Wu, "Energy-aware scheduling of virtual machines in heterogeneous cloud computing systems," *Future Generation Computer Systems*, vol. 74, pp. 142–150, 2017.

[160] D. M. Quan, R. Basmadjian, H. De Meer, R. Lent, T. Mahmoodi, D. Sannelli, F. Mezza, L. Telesca, and C. Dupont, "Energy efficient resource allocation strategy for cloud data centres," in *Computer and information sciences II*, pp. 133–141, 2011.

[161] T. Knauth and C. Fetzer, "Energy-aware scheduling for infrastructure clouds," in *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*, pp. 58–65, IEEE, 2012. http://dx.doi.org/10.1109/CloudCom.2012.6427569.

[162] A. Pahlavan, M. Momtazpour, and M. Goudarzi, "Variation-aware server placement and task assignment for data center power minimization," in *Parallel and Distributed Processing with Applications (ISPA), 2012 IEEE 10th International Symposium on*, pp. 158–165, IEEE, 2012. http://dx.doi.org/10.1109/ISPA.2012.29.

[163] A. Paya and D. C. Marinescu, "Energy-aware load balancing and application scaling for the cloud ecosystem," *IEEE Transactions on Cloud Computing*, vol. 5, no. 1, pp. 15–27, 2017.

[164] 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, 2016.

[165] L. Shi, Z. Zhang, and T. Robertazzi, "Energy-aware scheduling of embarrassingly parallel jobs and resource allocation in cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 6, pp. 1607–1620, 2017.

[166] P. M. Wells, K. Chakraborty, and G. S. Sohi, "Dynamic heterogeneity and the need for multicore virtualization," *ACM SIGOPS Operating Systems Review*, vol. 43, no. 2, pp. 5–14, 2009. http://dx.doi.org/10.1145/1531793.1531797.

[167] G. Vallee, T. Naughton, C. Engelmann, H. Ong, and S. L. Scott, "System-level virtualization for high performance computing," in *Parallel, Distributed and Network-Based Processing, 2008. PDP 2008. 16th Euromicro Conference on*, pp. 636–643, IEEE, 2008. http://dx.doi.org/10.1109/PDP.2008.85.

[168] A. Merkel and F. Bellosa, "Balancing power consumption in multiprocessor systems," in *ACM SIGOPS Operating Systems Review*, vol. 40, pp. 403–414, ACM, 2006. http://dx.doi.org/10.1145/1217935.1217974.

[169] A. Merkel and F. Bellosa, "Memory-aware scheduling for energy efficiency on multicore processors," in *Power Aware Computing and Systems, 2008 Conference on*, 2008.

[170] A. Merkel, *Task activity vectors: a new metric for temperature-aware scheduling.* PhD thesis.

[171] M. Hussin, Y. C. Lee, and A. Y. Zomaya, "Priority-based scheduling for large-scale distribute systems with energy awareness," in *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*, pp. 503–509, IEEE, 2011. http://dx.doi.org/10.1109/DASC.2011.96.

[172] D. Kliazovich, P. Bouvry, and S. U. Khan, "Dens: data center energy-efficient network-aware scheduling," *Cluster computing*, vol. 16, no. 1, pp. 65–75, 2013.

[173] M. Kocaoglu, D. Malak, and O. B. Akan, "Fundamentals of green communications and computing: modeling and simulation," *Computer*, vol. 45, no. 9, pp. 40–46, 2012. http://dx.doi.org/10.1109/ MC.2012.2.

[174] M. A. Salehi, P. R. Krishna, K. S. Deepak, and R. Buyya, "Preemption-aware energy management in virtualized data centers," in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pp. 844–851, IEEE, 2012. http://dx.doi.org/10.1109/CLOUD.2012.147.

[175] 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.

[176] I. Takouna, W. Dawoud, and C. Meinel, "Energy efficient scheduling of hpc-jobs on virtualize clusters using host and vm dynamic configuration," *ACM SIGOPS Operating Systems Review*, vol. 46, no. 2, pp. 19–27, 2012. http://dx.doi.org/10.1145/2331576.233158.

