RENEWABLE ENERGY BASED EFFICIENT FRAMEWORK FOR SUSTAINABILITY OF DATA CENTRES

A Thesis submitted in fulfillment of the requirement for the award of the degree of

DOCTOR OF PHILOSOPHY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted By

Gagangeet Singh Aujla

(Registration No: 901503005)

Under the guidance of

Dr. Neeraj Kumar

Associate Professor,

Computer Science and Engineering Department

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR INSTITUTE OF ENGINEERING AND TECHNOLOGY,
PATIALA – 147004

JUNE 2018

CERTIFICATE

I, Gagangeet Singh Aujla, Regn. No. 901503005, hereby declare that the thesis entitled **"Renewable Energy based Efficient Framework for Sustainability of Data Centres"** submitted to the Computer Science and Engineering Department at Thapar Institute of Engineering and Technology, Patiala, Punjab, India is an authenticated record of my own work for the award of the degree of "Doctor of Philosophy" under the supervision of Dr. Neeraj Kumar. This report has not been submitted to any other institution for award of any other degree.

Gagangeet Singh Aujla

Regn. No. 901503005

Place: Patiala

Date: 15.06.2018

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

Verified by:

Dr. Neeraj Kumar,

Associate Professor,

Computer Science and Engineering Department,

Thapar Institute of Engineering and Technology, Patiala.

ABSTRACT

Cloud computing (CC) has emerged as one of the most powerful technologies from past few years in which end users can access various services as per their demands on pay per basis concept. The need for on-demand state-of-art services (smart sensing, e-healthcare and smart transportation) and computing infrastructure have paved way to the powerful paradigm of CC. These enhanced flexible and reliable attributes offered by the CC platform have led to its widespread popularity amongst the academia and industry. Using virtualization technology, cloud service providers (CSPs) create multiple copies of virtual resources deployed over a physical server to provide various services to the end users. Such a virtualized environment and resources are hosted on large geo-distributed, service-oriented and critical computing infrastructure known as data centers (DCs). Ever since its inception in 2000, CC paradigm has witnessed significant transitions in its overall usage, size, computational ability and underlying technology used for accessing various services. However, the huge amount of data generated by various smart devices such as-smart phones, tablets, smart meter, body sensors and wearable devices has escalated the load on DCs to a great extent. Moreover, with the emergence of Internet of things (IoT), the demand of real-time data storage, access and processing at cloud has increased manifold. In recent years, data-intensive applications such as–e-health, e-commerce and e-banking generate a huge volume of heterogeneous data which varies with respect to time and its location. Such a huge amount of data needs to be collected, stored, analyzed and processed effectively using DC infrastructure. To handle such massive data streams generated from all these applications, exist-

ing DCs infrastructure needs to be expanded both horizontally and vertically. The dependence of smart communities also make it critical to expand the DCs with millions of servers operating at geo-located sites. The expansion of DCs has a strong impact on economy, environment, and performance of services offered by the service providers. Therefore, this may lead to many associated challenges such as-high energy consumption, high operational cost, potential source of carbon footprints, grid instability and inefficient utilization of DC resources. However, the growing popularity of renewable energy sources (RES) has opened a new frontier for tackling the increasing level of carbon footprints globally. For the survival of mankind and other organisms, healthy and sustainable ecosystems are necessary. For this purpose, considerable efforts are being put forth globally for the energy transition from fossil fuels to ecologically sustainable systems. Sustainable energy is one of the best ways to serve the present needs without compromising the ability of future generations to cope with their needs. According to a recent survey, carbon emissions associated to energy consumption of DCs are expected to be double as compared to the previous decade. With such an increase in the growth rate of carbon emissions, the environmental sustainability level can also degrade proportionately. Therefore, to cope up with this challenge, the focus of DCs is slowly shifting from the mere usage of renewables towards sustainability of DCs. This is due to the reason that sustainable energy can provide myriad benefits such as-reduced carbon emissions, grid load, and operational cost along with participation in environmental commitment. For above reasons, the major objective of this work is focused on sustaining the energy consumption of DC using RES. However, due to intermittent nature of RES, it becomes a challenging task. For this reason, four techniques in different environments are designed in this thesis to handle the intermittency issues of RES. In the first technique, an SDN-based DC energy management scheme using RES has been proposed. The purpose of the proposed system is to sustain the energy consumption of DC using RES which it accomplishes successfully. To handle the intermittent nature of RES, EVs are used to support the energy consumption of DC.

Moreover, SDN provides energy efficient flow scheduling in the designed communication sensitive environment. In the second technique, an SDN-based edge-cloud environment has been designed for achieving sustainability of DCs using RES. In this technique, SVM classifies the incoming jobs on the basis of priority and delay-sensitivity. The classified jobs are then routed using an optimal flow path to different cloud DCs or edge devices having sufficient amount of renewable energy for execution. For this purpose, a two stage workload scheduling scheme has been designed for sustainability of DCs using RES while ensuring lower SLA violations. In the third technique, a multi-leader multi-follower Stackelberg game has been designed for renewable energy-aware resource allocation. The objective of the scheme is to sustain the energy consumption of DCs using RES alone. For this purpose, an optimal utilization of cloud resources was achieved by aligning them with energy consumption of DCs. The results obtained show that an optimal resource utilization helps to reduce the number of servers provisioned, which in turn minimizes the energy consumption of DCs. In this way, the objective of sustaining the DCs using RES is achieved. Finally, a container-as-a-service ($CoaaS$) model has been deployed in a geo-distributed cloud environment. To achieve the objective of sustainability, a renewable energy-aware multi-indexed job classification and scheduling approach has been designed. Using the global controller, the proposed scheme allocates the incoming workload to those DCs which have sufficient amount of renewable energy to handle the incoming jobs. All the proposed schemes are evaluated using realistic workload and weather traces and the results obtained show that the proposed schemes successfully achieve their objectives in contrast to the existing schemes.

ACKNOWLEDGMENTS

I deem it my duty to express a word of hearty gratitude to all those helping hands that in the process of writing this thesis have become part and parcel of this endeavor. First of all, I wish to acknowledge the benevolence of omnipotent Almighty who gave me strength, courage and patience to overcome all obstacles.

With profound sense of gratitude and heartiest regard, I express my sincere feelings of indebtedness to my supervisor Dr. Neeraj Kumar, Associate Professor, Computer Science Engineering Department, Thapar Institute of Engineering and Technology for his valuable guidance, motivation, encouragement, moral support and invaluable co-operation. The generous and positive attitude with which he solved my queries will always have a shadow on my character. I deeply admire the delightful ambiance for learning provided by him that made this thesis possible. It has been a great pleasure and experience to work under his sanctuary.

I am grateful to Head of Department, Prof. Maninder Singh and Prof. Deepak Garg (former HOD), who made my study a knowledgeable experience during my stay in the department. I am much beholden to the Director, Dean (RSP) and the Management of Thapar Institute of Engineering and Technology, who provided me all the necessary resources and encouraged to produce results. I am thankful to my doctoral committee members Prof. Anil Kumar Verma, Dr. Ajay Kumar Loura, and Dr. Ravi Kumar for their constructive suggestions and ensuring the correct pace of the progress of my research work. I sincerely thank the faculty and support staff of Computer Science and Engineering department for their constant motivation. I would also like to thank Dr. Mukesh Singh for providing his able guidance to me in the initial phase of this journey.

I would like to pay deep gratitude to my parents Prof. K. S. Aujla and Mrs. Surjit Kaur for their unconditional love, affection, and encouragement through all the good and bad times. I would like to shower hearty gratitude, feeling of indebtedness and endless love to my wife Navneet Kaur for her unconditional love, moral support and sacrifices which helped me achieve this target. I would also like to acknowledge my parent-in-laws Mr. Datar Singh Mann and Mrs. Kuljeet Kaur for the encouragement and support they provided to me during this journey. I want to pay humble gratitude to my brotherly friend Mr. Sahil Vashist for being my strength at every moment.

Words can not truly express my feelings and appreciation to all friends and colleagues in lab, Dr. Amit Dua, Mr. Anish Jindal, Ms. Kuljeet Kaur, Mr. Rajat Chaudhary, Mr. Sahil Garg, Ms. Aisha Makkar, and Dr. Sudhanshu Tyagi who shard their knowledge and worked as a team to make my journey during this work memorable and pleasant. I would also like to thank Dr. Shalini Batra for providing me the support during the tough time of this journey. These words of gratitude may have missed the names of lots of well wishers, critics, friends and beloved ones. I pay regard to one and everyone who knowingly or unknowingly supported me during this journey of knowledge.

Finally, i want to bestow all my love to my world, my son, Avitaj Singh Aujla who came into my life during this journey and showed me with the path of positiveness with his heavenly and divine smile. Love you son.

(**Gagangeet Singh Aujla**)

Contents

List of Figures

List of Tables

LIST OF IMPORTANT ABBREVIATIONS

Acronym	**Meaning**
aDCL	Available data center list
BIOS	Basic input/output system
BESS	Battery energy storage system
CC	Cloud computing
CSP	Cloud service providers
CPU	Central processing unit
CoaaS/CaaS	Container-as-a-service
CS	Charging station
CML	Container migration list
cDC	Cloud Data center
DC	Data center
D2D	Device to device
DRAM	Dynamic random access memory
DCiE	Data center infrastructure efficiency
DPPE	Data center performance per energy
DCeP	Data center energy productivity
DHL	Destination host list
DHS	Destination host selector
EV	Electric vehicles
EaaS	Everything-as-a-service
ESU	Energy storage unit
ESD	Energy storage device

Acronym	**Meaning**
ESS	Energy storage system
FCFS	First come first serve
GT	Gillian ton
GW	Giga Watt
GHz	Giga Hertz
GB/s	Giga bytes per second
HA	Host activator
HD	Host de-activator
HDD	Hard disk drive
IoT	Internet of things
IT	Information technology
ICT	Information and communication technology
ITUE	Information technology usage effectiveness
IaaS	Infrastructure-as-a-service
I/O	Input/output
kWh	Kilo Watt
KB/s	Kilo bytes per second
LIBSVM	Library of support vector machine
LD	Load detector
MW	Mega Watt
MT	Million ton
M2M	Machine to machine
M/M/m	Markov machine with m servers
M/M/k	Markov machine with k servers
MEnSus	Energy management for sustainability
mDC	micro data center
MATLAB	Matrix laboratories

Acronym	**Meaning**
MIPS	Million instructions per second
NIST	National institute of standards and technology
NRDC	National resource defense council
nDC	nano data centers
OF	Open flow
OS	Operating system
OOL	Overloaded open flow switch list
OL	Overload limit
OHL	Overloaded host list
PDA	Personnel digital assistant
PPRR	Preemptive prioritized round robin
PaaS	Platform-as-a-service
PUE	Power usage effectiveness
PV	Photovoltaics
QoS	Quality of service
RES	Renewable energy sources
RP	Reward point
SSD	Solid state disks
SOA	Service oriented architecture
SDN	Software defined networks
SG	Smart grid
SoC	State of Charge
SaaS	Software-as-a-service
SVM	Support vector machines
TUE	Total power usage effectiveness
TOU	Time of use
US	United states

Acronym	**Meaning**
UHL	Underloaded host list
UDL	Underloaded destination list
UL	Underload limit
UOL	Underloaded open flow switch list
VM	Virtual machines
XaaS	Everything-as-a-service
5G	Fifth generation

Chapter 1

Introduction

With the rapid popularity of latest technologies such as- cloud computing (CC), Internet of things (IoT), and smart grid (SG) in modern smart cities environment , there is a steep increase in the design and deployment of various pilot projects in smart environments across the globe. Among all the aforementioned technologies, CC has emerged as one of the most powerful technologies from past few years in which end users can access various services as per the their demands on pay per basis concept [1]. The need for on-demand state-of-the-art services (smart sensing, e-healthcare and smart transportation) and computing infrastructure have paved the way to the powerful paradigm of CC [2]. CC provides various types of services (such as-IaaS, PaaS, SaaS, and in large everything as a service (EaaS)) to the end users connected to Internet using remote servers. Specifically, it provides shared pool of resources such as–servers, storage, and networks to the geo-located end users such as–human, vehicles, and smart devices [3].

The enhanced flexibility and reliability offered by the cloud computing platform leads to its widespread popularity amongst the academia and industry [4, 5]. This is evident from the wide scale adoption of CC infrastructure by the information technology (IT) vendors. According to 451 Research, the cloud services would witness an overall increase in their worldwide market from $21.9B in 2016 to $44.2B by 2020 [6]. This tremendous popularity can be attributed to the cloud's "pay-as-per-

use" model, wherein users utilize the available resources of storage, computation and networking as per their demands. The National Institute of Standards and Technology (NIST) has listed five essential attributes of cloud computing. These are namely-i) On-demand self-service, ii) Broad network access, iii) Resource pooling, iv) Rapid elasticity, and v) Measured service. These attributes are achieved by the underlying service oriented architecture (SOA) that supports services as per the respective enterprise model, *i.e.*, "EaaS".

Using virtualization technology, cloud service providers (CSPs) create multiple copies of virtual resources which may be deployed over a physical server to provide various services [7]. Such a virtualized environment and resources are hosted at large geo-distributed, service-oriented and critical computing infrastructure known as data centers (DCs). DCs are massive computing infrastructure that has transformed the information technology sector [8]. Ever since its inception in 2000, CC paradigm has witnessed significant transitions in its overall usage, size, computational ability and underlying technology [9]. The huge amount of data generated by smart devices such as-smart phones, tablets, smart meter, body sensors, wearable devices, etc has exaggerated the load on DCs to a great extent. According to Cisco Cloud Index (2013-2018) [10], almost one-third workload of IT sector is handled by the cloud. According to this report, data analytics and IoT based applications are projected at a 2.7-fold growth by 2020. Moreover, this huge amount of data will also witness a 10-fold escalation to 247 EB by 2020 [10].

Today, most of mobile devices equipped with communication and computing facility can avail cloud services on the move using high-speed Internet [11]. Moreover, with an emergence of IoT, the demand of real-time data storage, access and processing at cloud has exaggerated manifold [4, 12, 13]. Moreover, the big data generated by the connected devices (smart phones, PDAs, wireless body sensors, smart meters, etc.) would be in the order of zettabytes in the near future. This is evident as per the latest statistics shared by Gartner in its annual report [14]. It advocates that nearly 50 billion devices would be connected to the Internet by 2020. Hence, the re-

laying of such huge data to the cloud infrastructure may create network bottlenecks in the future.

For effective management of big data storage, cloud computing has emerged as a latest computing paradigm that offers the potential of storing bulk data remotely which can be accessed from anywhere using the Internet. Some of the basic factors that paved the way for the cloud-based big data storage are given as below.

- Data needs to be stored, processed and retrieved closer to the position of frequently using entities (regional data).
- Data is gathered from various heterogeneous devices and organizations and stored at different locations which can be accessed at any time.
- Data need to be replicated across DCs for easy availability, fault tolerance, and backup.

The emerging cloud computing paradigm provides a ubiquitous, pay-per-use, on-demand access to a shared pool of scalable computing resources that requires minimum service provider involvement and management efforts. Nowadays, organizations and business enterprises generate a huge amount of sensitive data, such as personal information, financial data, and e-health records. Moreover, the volume of digital data produced has shown a tremendous growth in recent years. The volume of data generated by IoT devices has overwhelmed the data storage capacity of many organizations. Hence, the management of such a huge amount of data in local storage system is difficult. Moreover, this incurs high expenses because of the requirement of high-capacity storage systems and the expert personnel to manage them. Recent years have witnessed a tremendous drop in the cost of storage hardware. But, still almost 75 percent of the total ownership cost is related to the management of data storage. However, now the organizations and business enterprises have an option to outsource their data storage to cloud. Such an outsourcing of data storage to cloud provides multi-fold benefits such as- reduced burden on local data storage, zero maintenance effort and cost.

1.1 Expansion of DCs

In recent years, data-intensive applications such as–e-health, e-commerce and e-banking generates a huge volume of heterogeneous data which varies with time and location. Such a huge amount of data needs to be collected, stored, analyzed and processed effectively using DC infrastructure. To handle such massive data streams generated from these applications, existing DCs infrastructure needs to be expanded both horizontally and vertically. Moreover, with an increase in smart communities such as-smart cities, smart grid, smart healthcare and IoT, the dependence of users on DCs has increased many folds. A scenario of a typical smart city with 1 million population is depicted in Fig. 1.1. The huge amount of data generated by various smart communities is illustrated in the figure. This huge amount of data and the associated services has paved the way for expansion of DCs to a large extent.

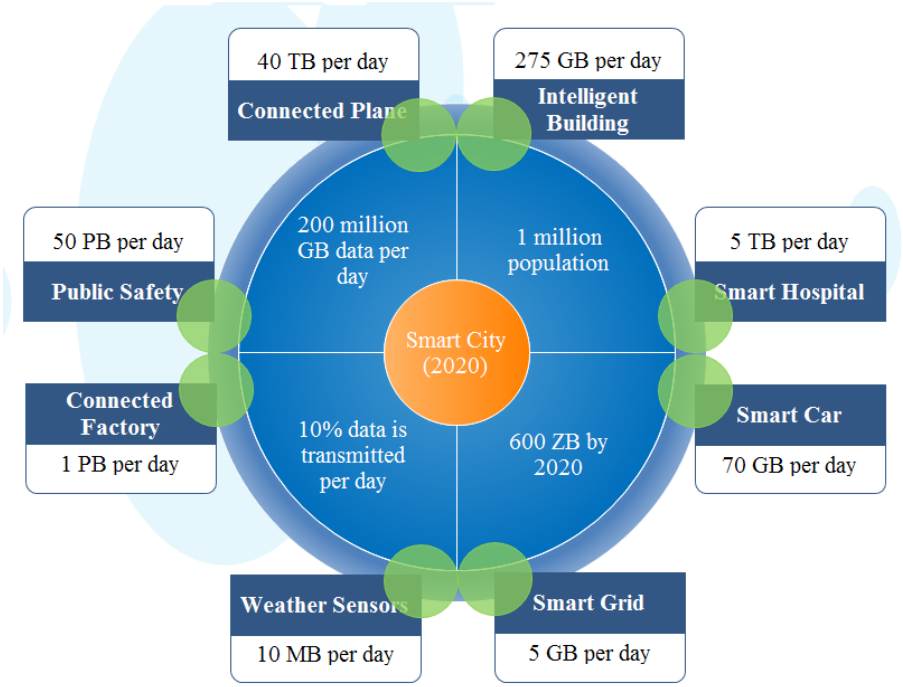


Figure 1.1: Data generated in a typical connected smart city

Therefore, the dependence of smart communities makes it critical to expand the DCs with millions of servers operating at geo-located sites. For example, over 12 million servers are installed in nearly 3 million DCs to manage the on-line activities

such as email, social media, and e-commerce across US only [15]. With an exponential growth of 566 percent (Fig. 1.2), every second 2.5 billion people are on-line all over the world including 70 percent Internet users. Nearly, 204 million email messages are exchanged, 5 million Google searches are made, 1.8 million users make likes on Facebook, 350,000 tweets are made, $272,000 amount is spent on Amazon for purchases, 15,000 tracks are downloaded through i-tunes on daily basis [15]. DCs act as backbone to handle all such on-line activities. Some of the largest DCs deployed by CSPS across the globe are shown in Fig. 1.2. For this reason, DCs have emerged as crucial promoter for growing IT sector with a global market size of $ 152 billion by 2016 [8].

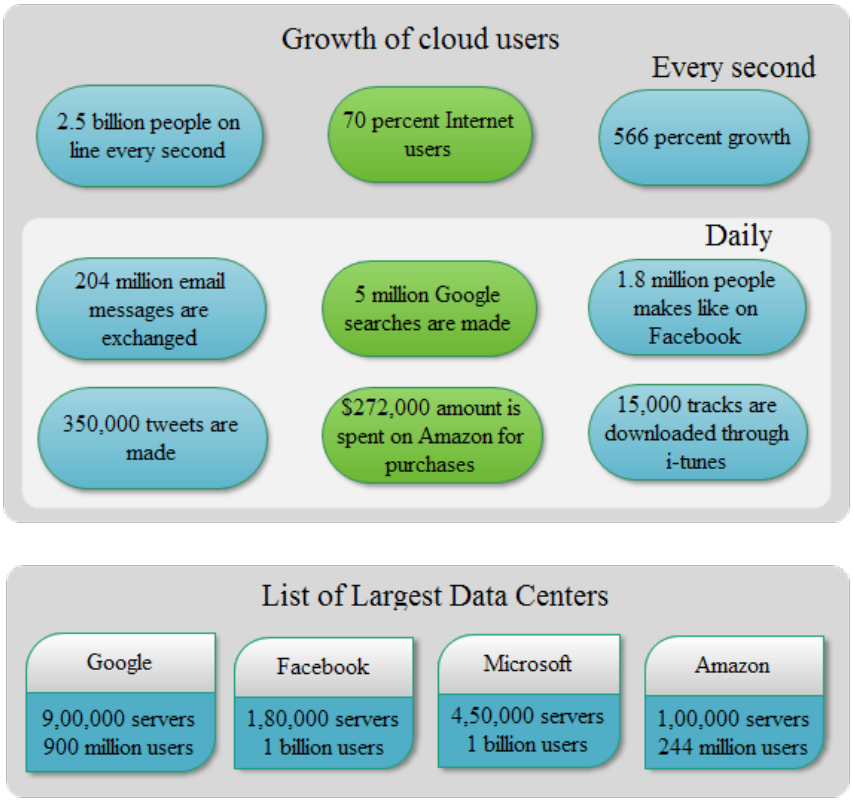


Figure 1.2: Growth of cloud users and largest DCs

Therefore, for these reasons, the DC infrastructure has witnessed a huge expansion in recent years. Moreover, this expansion is going to grow many folds in coming years. According to a recent announcement, IT sector is pledged to increase

the spending on DC infrastructure by nearly $2 billion. India is projected as second largest market for DC infrastructure in the Asia/Pacific region [16]. According to a survey (2014) [17], beyond 2015, only 8.5 percent of total global DCs are capable enough to handle the user's requests with its existing infrastructure. So, by 2020, 75 percent DCs are expected to expand to double the size as was in 2010.