[177] G. Von Laszewski, L. Wang, A. J. Younge, and X. He, "Power-aware scheduling of virtual machines in dvfs-enabled clusters," in *Cluster Computing and Workshops, 2009. CLUSTER'09. IEEE International Conference on*, pp. 1–10, IEEE, 2009. http://dx.doi.org/10.1109/CLUSTR.2009.5289182.

[178] C. De Alfonso, M. Caballer, F. Alvarruiz, and G. Moltó, "An economic and energy-aware analysis of the viability of outsourcing cluster computing to a cloud," *Future Generation Computer Systems*, vol. 29, no. 3, pp. 704–712, 2013. http://dx.doi.org/10.1016/j.future.2012.08.014.

[179] C. De Alfonso, M. Caballer, F. Alvarruiz, and V. Hernández, "An energy management system for cluster infrastructures," *Computers & Electrical Engineering*, vol. 39, no. 8, pp. 2579–2590, 2013.

[180] C. Xian, Y.-H. Lu, and Z. Li, "Energy-aware scheduling for real-time multiprocessor systems with uncertain task execution time," in *Design Automation Conference, 2007. DAC'07. 44th ACM/IEEE*, pp. 664–669, IEEE, 2007.

[181] K. H. Kim, A. Beloglazov, and R. Buyya, "Power-aware provisioning of virtual machines for real-time cloud services," *Concurrency and Computation: Practice and Experience*, vol. 23, no. 13, pp. 1491–1505, 2011.

[182] Q. Chen, "Towards energy-aware vm scheduling in iaas clouds through empirical studies," *Yüksek Lisans Tezi,, University of Amsterdam, Hollanda*, pp. 10–11, 2011.

[183] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, "Power and performance management of virtualized computing environments via lookahead control," *Cluster computing*, vol. 12, no. 1, pp. 1–15, 2009.

[184] F. Ahmad and T. Vijaykumar, "Joint optimization of idle and cooling power in data centers while maintaining response time," in *Proceedings of the fifteenth edition of ASPLOS on Architectural support for programming languages and operating systems*.

[185] 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. http://dx.doi.org/10.1109/TPDS.2008.111.

[186] I. Rodero, H. Viswanathan, E. K. Lee, M. Gamell, D. Pompili, and M. Parashar, "Energy-efficient thermal-aware autonomic management of virtualized hpc cloud infrastructure," *Journal of Grid Computing*, vol. 10, no. 3, pp. 447–473, 2012.

[187] E. Pakbaznia, M. Ghasemazar, and M. Pedram, "Temperature-aware dynamic resource provisioning in a power-optimized datacenter," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2010*, pp. 124–129, IEEE, 2010. http://dx.doi.org/10.1109/DATE.2010.5457223.

[188] M. R. V. Kumar and S. Raghunathan, "Heterogeneity and thermal aware adaptive heuristics for energy efficient consolidation of virtual machines in infrastructure clouds," *Journal of Computer and System Sciences*, vol. 82, no. 2, pp. 191–212, 2016.

[189] F. Dressler and O. B. Akan, "A survey on bio-inspired networking," *Computer Networks*, vol. 54, no. 6, pp. 881–900, 2010.

[190] M. Meisel, V. Pappas, and L. Zhang, "A taxonomy of biologically inspired research in computer networking," *Computer Networks*, vol. 54, no. 6, pp. 901–916, 2010. http://dx.doi.org/10.1016/j.comnet.2009.08.022.

[191] Y. Kessaci, M. Mezmaz, N. Melab, E.-G. Talbi, and D. Tuyttens, "Parallel evolutionary algorithms for energy aware scheduling," in *Intelligent Decision Systems in Large-Scale Distributed Environments*, pp. 75–100, 2011.

[192] X. Wang, Y. Wang, and H. Zhu, "Energy-efficient multi-job scheduling model for cloud computing and its genetic algorithm," *Mathematical Problems in Engineering*, vol. 2012, 2012.

[193] N. Quang-Hung, P. D. Nien, N. H. Nam, N. H. Tuong, and N. Thoai, "A genetic algorithm for power-aware virtual machine allocation in private cloud," in *Information and Communication Technology-EurAsia Conference*, pp. 183–191, 2013.

[194] F. F. Moghaddam, M. Cheriet, and K. K. Nguyen, "Low carbon virtual private clouds," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, pp. 259–266, IEEE, 2011. http://dx.doi.org/10.1109/CLOUD.2011.36.

[195] L. Chimakurthi *et al.*, "Power efficient resource allocation for clouds using ant colony framework," *arXiv preprint arXiv:1102.2608*, 2011.

[196] E. Feller, L. Rilling, and C. Morin, "Energy-aware ant colony based workload placement in clouds," in *Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing*, pp. 26–33, IEEE Computer Society, 2011. http://dx.doi.org/10.1109/Grid.2011.13.

[197] R. Babukarthik, R. Raju, and P. Dhavachelvan, "Energy-aware scheduling using hybrid algorithm for cloud computing," in *Computing Communication & Networking Technologies (ICCCNT), 2012 Third International Conference on*, pp. 1–6, IEEE, 2012. http://dx.doi.org/10.1109/ICCCNT.2012.6396014.

[198] Z. Wang, K. Shuang, L. Yang, and F. Yang, "Energy-aware and revenue-enhancing combinatorial scheduling in virtualized of cloud datacenter," *Journal of Convergence Information Technology*, vol. 7, 2012.

[199] N. Bergmann, Y. Y. Chung, X. Yang, Z. Chen, W. C. Yeh, X. He, and R. Jurdak, "Using swarm intelligence to optimize the energy consumption for distributed systems," *Modern Applied Science*, vol. 7, no. 6, pp. 59–66, 2013. http://dx.doi.org/10.5539/mas.v7n6p59.

[200] S. K. Garg, C. S. Yeo, A. Anandasivam, and R. Buyya, "Environment-conscious scheduling of hpc applications on distributed cloud-oriented data centers," *Journal of Parallel and Distributed Computing*, vol. 71, no. 6, pp. 732–749, 2011. http://dx.doi.org/10.1016/j.jpdc.2010.04.004.

[201] L. Luo, W. Wu, D. Di, F. Zhang, Y. Yan, and Y. Mao, "A resource scheduling algorithm of cloud computing based on energy efficient optimization methods," in *Green Computing Conference (IGCC), 2012 International*, pp. 1–6, IEEE, 2012. http://dx.doi.org/10.1109/IGCC.2012.6322251.

[202] Y. Shi, X. Jiang, and K. Ye, "An energy-efficient scheme for cloud resource provisioning based on cloudsim," in *Cluster Computing (CLUSTER), 2011 IEEE International Conference on*, pp. 595–599, IEEE, 2011. http://dx.doi.org/10.1109/CLUSTER.2011.63.

[203] J. Shuja, K. Bilal, S. A. Madani, and S. U. Khan, "Data center energy efficient resource scheduling," *Cluster Computing*, vol. 17, no. 4, pp. 1265–1277, 2014.

[204] S. Singh and I. Chana, "Earth: Energy-aware autonomic resource scheduling in cloud computing," *Journal of Intelligent & Fuzzy Systems*, vol. 30, no. 3, pp. 1581–1600, 2016.

[205] Q. Zhao, C. Xiong, C. Yu, C. Zhang, and X. Zhao, "A new energy-aware task scheduling method for data-intensive applications in the cloud," *Journal of Network and Computer Applications*, vol. 59, pp. 14–27, 2016.

[206] S. K. Mishra, M. A. Khan, B. Sahoo, and S. K. Jena, "Time efficient task allocation in cloud computing environment," 2017.

[207] V. Sontakke, P. Patil, S. Waghamare, R. Kulkarni, N. Patil, M. Saravanapriya, and U. Scholar, "Dynamic resource allocation strategy for cloud computing using virtual machine environment," *International Journal of Engineering Science*, 2016.