1.2 Challenges of DC expansion

The drastic expansion of DCs has a strong impact on economy, environment, and performance of services offered by service providers [8]. Therefore, this may lead to many associated challenges. Some of the major challenges are discussed as below.

1.2.1 High Energy Consumption

According to New York times (2012), DCs consumed energy equal to generation of 30 nuclear plants [18]. Similarly, Mostelic and Brandic [17], highlighted that the energy consumed by DCs increased from 70 billion kWh to 330 billion kWh from 2000 to 2007. By 2020, this figure is expected to reach 1000 billion kWh [17]. Moreover, according to a report [19], DCs consume about 2 percent of electricity annually in US only. The aforementioned facts reveal that it requires 34 power plants capable of generating 500 MW of energy to power DCs in US only. According to National resources defense council (NRDC) [15], in 2013,DCs consumed nearly 91 billion kWh of energy in US only. This huge amount of energy is sufficient to power New York city for about 2 years. Moreover, the energy consumption of US DCs is projected at a growth of 53 percent, i.e., 140 billion kWh till 2020.

An increase in the demand of data-oriented computing, storage, and processing has made way for creation of massive DCs. Two major contributors of energy consumption in a DC are infrastructure facilities (cooling systems) and IT equipments (servers, storage, and network). The energy consumption breakup for a typical DC is shown in Fig. 2.1 [8].

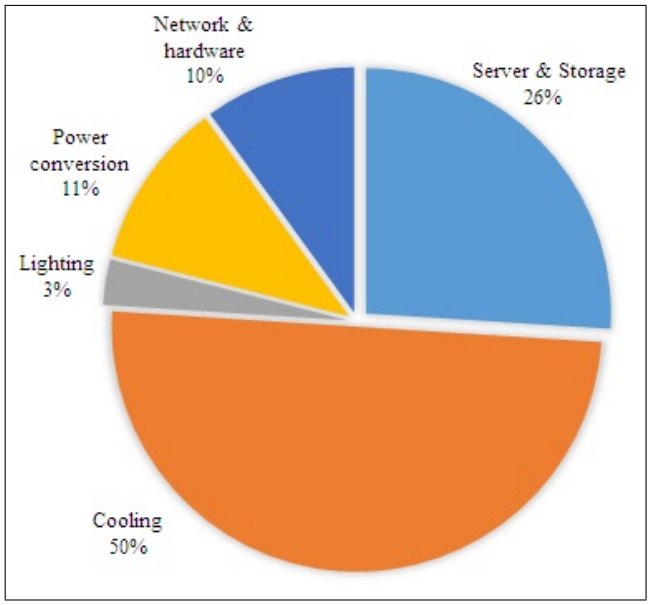


Figure 1.3: Energy consumption breakup for a typical DC

1.2.2 High Operational Cost

There is an operational cost associated to the routine activities (data storage, on-line data analytics etc.) of DCs. Among various factors contributing to the operational cost of these DCs, energy is one of the major concerns for management of routine activities of DCs. These DCs consume a huge amount of energy for their routine activities [3]. With an increase in size of DCs, their energy consumption has increased drastically which results in a overall high operational cost. For example, a typical server room in single DC consume energy equivalent to energy required to power 180 thousand homes. According to NRDC [15], in 2013, the amount of energy consumed by DC is equivalent to the energy required to power New York city for about 2 years. This amount of energy costs nearly $13 billion annually to a US businessman. Hence, CSPs have to bear a majority of operational cost on energy consumption for management of routine activities of DCs.

1.2.3 Largest Source of Carbon Footprints

According to the forecasting [15], by 2020, US will need 17 more power plants to meet the energy demand of DCs. Therefore, the use of fossil fuels to power such a

huge demand of DCs results in an increase in carbon emissions. About 97 million MT of harmful emissions related to DCs were released in 2013 and are projected to reach 147 million MT by 2020 [15]. For this reason, DCs are projected as the biggest contributors to carbon footprints. Moreover, energy sector powering DCs is projected as the largest source of global emissions with a 16 percent increase, i.e., 36.7 Gt by 2040 [20].

1.2.4 Grid Instability

According to world energy outlook 2014, global energy demand will grow annually by 1 percent making it one-third growth from 2013-2040 [20]. A considerable amount of energy demand is lead by physical infrastructure deployed by various ICT-based service providers. Being large contributors to the energy load, such infrastructure may lead to instability of grid [7]. Among various ICT, CC has emerged as one of the leading service providers to serve the requests of users. According to New York times (2012), DCs consumed energy equal to generation of 30 nuclear plants [18]. Mostelic and Brandic [17], highlighted that the DC energy consumption is expected to reach 1000 billion kWh by 2020. Therefore, such a huge growth in DC energy consumption bears additional load on the connected power grid which may lead to grid instability.

1.2.5 Inefficient utilization of DC resources

The expansion of DCs in terms of servers, network equipments, storage devices, etc may require effective resource management. At present, the resource management at most of the DCs is not up to mark. The over provisioning of resources is a common practice and there are various DCs which are under-provisioned. In general, a typical DC consume a large amount of energy in doing comparatively lesser work than its actual capacity. An average physical server deployed at a DC utilizes only 12-18 percent of its capacity [15]. Further, it has been said that generally each server

consumes about 60 percent of energy even when it is idle [21]. Hence, the inefficient utilization of DC resources may further elevate the above discussed challenges such as–energy consumption, carbon footprints and operational cost.

1.3 Sustainable DCs

The growing popularity of renewable energy sources (RES) has opened a new door for tackling the increasing level of carbon footprints globally. In regard to DCs, the use of renewables has traveled a long way from the use of RES as a secondary source of energy to power DCs towards the design of sustainable DCs. Some of the important facts and footsteps in this direction are discussed as below.

1.3.1 Impact of Renewable Energy Sources

The dependence on traditional fossil fuels has increased drastically in last few years. Due to this reason, energy sector remains the largest source of global emissions with a projected increase of 16 percent, i.e., 36.7 Gt by 2040. However, the growth of carbon emissions has been stable in recent years due to a shift towards alternate sources of energy generation. The decrease in annual growth rate of carbon emissions (2.4 percent: 2000 to 0.5 percent - 0.6 percent: 2020-30) is a positive sign. The global efforts to protect the environment against harmful emissions has made the way for renewables to overtake coal in order to become the largest source of electricity generation by 2040. The renewables are expected to overpower all other types of power generation with the deployment of 3600 GW renewables from 2013 to 2040 with most of it from wind and solar sources [20].

As per World energy outlook (2014) [20], global energy demand is expected to increase at a rate of 1 percent annually. By 2020, one-third increase in the energy demand is projected as compared to 2013. Hence, in coming years, the energy demand is going to increase many folds. Hence, to cope with the increasing demand, the energy generation has to be increased. However, traditional power generation

methods are not a preferred due to a decrease in fossil fuels and an increase in the carbon emissions. Therefore, RES are bound to lead the energy generation sector to handle these challenges. Hence, the integration of DCs with RES overcome the challenges related to its drastic expansion [22]. Generally, the DCs are powered by electric grid. But, nowadays, there are various DCs which utilize diesel, wind power, solar power and fuel cells as source of energy. Various CSPs such as– Amazon, Google, Facebook, etc. are utilizing clean energy to power their DCs.

1.3.2 Need of Sustainability

Sustainability is a way of reducing negative human impact on environment through various socio-ecological methods. For the survival of mankind and other organisms, healthy and sustainable ecosystems and environments are necessary. For this purpose, considerable efforts are being applied for the energy transition from fossil fuels to ecologically sustainable systems. Sustainable energy is one of the best ways to serve the present needs without compromising the ability of future generations to cope with their needs [23]. Moreover, it is projected that by 2040, renewables can expected to replace coal based energy generation [20]. According to a recent survey [24], by 2020, carbon emissions associated to energy consumption of DCs are expected to be double as compared to 2011. With such an increase in the growth rate of carbon emissions, the environmental sustainability level can also degrade proportionately. Therefore, to cope with this challenge, the focus of DCs is slowly shifting from the mere usage of renewables towards sustainability of DCs, ie., 100 percent renewable energy. For example, Apple tops the list of CSPs by achieving 100% clean energy index [24]. Moreover, the vision of green cities is incomplete without sustainable DCs. Hence, Sustainable energy can provide manifold benefits such as-reduced carbon emissions, grid load, and operational cost along with participation in environmental commitment.

1.4 Emergence of Edge Computing in Cloud Era

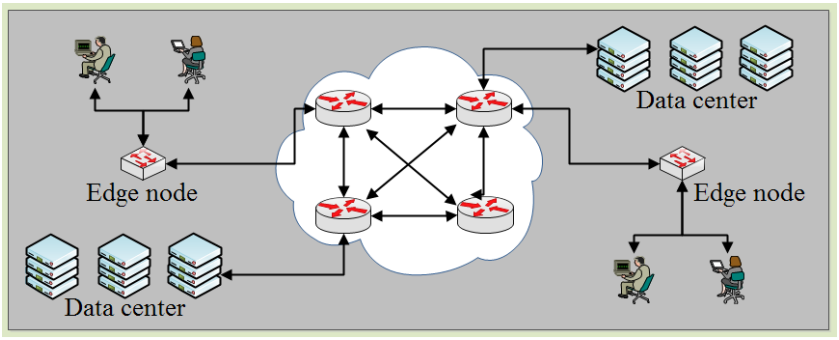
Today, most of mobile devices equipped with communication and computing facility can avail cloud services on the move using high-speed Internet. To handle such services, mobile cloud computing is one of the recent technologies [11]. However, limited computing resources and battery, high response time and cost make it difficult to provision various services to mobile devices using this platform. Moreover, device mobility poses an additional challenge for DCs to provide low-latency to mobile users. Recently, the usage of fifth-generation (5G) networks by such mobile devices have elevated the challenges to a higher level. Smart communities are converging with 5G to connect to cloud services. By adoption of such a high-speed network, computational time is bound to reduce significantly. But, energy consumption associated with provisioning of cloud services to 5G-enabled mobile devices using cloud DCs is expected to increase to a great extent [25].

Moreover, with the emergence of IoT, the demand of real-time processing and storage of data using CC has grown exponentially. The huge amount of data generated by smart devices such as-smart phones, tablets, smart meter, body sensors, wearable devices, etc has exaggerated the load on DCs to a great extent. According to Cisco Cloud Index (2013-2018) [26], almost one-third workload of information technology sector is handled by the cloud. So, relaying such a huge amount of data on underlying networks may lead to a performance bottleneck. Such an increase in network traffic in recent years may lead to additional burden on the underlying networks. In CC, to execute various applications smoothly, there is a movement of high volume of data across different geographically separated nodes which may create a huge burden on the network infrastructure.

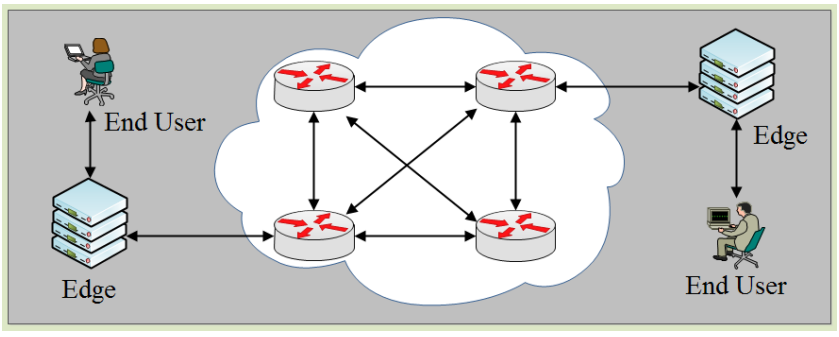
The existing network infrastructure is not capable to cater the demand of this high volume of data for all the above mentioned smart devices. Moreover, the interoperability of various mobile devices is one of the major concerns for effective network infrastructure usage. So, this may end up in latency problems which in

turn may lead to poor quality of service (QoS). Hence, to deal with service requests and data generated from smart devices (vehicles, IoT sensors, and actuators) locally, a recent paradigm popularly known as Edge computing has emerged.

Popularly known as '*cloud close to the ground*', edge computing provisions computing and processing resources at the edge of the network. This is realized through device-to-device (D2D) or machine-to-machine (M2M) communications. Edge devices comprise of nano-DCs (n-DCs), micro-DCs (m-DCs), gadgets, tablets, etc that lend their computing and processing resources to end users for real-time data processing. In this way, the need to transmit all the data to the core data centers can be avoided to a great extent. By using edge computing, multiple objectives such as-low latency, high bit-rate, faster transmission, and quick response time can be achieved successfully. Edge devices handles dual functionality such as- (1) creation of single hop or D2D communication between users without any involvement of central cloud, and (2) interplay with core cloud to utilize high-end functionality. Fig. 1.4 shows the typical cloud DC and edge DC scenario.



(a) Cloud computing scenario



(b) Edge computing scenario

Figure 1.4: Cloud and edge computing scenario

But, in order to provide low latency and high data rates, energy consumption of edge devices may surge. Hence, energy efficiency is matter of concern for both cloud and edge computing. In [27], the role of edge computing to meet the requirement of large-scale mobile services in terms of lower latency, geo-distribution, and mobility support is explored by the authors. But, at the same time, the impact of resource limitation on energy conservation has been pointed out. So, edge-cloud interplay brings many challenges which are listed below.

- Need for effective workload classification so that latency-hungry and realistic end user requests can be provisioned to edge devices and resource-hungry requests are relayed towards central cloud DCs.
- Optimal scheduling of classified workload in edge-cloud environment in order to achieve trade-off between energy efficiency and QoS.
- Resource-limited edge devices may migrate user requests to cDCs in order to meet service level of agreement (SLA). For this purpose, the underlying networks must be adaptive to meet the vendor-independent demand of different edge nodes. This may lead to additional load on underlying networks.

The comparative analysis of cloud and edge computing is given in Table 1.1.

Table 1.1: Comparative analysis of cloud and edge computing

Parameter/Constraint	**Cloud computing**	**Edge computing**
Virtual machine (VM) migration	Yes	Yes
VM consolidation	Yes	Yes
VM placement	Yes	Yes
Content placement	No	Yes
Content migration	No	Yes
Resource limitation	No	Yes
Resource overloading	High	Very-high
Distance from end-user	High	Low
Type of access network	Traditional	High speed
Geo-distribution	Wide area	Local area
Integration with RES	Yes	No
Sustainability	Yes	No

1.5 Evolution of Software Defined Networks

One of the major challenges with cloud computing is the limitation of bandwidth. This challenge is expected to leverage with growth in cloud users. In this regard, edge computing has come up to rescue of the CC. By utilizing distributed and scattered edge devices, the storage and processing in proximity to the end user premises can lead to stability of bandwidth consumption. However, with with the advent of edge computing, a huge amount of load on the network is generated due to numerous requests flowing across edge-cloud scenario. Therefore, an effective, programmatically configured, scalable, flexible, and adaptable underlying network backbone is need of the hour. Moreover, as per [28] network devices consume a large share of DC energy consumption after servers. Hence, apart from effective management of network resources, energy-efficiency is also a matter of concern in context of underlying networks [1]. In this direction, some of the challenges which makes traditional networks unsuitable for edge-cloud environment are given as below.

- Huge amount of streaming and batch workloads and Real-time provision of user requests.
- Interoperability and vendor independent edge nodes.
- Need of reconfigurable underlying network capabilities that can behave according to varying conditions.
- Programmable network capabilities to meet realistic bandwidth and latency requirement.
- Scalability of edge-cloud environment is very high.
- High cost of traditional network setup for edge-cloud environment.
- Energy efficiency, energy-aware flow consolidation and scheduling
- No scope of deployment of virtualized network resources.

Hence, to deal with above challenges, an emerging paradigm known as software defined networks (SDN) has emerged from past few years. SDN is a reconfigurable, flexible, and scalable network architecture that can be deployed in edge-cloud environment to handle bandwidth and latency issues along with energy-efficient transmission [29]. Moreover, SDN is agile, programmatically configured, open standards based, and vendor neutral, which makes it best suited to address the challenges faced by DCs. In SDN, the underlying internal infrastructure is abstracted from the applications and network control functions. Unlike traditional networks, in SDN, the control and data planes are decoupled to provide better level of abstraction. In this architecture, OpenFlow (OF) [30] protocol developed by OF Foundation is the core of flow management. The OF protocol is directly programmable and centrally managed. Fig. 1.5 depicts a SDN-based multi-cloud environment.

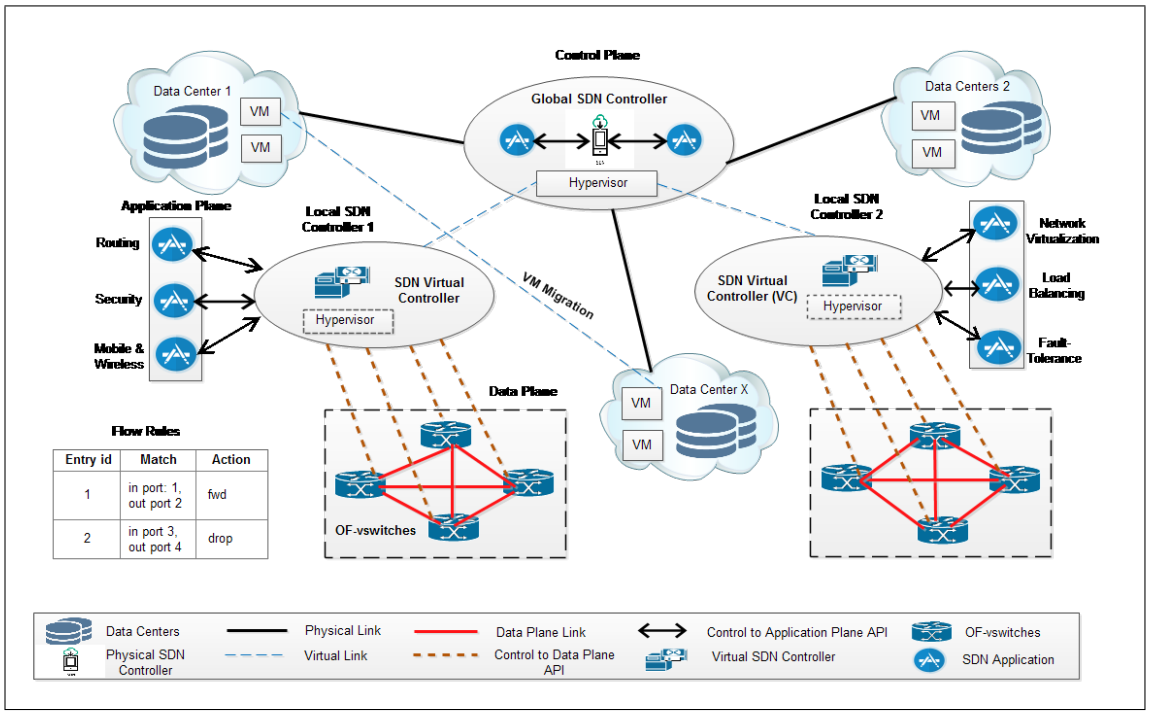


Figure 1.5: SDN-based multi-cloud environment

Such a platform not only improves the network performance, but it can also manage the network load efficiently. By adopting SDN, the network management becomes scalable as new functionalities and policies can be updated locally. Therefore, such network is a good choice to reduce complexity to improve network perfor-

mance. One of the most important capabilities of SDN is the ability to reconfigure flow management. In SDN-based systems, the control flow management can be programmed according to the user's requirements. But, in traditional networks, the control flow is fixed and cannot be programmed. Due to this reason, un-utilized switches and links consume huge amount of energy. But, energy consumption of such networks can be minimized by managing the control flow. In SDN, flow consolidation can also be used to minimize energy consumption of underlying networks. So, it is evident from above discussion that SDN is one of the most suitable technologies to manage the underlying networks in sustainable edge-cloud environment. Table 1.2 shows the major characteristics and comparative analysis of traditional network and SDN [31, 32].

Table 1.2: Traditional Networks vs SDN-based Networks

Parameters	**SDN**	**Traditional Networks**
Methodology	Centralized protocol	Dedicated protocol for each problem
Configuration	Automated and programmable	Manual and error-prone
Control	Cross layer and dynamic	Single layer and static
Implementation	Software-based environment	Hardware-based environment
Architecture	Decoupled layers (Data, control, and application)	Coupled layers
Extensibility	Open to new innovations due to software-based implementation	Limited implementation of new innovations due to hardware limitations
Forwarding	Flow-based	Routing table based
Control flow	Programmable and reconfigurable	Uses traditional protocols
Hardware	Vendor independent	Vendor dependent
Virtualization	Network and server	Not available
Network segmentation	Easy	Complicated
Data flow	Multiple paths for same flow	Fixed path

1.6 Motivation

According to [24], CC sector consumed about 684 billion kWh energy to handle its routine activities (data storage, on-line data analytics, etc.). Mostelic and Brandic [17], highlighted that the energy consumed by DCs increased from 70 billion kWh to 330 billion kWh from 2000 to 2007. By 2020, this figure is expected to reach 1000 billion kWh. Fig. 1.6 shows the energy consumption of different countries across the globe. It is evident from the figure that CC is ranked 6^{th} among globe with respect to energy consumption. These facts show that the amount of global energy usage is very high and DCs generates an additional load on the power grid. Hence, the grid to which DCs are connected have to bear an additional load.