[208] A. Saraswathi, Y. Kalaashri, and S. Padmavathi, "Dynamic resource allocation scheme in cloud computing," *Procedia Computer Science*, vol. 47, pp. 30–36, 2015.

[209] C. S. Pawar and R. B. Wagh, "Priority based dynamic resource allocation in cloud computing," in *Cloud and Services Computing (ISCOS), 2012 International Symposium on*, pp. 1–6, IEEE, 2012.

[210] I. R. K. Raju, P. S. Varma, M. R. Sundari, and G. J. Moses, "Deadline aware two stage scheduling algorithm in cloud computing," *Indian Journal of Science and Technology*, vol. 9, no. 4, 2016.

[211] M. A. Rodriguez and R. Buyya, "Deadline based resource provisioningand scheduling algorithm for scientific workflows on clouds," *IEEE transactions on cloud computing*, vol. 2, no. 2, pp. 222–235, 2014.

[212] R. N. Calheiros and R. Buyya, "Meeting deadlines of scientific workflows in public clouds with tasks replication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1787–1796, 2014.

[213] S. J. Cunningham and S. M. Dillon, "Authorship patterns in information systems," *Scientometrics*, vol. 39, no. 1, p. 19, 1997.

[214] S. Vialle, S. Contassot-Vivier, T. Jost, *et al.*, "Optimizing computing and energy performances in heterogeneous clusters of cpus and gpus.,"

[215] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya, "Virtual machine power metering and provisioning," in *Proceedings of the 1st ACM symposium on Cloud computing*, pp. 39–50, ACM, 2010. http://dx.doi.org/10.1145/1807128.1807136.

[216] H. P., "Microsoft joulemeter: Using software to green the data center."

[217] W. Chen, Y. C. Lee, and A. Y. Zomaya, *Handbook of Energy Aware and Green Computing*, vol. 2. Taylor & Francis Group, 2012.

[218] S. Gopalakrishnan, "Sharp utilization thresholds for some realtime scheduling problems," *ACM SIGMETRICS Performance Evaluation Review*, vol. 39, no. 4, pp. 12–22, 2012.

[219] "Parallel workload archives."

[220] R. Patton, F. Uppal, and C. Lopez-Toribio, "Soft computing approaches to fault diagnosis for dynamic systems: a survey," *IFAC Proceedings Volumes*, vol. 33, no. 11, pp. 303–315, 2000.

[221] C. Cruz, J. R. González, and D. A. Pelta, "Optimization in dynamic environments: a survey on problems, methods and measures," *Soft Computing*, vol. 15, no. 7, pp. 1427–1448, 2011.

[222] R. Hecht-Nielsen, "Counterpropagation networks," *Applied optics*, vol. 26, no. 23, pp. 4979–4984, 1987.

[223] Z. Lin, K. Khorasani, and R. Patel, "A counter-propagation neural network for function approximation," in *Systems, Man and Cybernetics, 1990. Conference Proceedings., IEEE International Conference on*, pp. 382–384, IEEE, 1990.

[224] T.-C. Liu and R.-K. Li, "A new art-counterpropagation neural network for solving a forecasting problem," *Expert Systems with Applications*, vol. 28, no. 1, pp. 21–27, 2005.

[225] A. Majeed and M. A. Shah, "Energy efficiency in big data complex systems: a comprehensive survey of modern energy saving techniques," *Complex Adaptive Systems Modeling*, vol. 3, no. 1, p. 6, 2015.

[226] M. Mao, J. Li, and M. Humphrey, "Cloud auto-scaling with deadline and budget constraints," in *Grid Computing (GRID), 2010 11th IEEE/ACM International Conference on*, pp. 41–48, IEEE, 2010.

[227] "Mathematical optimization." Available on: www.wikipedia.org/wiki/Mathematical optimization, Last accessed= August, 2015.

[228] S. Sivanandam and S. Deepa, *Principles of Soft Computing*. John Wiley & Sons, 2007.

[229] 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.

[230] D. Ballabio, V. Consonni, and R. Todeschini, "The kohonen and cp-ann toolbox: a collection of matlab modules for self organizing maps and counterpropagation artificial neural networks," *Chemometrics and intelligent laboratory systems*, vol. 98, no. 2, pp. 115–122, 2009.