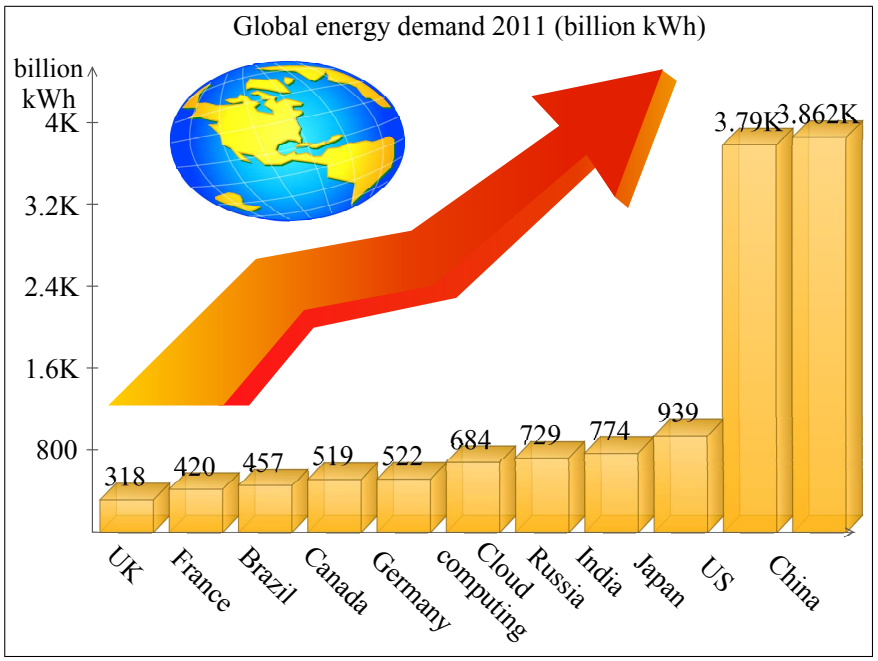


Figure 1.6: Energy consumption of different countries across the globe

With such a huge amount of energy consumption, DCs are major contributors towards the global carbon emissions. Fig. 1.7 depicts the country-wise representation of carbon emissions related to DC servers installed in each country [24]. As per this survey, projected carbon emissions for the year 2020 is almost double the carbon emissions pertaining to year 2011. Hence, such a huge amount of carbon emissions are alarming us to utilize RES to power DCs.

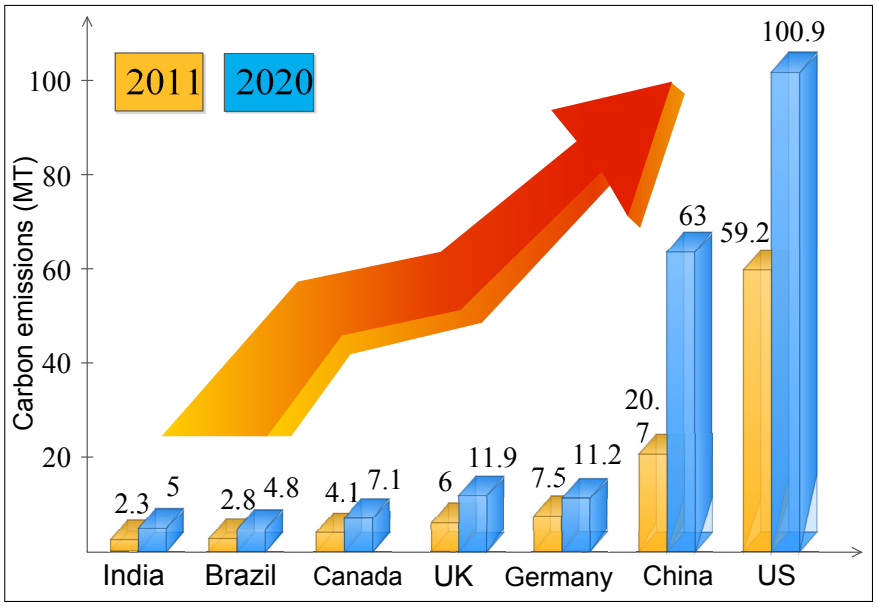


Figure 1.7: Representation of carbon emissions related to DC servers installed

Keeping in view of the above facts, the recent trends of large-scale service providers such as Apple, Yahoo, Facebook, Google, etc to shift towards the use of clean energy to power their respective DCs is a motivating aspect for the proposed work [24]. Fig. 1.8 shows the recent contribution of large-scale service providers towards the use of clean energy. Apple has already achieved 100 percent clean energy index for its DCs. Hence, the huge amount of energy consumption related to DCs, its direct relation to carbon emissions, and recent trends towards use of renewable energy are the factors for motivation of the proposed work.

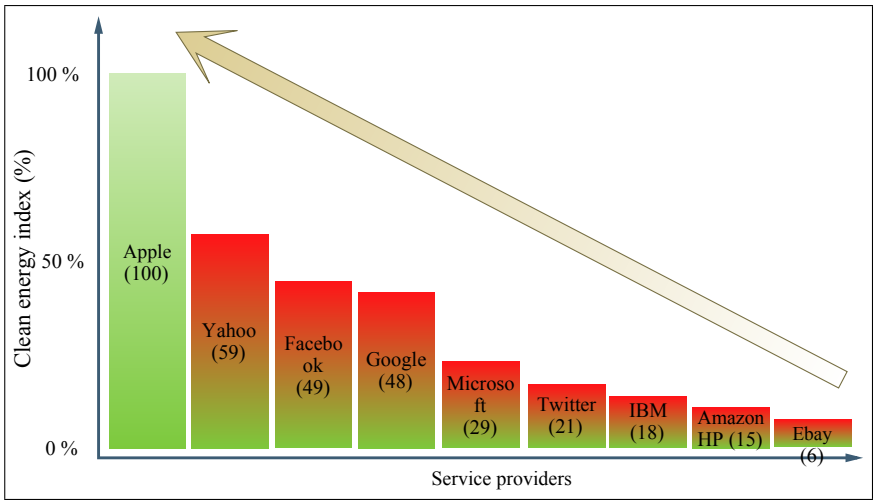


Figure 1.8: Recent contribution of CSPs towards the use of clean energy

Therefore, the major challenge is to provide a cost-effective solution for the power consumption of DCs to optimize the load on the grid. The design of sustainable DCs using RES can be an answer to the above raised issues [33]. However, the intermittent nature of RES needs to be handled effectively to achieve the above objective. For this purpose, renewable energy-aware resource allocation among geo-distributed DCs can be a viable solution. Moreover, latest technologies such as, SDN, edge computing and containers are promising solutions to overcome the different challenges arising during the design of sustainable DCs. SDN can be used to dynamically provision network traffics thereby reducing energy consumption and latency. Similarly, edge devices can act as a compliment to the central cloud infrastructure to handle the delay sensitive job requests closer to the location of the user. This not only reduces the delay and provides faster response time but also sheds the load on the central cloud infrastructure. Moreover, containers being quicker and energy efficient as compared to VMs can be beneficial for provision of faster migration with minimum dissipation of energy.

1.7 Thesis Organization

The thesis has been organized as per the following chapters:

- **Chapter 2: Literature Review**

  A comprehensive and illustrative literature review in the domain of energy efficient and sustainable data centers is provided in this chapter. Various existing approaches are compared with respect to different parameters. A detailed comparative analysis of the existing approaches related to data center energy efficiency and sustainability has also been provided in this chapter.

- **Chapter 3: An Electric Vehicle based Energy Management Scheme**

  In this chapter, the first technique, i.e., an SDN-based energy management scheme for sustainability of DCs is proposed using RES. To achieve the afore-

mentioned objectives, an energy-efficient flow scheduling algorithm is proposed using SDN. Moreover, a charging-discharging scheme for penetration of electric vehicles is also proposed to manage the intermittency of renewable energy. Finally, an energy trading and reward point scheme is designed to attract the electric vehicles to participate in the proposed energy management scheme.

- **Chapter 4: An Edge Computing based Energy Management Scheme**
  The second technique designed for sustainable DCs is proposed in this chapter. Here, an efficient scheme for energy management with sustainability (MEnSuS) of cloud data centers in edge-cloud environment using SDN is presented. In the proposed scheme, a support vector machine-based workload classification approach is presented. Moreover, a two-stage game for workload scheduling for sustainability of DCs is designed. In order to achieve energy efficiency and optimal utilization of network and computing resources, different consolidation schemes are also proposed.

- **Chapter 5: Renewable Energy-aware Resource Allocation Technique**
  This chapter provides the third technique, i.e., a renewable energy-aware resource allocation scheme is proposed using a Stackelberg game for energy management in cloud-based DCs. For this purpose, a cloud controller is used to receive the requests of users which then distributes these requests among geo-distributed DCs in such a way that the energy consumption of DCs is sustained by RES. However, if energy consumption of DCs is not sustained by RES then the energy is drawn from the grid. The requests of users are routed to the DC which is offered lowest energy tariff from the grid. For this purpose, a Stackelberg game for energy trading is also proposed to select the grid offering lowest energy tariff to DCs.

- **Chapter 6: Multi-indexed Classification and Scheduling Scheme**
  The final technique designed for sustainable DCs is proposed in this chapter. In this regard, a renewable energy-aware multi-indexed job classification and

scheduling scheme using container-as-a-service (CoaaS) is proposed for DC sustainability. In the proposed scheme, incoming workloads from different devices are transferred to the DC which has sufficient amount of renewable energy available with it. For this purpose, a renewable energy-based host selection and container consolidation scheme is also designed.

- **Chapter 7: Results and Evaluation**

  This chapter presents the experimental evaluation of all the proposed techniques. The proposed techniques have been simulated using MATLAB and CloudReports. All the proposed techniques were evaluated using interactive workload and realistic weather traces.

- **Chapter 8: Conclusion and Future Scope**

  This chapter concludes the thesis by highlighting the contributions made towards the proposed research domain. Moreover, this chapter also provides the future directions in this research area.

Summary of the Chapter: The Chapter 1 highlights the background of the cloud DCs. Moreover, all the challenges associated with the DC expansion are discussed in this chapter. The need of sustainable DCs and motivation of the work is also presented in this chapter followed by the organization layout of the thesis. This chapter clearly states the reason and the need of working in the area of sustainability of DCs using RES backed by various facts and surveys.

Chapter 2

Literature Review

In the past, a lot of existing research proposals have focused on designing energy-efficient DCs[1]. However, nowadays, the focus of large-scale service providers such as, Facebook, Google, Amazon, Apple and Microsoft has been shifted towards the design of green and sustainable DCs [34]. According to global surveys [7,8], the two major contributors towards energy consumption in a DC are infrastructure facilities (cooling and air conditioning systems) and IT equipments (servers, storage, and network). Almost half of the total energy consumption of DCs is lead by cooling systems and most of the other half includes majority of computing resources, computation jobs, and storage. The energy consumption breakup for a typical DC is shown in Fig. 2.1 [8].

[1]The content of this chapter is taken from:

- G. S. Aujla and N. Kumar, "SDN-based energy management scheme for sustainability of data centers: An analysis on renewable energy sources and electric vehicles participation", Journal of Parallel and Distributed Computing, Vol. 117, July 2018, pp. 228-245.
- G. S. Aujla and N. Kumar, "MEnSuS: An efficient scheme for energy management with sustainability of cloud data centers in edgecloud environment", Future Generation Computer Systems, 2017, doi:10.1016/j.future.2017.09.066.
- G. S. Aujla, M. Singh, N. Kumar and A. Zomaya, "Stackelberg Game for Energy-aware Resource Allocation to Sustain Data Centers Using RES," IEEE Transactions on Cloud Computing, 2017. doi: 10.1109/TCC.2017.2715817
- G. S. Aujla, N. Kumar, S. Garg, K. Kaur, R. Ranjan and S. K. Garg, "Renewable Energy-based Multi-Indexed Job Classification and Container Management Scheme for Sustainability of Cloud Data Centers," IEEE Transactions on Industrial Informatics, 2018. doi: 10.1109/TII.2018.2800693

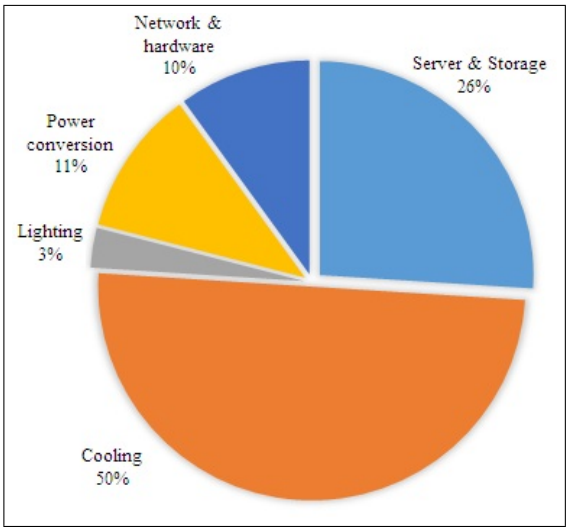


Figure 2.1: Energy consumption breakup for a typical DC

In this direction, Fig. 2.2 shows the classification of various existing proposals related to energy management of DCs.

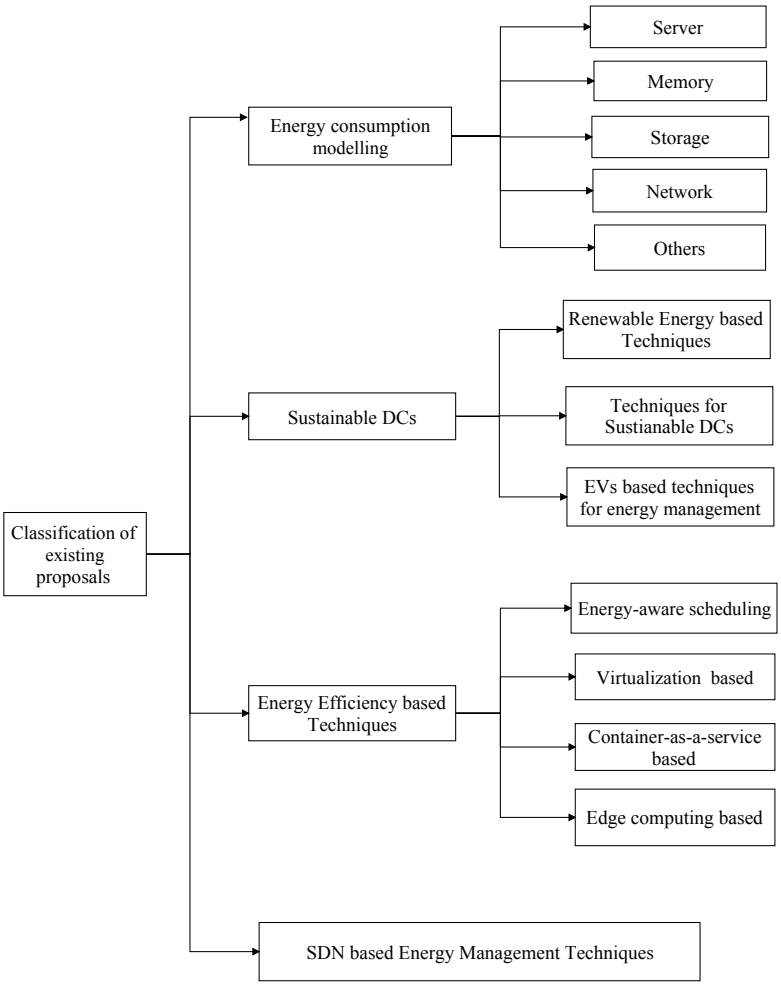


Figure 2.2: Classification of existing proposals

On the basis of this classification, the review of status of research and development in the subject related to various aspects is illustrated as below.

2.1 Energy consumption modelling for DCs

Before designing an optimal solution for energy management of DCs, the energy usage patterns at various levels and components in a DC have to be analyzed. In this regard, Dayarathna *et al.* [8] suggested that energy modeling is a key step towards the design of an energy-efficient DCs. Authors surveyed different techniques related to energy management of DCs. After performing deep analysis of these techniques at various levels (hardware, server and system), the authors suggested that there is lot of potential to be explored at higher level of energy consumption modeling. In this direction, the energy consumption modeling at various levels such as–servers, processors, memory, storage, network and software is discussed as below.

Servers are responsible for various types of productive work in DCs. For this reason, they consume a large amount of energy for their routine jobs. Various power models for servers are available. For example, Roy *et al.* [35] proposed one of the simplest model which represented the server as a combination of central processing unit (CPU) and memory. Further, Jain *et al.* [36] proposed a power model by dividing the energy consumption of CPU and memory into data and instruction. In similar direction, Tudor *et al.* [37] proposed a conflux of above discussed power models with input/output parameters. Similarly, Ge *et al.* [38] proposed a power model comprising of CPU, memory, and other components of a system.

The energy usage by various components of a typical server in a Google DC is shown in Fig. 2.3. As per the figure, the energy consumption of CPU and memory leads the overall energy consumption of servers in a typical DCs. However, according to [8], the energy consumption of two servers with different configuration is also different. From the above proposals, it is clear that a sever and its configuration has a strong impact on energy consumption of DC.

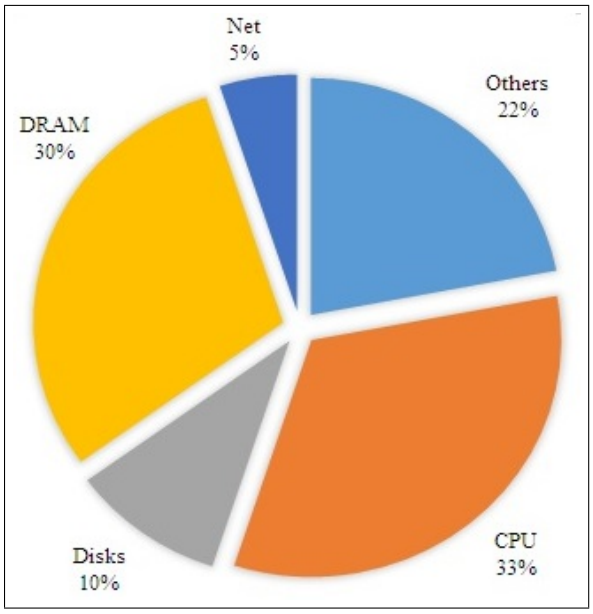


Figure 2.3: Breakdown of energy consumption of a typical server

Moving further in this direction, the amount of utilization of system resources and its components has a direct impact on energy consumption of DC. Various researchers have proposed power models for understanding the impact of *utilization level* of various components of a server on overall energy consumption of DCs. For example, Zhang *et al.* [39] highlighted the impact of processor utilization on energy consumption of a DC. Similarly, Wang *et al.* [40] suggested that CPU utilization is a proxy for the effect of an active workload management of DCs. Li *et al.* [41] presented a power consumption model of a server in a DC based on CPU utilization. The authors used the normalized power unlike existing power models.

On the similar trends, Tang *et al.* [42] designed a sophisticated power model with server dependent parameters and CPU utilization. The major difference of this power model from other power models is the addition of a temporal parameters and accounting feature for multiple servers. In an another work, Yao *et al.* [43] presented power consumption modeling of a server based on the utilization and rate of operations. Similarly, Tian *et al.* [44] presented a power consumption modeling scheme of a server based on the service rate and the utilization of servers. The authors replaced the frequency parameter with the above mentioned parameters.

As visible in Fig. 2.3, CPU is the largest consumer of energy in a server. With wide variety of CPUs available from single-core to multi-core, its energy consumption is directly dependent of *type of processor*. The power consumption of a CPU largely depends on the micro architecture of the processor. To a great extent, CPU frequency is responsible for deciding the energy utilization of a processor. The power breakdown of single-core and multi-core processor is different. The multi-core processors witness large amount of power leakage in contrast to the single-core processors. Apart from other aspects, processor cache also accounts to large amount of energy consumption. In this direction, Shin *et al.* [45] presented an additive power model for a single-core processor. The authors presented that the power models for single-core processors can be build by adding the power consumption of each component of CPU. Similarly, Bertran *et al.* [46] presented an energy model for a single core CPU. The authors represented the dynamic power consumption in terms of activity ratio and total number of components.

In an another work, Bircher *et al.* [47] discussed that the performance related events (for example, cache miss) that are very known within a microprocessor highly impacts and relates to the power consumption activities happening outside the microprocessor. Therefore, the internal events happing inside a processor (within each subsystem) are used to model the energy consumption of a processor. On the basis of such prediction based power models, the power consumption behavior of a system can be optimized. Moving ahead, Li *et al.* [48] utilized queuing theory for power modeling of multi-core processors. The authors presented multi-core processor as an M/M/m queuing model comprising of multiple servers. Two different types of core speeds (idle and constant) are considered in this power model. However, the authors improved the existing power model for the core by using different speeds for each core. The authors suggested that the core speed, number of cores, arrival rate has high impact on the power consumption of a server. Hence, from the above discussion, it is evident that the architecture and type of the processor has an impact on energy consumption of DC.

To summarize the power models for servers and CPU, Table 2.1 is given below.