[231] D. Ballabio and M. Vasighi, "A matlab toolbox for self organizing maps and supervised neural network learning strategies," *Chemometrics and intelligent laboratory systems*, vol. 118, pp. 24–32, 2012.

[232] L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, 2007.

[233] S. Balasundram, J. Roshy, B. Ramadoss, and T. Balasubramanian, "Smart learning using pervasive computing devices," *Encyclopedia on Information Science and Technology*, 2007.

[234] N. Mishra, "Cloud computing and the banking industry a leap of faith," 2015. https://www.netmagicsolutions.com/blog/cloud-computing-and-the-banking-industry-a-leap-of-faith.

[235] S. Ghule, R. Chikhale, and K. Parmar, "Cloud computing in banking services," *International Journal of Scientific and Research Publications*, vol. 4, no. 6.

[236] "Benefits of cloud based banking infrastructure," Available on: https://letstalkpayments.com/the-benefits-of-cloud-based-banking-infrastructure/, Last accessed= July, 2016.

[237] "Collaboration slashes admin burden for small biz." Available on: https://www.commbank.com.au/guidance/newsroom/commbank-xero-collaboration-201607.html, Last accessed= October, 2017.

[238] "About salesforce financial services." Available on: https://www.salesforce.com/au/financialservices/, Last accessed= June, 2017.

[239] "Cloud computing expo." Available on: http://www.cloudcomputingexpo.com/node/1706833, Last accessed= July, 2017.

[240] "Deutsche bank and hewlett-packard sign agreement," tech. rep., 2015. Available on: http://www8.hp.com/in/en/hp-news/press-release.html?id=1919059.WZgaqCgjHDc.

[241] "Nedbank uses cloud computing to improve business performance." Available on: http://www-01.ibm.com/software/info/television/html/F403734K26932U49.html, Last accessed= October, 2017.

[242] "Mizuho bank begins api banking on ibm cloud to help drive innovation with partners." Available on: https://www-03.ibm.com/press/us/en/pressrelease/52559.wss.

[243] "Philippines unionbank selects ibm cloud for app overhaul." Available on: https://www-03.ibm.com/press/us/en/pressrelease/52249.wss, Last accessed= January, 2017.

[244] "Softwares used by different banks in india." Available on: http://latestbankupdate.blogspot.in/2016/02/software-used-by-different-banks-in_18.html, Last accessed= February, 2017.

[245] "About finacle." Available on: https://www.edgeverve.com/finacle/, Last accessed= March, 2017.

[246] "Oracle flexcube direct banking." Available on: http://www.oracle.com/us/products/applications/financial-services/flexcube-direct-banking/index.html, Last accessed= April, 2017.

[247] "Tcs bancs banking solution." Available on: http://sites.tcs.com/tcsbancs/banking/ , Last accessed= February, 2017.

[248] "Xero- beautiful accounting software." Available on: https://www.xero.com/au/about/, Last accessed= April, 2017.

[249] "Xero- features and tools." Available on: https://www.xero.com/uk/features-and-tools/accounting-software/, Last accessed= February, 2017.

[250] "Tibco- hybrid cloud integration platform." Available on: https://www.tibco.com/solutions/hybrid-cloud-integration.

[251] "About bank of montreal- tibco." Available on: https://www.tibco.com/customers/bank-montreal.

[252] "First citizens bank- tibco." Available on: https://www.tibco.com/customers/first-citizens-bank, Last accessed= February, 2017.

[253] "Kuveytturk bank- tibco client." Available on: https://www.tibco.com/customers/kuveytturk-bank, Last accessed= February, 2017.

[254] G. P. Gupta, M. Misra, and K. Garg, "Performance evaluation of agent and non-agent based data dissemination protocols for wireless sensor networks," in *Networks (ICON), 2011 17th IEEE International Conference on*, pp. 123–128, IEEE, 2011.

[255] "Green banking for indian banking sector," tech. rep., Institute for Development and Research in Banking Technology, 2013.