Table 2.1: Energy modelling of servers in DCs

Proposal	**Focus area**	**Description**
Daya *et al.* [8]	Energy models	Suggested that energy modeling is key towards designing energy-efficient DCs.
Roy *et al.* [35]	Server	Represented the power consumption model of server as integration of CPU and memory
Jain *et al.* [36]	CPU and Memory	Proposed a power model for CPU and memory according to data and instruction
Tudor *et al.* [37]	Server	Augmented I/O parameters to the above discussed power model
Ge *et al.* [38]	Server	Presented a power model comprising of CPU, memory, and other components of system
Zhang *et al.* [39]	CPU utilization	Highlighted the impact of processor utilization
Wang *et al.* [40]	CPU utilization	CPU utilization is a proxy for effect of active workload management
Li *et al.* [41]	CPU utilization	Server power consumption modeling based on CPU utilization
Tang *et al.* [42]	CPU utilization	Designed a power model with server dependent and temporal parameters.
Yao *et al.* [43]	CPU utilization/Service rate	Power consumption modeling of a server based on utilization/rate of operation
Tian *et al.* [44]	CPU utilization/Service rate	Power consumption modeling of a server based on service rate and utilization of server
Shin *et al.* [45]	Single-core Processor	Power model for single-core processors
Bertran *et al.* [46]	Single-core Processor	Presented dynamic power consumption in terms of activity ratio and number of components
Bircher *et al.* [47]	Microprocessor	Performance within microprocessor highly impacts the power consumption happening outside microprocessor
Li *et al.* [48]	Multi-core Processor	Power model for multi-core processors

Further, it has been noted that the impact of memory and storage has increased in recent times. *Memory* has become the second largest energy consumer of a typical server [8]. It consumes nearly 30 percent of total DC server power. This is due to a rapid increase in the memory capacity of servers deployed in DCs [49]. In this direction, Ahn *et al.* [50] conducted a power modelling for memory. The authors classified power into static and dynamic categories. Further, the authors classified the dynamic power into activate pre-charge power and read-write power. In an another work, Malladi *et al.* [51] designed a power model with respect to expectations for memory energy. In this model, the authors initially modeled a stream of memory requests as a Poison process and then they represented the energy dissipation during power up and power down states. Lin *et al.* [52] proposed a power model for energy consumption of dynamic random-access memory (DRAM). In similar direction, Lewis *et al.* [53] proposed an energy consumption model with an aim on the number of memory devices involved at the time of operation.

Apart from memory, *storage systems* also consume a hefty amount of energy. Hard disk drive (HDD) is one of the main power consumer in the storage segment. However, the power modeling of HDD is a very difficult process due to the invisibility of the hard disk states [54]. In [55], the authors divided the HDD power consumption into two categories, 1) static and 2) dynamic. In an another work, Hylick *et al.* [56] suggested that the energy consumption related to read operation of multiple HDD's has a cubical relation with logical block numbers. Apart from HDD, flash memory based solid state disks (SSD) are leading the modern DC memory infrastructure because of its faster access and energy efficiency. SSD's add to lower power consumption as compared to HDDs but still they account for the same. In this direction, Park *et al.* [57] described the two-way energy consumption model for write operations based on the number of active flash dies during the plateau period. In an another work, Mohan *et al.* [58] proposed an analytical power model termed as FlashPower for two major variants known as single level cell and two-bit multi level cell based memory chips. Further, Poess *et al.* [59] discussed the energy

consumption modeling of data-intensive applications for enterprise warehouses. Table 2.2 shows power models for memory and storage systems.

Table 2.2: Energy models related to memory and storage systems for DCs

Proposal	**Focus area**	**Description**
Lin *et al.* [49]	Memory	Suggested that memory is major energy consumer.
Ahn *et al.* [50]	Memory	Classified power into static and dynamic categories. The dynamic power is further classified into activate pre-charge power and read-write power.
Malladi *et al.* [51]	Memory	Initially a stream of memory requests is modeled as a Poison process and then as energy dissipation during power up and power down states.
Lin *et al.* [52]	Memory	Designed a power modelling scheme to estimate the DRAM power at an instance.
Lewis *et al.* [53]	Memory	Energy consumed by DRAM bank is directly related to the number of read/write operations.
Kansal *et al.* [54]	HDD	Suggested that the power modeling of HDD is a very difficult process due to the invisibility of the underlying hard disk states.
Zhang *et al.* [55]	HDD	Divided the HDD power consumption into two categories, i.e., static and dynamic
Hylick *et al.* [56]	HDD	Suggested that the energy consumption related to read operation of multiple HDD's has a cubical relation with logical block numbers
Park *et al.* [57]	SSD	Described the two-way energy consumption model for write operations based on the number of active flash dies during the plateau period
Mohan *et al.* [58]	SSD	Proposed an analytical power model termed as FlashPower for two major variants known as single level cell and two-bit multi level cell based memory chips.
Poess *et al.* [59]	Storage	Energy consumption modeling of data-intensive applications for enterprise data warehouses.

All the above discussed proposals focused on lower-level energy consumption with an aim to design an efficient higher-level power models. In another direction, the power consumption of *group of servers* has been elaborated in various research proposals using queuing theory [60] and power efficiency [61]. In [60], the authors considered the use of queuing theory for power modeling of a DC with respect to group of servers. The authors stressed that multiple parameters such as time delay and power penalty need to be considered for the group of servers in contrast to a single server. The authors modeled the server farm with a setup cost on the basis of M/M/k queuing model. In another direction, Mitrani *et al.* [61] focused on the optimization of power management policies related to active and idle servers in a server farm. The authors suggested that there is no major benefit of micro managing the ON/OFF process of servers in a large-scale setup. Apart from queuing theory, some of the authors modeled group of servers on the basis of power efficiency. For example, Qureshi *et al.* [62] designed a power model for group of servers in a large scale system on the basis of power usage effectiveness (PUE).

Moving ahead, various power models have been designed based on *DC performance metrics*. These power metrics play an effective role in design of power models for DCs. In this direction, PUE is one of the most popular metrics to compare different designs of DCs with respect to energy consumption [62]. In the similar direction, DC infrastructure efficiency (DCiE) is another metric for measuring the efficiency of DCs [63]. Similarly, Patterson *et al.* [64] proposed two performance metrics for DCs namely, IT power usage effectiveness (ITUE) and total power usage effectiveness (TUE). ITUE is dependent on the power consumption of IT infrastructure, whereas, TUE is the integration of ITUE and PUE. In [65], energy efficiency of an entire DC is modeled using another metric known as DC performance per energy (DPPE). Sego *et al.* [66] proposed DC energy productivity (DCeP) metric to model performance of DCs with respect to relative energy consumption. Apart from the above discussed power efficiency metrics, various other DC power metrics exists which are not discussed above.

Table 2.3 is presented as below to summarize the power models with respect to the group of servers and DC performance metrics.

Table 2.3: Energy models for group of servers and performance metrics

Proposal	**Focus area**	**Description**
Gandhi *et al.* [60]	Group of servers	Developed a power model for group of servers. The authors stressed that multiple parameters such as time delay and power penalty need to be considered for the group of servers in contrast to a single server.
Mitrani *et al.* [61]	Group of servers	Optimized the power management policies for group of servers. The authors suggested that there is no major benefit of micro managing the ON/OFF process of servers in a large-scale setup.
Qureshi *et al.* [62]	Group of servers/ Performance metrics	Designed a power model for server farm in a large scale system on the basis of PUE.
xyz *et al.* [63]	Performance metrics	Designed a new parameter namely DCiE is another metric for measuring the efficiency of DCs.
Patterson *et al.* [64]	Performance metrics	Proposed two performance metrics for DCs namely, ITUE and TUE.
Obaidat *et al.* [65]	Performance metrics	Modeled energy efficiency of an entire DC using DPPE.
Sego *et al.* [66]	Performance metrics	Proposed a metric known as DCeP to model performance of DCs with respect to relative energy consumption.

The above discussed proposals related to group of servers are connected to each other through DC *networks*. Moreover, multiple DCs are also connected to each other using wide area networks. In this direction, communication links and network infrastructure consumes a large amount of energy. Various researchers have modeled the energy consumption related to DC networks exploiting its dynamic nature. For example, the power consumption model at DC network level based on level of

power consumption by a link was described by Heller *et al.* [67]. Extending this work, Widjaja *et al.* [68] proposed an optimization scheme for the network power consumption. In an another work, Li *et al.* [69] presented two power models related to the DC network traffic flow. The authors modeled the energy consumption of DC traffic flow on the basis of set of switches and ports that are active.

Vishwanath *et al.* [70] proposed a power consumption model for network devices. The authors modeled the energy consumption of router/switch on the basis of control, environment, and data planes. Pedram [71] designed an additive power model which presents the total energy consumption as a sum of different switches in a DC network. In another work, Hlavacas *et al.* [72] proposed an energy model for switches (residential and professional) on the basis of linear regression and bandwidth. Similarly, Ahn *et al.* [73] measured tha the energy consumption of an edge router is directly proportional to the link utilization and packet size. In a different work, Basmadjian *et al.* [74] classified the energy consumption of a network interface into two parts with respect to dynamic and idle status.

Moving ahead, the optical networks are also used in DC networks as they are more energy efficient and offer high throughput and lower latency. The power model for such networks range from low to high complexity. For example, Kachris and Tomkos [75] presented a network architecture for commodity switches with respect to different switches and transceivers. In a similar work, Kachris and Tomkos [76] proposed energy models with respect to six different optical DC networks. In an another work, Heddeghem *et al.* [77] combined the constituting layers of an optical network to model the total energy dissipation. In [78], the energy consumption model for transformation process for uninterrupted supply of electric power has been described. In another context, Diouri *et al.* [79] proposed an energy consumption technique based on broadcast algorithms. The aim of the scheme was to estimate the energy consumption of broadcast algorithms for communication-intensive applications. The summary of all the above discussed power models for DC networks is provided in Table 2.4.

Table 2.4: Energy models for DC networks

Proposal	**Focus area**	**Description**	**Limitations**
Heller *et al.* [67]	Network link	Based on level of power consumption by a link	Simpler model
Widjaja *et al.* [68]	Entire network	Total energy consumption is presented as a sum of energy consumed by each network link	Simpler model
Li *et al.* [69]	Entire network	Energy model presented with respect to traffic flow in a DC	Assumptions such as fixed energy consumption for switches and ports
Vishwanath *et al.* [70]	Network infrastructure	Power consumption model for network device	It is quiete challengeing to measure power consumed by different units
Pedram [71]	Switch	Additive power model	Depends on constants
Hlavacas *et al.* [72]	Switch	Linear regression based power model	Depends on constants
Ahn *et al.* [73]	Router	Power model based on link utilization and packet size	Need to calculate some values in advance
Kachris and Tomkos [76]	Optical network	Additive power model	Depends on accuracy of various parameters
Kachris and Tomkos [75]	Optical network	Additive power model	Depends on accuracy of various parameters
Heddeghem *et al.* [77]	Optical network	Based on static and dynamic energy consumption of network interfaces	Depends on accuracy of idle and dynamic time period
Diouri *et al.* [79]	Communication	Energy consumption model based on broadcast algorithms to highlight the impact of energy consumption for communication-intensive applications	Depends on multiple assumptions

The energy consumption of a DC also depends on the type of applications and jobs it handle. Various existing proposals highlighted that the *software-level* has a strong impact on energy consumption of a DC. Li *et al.* [80] discussed the behavior of operating system (OS) with respect to various applications to understand the energy pattern. According to the paper, data path and pipeline structures consume half of the total OS power consumption. Davis *et al.* [81] proposed an energy model based on OS for high accuracy. The authors presented a highly accurate and composable power model consisting of server and clusters. The authors evaluated four different modelling techniques with different complexities. Kim *et al.* [82] proposed a power model for VMs stating that the energy consumed by VMs depends on the number of events happening. The authors presented that the VM energy consumption depends on the number of active cores, memory accesses, and retired instructions. In similar direction, Liu *et al.* [83] proposed an energy consumption model based on VM migration. The proposed energy model is applicable for heterogeneous servers also. The summary of software-level power models is given in Table 2.5.

Table 2.5: Software-level energy models

Proposal	**Focus area**	**Description**
Li *et al.* [80]	OS	Energy model based on behavior of operating system energy profiles
Davis *et al.* [81]	OS	Evaluated different modeling schemes with different complexities
Kim *et al.* [82]	VM	Power model for VMs stating that the energy consumed by a VM is based on number of events happening at that time
Liu *et al.* [83]	VM	Energy consumption model for VM migration

After analysis of above discussed power models, it is evident that managing the energy consumption at various levels can be beneficial. In this context, various existing energy efficiency techniques for DCs are discussed ahead.

2.2 Energy-efficient Techniques for DCs

Shuja *et al.* [7] discussed various techniques and architecture for designing energy-efficient DCs. The authors discussed that high energy consumption make it necessary to design energy-efficient DCs. In this regard, many existing proposals have proposed energy efficient techniques for DCs. Such proposals are discussed as below.

2.2.1 Energy-aware scheduling in geo-distributed DCs

One of the major problem related to DCs, i.e., cost related to energy can be reduced by efficient resources allocation among geo-distributed DCs. Moreover, an efficient power-aware resources provisioning could be one of the options to minimize the energy consumption of DCs. For this purpose, DCs use the diversity of energy pricing to reduce the operational cost incurred to provide power to its physical infrastructure [84]. By capitalizing the concept of dynamic price provided by SG, the electricity cost of DCs could be reduced. For example, Naryan *et al.* [2] proposed a power consumption model to align the dynamic pricing offered by SG with cost model of cloud service providers. The authors proposed a cloud-based power metering system which accounts for the varying input cost of electricity offered by SG to decide the price of a cloud services to be charged from end users.

In a similar work, Quereshi *et al.* [62] proposed an scheme which used the time and location-dependent electricity price to reduce the operational expenditure of DCs. The authors utilized the variation in temporal and location dependent energy pricing in SG. The end user request were routed to the DCs where energy price was low. However, the scheme has to suffer from additional delay due to migration of requests among geo-distributed DCs. In similar direction, Rao *et al.* [85] addressed the above raised issue of delay by designing an optimization problem. In an another work, Li *et al.* [86] proposed an electricity demand management scheme for price-sensitive batch computing workload with an aim of energy cost minimization. The authors proposed a scheme for cooling efficiency based demand management to

reduce the electricity consumption of DCs. Similarly, Ghasemi-Gol *et al.* [87] presented an energy management scheme to utilize the day-ahead pricing announced by SG. In this way, the incoming requests were routed to DCs where energy price was lowest in order to reduce operational cost.

In an another work, Wang *et al.* [88] proposed a two-stage price optimization problem by considering active decisions between DCs and SG to migrate the workload. Tran *et al.* [89] formulated a game between utility and DC operator to minimize the electricity cost by shifting the request load dynamically among DCs. Authors proposed a Stackelberg game for demand response management between DCs and grid. The proposed scheme uses dynamic server allocation and workload shifting among DCs with an aim to minimize the operational cost. Qiu *et al.* [90] proposed an optimization scheme for cost minimization by migrating the user requests dynamically among geo-distributed DCs. Forestiero *et al.* [91], proposed a hierarchical approach for workload assignment and migration to reduce the energy cost of geo-distributed DCs. Wang *et al.* [92] proposed a CMC algorithm to optimize the workload distribution with respect to economics of DCs.

However, all these proposals focused on scheduling workload among geo-distributed DCs that may incur additional migration delay and in turn result in lower QoS. To overcome this challenge, Jin *et al.* [93] proposed an optimization problem between operational cost an performance degradation. Similarly, Rao *et al.* [94] proposed a scheme for cost minimization using the electricity price variations of deregulated market. The authors minimized the risks associated with uncertain energy pricing in deregulated energy markets while ensuring the required performance of DCs.

2.2.2 Virtualization-based energy-efficiency schemes for DCs

Dabbagh *et al.* [95] highlighted the potential solutions related to energy minimization of DCs. Authors exploited different energy management techniques associated to virtualization technology. The major finding of the proposals aimed at virtualiza-

tion, resource utilization and consolidation to achieve energy savings. The authors suggested that resource overloading is one of the major reasons for excessive energy consumption of DCs. In this direction, various research proposals focused on virtualization, resource utilization and consolidation to achieve energy efficiency.

In this direction, Wang *et al.* [96] proposed an energy-efficient VM placement approach for data-intensive services ensuring global QoS in national DCs. The authors proposed an optimization scheme for minimization of energy along with QoS guarantee. Authors used particle swarm optimization algorithm to design a VM placement scheme to achieve optimal trade-off between energy consumption and global QoS. But, the focus of this work was limited to just data-intensive services rather than user-centric services. The instability of arrival rate of user request was a big challenge to be addressed while guaranteeing QoS with energy minimization. In this direction, Goudarzi and Pedram [97] proposed a scheme using dynamic programming by creating multiple copies of VM to trade off between energy minimization and QoS. However, the above schemes considered homogeneous nature of physical servers apart from lacking in providing global QoS. But, this cannot be true for each and every DC such as, National DCs which allow different servers configurations along with addition in the number of servers whenever required. Hence, the minimum number of active servers do not guarantee minimum energy consumption.

In an another work, Islam *et al.* [98] proposed an on-line resource management scheme for workload minimization of DC by optimal resource allocation. In a similar work, Gu *et al.* [99] presented a cost minimization scheme with focus on task assignment, data placement, and data movement in DCs. Most of the aforementioned works only focused on minimizing the energy consumption of DCs by proposing various resource allocation schemes. However, with increasing dependence on CC and huge increase in user's requests day by day, the QoS guarantee is must. While migrating the jobs among geo-distributed DCs, the QoS such as response time, availability, and throughput has to be compromised to some extent. None of the above discussed schemes provided the QoS guarantee to the end users requesting for re-

sources. However, with increasing dependence of end users on computing resources being provided by CSPs, it becomes necessary to provide QoS guarantee. Moreover, to maintain SLA between service providers and service consumers, QoS with respect to the parameters such as response time, delay, availability, reliability, and throughput needs to be preserved at various levels.

Shuja *et al.* [28] discussed the conflicting aspects related to trade-off between maximizing QoS and minimizing energy consumption of DCs. Authors highlighted that inception of resource consolidation with minimal effect on QoS using a controller. Moreover, the authors discussed various hardware and software level methods to achieve energy conservation. They highlighted that energy consumption of a typical server accounts about 50-90 percent of total IT energy usage. This is followed by energy usage by network devices. To preserve QoS while designing energy-efficient DCs, Beloglazov and Buyya [100] presented a live migration scheme using simulation-driven evaluation of heuristics for dynamic reallocation of VMs. Authors designed an energy-efficient resource management scheme based on continuous VM consolidation as per the current utilization of the resources in order to reduce operational cost and ensure QoS.

Chiang *et al.* [101] proposed an efficient green control algorithm to solve constrained optimization problem in order to make trade-off between performance and energy saving policies. Feller *et. al.* [102] proposed an energy management scheme based on energy-aware VM management for private clouds. The proposed scheme achieved substantial energy savings by dynamically scaling the energy consumption of DCs with respect to the load with a minimal effect on performance. Beloglazov *et al.* [103], proposed an adaptive heuristics for dynamic VM consolidation to reduce energy consumption while preserving SLA. Wang *et al.* [104] presented a virtual batching technique for virtualized servers. The authors allocated the CPU resources dynamically in such a way that all VMs can achieve same performance level with respect to their allowed peak values. In an another work, Beloglazov *et al.* [105] proposed an energy-aware resource allocation for efficient management of DCs. Some

of the existing proposals focusing on both energy and QoS are listed in Table 2.6

Table 2.6: Energy and QoS aware Techniques for DCs

Author	**Description of work**	**Drawbacks**
Shuja *et al.* [7]	Energy efficiency of DCs with balance between energy consumption and QoS	Only highlighted the issue
H. Wang *et al.* [88]	Two-stage optimization problem to model interaction between grid and DCs	No focus on resource allocation and utilization, stable workload, homogeneous server, No integration with RES
Jin *et al.* [93]	Efficient VM consolidation and scheduling to optimization operational cost and performance degradation	Single DC, stable workload, homogeneous server, No integration with RES
S. Wang *et al.* [96]	Provision of data-intensive services through energy-and QoS-aware VM placement	Limited to just data-intensive services, National DCs, Instability of arrival rate of requests not addressed
Goudarzi *et al.* [97]	Energy-efficient VM replication and placement	Single DC, performance degradation, No integration with RES
Buyya *et al.* [100]	Energy-efficient resource management by consolidation of VMs according to current utilization of resources	Single DC, No integration with RES
Beloglazov *et al.* [105]	Energy-aware resource allocation for efficient management of DCs	Single DC, No integration with RES

2.2.3 Container-based energy-efficient techniques for DCs

CSPs are adopting energy-efficient techniques (such as VM consolidation, migration, and workload prediction) to handle the routine activities of DCs [106]. By adopting energy-efficient techniques, CC sector can witness manifold benefits such as reduction in energy consumption, operational cost, and carbon emissions. How-

ever, CSPs have to also simultaneously face a tough challenge of preserving SLA, QoS, and reliability of the cloud services. Therefore, to handle these issues, nowadays, a lightweight technology; *CoaaS* is being adopted to design energy-efficient cloud DCs [107].

CoaaS is an emerging technology in which the resources are kept inside a container from where different applications can access them as per their requirements. Generally, the resources in a VM are emulated over a hypervisor and a guest OS. However, *CoaaS* runs on a Docker engine rather than a hypervisor. Hence, they can also be utilized as an alternative to the hypervisor-based virtualization to execute time-critical applications. Container orchestrator, a run-time scheduler, is one of the components of container that handles various scheduling decisions as per service availability. In a *CoaaS* model, all the containers share single OS kernel with different levels of abstraction for all the applications. On contrary, the VMs require their own virtualized network, BIOS, CPU, and OS. Hence, PaaS and SaaS use containers instead of VMs or bare-metal systems [108].

Table 2.7: Comparison of VM and container technologies

Parameter	**Containers**	**VMs**
Virtualization type	OS-level virtualization	Completely virtualized
Quality assurance	High	Relatively low
User space	Separate	Common
Kernel state	Require booting	Operational
Start and stop time	Less than 50 ms	30-40 and 5-10 secs
Startup overhead	Low	Relatively high
Memory usage	Low	High
CPU usage	Low	High
Migration speed	High	Low
Cost (overheads)	Low	High
Guest OS	Not required	Multiple guest OS
Resource sharing	YES (Kernel instances, OS, bare file system)	NO (own BIOS, CPU, OS and disk storage.)
Tools	Docker, CoreOS-RKT, OpenVZ, Sandboxie	Virtual Box, Xen, VM-Ware
Standardization	Runtime and Image Format Specification v1.0: OCI	Well standardized system

Table 2.7 depict the comparison of containers and VMs. Containers are smaller in size as VMs, so they provide a quick startup with an improved performance, and better compatibility. These can run either on the host or inside a VM [108]. Moreover, container based virtualization can also support lightweight and energy-efficient migration within and outside VMs. This helps in live system updates using the underlying host OS thereby eliminating the requirement of hardware emulation. According to [109], containers migration consumes less energy with minimum resource wastage as compared to the entire VM. Moreover, containers use few number of OSs for higher level of application execution in contrast to the VMs [108]. After analysis of the above facts, it is evident that containers can provide increased performance, higher efficiency and lower energy dissipation through higher execution speed, quick startup and shared kernel instances. These benefits makes containers adequate for real-time applications. In other words, these light weight containers are better options than typical virtualization utilized in current cloud DCs.

Recently some research proposal have used CoaaS to minimize the energy consumption of DCs. For example, Kaur *et al.* [108] proposed an energy-efficient scheme using container migrations at edge devices. The authors presented a broker based resource allocation and management scheme using containers. Similarly, Piraghaj *et al.* [109] presented energy-efficient container consolidation scheme for cloud DC. The authors suggested that containers can act as an alternative to VMs for design of energy efficient DCs. Dai *et al.* [110] proposed a QoS-aware approach for implementing CoaaS model to provide energy efficiency in DCs. In an another work, Gutierrez-Garcia *et al.* [111] presented a load management technique using VM migrations to save energy. But, the authors highlighted that use of containers can be more beneficial and energy efficient.

In an another work, Piraghaj *et al.* [112] used CaaS for improving the energy-efficiency of servers using container consolidation in cloud DC. In an another work, Piraghaj *et al.* [113] proposed that the size of the VM for efficiently hosting of CaaS. The authors proposed an efficient techniques to design variable load-aware

VM design in which multiple containers can be deployed. These containers are used for execution of jobs of different sizes which increased the overall efficiency of any designed solution in this environment. The authors proposed efficient virtual machine sizes for hosting containers in such a way that the workload is executed with minimum wastage of resources on VM level. Moving a step ahead, Piraghaj *et al.* [114] proposed ContainerCloudSim for modeling and simulation of cloud environment using containers. Moreover, the authors designed an architecture for container simulation using different use cases. The authors suggested that containers performs well with respect to SLA adherence and energy efficiency. Hence, after analyzing the above discussed proposals, it is evident that containers can be useful to reduce energy consumption of DCs while ensuring SLAs. However, none of the above proposals have focused on the sustainability DCs using containers. After analysis of the above discussed proposals, Table 2.8 shows the summary of various aspects related to containers.

Table 2.8: CoaaS for DC energy efficiency

Proposal	**Focus area**	**Environment**	**VM/ Server**	**Energy**	**Performance**
Piraghaj *et al.* [112]	Container consolidation	Cloud DCs	VM	Yes	Yes
Piraghaj *et al.* [113]	VM sizing for hosting containers	Cloud DCs	VM	Yes	Yes
Piraghaj *et al.* [114]	Container CloudSim	Cloud DCs	VM	No	No
Kaur *et al.* [108]	Container migration	Fog DCs	VM	Yes	Yes
Piraghaj *et al.* [109]	Container consolidation	Cloud DCs	VM	Yes	Yes
Dai *et al.* [110]	CoaaS model	Cloud DCs	VM	Yes	Yes
Gutierrez-Garcia *et al.* [111]	CoaaS model	Cloud DCs	VM	Yes	Yes

2.2.4 Edge computing for energy efficiency in DCs

All the above works focused on the concept of VM migrations and placement, and resource consolidation to achieve energy efficiency along with ensuring QoS. However, none of the above discussed proposals explored the concept of edge-cloud interplay to minimize energy consumption and ensure high QoS.

In this direction, some of the recent proposals have used edge-cloud environment to handle the millions of end user requests. For example, Deng *et al.* [115] presented a workload allocation scheme in edge-cloud environment with a focus on energy minimization. Authors designed an optimization problem for trade-off between power consumption and delay. They stressed on deep analysis of cooperation between edge and cloud with respect to energy and QoS. In an another work, Jalali *et al.* [116] presented a comparative analysis of edge DCs and centralized DC. Authors highlighted different aspects related to energy consumption in edge-cloud environment. They suggested that provisioning the user requests closer to its premises can lead to lower latency and effective bandwidth utilization. But, they suggested that underlying network infrastructure has a major role to play in this direction. Moreover, effective network architecture and management can lead to maximum energy savings. In the above discussed proposals, it may be concluded that efficient cooperation between cloud and fog may result in energy saving and higher QoS in terms of latency and bandwidth consumption. Table 2.9 summarizes proposals related to edge computing for energy efficient DCs.

Table 2.9: Edge computing for energy-efficient DCs

Proposal	**Focus area**	**Description**
Deng *et al.* [115]	Workload allocation	Designed an optimization problem for trade-off between power consumption and delay
Jalali *et al.* [116]	Cooperation between edge and cloud	Suggested that provisioning the user requests closer to its premises can lead to lower latency and effective bandwidth utilization

2.3 Sustainable DCs

Various research proposals focusing partially or completely towards the design of sustainable DCs are discussed as below.

2.3.1 Renewable energy-based techniques for DCs

Above discussed proposals focused on energy minimization or cost effectiveness. However, none of them have used the RES for making DCs self-sustainable. Some of the existing proposals which utilized the RES in DC are discussed as follows. For example, Paul *et al.* [21] proposed a demand response in DCs using an energy-efficient scheduling. The authors have categorized the user requests for resources as real-time and deferrable. They have offered incentives as price for deferrable requests. The authors considered a DC where various diverse services are deployed in three tiers with focus on capitalizing deferrable jobs. They have also proposed a mechanism to integrate RES into power grid. They concluded that each server consumes about 60 % of energy even when it is idle. In a similar work, Kaewpuang *et al.* [117] proposed a cooperative game for virtual machine management to reduce the total operational cost of DCs by fair resource allocation scheme for VM management using RES. The authors considered uncertainty in electricity price, user demand along with RES to formulate a coalitional game.

In an another work, Nguyen *et al.* [118] developed a green testbed for powering a DC through RES. The authors designed a GreenStar network testbed using a wide area network of renewable-energy powered DCs. Authors presented different aspects related to green ICT such as-virtualization, unified computation management, carbon assessment and power control. In an another work, Erol-Kantarci and Mouftah [33] suggested that efficient resource scheduling by cloud operators and RES helps to shift the load of massive power consumers from grid and thereby, reducing harmful carbon emissions. The authors also suggested that utilization of RES to power DCs could be an effective solution to cost minimization of DCs while

reducing harmful carbon emissions. In this context, Yu *et al.* [119] designed an optimization technique for energy management system to deal with power outages using RES, backup generators, and battery management.The authors utilized RES and power generators to design an optimization scheme for energy management of DCs in order to handle the situations of power outages.

Similarly, Paul *et al.* [120] also proposed an energy-efficiency scheme for load distribution and energy price control using RES. Chen *et al.* [121] proposed a scheme for operational cost minimization in DCs using renewables. Wang *et al.* [122] formulated an energy-efficient game-theoretic scheme between cloud and DC controllers using dynamic pricing of SG with integration of renewables. In an another work, Sharifi *et al.* [123] presented cloud of energy paradigm that exploits green energy and cloud. Authors designed an everything as a service framework to handle service exchange for computing and power grid through an economic middleware. Similarly, Nguyen and Cheriet [124] presented an environment-aware virtual slicing scheme to achieve energy efficiency in green cloud environment. The proposed scheme deals with the intermittent RES by optimizing flow management to VMs in DCs as per actual traffic conditions, VM location, network capacity, and available renewable energy. Moreover, a VM assignment scheme is presented to minimize network footprints and improve performance. Khosrovi *et al.* [125], proposed a novel VM placement algorithm by considering different carbon footprint rates of various energy sources and their power usage effectiveness to provide environment sustainability.

Table 2.11 shows an analysis of RES-based schemes with respect to various parameters. Aforementioned works considered RES as a secondary source of energy to handle the power consumption of DCs. But, none of these solutions tried to sustain the energy demand of DCs with RES. Due to intermittent nature of RES, all the above discussed proposals utilized RES as secondary source of energy to power DCs. However, none of the above discussed proposals utilized RES as primary energy source to sustain energy consumption of DCs.

2.3.2 Techniques to handle intermittent nature of RES

All the above discussed works have used grid as a primary source of energy. Due to intermittent nature, none of the authors used RES as a primary source of energy. But, none of these proposals have focused on sustainability of DCs using RES. All these proposals have used renewable energy as a secondary source to power DCs due to the intermittent nature of RES. All the above discussed research proposals utilized RES as an additional source of energy for DCs. However, these proposals have not focused on complete sustainability of DCs using RES. Aforementioned proposals have used utility grid as a major source of energy to power DCs rather than RES. The intermittent nature of RES is one of the biggest challenges for sustaining the energy consumption of DCs [126, 127].

To address this issue, Chen et al. [128] proposed an unified management approach to deal with the intermittent nature of RES. In a similar work, Gou *et al.* [129] utilized the thermal storage to manage the intermittent nature of RES. However, none of the proposals tried to tackle the intermittent nature of RES using electric vehicles (EVs). However, with penetration of EVs, the energy consumption of DCs may be effectively managed. In this context, Milano and Hersent [130] proposed a scheme for an optimal load management of grid with an inclusion of EVs. They proposed optimization scheme for demand management while preserving the security of system. In similar direction, Kaur *et al.* [131] proposed a charging-discharging scheme to manage the load of grid using EVs. The proposed scheme regulates the charging and discharging activities of EVs using day-ahead load curves obtained using load forecasting schemes.

In another work, Kaur *et al.* [9] proposed an aggregator-based load management by using a feet of EVs for efficient frequency support. The authors stressed that EVs can play a major role in reduction the frequency fluctuations. In a similar work, Rana et al. [132] proposed a scheme to utilize EVs for providing frequency support to grid. Druitt *et al.* [133] investigated the role of EVs for variable generation sources

such as wind power. The authors proposed a model for EVs to provide flexible demand management for dealing with intermittent nature of wind power. Further, Caramanis *et al.* [134] managed EVs load to cope with the indeterminacy of RES as well as challenges of distributed generation. In similar direction,Freire *et al.* [135] proposed a scheme for load balancing of grid using RES and EVs. Further, Saber *et al.* [136] presented a scheme for maximum utilization of EVs and RES for cost and emission reductions. The authors suggested that the penetration of EVs in order to tackle the intermittency of RES could be beneficial and economical. Hence, after analyzing the aforementioned research proposals, it is evident that EVs could be utilized to manage the intermittent nature of RES effectively. Such an integration would not only be environment friendly and a cost effective solution for DCs. Table 2.10 summarizes various ways for managing intermittency of RES.

Table 2.10: Techniques for handling intermittency of RES

Author	**Description of work**
Chen et al. [128]	Unified management approach to deal with the intermittent nature of RES
Gou *et al.* [129]	Utilized the thermal storage to manage the intermittent nature of RES
Milano and Hersent [130]	Optimal load management of grid with an inclusion of EVs
Kaur *et al.* [131]	Charging-discharging scheme to manage the load of grid using EVs using day-ahead load curves obtained using load forecasting schemes
Kaur *et al.* [9]	An aggregator-based load management by using a feet of EVs for efficient frequency support
Druitt *et al.* [133]	Proposed a model for EVs to provide flexible demand management for dealing with intermittent nature of wind power
Caramanis *et al.* [134]	Managed EVs load to cope with the indeterminacy of RES as well as challenges of distributed generation
Delgado *et al.* [135]	Integrated RES with charging strategies of EVs to optimize the load balancing of grid
Sabar *et al.* [136]	Presented a scheme for maximum utilization of EVs and RES for cost and emission reductions. The authors suggested that the penetration of EVs in any environment in order to tackle the intermittency of RES could be beneficial and economical

2.3.3 Techniques for sustainable DCs

All the above discussed research proposals utilized RES as an additional source of energy for DCs. However, these proposals have not focused on sustainability of DCs using RES. Aforementioned proposals have used utility grid as a major source of energy to power DCs. The intermittent and variable nature of renewable energy poses a challenge to manage the energy consumption of DCs. However, some of the recent proposals have presented various techniques for sustainable DCs. In this direction, Guo *et al.* [137] proposed an energy and network aware workload management scheme for sustainable DCs using thermal storage. The authors targeted the opportunities offered by the geographical load balancing and opportunistic scheduling of delay-tolerant workloads using thermal storage to manage the intermittency of RES. Authors exploited thermal storage to facilitate integration of green energy in order to reduce the usage of brown energy. In this regard, a stochastic problem is formulated followed by Lyapunov optimization based on-line algorithm to solve it. Patil and Duttagupta [138], discussed that a hybrid power generation system can act as a platform for sustainable DCs using solar energy and fuel cells. In an another work, Chen *et al.* [139], presented a unified management approach in which DCs can respond dynamically to the various situations with respect to renewable availability, workload shifting, and energy price fluctuations in order to provide long-term QoS.

In a similar work, Chen *et al.* [140] proposed a workload and energy management scheme for sustainability of DCs. The authors highlighted the need of energy efficient and sustainable DCs to tackle the growing energy demand of the massive data processing infrastructure. The authors presented a framework for integration of RES, distributed storage units, and colling facilities in DCs. They presented an optimization problem for resource allocation to manage with the intermittency of RES. However, some of the major CSPs have shifted their focus from the mere usage of renewables towards sustainability of DCs. Polverini *et al.* [141] proposed a thermal-aware scheduling of batch jobs in geo-distributed DCs. Moving a step

ahead, some of the recent proposals have proposed different methods for designing 100 percent sustainable DCs.

Gu *et al.* [142] proposed an optimization scheme to design green DCs with an aim of cost and carbon reductions. For this purpose, the authors utilized sustainable energy along with exploiting time and location varying electricity pricing. In an another work, Callou *et al.* [143] presented an evaluation for sustainability and dependability of DC architectures. The major focus of the authors was redundant infrastructure, sustainability growth and reduction in the operational cots of DCs. He *et al.* [144] proposed a socially responsible algorithm for scheduling of loads over smart grid for sustainable DCs. The authors designed an optimization problem for electricity, performance and social costs. The authors utilized dynamic pricing, renewable energy sources and latency.

In an another work, Chen *et al.* [145] presented a load balancing scheme for design of sustainable DCs located at different geographical locations. The authors used renewable energy, colling infrastructure and distributed storage system to achieve sustainability. Moreover, the authors also focused on energy and workload management along with the utilization of the dynamic pricing to achieve the ultimate goal of sustainable DCs. Moving ahead in the similar direction, Abbasi *et al.* [146] designed an holistic approach for carbon, energy and cost optimization. The authors integrated the Lyapunov optimization with predictive solution for achieving an optimal trade off among energy costs and carbon footprints. The proposed solution suggested that the predicted parameters are beneficial for efficient energy management of DCs. In an another work, Klien *et al.* [147] designed sustainable DCs using a three fold mechanism including, 1) integration with RES to reduce carbon footprints, 2) workload consolidation and optimization for improved energy efficiency, and 3) use of multi-functional sensor for effective management of cooling facilities and other infrastructure. In this work, the authors highlighted the important insights towards the sustainable path for DCs. In an another work, Tossi *et al.* [148] proposed a geographical load balancing scheme for sustainable DCs using RES.

Table 2.11: Comparison analysis of existing techniques for sustainable DCs

Existing techniques	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Kaewpuang *et al.* [117]	✓	S	Partial	✓	×	✓	×	×	×	×	×
Nguyen *et al.* [118]	✓	S	Partial	×	Battery	×	×	✓	✓	×	✓
Erol-Kantarci and Mouftah [33]	✓	RES	Survey	✓	✓	✓	✓	✓	✓	✓	✓
Yu *et al.* [119]	✓	W	Partial	✓	Storage	✓	✓	✓	×	×	×
Paul *et al.* [120]	✓	S/W	Partial	✓	–	✓	✓	✓	×	×	✓
Chen *et al.* [121]	×	×	Partial	✓	✓	×	×	×	×	×	×
Wang *et al.* [122]	✓	RES	–	✓	Partial	✓	✓	×	×	×	×
Sharifi *et al.* [123]	×	✓	Partial	✓	✓	×	×	×	×	×	✓
Nguyen and Cheriet [124]	✓	S	Partial	×	Battery	✓	✓	✓	✓	✓	✓
Khosrovi *et al.* [125]	✓	S/W/G	Partial	×	Storage	×	✓	×	×	✓	✓
Guo *et al.* [137]	✓	S/W	Full	✓	Thermal	✓	✓	✓	✓	×	✓
Patil and Duttagupta [138]	×	S/F	RES	×	ESU	×	×	×	×	×	✓
Chen *et al.* [139]	✓	RES	RES	✓	Storage	✓	✓	✓	×	×	✓
Chen *et al.* [140]	✓	S/W/G/Gen	Full	✓	ESU	×	✓	✓	✓	×	✓
Polverini *et al.* [141]	✓	×	Thermal	✓	×	✓	✓	×	×	×	×
Gu *et al.* [142]	✓	S/W/G	RES	✓	ESU	✓	✓	✓	×	Delay	✓
Callou *et al.* [143]	×	×	Chillers	×	×	✓	×	✓	×	×	✓
He *et al.* [144]	✓	S/W	RES	✓	×	✓	✓	✓	✓	✓	×
Chen *et al.* [145]	✓	RES	RES	✓	×	Battery	✓	✓	✓	✓	×
Abbasi *et al.* [146]	✓	S/W/G	RES	✓	ESD	✓	✓	×	×	×	✓
Klien *et al.* [147]	×	S	RES/ Cooling	×	Thermal	✓	✓	×	×	×	✓

Note- 1: Geo-distributed DCs, 2: RES, 3: Sustainable DCs, 4: Dynamic pricing, 5: Storage, 6: Cost minimization, 7: Workload management, 8: Energy management, 9: Network management, 10: QoS, 11: Carbon reduction

Notations-✓: considered, ×: not-considered, –: No information provided, S: Solar energy, W: Wind energy, G: Grid, F: Fuel cell, Gen: Generator, ESU: Energy storage unit, and ESD: Energy storage device.

2.4 SDN-based Energy Management of DCs

As already discussed, inter-DC and intra-DC communication links and network infrastructure consumes a large amount of energy. Apart from energy, DC networks play an important role in the success of the cloud computing architecture [149]. Therefore, various researchers have utilized SDN for energy efficient and QoS adhered DC networks. The flexible and dynamic nature of SDN makes it possible to design network policies and flow management according to the requirements. Moreover, Vishwanath *et al.* [70] highlighted the fact that network devices and efficient network has a huge impact on energy consumption of DCs. In SDN-based systems, the control flow management can be programmed according to the user's requirements [150]. But, in traditional networks, the control flow is fixed and cannot be programmed. Due to this reason, underutilized switches and links consume huge amount of energy. But, energy consumption of such networks can be minimized by managing the control flow [151].

In this direction, various existing proposals have highlighted the benefits of SDN with respect to DCs. For example, Tu *et al.* [152] proposed an energy saving model for DCs using SDN. The authors presented two algorithms for the proposed energy saving model. The authors suggested that traffic preprocessing and centralized control can provide energy efficiency and thereby save a large amount of energy related cost. In an another work, Huang *et al.* [129] proposed a Green DataPath architecture by introducing a dynamic voltage and frequency scaling technique in software-defined networks. Authors also presented an energy-efficient routing algorithm in SDN. Similarly, Xu *et al.* [153] proposed a bandwidth-aware routing scheme to achieve energy efficiency in SDN-based DC networks. In a another work, Xu *et al.* [154] presented a bandwidth-aware flow scheduling scheme for energy-efficiency and optimal bandwidth utilization using SDN in DC networks. Authors highlighted the role of SDN in network management services such as- routing, flow management, and flow consolidation to design deterministic, scalable, and energy-efficient DCs.

Similarly, Wang *et al.* [155] presented a VM placement scheme for hop reduction and energy savings in SDN-based DC networks. So, after analyzing above aspects related to the role of SDN in DC networks, it is concluded that SDN can be a viable solution to handle the streaming traffic in DC networks to provide energy efficiency and optimal bandwidth utilization. In an another work, Son *et al.* [156] proposed an SLA aware dynamic overbooking approach to design energy efficient DCs using SDN. The authors presented a dynamic overbooking scheme for traffic consolidation using virtualization capabilities of SDN. In the similar direction, Habibi *et al.* [157] proposed an energy efficient VM placement and QoS aware routing scheme for DC networks using SDN. In an another work, Borylo *et al.* [158] designed an anycast based routing scheme for the reduction of carbon footprints related to DC networks.