[256] "Finacle core banking solution." Available on: https://www.edgeverve.com/finacle/finacle-core-banking-solution/, Last accessed= January, 2017.

[257] "Infosys tie-up with microsoft for cloud services." Available on: https://ibsintelligence.com/ibs-journal/ibs-news/infosys-finacle-heads-for-cloud-with-microsofttie-up-and-qantas-credit-union-deal-2/.

[258] "Eclipse mars." Available on: https://projects.eclipse.org/releases/mars, , Last accessed= March, 2016.

[259] A. W. Lewis, S. Ghosh, and N.-F. Tzeng, "Run-time energy consumption estimation based on workload in server systems," pp. 17–21, 2008.

List of Publications

International Journals (indexed by SCI)

1. Tarandeep Kaur and Inderveer Chana, "Energy efficiency techniques in cloud computing: A survey and taxonomy", ACM Computing Surveys (CSUR), Volume 48, No. 2, Article 22, 2015, doi: http://dx.doi.org/10.1145/2742488, (Published by ACM, Impact Factor: 6.7, ISSN: 0360-0300), cited by 80.

2. Tarandeep Kaur and Inderveer Chana, "Energy Aware Scheduling of Deadline-Constrained Tasks in Cloud Computing", Cluster Computing, Volume 19, No. 2, 2016, pp. 679-698. doi: http://dx.doi.org/10.1007/s10586-016-0566-9, (Published by Springer, Impact Factor: 2.04, ISSN: 1386-7857), cited by 14.

3. Tarandeep Kaur and Inderveer Chana, "GreenSched: An Intelligent Energy Aware Scheduling For Deadline-and-Budget Constrained Cloud Tasks", Simulation Modeling Practice and Theory, Volume 82, 201, 2018, pp. 55-83. doi: https://doi.org/10.1016/j.simpat.2017.11.008, (Published by Elsevier, Impact Factor: 1.9, ISSN: 1569-190X).

International Journals (not indexed by SCI)

1. Inderveer Chana and Tarandeep Kaur, "Delivering IT as a Utility- A Systematic Review", International Journal in Foundations of Computer Science Technology, Volume 3, No. 3, May 2013. doi: 10.5121/ijfcst.2013.3302, (Published by AIRCC Publishing Corporation, ISSN: 1839-7662), cited by 5.

2. Inderveer Chana and Tarandeep Kaur, "Resource Scheduling Techniques in Utility Computing- A Survey", International Journal of Systems and Service-Oriented Engineering (IJSSOE), IGI Global, USA, Volume 4, No. 2, pp. 44-65, (2014). doi: 10.4018/ijssoe.2014040104, (Published by IGI-Global, USA, Impact Factor: 0.37, ISSN: 1947-3052), cited by 2.

International Conferences

1. Tarandeep Kaur and Inderveer Chana, "Energy Efficient Cloud: Trends, Challenges and Future Directions", International Conference on Next Generation Computing and Communication Technologies, 2014 (ICNGCCT ,14), Dubai, UAE.

2. Tarandeep Kaur and Inderveer Chana, "Energy Conscious Allocation and Scheduling of Tasks in ICT Cloud Paradigm", Advances in Intelligent Systems, Computing, Vol. 409, Proceedings of International Conference on ICT for Sustainable Development (ICT4SD, 2015), Ahmadabad, Gujarat, Advances in Intelligent Systems and Computing, Vol. 409, Springer, Retrieved from http://www.springer.com/in/book/9789811001277aboutBook, doi: https://doi.org/10.1007/978-981-10-0135-2_57

3. Tarandeep Kaur, "Energy Aware Scheduling of Deadline-Constrained Tasks in Cloud Computing", Grace Hopper Conference of Women in Computing (GHC), Poster Presentation, Houston, Texas, USA (2015).

4. Tarandeep Kaur, "IntelGreen- An Intelligent Model for Allocation and Scheduling of Tasks in Energy aware Cloud", Grace Hopper Conference of Women in Computing (GHC), Poster Presentation, Orlando, Florida, USA (2017).