In a different direction, Borylo *et al.* [159] proposed a dynamic resource allocation scheme with respected to energy-aware edge-cloud interplay. Authors suggested that SDN based edge-cloud interplay can lead to maximum energy conservation in DC networks. Authors utilized SDN for effective network management in edge-cloud interplay. They used SDN to design a latency-aware scheme to handle traffic routed to fog devices. However, the above proposal has not focused on energy management of DCs. Apart from energy efficiency, SDN has been successfully deployed in DC networks for different objectives such as bandwidth utilization and latency provisioning. For example, Tu *et al.* [160] stressed that SDN can be a viable solution to handle the latency and bandwidth issues in communication-sensitive DCs. But, the authors did not focus on the energy aspects of DCs. In an another work, Sadasivarao *et al.* [161] presented a use case for transport SDN for handling data between DCs. The authors proposed an SDN based architecture for optical transport. In this work, the authors programmable virtual switch using openflow protocol. Further, Truncer *et al.* [162] suggested that SDN can handle heterogeneous applications, equipments and controls effectively. The authors suggested that effective centralized control and resource management using SDN can improve the performance of any environment. Table 2.12 summarizes various SDN-based energy management schemes.

Table 2.12: Comparison analysis of existing techniques for SDN-enabled DCs

Existing techniques	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Wang *et al.* [149]	Survey on SDN in cloud computing	✓	✓	✓	×	×	✓	✓	✓
Kachris *et al.* [150]	Optical network for DC	×	×	×	×	✓	×	×	×
Son *et al.* [151]	Survey on DC networks	✓	×	×	×	×	✓	✓	✓
Tu *et al.* [152]	0-1 Integer programming model	✓	✓	×	×	✓	✓	×	×
Xu *et al.* [153]	Linear programming	✓	×	✓	×	✓		×	×
Xu *et al.* [154]	Linear programming	✓	×	✓	×	✓	✓	×	✓
Wang *et al.* [155]	Energy efficient routing and QoS aware VM placement	✓	×	×	×	✓	×	✓	
Son *et al.* [156]	Linear power model	✓	✓	×	✓	✓	×	✓	✓
Habibi *et al.* [157]	Routing and energy efficient VM placement	✓	✓	✓	×	✓	×	✓	×
Borylo *et al.* [158]	Routing and wavelength assignment	✓	✓	×	×	×	×	×	✓
Borylo *et al.* [159]	Latency aware algorithm	✓	×	×	×	✓	×	×	×
Tu *et al.* [160]	Integer linear programming	×	✓	×	×	✓	×	×	×
Sadasivarao *et al.* [161]	Optical transport DC network	×	×	✓	×	×	×	×	×
Truncer *et al.* [162]	Controller placement algorithm	×	✓	×	×	×	×	×	✓

Note- 1: Technique, 2: Energy, 3: Load balancing, 4: Flow management, 5: Flow consolidation, 6: Latency, 7: Traffic engineering, 8: Virtualization, 9: Resource management

Notations-✓: considered, and ×: not-considered.

Table 2.13: Comparison analysis of existing techniques and proposed work

Existing techniques	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Chen *et al.* [140]	✓	×	×	×	×	×	×	✓	✓	×	×	×	×	×	×
Quereshi *et al.* [62]	×	×	✓	✓	×	×	✓	×	×	×	×	×	×	×	×
Li *et al.* [86]	✓	✓	✓	×	✓	×	✓	×	×	×	×	×	×	×	×
Xu *et al.* [154]	×	×	✓	✓	✓	✓	×	×	×	×	✓	×	×	×	×
Guo *et al.* [137]	✓	×	×	×	×	×	×	✓	✓	×	×	×	×	×	×
Qiu *et al.* [90]	✓	✓	×	×	✓	×	×	×	×	×	×	×	×	×	×
Tran *et al.* [89]	✓	✓	×	×	✓	×	✓	×	×	×	×	×	✓	×	×
Polverini *et al.* [141]	✓	✓	×	✓	✓	×	✓	×	×	×	×	×	×	×	×
Paul *et al.* [21]	×	×	✓	✓	✓	×	✓	✓	×	×	×	×	×	×	×
Chen *et al.* [121]	×	×	✓	✓	✓	×	×	×	×	×	×	×	×	×	×
Kaewpuang *et al.* [117]	×	✓	✓	✓	✓	✓	×	✓	✓	×	×	×	×	×	×
Rao *et al.* [85]	✓	×	×	×	✓	×	✓	×	×	×	×	×	×	×	×
Narayan *et al.* [2]	×	✓	×	✓	×	×	✓	×	×	×	×	×	×	×	×
Wang *et al.* [163]	✓	×	✓	×	×	×	✓	✓	×	×	×	×	✓	×	×
Piraghaj *et al.* [112]	×	✓	✓	✓	×	×	×	✓	×	×	×	×	×	×	✓
Piraghaj *et al.* [113]	×	✓	✓	✓	×	×	×	×	×	×	×	×	×	×	✓
Piraghaj *et al.* [114]	×	✓	✓	✓	×	×	×	×	×	×	×	×	×	×	✓
Son *et al.* [156]	×	✓	✓	×	×	✓	×	×	×	×	×	×	×	×	×
Paul *et al.* [120]	✓	✓	✓	×	✓	×	✓	✓	×	×	×	×	×	×	×
Buyya res *et al.* [148]	×	✓	✓	✓	×	×	✓	✓	✓	×	×	×	×	×	×
Beloglazov *et al.* [164]	×	✓	✓	✓	×	×	×	×	×	×	×	×	×	×	×
Proposed work	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Note- 1: Geo-distributed DCs, 2: Heterogeneous servers, 3: Energy efficiency, 4: Resource utilization, 5: Cost savings, 6: SDN, 7: SG Dynamic pricing, 8: RES, 9: Sustainability, 10: Edge computing, 11: Flow management, 12: Flow consolidation, 13: Stackelberg game, 14: EVs, 15: CoaaS

Notations-✓: considered, and ×: not-considered

2.5 Research Gaps

After the detailed analysis of the existing proposals (Table 2.13) with respect to current status of the DCs, the following research gaps have been identified.

- **Vision of green DCs is moving at a slow pace:** With an increase in the harmful emissions globally, the vision of green DCs is still in infancy. The dream of zero emission DCs is moving at a slow pace. Lot of studies and projects are going on world-wide but still a lot need to be done.

- **Focus on reducing operational cost rather than making DCs self-sustainable:** The focus of majority of the existing studies was only on the cost minimization of DCs rather than making DCs the self-sustainable. None of the existing works attempted to ease the load on grid.

- **No focus on use of RES as primary source of energy to power DCs:** Existing studies utilized RES as secondary source of energy. None of the work tried to propose an energy management scheme to sustain DCs using RES.

- **Inefficient utilization of resources and servers in DCs:** Even after vast research in cloud computing worldwide, the over provisioning of resources is a common practice and there are various DCs which are under-provisioned. This leads to an inefficient utilization of resources which increases energy consumption and operational cost of DCs.

- **QoS suffers while minimization of energy consumption:** While developing energy efficient DCs, the QoS parameters such as, response time, availability and latency have to be been ignored or compromised in majority of existing proposals. However, with increasing dependence of end users on CC, it becomes necessary to provide QoS guarantee.

- **No optimal trade-off scheme for energy minimization and QoS:** The existing proposals fail to achieve an optimal trade off between energy minimiza-

tion and QoS. The problem of optimal trade off between energy minimization and QoS guarantee still remains unaddressed.

- **Heterogeneous nature of servers and resources not fully addressed:** Most of the existing proposals considered homogeneous nature of physical servers and resources apart from lacking in providing global QoS. But, this may not be true for each and every DC such as, National DCs which allow different servers configurations apart from allowing addition of servers if required to provide services. Hence, it implies that minimum number of active servers do not guarantee minimum consumption of energy.

- **Limited utilization of latest technologies**. The existing proposals have either not utilized or made limited use of latest technologies for the design of energy-efficient DCs or sustainable DCs. Such technologies include, SDN, network function virtualization, edge computing, and CoaaS. Some of the recent proposals have utilized SDN and CoaaS but not to great extent. However, none of the proposals have considered SDN, edge computing and CoaaS for designing sustainable DCs.

2.6 Objectives

To address the aforementioned research gaps, the following objectives were proposed:

- To study and analyze various existing data centre energy management schemes for sustainability.

- To analyze the impact of renewables as an alternative source of energy in DCs.

- To design an efficient framework for sustainability of data centres.

- To validate the proposed solutions by selecting various performance evaluation metrics in different scenarios.

2.7 Thesis Contributions

To achieve the aforementioned objectives, following contributions are presented in the thesis which are arranged in different chapters.

- In this thesis, a renewable energy based framework for sustainability of DCs comprising of four different techniques is designed. All the four techniques are given as below.
  - *An Electric Vehicle based Energy Management Scheme:* An SDN-based energy management scheme for sustainability of DCs is proposed using RES. To achieve the aforementioned objectives, an energy-efficient flow scheduling algorithm is proposed using SDN. Moreover, a charging-discharging scheme for penetration of electric vehicles is also proposed to manage the intermittency of renewable energy. Finally, an energy trading and reward point scheme is designed to attract the electric vehicles to participate in the proposed energy management scheme.
  - *An Edge Computing based Energy Management Scheme:* An efficient scheme for energy management with sustainability of cloud data centers in edge-cloud environment using SDN is presented. In the proposed scheme, a support vector machine-based workload classification approach is presented. Moreover, a two-stage game for workload scheduling for sustainability of DCs is designed. In order to achieve energy efficiency and optimal utilization of network and computing resources, different consolidation schemes are also proposed.
  - *Renewable Energy Aware Resource Allocation Technique:* A renewable energy-aware resource allocation scheme is proposed using a Stackelberg game for energy management in cloud-based DCs. For this purpose, a cloud controller is used to receive the requests of users which then distributes these requests among geo-distributed DCs in such a way that

the energy consumption of DCs is sustained by RES. However, if energy consumption of DCs is not sustained by RES then the energy is drawn from the grid. The requests of users are routed to the DC which is offered lowest energy tariff from the grid. For this purpose, a Stackelberg game for energy trading is also proposed to select the grid offering lowest energy tariff to DCs.

– *Multi-indexed Classification and Scheduling Scheme:* A renewable energy-aware multi-indexed job classification and scheduling scheme using CoaaS is proposed for DC sustainability. In the proposed scheme, incoming workloads from different devices are transferred to the DC which has sufficient amount of renewable energy available with it. For this purpose, a renewable energy-based host selection and container consolidation scheme is also designed.

- The proposed DC sustainability framework has been implemented using MATLAB and CloudReports in comparison to other schemes of its category.

- The proposed techniques have been evaluated using various benchmark datasets.

  1. *Technique 1* is evaluated using interactive workload traces rescaled from real traffic from Wikipedia [140]. The weather traces, i.e., solar radiations [165] and wind speed [166] are used. Several performance metrics such as–energy consumption, revenue generated, reward points gained, etc are used to evaluate the effectiveness of the proposed scheme.

  2. *Technique 2* is tested using a workload trace of 500 jobs extracted from Google workload traces [167]. The realistic weather traces are extracted from [165] [166] The performance evaluation metrics such as–energy consumption, energy cost, energy savings, SLA violations, migration delay, migration cost, etc.

  3. *Technique 3* is evaluated using Google workload traces [167] and realis-

tic weather traces, i.e., solar radiations [165] and wind speed [166]. To evaluate the effectiveness of the proposed technique, several performance evaluation criterion such as–energy consumption, energy savings, profit gained, SLA violations, response time, etc are employed.

4. *Technique 4* is evaluated using 200 heterogeneous jobs from Google workload traces [167]. The two different renewable energy traces from [129] and [165] [166] are used. For testing the effectiveness of the proposed scheme, performance evaluation metrics such as–energy consumption, SLA violations, container migration rate, overhead are employed.

Summary of the Chapter: The Chapter 2 highlights the existing proposals related to the energy consumption of DCs. The existing proposals are classified into various categories and each of these categories are elaborated for better understanding of the problem. The research findings in form of gaps are also presented in this chapter followed by the the research objectives of the thesis. This chapter presents the discussion of the existing proposals in the similar research area, which acted as a backbone for the design of various techniques in this thesis.

Chapter 3

EV-based Energy Management Scheme for Sustainable DCs

In this chapter, an EV-based energy management scheme for sustainability of DCs using RES is proposed[1]. Moreover, an energy-efficient flow scheduling algorithm is proposed using SDN. In order to manage the intermittency of renewable energy, a charging-discharging scheme for penetration of EVs is also presented. Finally, an energy trading and reward point scheme is designed to attract the EVs to participate in the proposed energy management scheme.

3.1 System Model

Fig. 3.1 shows the system model comprising of a typical DC connected to an ESU and three different sources of energy. A typical DC consist of servers, memory,

[1]The content of this chapter is taken from:

- G. S. Aujla and N. Kumar, "SDN-based energy management scheme for sustainability of data centers: An analysis on renewable energy sources and electric vehicles participation", Journal of Parallel and Distributed Computing, Vol. 117, July 2018, pp. 228-245.
- G. S. Aujla, A. Jindal, N. Kumar and M. Singh, "SDN-Based Data Center Energy Management System Using RES and Electric Vehicles," 2016 IEEE Global Communications Conference (GLOBECOM), Washington, DC, 2016, pp. 1-6.
- G. S. Aujla, N. Kumar, M. Singh and A. Y. Zomaya, "Energy Trading with Dynamic Pricing for Electric Vehicles in a Smart City Environment", Journal of Parallel and Distributed Computing, 2018.

storage, and network devices which are utilized to accomplish the routine jobs based on end user requests. The DC consumes large amount of energy while accomplishing these routine jobs based on end user requests. Generally, a DC is connected to a grid to meet its energy demand. However, in the proposed scheme, the DC is connected to three sources of energy to manage its power demand. The three energy sources and ESU that are connected to DC are discussed in the subsequent sub-sections.

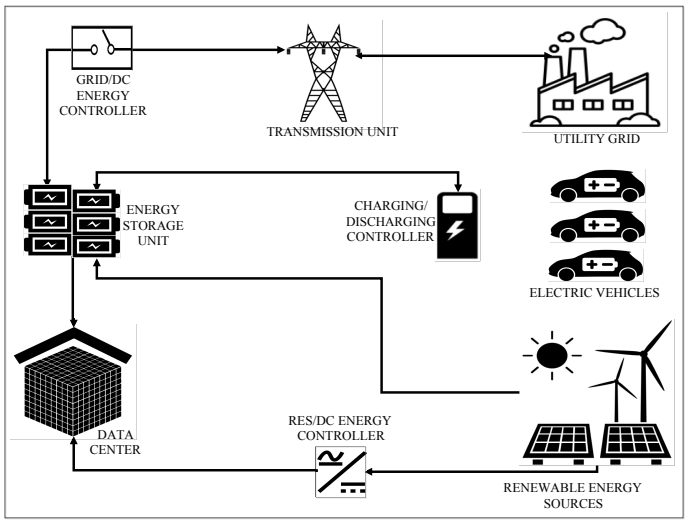


Figure 3.1: System model

3.1.1 Renewable energy sources

Here, the primary source of energy is RES which consist of solar and wind energy. Solar and wind energy are one of the most popular RES which could be deployed in-house for energy management. However, the proposed scheme for energy management of DCs is extensible to other RES also. The energy generated (E_{dc}^{ren}) by RES is utilized by DC directly. However, when the energy generation by RES is higher than the consumption of DC then it is stored in ESU connected to DC. The energy generated (E_{dc}^{ren}) by RES is given as below.

$$E_{dc}^{ren} = E_{dc}^{pv} + E_{dc}^{wn} \tag{3.1}$$

where, E_{dc}^{pv} and E_{dc}^{wn} are the energy generated by photovoltaics (PV) panels and wind turbines.

The two most popular RES used in the proposed work are discussed as below.

- **Solar energy:** The most popular alternative for traditional non-renewable energy sources is Solar energy. The PV panels are used to capture radiant heat and convert it into solar energy. However, solar energy generated by PV panels is variable and depends on various factors such as solar radiations (ɾ), conversion efficiency of panel (η), radiation angle of Sunlight ($cos\ \alpha$), size of panel (S_{dc}^{pv}), and temperature exedence loss (L_{pv}^{temp}). The ratio of energy received from solar radiations and amount of energy generated is known as conversion efficiency. The radiant angle of solar radiations on PV panel depends on the angular deviation (α) of sunlight. The temperature exedence loss occurs due to cold weather. The energy (E_{dc}^{pv}) generated by capturing Sunlight using PV panels is given as below [168].

$$E_{dc}^{pv} = \left(1 - L_{pv}^{temp}\right)\ \eta\ S_{dc}^{pv}\ cos(\alpha)\ \text{ɾ} \tag{3.2}$$

- **Wind energy:** Apart from solar energy, wind is one of the widely used renewable source of energy which use wind turbines to generate wind energy. The energy (E_{dc}^{wn}) generated by a typical wind turbine is given as below [169].

$$E_{dc}^{wn} = \frac{1}{2}\ \left[\text{Ç}\ \rho\ \hat{a}\ \nu^3\right]\ L_{wn}^{temp} \tag{3.3}$$

where, Ç is the rotar efficiency, ρ is the air density, $\hat{a}$ is the area swept by rotar blades, ν is the wind speed, and L_{pv}^{temp} is the temperature exedence loss.

3.1.2 Electric Vehicles

The secondary source of energy are i EVs which can charge/discharge the energy through a charging station (CS) located at DC. This CS also act as charging/discharging controller to manage the inflow/outflow of energy to/from EVs.

The energy stored in i^{th} EVs battery is given as below.

$$E_i^{avl} = V \ E_i^{rat} \ SoC_i^{inl} \tag{3.4}$$

where, V is the voltage, E_i^{rat} is the rated capacity of i^th EVs battery, and SoC_i^{inl} is the initial state of charge (SoC) of i^th EVs battery.

The energy available (E_{dc}^{avl}) with all the EVs connected to DC is given as below.

$$E_{dc}^{avl} = \sum_{i=0}^{n} \left(V \ E_i^{rat} \ SoC_i^{prs} \right) \tag{3.5}$$

3.1.3 Utility grid

The utility grid acts as last energy source and is connected to DC with a switch managed by a controller. The supply of energy from grid to DC is switched ON by the controller when there is deficit in energy supply from RES and EVs. The energy is supplied from DC to grid when it has excess energy supply from RES and EV which cannot be stored in ESU. The utility grid collects the excess energy from DC to ease the load of its own customers. However, it have to return back the amount of energy it has been supplied along with an advance credit of energy if required by DC. Such an advance energy received from grid is supplied back to it when DC has excess of energy available with it. The energy (E_{dc}^{grid}) available at utility grid that can be drawn to power the DC is given as below.

$$E_{dc}^{grid} = E_{dc}^{drawn} - E_{dc}^{sup} \tag{3.6}$$

where, E_{dc}^{drawn} is the energy drawn from DC and E_{dc}^{sup} is the energy supplied to DC.

3.1.4 Energy storage unit

The RES are intermittent in nature due to variability of sunshine in a day and wind speed [127]. Hence, at some period of time, the energy generated by RES

may not meet the energy consumption of DC. However, at other period of time, the energy generated by RES could be excess with respect to energy demand of DC. Hence, at such times, the excess energy is stored in ESU connected to DC. Now, to effectively map the excess/deficit of energy generated by RES with energy demand of DC, an RES/DC energy controller is used. Further, there may be a situation when the energy demand of DC is not meet by energy stored in ESU also. To deal with such situation, energy stored in batteries of EVs is utilized by DC. In such case, the charging/discharging of energy from/to EVs in ESU is controlled by a charging/discharging controller located at CS.

Finally, if the energy generated by RES and energy charged from EVs is excess with respect to energy demand of DC and storage capacity of ESU, then the excess energy could be supplied to utility grid from ESU. For this purpose a DC/grid energy controller is used. To manage all the situations of excess and deficit of energy from various sources and to control the inflow/outflow of energy from/to various sources, ESU has an important role to play. Moreover, due to real-time execution of various jobs at DC, the continuous power connectivity is essential. ESU consists of various batteries that are modeled similar to [170]. The energy stored in an ESU depends on the terminal voltage (V_t) and capacity (E_{bat}) of the battery and is given as below.

$$E_{cap}^{esu} = V_t E_{bat} SoC_{ch} \tag{3.7}$$

where. E_{cap}^{esu} is the energy stored in the battery of ESU, and SoC_{ch} is the SoC required for charging of a battery.

The energy stored in the battery is dependent on the number of cycles used. As the number of cycles increases, the energy capacity of battery decreases. So, the efficiency of the battery (δ) depends on the battery cycles and is given as below [170].

$$\delta = d \times \frac{N_t - N_c}{N_t} \tag{3.8}$$

where, N_c is the number of full cycles, N_t depict the life cycle of a battery, and d is the constant parameter for battery degradation.

However, a self-discharge loss which is almost 1-5 % per hour occurs at ESU [171]. So, the energy stored (E_{dc}^{esu}) in ESU at any instant is given as below.

$$E_{dc}^{esu}(t) = (E_{dc}^{esu}(t-1) + E_{dc}^{ren} \mp E_{dc}^{ev} - E_{dc}^{cons})\, L_{dc}^{esu} \tag{3.9}$$

where, L_{dc}^{esu} is the self-discharge rate and E_{dc}^{cons} is the energy consumed from ESU.

3.2 Problem Formulation

A typical DC consisting of k servers consumes some energy (E_{dc}) to accomplish its routine job execution. The energy consumption of DC depends directly on the amount of utilization of servers. The energy consumed (E_{dc}^{k}) by k^{th} server of a DC is given below [121].

$$E_{dc}^{k} = E_{idl}^{k} + (E_{mx}^{k} - E_{idl}^{k})\, U_{dc}^{k} \tag{3.10}$$

where, E_{idl}^{k} , E_{mx}^{k}, U_{dc}^{k} denotes the idle energy consumed, maximum energy consumed, and utilization level of k^{th} server of DC, respectively.

The utilization level of k^{th} server of a DC is given as below.

$$U_{dc}^{k} = \left(\frac{R_{t}^{k}}{R_{mx}^{k}}\right) \times 100 \tag{3.11}$$

where, R_{t}^{k} is the status of resource utilization at time t and R_{mx}^{k} is the capacity of resources at k^{th} server of DC.

Apart from servers, the second most energy consuming infrastructure based on computing are network devices. The network devices consisting of switch and ports consume energy with respect to the traffic load. However, the network devices consumes energy in two parts, i.e., fixed part (energy consumed by working switch components such as fan, chassis, and switching fabric) and dynamic part (energy consumed by working ports). The total energy consumption of network devices in

a DC is given as below.

$$E_{dc}^{n} = E_{sw}^{n} + E_{port}^{n} \tag{3.12}$$

where, E_{sw}^{n}, and E_{port}^{n} is the energy consumed by network switches and ports, respectively in a DC.

The energy consumed by the network infrastructure in a DC depends upon the working time of the network devices.

$$E_{dc}^{n} = \sum_{q \in S} P_q \times T_q + \sum_{r \in Pq} P_r^q \times T_r^q \tag{3.13}$$

where, S and P_q are set of switches and set of ports in switch q, P_q, T_q, P_r^q, and T_r^q are the fixed power consumed by q^{th} switch, working time of q^{th} switch, dynamic power consumed by r^{th} port of q^{th} switch, and working time of r^{th} port of q^{th} switch.

The energy consumption in a typical DC depends on servers, cooling, and various other activities. The overall energy consumption of typical DC is given as below.

$$E_{dc} = \sum_{k} E_{dc}^{k} + E_{dc}^{n} + E_{dc}^{c} + E_{dc}^{o} \tag{3.14}$$

where, E_{dc}^{c}, E_{dc}^{c}, and E_{dc}^{o} is the energy consumed by network devices, cooling infrastructure and other activities, respectively in a DC.

A typical DC draws energy from grid to which it is connected. But, the proposed work aims to draw energy form RES (solar and wind) to power a DC. However, the intermittent nature of RES pose a serious threat to achieve this objective. Hence, to manage the intermittency of RES, EVs are used. In this context, the energy consumption of a DC using RES and bi-directional EVs is given as below.

$$E_{dc} = E_{dc}^{ren} \pm E_{dc}^{ev} \tag{3.15}$$

Hence, after equating Eq. 3.14 and 3.15, we get following scenario.

$$\sum_{k} E_{dc}^{k} + E_{dc}^{n} + E_{dc}^{c} + E_{dc}^{o} = E_{dc}^{ren} \pm E_{dc}^{ev} \tag{3.16}$$

After analyzing above aspects, different cases are discussed as below.

- **Case 1**: If $E_{dc}^{ren} > E_{dc}$, then the excess energy (E_{ex}) generated by RES is stored in ESU for future use and is given as below.

$$E_{ex} = E_{dc}^{ren} - E_{dc} \tag{3.17}$$

However, if $E_{ex} > E_{mx}^{esu}$, then the excess energy is supplied to EVs for charging and is given as below.

$$E_{sup}^{ev} = E_{ex} - E_{mx}^{esu} \tag{3.18}$$

where, E_{mx}^{esu} is the maximum storage capacity of ESU.
The EVs have to pay a price (P_{dc}^{ev}) to charge for the energy. For this purpose, the DC have to fix an optimal price to attract EVs to charge from DC rather than any other CS. Further, if there are no EVs to charge the excess energy of their is still some energy available after all EVs charge, then the excess energy is supplied to grid and is given as below.

$$E_{dc}^{drawn} = E_{ex} - E_{mx}^{esu} - E_{sup}^{ev} \tag{3.19}$$

The DC charge a price (P_{dc}^{grid}) to supply the excess amount of energy to grid.

- **Case 2:** If $E_{dc}^{ren} < E_{dc}$, then the required energy (E_{req}) is drawn from ESU and EVs. The required energy is given as below.

$$E_{req} = E_{dc} - E_{dc}^{ren} \tag{3.20}$$

After drawing energy from ESU, if energy is still required by DC then it is drawn from EVs. The energy drawn from EVs is stored in ESU and is supplied to DC to fulfill the energy deficit of DC. The required energy is given as below.

$$E_{req}^{ev} = E_{dc} - E_{dc}^{ren} - E_{dc}^{esu} \tag{3.21}$$

The EVs supply the energy required to DC in proportion to some price (P_{ev}^{dc}) or reward point (RP_{ev}^{dc}) benefits. However, at some instant, the energy required may not be available with EVs. Hence, in such case, an energy deficit (E_{def}) is drawn from grid to which DC is connected and is given as below.

$$E_{def} = E_{req}^{ev} - E_{drw}^{ev} \tag{3.22}$$

where, E_{drw}^{ev} is the energy drawn from EVs to ESU for powering DC.

- **Case 3:** If $E_{dc}^{ren} = E_{dc}$, then the required energy (E_{req}) is sustained by the energy available with RES (E_{dc}^{ren}). In this case, neither deficit nor excess of energy exists. So, this case is considered as an expected sustainable case.

Hence, using the above aspects, the objective function of the proposed scheme for each time instant t is illustrated as below.

$$E_{obj} = min\left(E_{dc}(t) - E_{dc}^{ren}(t) \mp E_{dc}^{ev}\right) : \forall t \tag{3.23}$$

subject to following constraints

$$E_{dc}^{ren}(t), E_{dc}, E_{dc}^{ev}(t) > 0 : \forall t \tag{3.24}$$

$$0 < E_{dc}^{ren}(t) < E_{dc}^{rren}(t) : \forall t \tag{3.25}$$

$$0 < E_{i}^{avl}(t) < E_{i}^{rated}(t) : \forall i, t \tag{3.26}$$

$$E_{mn}^{esu}(t) < E^{esu}(t) < E_{mx}^{esu}(t) : \forall t \tag{3.27}$$

$$P_{dc}^{ev}(t) > P_{ev}^{dc}(t) : \forall t \tag{3.28}$$

where, E_{dc}^{rren} is the rated power of RES, E_{mn}^{esu} is minimum capacity of ESU, P_{dc}^{ev} is the price charged by DC to supply energy to EVs, and P_{ev}^{dc} is the price charged by EVs to supply energy to DC.

3.3 EV-based Energy Management Framework

An SDN-based renewable energy and network-aware framework for sustainability of DCs using RES and EVs is proposed. The proposed scheme is divided into various sub-systems which are elaborated in the subsequent sub-sections.

3.3.1 SDN-based Control Framework

In the SDN-based control framework, the underlying infrastructure and the network control services are separated using various planes such as, data plane, control plane, and application plane. The proposed SDN-based framework is shown in Fig. 3.2.

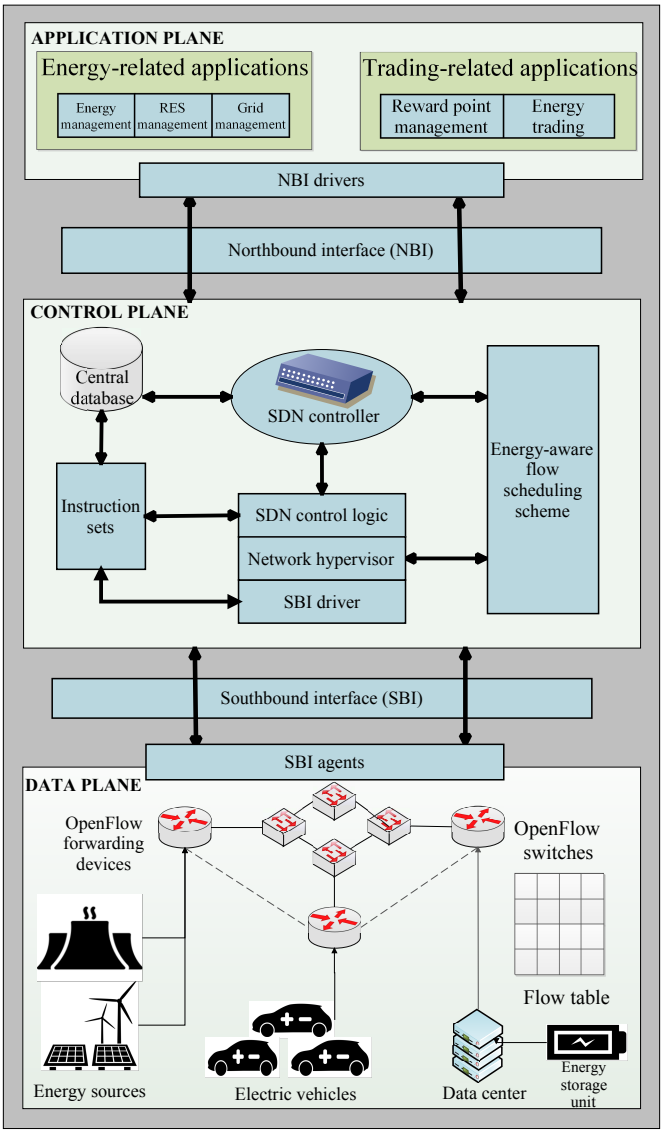


Figure 3.2: SDN-based control framework

All the communication infrastructure in the proposed framework is SDN-enabled using Open flow protocol [172]. The underlying network devices follow the proposed control algorithm to automatically configure itself as per various situations. For better insight of the proposed scheme, three planes of the proposed SDN-based scheme are described as below.

- **Data plane:** The data plane consist of various sources such as DC, ESU, EVs, and energy sources (RES and grid) which are connected using various forwarding devices such as OF physical switches, OF virtual switches, OF routers, and OF gateways. The data generated from the various sources is routed over the energy-aware path using forwarding tables based on the policy prescribed by SDN controller. A list of forwarding tables and group table linked by a pipeline are available with each forwarding device. The flow table available with the forwarding devices is small in size in order to keep the SDN architecture simple and consistent. Based on the programmable logic of SDN controller, an instruction set containing the forwarding decisions is installed on the forwarding devices. The instruction set installed at the forwarding devices is mapped with the list of flow tables and forwarding decision is taken accordingly. In the proposed SDN-based framework, an energy-aware flow scheduling algorithm is proposed to make various flow related decisions. The data acquired from all these sources is used by the proposed scheme to make decisions and computations in the control plane. The DC consists of heterogeneous servers, network devices, and various other equipments. Further, the ESU is associated with data related to storage of energy, inflow/outflow of energy to/fro various sources such as EVs, RES, grid. The data acquired from EVs comprise of battery capacity, state of charge (SoC) and other related data. The energy sources are responsible for data related to energy generated from renewables and price of energy at grid at a particular instance.

- **Control Plane:** The control plane is the decision making plane where the

SDN controller is located and is also known as the brain of network. SDN controller resides in a centralized server located at this plane with a network operating system installed in it. The basic objective of this plane is to enable the control commands on various forwarding devices and manage the information related to all the SDN applications residing at the application plane. The control plane is also responsible for receiving the feedback from various forwarding devices. The network policies are decided at this plane which are implemented at other two planes. The data acquired from various sources and equipments is used by control algorithms at this plane for decisions making and trade-offs. The control plane makes various decisions with respect to various applications related to ESU, RES, EVs charging-discharging, grid, and energy trading.

- **Application plane:** The application plane consists of various end user applications. All the SDN applications run on the application plane. The proposed scheme comprise of various applications related to RES, EVs, DC, ESU, and grid as shown in Fig. 3.2. The EVs owners utilize this platform to interact with DC form charging-discharging purpose. The access to reward point scheme for EVs is also through application plane.

The proposed energy-aware flow scheduling scheme is described as below.

3.3.2 Energy-aware Flow Scheduling Scheme

The proposed SDN-based scheme take various flow scheduling decisions related to the data generated from various sources using an energy-aware flow scheduling algorithm. The proposed Algorithm 1 is designed with focus on improving network performance and minimizing the energy consumption of network devices. In the system model, the SDN controller schedules traffic flows. The traffic flow is divided into three types on the basis of their status as-active, queued, and suspended. The traffic flow required to be scheduled is put in a specific queue. The flow is considered

as active only when a valid path is available for it without any other flows. The flow becomes active when it reaches to the top of the queue. Otherwise, it is suspended when no valid path is available. In order to make the flow scheduling process energy-efficient, ports on an inactive link are put into sleep mode. Moreover, when all ports of a specific switch are in sleep mode, then the concerned switch is also put into sleep mode. This action is performed to minimize the energy consumption of unused ports and switches [154]. In order to synchronize the shifting of switch into sleep mode, a decision variable (d_{syn}, $\forall t$) is defined as below.

$$d_{syn} = \begin{cases} 1 \quad \text{for} \quad active \\ 0 \quad \text{for} \quad idle \end{cases} \tag{3.29}$$

If ($d_{syn} = 0$), then the switch shifts to sleep mode. For this purpose, a threshold time (t_{thr}) is considered. The value of d_{syn} become 0 only if the switch is idle for the threshold time (t_{thr}). The switch shifts back to active mode if the value of d_{syn} becomes 1. The working of proposed Algorithm 1 is shown as below.

Consider a flow (f_p) having size (s_p) with a release time (T_p^r), and deadline time (T_p^d) which is to be scheduled. The flow (f_p) will be scheduled to links with minimal energy consumption and satisfying deadline time. A guaranteed flow rate (g_p) is calculated for the incoming flow (f_p) (line 1). Now, the first step is to search for a valid path with respect to network topology (G), active flow (F_{act}), queued flows (F_{que}), suspended flows (F_{sus}), incoming traffic flow (f_p), and guaranteed flow rate (g_p) (line 2). If a valid path exists, then the incoming flow could be scheduled over a link with minimal energy consumption. For this purpose, each flow available is divided into set of flows. Once the flow set is available, the active time (T_{act}) of each flow set element (F_{set}) is calculated. The energy consumed by link is calculated using Eq. 3.13. Now, the incoming flow is scheduled for the flow set element with minimum active time and energy consumption. The flow is entered into top of the queue and becomes active to be scheduled. Otherwise, the incoming flow is suspended (line

3-18). However, if the path do not exist, the incoming flow is suspended directly (line 19-22). Now, once the scheduled flow is finished, the flow is removed from active flows and the next scheduled flow is brought to the front of the queues in active status.

Algorithm 1 Energy-aware flow scheduling algorithm

Input: F, f_p, s_p, T_p^d, T_p^r, G, F_{act}, F_{que}, and F_{sus}
Output: *path* p, g_p

1: Calculate guaranteed flow rate ($g_p = \frac{s_p}{T_p^d - T_p^r}$)
2: p $\leftarrow$ *FindPath*(G, F_{act}, F_{que}, F_{sus}, f_p, g_p)
3: **if** path p exists **then**
4: Schedule f_p over path p with minimum energy consumption
5: **for** Each flow in F **do**
6: Divide each flow in F into flow sets F_{set} with no shared links
7: **for** $F_{set} \in$ F **do**
8: Calculate $T_{act} = activetime(F_{set})$
9: Compute energy consumed using Eq. (3.13)
10: **if** (T_{act} is minimum) **then**
11: $F_{que} \leftarrow F_{que} + f_p$
12: Schedule flow f_p
13: **else**
14: Suspend f_p
15: $F_{sus} \leftarrow F_{sus} + f_p$
16: **end if**
17: **end for**
18: **end for**
19: **else**
20: Suspend f_p
21: $F_{sus} \leftarrow F_{sus} + f_p$
22: **end if**
23: **if** flow f_p finishes **then**
24: $F_{act} \leftarrow F_{act}$ - f_p
25: Move f_{p-1} to front of queue
26: $f_{p-1} \leftarrow$ Active
27: **end if**

3.3.3 Energy Management Scheme

An energy management scheme is proposed to control the energy generated from various sources such as RES, EVs, and grid. The major aim of the proposed scheme is to sustain the energy consumption of DC using RES. However, due to intermit-

tent nature of RES, the energy deficit is managed using EVs. The grid is used as a back-up for worst case scenario. The proposed scheme intend to supply excess generated energy to grid using an energy trading scheme. The proposed Algorithm 2 for energy management scheme is given as below.

Algorithm 2 DC energy management algorithm

Input: E_{mx}^{esu}, P_{grid}^{dc}, P_{grid}^{dc}, E_{dc}^{rated}

Output: E_{dc}, E_{dc}^{ren}, E_{req}, E_{req}^{ev}, $\hat{E}_{req}$, E_{i}^{avl}, E_{dc}^{avl}, E_{def}, E_{ex}

1: Calculate (E_{dc}) using Eq. (3.14)
2: Calculate (E_{dc}^{ren}) using Eq. (3.1)
3: **if** ($E_{dc} > E_{dc}^{ren}$) **then**
4: Calculate (E_{req}) using Eq. (3.20)
5: **if** ($E_{req} < E_{dc}^{esu}$) **then**
6: Charge (E_{req}) from ESU
7: **else**
8: Calculate the (E_{req}^{ev}) using Eq. (3.21)
9: **for** (i=1; i $\leq$ n; i++) **do** ▷ $i \leftarrow$ Number of EVs
10: Calculate (E_{i}^{avl}) using Eq. (3.31)
11: Calculate (E_{dc}^{avl}) using Eq. (3.5)
12: Charge (E_{req}^{ev}) from i EVs proportionately using Eq. (3.32)
13: Store energy charged from EVs in ESU for supply to DC
14: **end for**
15: **if** ($E_{dc}^{avl} < E_{req}^{ev}$) **then**
16: Calculate (E_{def}) using Eq. (3.22)
17: Draw (E_{def}) from grid at a price (P_{grid}^{dc})
18: Update the energy credit account with grid
19: **else**
20: Store excess available energy in ESU
21: **end if**
22: **end if**
23: **else**
24: Calculate (E_{ex}) using Eq. (3.17)
25: Store (E_{ex}) in ESU
26: **if** ($E_{ex} > E_{mx}^{esu}$) **then**
27: Supply energy to EVs proportionately using Eq. (3.35)
28: **else if** ($E_{ex} > E_{dc}^{rated}$) **then**
29: Supply excess energy to grid at price (P_{dc}^{grid})
30: Update the the energy credit account with grid
31: **end if**
32: **end if**

The energy required by DC to accomplish its routine jobs is calculated using Eq. (3.14) and energy generated by RES is calculated using Eq. (3.1) (line 1-2). If the

energy consumption of DC is greater than energy generated by RES, then calculate (E_{req}) using Eq. (3.20). Further, if (E_{req}) is less than energy available with ESU (E_{dc}^{esu}), then charge the required energy from ESU (line 3-6). Otherwise, the energy is discharged from EVs. For this purpose, calculate energy required from EVs (E_{req}^{ev}) using Eq. (3.21) (line 7-8). The energy available with each EV is calculated using Eq. (3.31). Further, the total energy available with all EVs is calculated using Eq. (3.5). The required energy is discharged from EVs proportionately using Eq. (3.32) and it is stored in ESU for use by DC (line 9-14). Now, if energy available with all EVs is less than energy required from EVs, then deficit energy (E_{def}) calculated using Eq. (3.22) is drawn from grid after paying price (P_{grid}^{dc}) (line 15-17). Otherwise, store excess energy in ESU for use by DC (line 18-22).

If the energy consumption of DC is less than energy generated by RES, then calculate (E_{ex}) using Eq. (3.17). This excess energy is stored in ESU till it reaches the maximum capacity of ESU. If the excess energy is more than the maximum capacity of ESU, then it is supplied to EVs proportionately using Eq. (3.35). However, if there are no EVs available to charge energy, then the excess energy is supplied to grid at a price (P_{dc}^{grid}) (line 23-30).

3.3.4 EV charging-discharging management scheme

In the proposed scheme, EVs are used to tackle the intermittent nature of RES. In a situation when energy generation from RES is low, then the EVs discharge the required energy. Further, at times when generation is high, then EVs are charged. In this regard, it is assumed that there are two categories of EVs that charge-discharge at CS located at DC. The first category consist of i EVs which are reward point member such as DC employees and regular customers. The second category is of j EVs which take part in the energy trading scheme to charge and discharge energy on the basis of price computed using time-of-use (TOU) scheme. The working of charging-discharging system is elaborated as below.

- **EV discharging:** The energy stored in EVs battery is used to support the intermittency of RES connected to DC. For this purpose, EVs joining this scheme decide a threshold SoC (SoC_i^{thr}) which they definitely require once they depart from the DC. The SoC (SoC_i^{avl}) available with i^{th} EV to discharge is given as below.

$$SoC_i^{avl} = SoC_i^{inl} - SoC_i^{thr} \tag{3.30}$$

Further, the energy available with i^{th} EV to discharge is given as below.

$$E_i^{avl} = \left(SoC_i^{inl} - SoC_i^{thr}\right) E_i^{rat} \tag{3.31}$$

The energy required by DC from EVs (E_{req}^{ev}) is drawn from EVs in proportion to the energy available with each participating EV and is given as below.

$$E_i^{drw} = \left(\frac{SoC_i^{avl}}{\sum_i (SoC_i^{avl})}\right) E_{req}^{ev} \tag{3.32}$$

- **EV charging:** The EVs charge from CS located at DC to acquire the pre-decided threshold level of SoC if they are part of the reward point scheme. Other EVs can also charge the required energy from DC after paying price calculated using energy trading scheme. The energy that i^{th} EV (part of reward point scheme) can charge from DC is given as below.

$$E_i^{ch} = (SoC_i^{mx} - SoC_i^{up})\, E_i^{rat} \tag{3.33}$$

where, SoC_i^{mx} is the maximum SoC that i^{th} EV can achieve, SoC_i^{up} is the updated SoC of i^{th} EV after discharging the energy to DC.
The SoC (SoC_i^{gvn}) that must be given to i^{th} EV (part of reward point scheme) to reach its threshold SoC is given as below.

$$SoC_i^{gvn} = SoC_i^{thr} - SoC_i^{up} \tag{3.34}$$

Using the above equation, the energy (E_i^{gvn}) is given proportionately to all the participating EVs as given below.

$$E_i^{gvn} = \left(\frac{SoC_i^{gvn}}{\sum_i (SoC_i^{gvn})} \right) E_{ex} \tag{3.35}$$

After the threshold SoC is achieved by EVs and the excess energy if still available with DC is supplied to them till they achieve maximum level of SoC. The EVs will have to pay for energy they charge above the SoC that they had before they joined the reward point scheme. The energy (E_i^{ex}) for which EVs have to pay is given as below.

$$E_i^{ex} = \left(SoC_i^f - SoC_i^{inl} \right) E_i^{rat} \tag{3.36}$$

where, SoC_i^f is the final SoC that i^{th} EV achieve after charging.

The EVs that are not a part of reward point scheme can charge from DC using excess energy is available after meeting the demand of participating EVs by paying some price. The excess energy (E_j^{ex}) available to charge EVs that are not a part of reward point scheme is given as below.

$$E_j^{ex} = E_{ex} - \sum_i E_i^{gvn} - \sum_i E_i^{ex} \tag{3.37}$$

The energy that j^{th} EV (not a part of reward point scheme) can charge from DC is given as below.

$$E_j^{ch} = \left(SoC_j^{mx} - SoC_j^{inl} \right) E_j^{rat} \tag{3.38}$$

Such EVs trade for energy with DC using single-leader multi-follower Stackelberg game which is described in next section.

Using the above discussed EV charging and discharging aspects, an EV charging-discharging algorithm (Algorithm 3) is designed which is given as below.

Algorithm 3 EV charging-discharging algorithm

Input: SoC_i^{inl}, E_i^{rated}
Output:

1: **for** (i=1; i ≤ n; i++) **do** ▷ $i \leftarrow$ Number of reward point member EVs
2: Check (SoC_i^{inl})
3: **if** ($SoC_i^{inl} > SoC_i^{thr}$) **then**
4: Calculate (SoC_i^{avl}) using Eq. (3.30)
5: Calculate (E_i^{avl}) using Eq. (3.31)
6: **while** ($SoC_i^{avl} > SoC_i^{thr}$) **do**
7: Draw (E_i^{drw}) proportionately from EVs
8: Supply (E_i^{drw}) to ESU
9: Calculate (SoC_i^{upd})
10: **end while**
11: **else**
12: Calculate (SoC_i^{gvn}) using Eq. (3.34)
13: Calculate (E_i^{gvn}) using Eq. (3.35)
14: **while** ($SoC_i^{upd} < SoC_i^{thr}$) **do**
15: Supply (E_i^{gvn}) proportionately to EVs
16: **end while**
17: Calculate (E_i^{ex}) using Eq. (3.36)
18: **while** ($SoC_i^{thr} < SoC_i^{mx}$) **do**
19: Supply (E_i^{ex}) to EVs
20: **end while**
21: **for** (j=1; j ≤ n; j++) **do** ▷ $j \leftarrow$ Number of other EVs
22: Calculate (E_j^{ex}) using Eq. (3.37)
23: **while** ($SoC_j^{inl} < SoC_j^{mx}$) **do**
24: Supply (E_j^{ex}) to j EVs using Algorithm 4
25: **end while**
26: **end for**
27: **end if**
28: **end for**

In this algorithm, the initial SoC (SoC_i^{inl}) of i EVs participating in reward point scheme is checked. If the initial SoC is greater than threshold SoC (SoC_i^{thr}), then SoC available (SoC_i^{avl}), then the EVs can discharge energy available with them. For this purpose, SoC and energy available with each EV is calculated using using Eq. (3.30) and Eq. (3.31) respectively. The available energy with each EV is supplied to DC and stored in ESU proportionately using Eq. (3.32) and the SoC for each EV is updated accordingly (line 1-10). However, if initial SoC of EVs is less than the threshold SoC, then the energy is charged by EVs from CS connected to DC. For this purpose, SoC and energy given by each EV is calculated using using Eq.

(3.34) and Eq. (3.35) respectively. The energy is supplied to EVs till they achieve their threshold SoC. After this, the excess available energy (E_i^{ex}) is calculated using Eq. (3.36). If excess energy is still available, then it is supplied to participating EVs till they reach the desired SoC or maximum SoC (line 11-20). Now, if still there is excess energy available with ESU, then is is supplied to non-participating j EVs using energy trading scheme as per Algorithm 4 (line 21-28).

3.4 Reward-Point and Energy Trading Scheme

In this section, a reward point management mechanism and a single-leader multi-follower Stackelberg game for energy trading is designed. The first case is designed for EVs which agree to participate in reward point mechanism. The second case is designed to decide price of energy to be charged from DC by requesting EVs other than the reward point members. The reward point (RP) management scheme and an energy trading scheme is described in detail in subsequent sections given below.

3.4.1 EV reward point management scheme

The reward point management mechanism is designed to attract EVs to be a part of charging-discharging scheme for managing the intermittent nature of RES. The employees of DC and other offices located near DC are attracted by giving luring offers using the proposed scheme. Each participating member has to register using SDN-based on-line user interface. Once an EV becomes member of the scheme, it agrees to park at the DC and supply energy available with it to DC whenever required. Apart from providing free parking facility, DC ensures RP to the EVs which could be redeemed whenever required to access cloud services, food in cafeteria, money, etc. The RPs are updated into the account of EV whenever it charge or discharge from the DC. The RP earned (RP_i^{ern}) by i^{th} EV varies with respect to

the initial SoC and final SoC and is given as below.

$$RP_i^{ern} = \begin{cases} \beta^n & \text{for} \quad SoC_i^{fnl} = SoC_i^{thr} \\ \beta & \text{for} \quad SoC_i^{thr} < SoC_i^{fnl} \leq SoC_i^{inl} \\ 0 & \text{for} \quad SoC_i^{fnl} > SoC_i^{inl} \end{cases} \tag{3.39}$$

where, SoC_i^{fnl} is the level of energy with i^{th} EV when it departs from DC.

The value of n varies with respect to the maximum discharging capacity (D_{dis}^{max}), (SoC_i^{inl}), and (SoC_i^{fnl}) and is given as below.

$$n = 1 + \left(\frac{1}{D_{dis}^{max}}\right)\left(SoC_i^{fnl} - SoC_i^{inl}\right) \tag{3.40}$$

These RPs could be utilized by EVs to charge from DC at later stage. The RPs utilized (RP_i^{utl}) by i^{th} EV to charge at DC is given as below.

$$RP_i^{utl} = \begin{cases} \beta^m & \text{for} \quad SoC_i^{inl} = SoC_i^{mx} \\ \beta & \text{for} \quad SoC_i^{inl} < SoC_i^{mx} \end{cases} \tag{3.41}$$

The value of m varies with respect to the maximum charging capacity (D_{ch}^{max}), (SoC_i^{inl}), and (SoC_i^{fnl}) and is given as below.

$$m = 1 + \left(\frac{1}{D_{ch}^{max}}\right)\left(SoC_i^{fnl} - SoC_i^{inl}\right) \tag{3.42}$$

After the EV charges or discharges from CS located at DC, the RP earned by EV owner are updated accordingly as shown below.

$$RP = RP_i^{inl} + RP_i^{ern} - RP_i^{utl} \tag{3.43}$$

where, RP_i^{inl}, RP_i^{ern} and RP_i^{utl} are initial, earned and the utilized values of RPs, respectively.

The Algorithm 4 depicting the flow of RP scheme is given as below.

Algorithm 4 EV reward point management algorithm

Input: RP_i^{inl}, SoC_i^{inl}, SoC_i^{fnl}, SoC_i^{thr}
Output: RP_i^{ern}, RP_i^{utl}, RP

1: **for** (i=1; i $\leq$ n; i++) **do** ▷ $i \leftarrow$ Number of reward point member EVs
2: Check (RP_i^{inl})
3: **if** ($SoC_i^{fnl} == SoC_i^{thr}$) **then**
4: ($RP_i^{ern} == \beta^n$)
5: **else if** ($SoC_i^{thr} < SoC_i^{fnl} < SoC_i^{inl}$) **then**
6: ($RP_i^{ern} == \beta$)
7: **else**
8: ($RP_i^{ern} == 0$)
9: Calculate (RP_i^{utl}) using Eq. (3.41)
10: Update RP using Eq. (3.43)
11: **end if**
12: **end for**

This algorithm is meant for only i EVs who are participating in RP scheme. The initial value of RP's (RP_i^{inl}) is check for i EVs (line1-2). If the final SoC (SoC_i^{fnl}) of EVs is equal to threshold SoC (SoC_i^{thr}), then β^n RP's are earned which are updated in the account of concerned EV (line 3-4). However, if the final SoC (SoC_i^{fnl}) of EVs is greater than threshold SoC (SoC_i^{thr}) but less than initial SoC (SoC_i^{inl}), then β RP's are earned which are updated in the account of concerned EV (line 5-6). Further, no reward point is given if the final SoC (SoC_i^{fnl}) of EVs is greater than the initial SoC (SoC_i^{inl}) (line 7-8). Similarly, the RP's utilized by EVs is calculated using Eq. (3.41). Finally, the reward points earned or utilized are updated in the user account using Eq. (3.43) (line 9-12).

3.4.2 Stackelberg game-based energy trading scheme

In this section, a single-leader multi-follower Stackelberg game for energy trading is formulated. The CS located at DC act as single-leader and EVs act as multi-followers. The CS is connected to ESU to supply energy to EVs. The CSs supply the energy (E_j^{req}) to EVs at a price (P_{dc}^{ev}) calculated using time of use (TOU) pricing scheme. An user interface such as Plug Share is designed to facilitate EVs to trade for energy [173]. CS decides the price for energy at a particular time slot. EVs that

wants to charge for energy check for the announced price. If the announced price suite the requirement of EVs, they decide the energy demand. EVs announce the energy demand to CS which supply the required energy once the price is received.

The energy available (E_{cs}^{avl}) at CS connected to DC that can be supplied to EVs depends upon energy consumption of DC (E_{dc}), energy stored in ESU (E_{dc}^{esu}), and energy generated by RES (E_{dc}^{ren}). The energy available with CS is given as below.

$$E_{cs}^{avl} = E_{dc}^{esu} + E_{dc}^{ev} + E_{dc}^{ren} - E_{dc} \tag{3.44}$$

In general, the trading of energy between EVs and CSs occurs in sequential way. Hence, the Stackelberg game is the most suited technique which follows the similar movement trend. Further, it is a two period game which involves a leader and follower [174]. The leader has preference to announce its move before follower. The follower can opt out of the competition while the leader cannot back-out from it. So, The leader initiates the game with best move which the follower reply with best response [175]. The aim of leader and follower is to maximize their profit, i.e., utility of players. Moreover, the players in Stackelberg game define their strategies aimed at economic benefit. The various aspects related to the proposed Stackelberg game are given as below.

- **Players:** In this scheme, j EVs (where $j \in \hat{J}$) and single CS is considered.
- **Strategy:** The strategy profile for EVs is given as $S_{ev} = \{CS, j \in \hat{J}\}$. For EVs, the strategy is to decide the amount of energy to be charged from CS which maximizes its utility. For CSs, the strategy is to decide the optimal price of energy using TOU which maximizes its utility.
- **Payoff:** Both EVs and CSs decide their strategies on the basis of the payoffs they receive. The payoff or utility is described on the basis of three entities; price, cost, and revenue. These three entities are used to design the utility functions of EVs and CS. In this context, price is refered as the amount of

money charged to sell a energy, revenue is used for the amount of money collected after selling energy, and cost is refered as the amount money incurred on purchase or generation of energy. After incorporating the above entities, the utility functions for EVs and CSs are defined as below.

1. **Utility function of EV:** The utility of EVs represents the gain or profit achieved after charging required amount of energy from CS. A positive concave function R_j^{ev} is considered as revenue available with j^{th} EV in terms of energy. EVs pay a certain amount of money (P_{dc}^{ev}) to charge the required amount of energy from CS. Hence, the utility function (U_j^{ev}) of j^{th} EV is given as below.

$$U_j^{ev} = R_j^{ev} - P_{dc}^{ev} \tag{3.45}$$

2. **Utility function of CS:** The utility of CS represents the profit or gain achieved after selling energy to EVs. The revenue generated by a CS after selling energy to j^{th} EV is given by R_j^{cs} as follows.

$$R_j^{cs} = \sum_{j=o}^{m} P_{dc}^{ev} \tag{3.46}$$

The total cost (C_{dc}^{tot}) of energy comprise of cost incurred on generation (C_{dc}^{ren}) and maintenance of energy (C_{dc}^{mn}) and is given as below.

$$C_{dc}^{tot} = C_{dc}^{ren} + C_{dc}^{mn} \sum_{j=0}^{n} E_j^{ch}. \tag{3.47}$$

Hence, using the revenue and cost, the utility function (U_{dc}^{cs}) of CS is given below.

$$U_{dc}^{cs} = R_j^{cs} - C_{dc}^{tot} \tag{3.48}$$

The price of energy varies with respect to time [163, 176]. Hence, using fixed price for energy may not suite every EV. So, the price based on TOU is considered in the

proposed energy trading scheme. The TOU is divided into *peak time* and *off-peak time* on the basis of energy available with the CS and the energy being charged by EVs at present time slot. For this purpose, a threshold limit (E_{dc}^{thr}) is considered by CS to differentiate between both the times. The threshold limit (E_{dc}^{thr}) is calculated as given below.

$$E_{dc}^{thr} = E_{dc}^{esu}/j \tag{3.49}$$

With an increase in the number of EVs charging from CS and decrease in generation of energy by RES, the energy availability at CS is low. If the energy available at CS is less than threshold limit, then it is *off-peak time*. Otherwise, it is *peak time* [177]. The price (P_{dc}^{ev}) of energy decided using TOU is given as follows.

$$P_{dc}^{ev} = \begin{cases} P_p & \text{for} \quad E_{dc}^{avl} \leq E_{dc}^{thr} \\ P_{op} & \text{for} \quad E_{dc}^{avl} > E_{dc}^{thr} \end{cases} \tag{3.50}$$

The price (P_p) of energy during peak time is given as below.

$$P_p = \zeta E_j^{ch}/E_{cs}^{avl} \tag{3.51}$$

where, ζ is the constant decided by DC which is used to decide the price of energy during peak time.

The price (P_{op}) of energy during off-peak time is given as below.

$$P_{op} = E_j^{ch}/E_{cs}^{avl} \tag{3.52}$$

The proposed single-leader multi-follower Stackelberg game for energy trading is described in Algorithm 5. In this algorithm, initially the number of EVs requesting for energy are calculated (line 1). Further, energy available (E_{dc}^{avl}) is calculated using Eq. (3.31). On the basis of energy available and number of requesting EVs, the threshold value (E_{dc}^{thr}) is calculated using Eq. (3.49) (line 2-3). Now, if TOU is peak, then calculate price P_p using Eq. (3.51). But, if TOU is off-peak, then calculate price P_{op} using Eq. (3.52) (line 4-8). Once price is decided, the CS connected to

DC check its utility using using Eq. (3.48). If the utility is more as compared to utility at previous instance, then announce price. Otherwise, recompute price till the utility shows growth (line 9-13). Once the price is announced, the requesting EVs calculate the energy required from CS using Eq. (3.38). Using price and energy, EVs calculate their utility using Eq. (3.45). If the utility is more than the utility at previous instance, then announced price is accepted. The EVs pay the accepted price and charge the required energy. Otherwise, wait for next announcement of price or recompute the energy requirement and check the utility again (line 14-22).

Algorithm 5 Stackelberg game-based energy trading algorithm

Input: j

Output: E_{dc}^{avl}, E_{dc}^{thr}, P_p, P_{op}, U_j^{ev}, U_{dc}^{cs}, E_i^{ch}

1: Check for number of requesting EVs ▷ Leaders move (CS)
2: Calculate (E_{dc}^{avl}) using Eq. (3.31)
3: Calculate threshold value (E_{dc}^{thr}) using Eq. (3.49)
4: **if** time-of-use == peak **then**
5: Calculate P_p using Eq. (3.51)
6: **else**
7: Calculate P_{op} using Eq. (3.52)
8: **end if**
9: **if** $(U_j^{ev}(t) \geq U_i^{ev}(t-1))$ **then**
10: Announce price to EVs
11: **else**
12: Recompute price
13: **end if**
14: **for** (j=1; j≤n; j++) **do**
15: Calculate E_i^{ch} using Eq. (3.38) ▷ Followers move (EVs)
16: Calculate utility of all EVs using price announced
17: **if** $(U_{dc}^{cs}(t) \geq U_{dc}^{cs}(t-1))$ **then**
18: Accept and pay price, charge for required energy ▷ Nash equilibrium
19: **else**
20: Wait for next announcement
21: **end if**
22: **end for**

Summary of the Chapter: The Chapter 3 presents an EV-based energy management scheme for sustainability of DCs using RES. The entire flow of data in the proposed scheme depends on SDN-enabled energy-efficient flow scheduling algorithm. Moreover, a charging-discharging scheme for penetration of EVs is also presented to handle the intermittency of RES.

Chapter 4

Edge computing-based Energy Management Scheme

In this chapter, an efficient scheme for energy management for sustainability of Cloud DCs in Edge-Cloud Environment using SDN is presented[1]. In the proposed scheme, a support vector machine (SVM)-based workload classification approach is presented. Moreover, a two-stage game for workload scheduling for sustainability of DCs is designed. In order to achieve energy efficiency and optimal utilization of network and computing resources, different consolidation schemes are also presented.

4.1 System Model

The proposed system model of edge-cloud environment consists of multiple edge nDCs and cloud DCs (cDCs). Fig. 4.1 shows the architecture of the geo-distributed edge-cloud environment in a typical green city. The proposed model comprises of

[1]The content of this chapter is taken from:

- G. S. Aujla and N. Kumar, "MEnSuS: An efficient scheme for energy management with sustainability of cloud data centers in edgecloud environment", Future Generation Computer Systems, 2017, doi:10.1016/j.future.2017.09.066.
- G. S. Aujla, N. Kumar, A. Y. Zomaya and R. Rajan, "Optimal Decision Making for Big Data Processing at Edge-Cloud Environment: An SDN Perspective," IEEE Transactions on Industrial Informatics, vol. 14, no. 2, pp. 778-789, 2018.

four layers. The lowest layer consists of various end users who want to access various cloud resources to run their applications. The second layer is network layer, where a SDN-based forwarding scheme is used to effectively manage latency issues and network resource utilization. The third layer consists of cloud infrastructure such as-multiple edge nDCs and cDCs along with a cloud controller. The cDCs comprise of large-scale resource abundant geo-distributed DCs. But, nDCs comprise of small-scale resource limited DCs and edge devices such as-gadgets, PDAs, etc. The last layer is energy layer consisting of RES (solar panels and wind turbines), battery energy storage system (BESS), and power grid. Different aspects related to the proposed system model are discussed in subsequent sections.

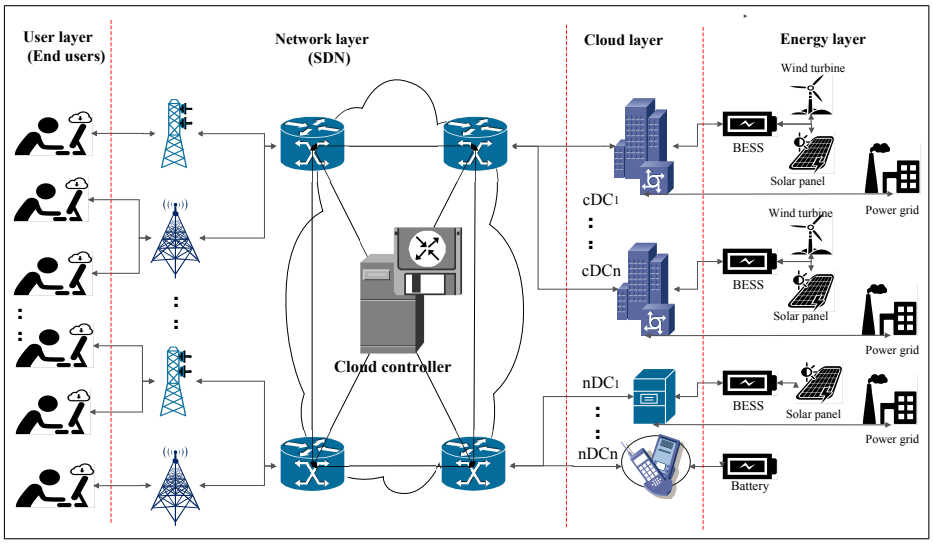


Figure 4.1: System architecture of edge-cloud environment

4.1.1 Queuing model

Let us consider r incoming requests comprising of j type of jobs which need to be scheduled in edge-cloud environment. The cloud controller classifies the incoming requests into appropriate categories and added to different queues on the basis of priority, latency, resources required, resources available, renewable energy, etc. The request in different queues are scheduled using nDCs and cDCs by cloud controller. For this purpose, at time t, the type j jobs are modeled using poisson distribu-

tion with an arrival rate of $(A_j(t))$. The queues dynamics followed by type j jobs scheduled at i^{th} nDC/cDC is given as below [137].

$$q_i^j(t+1) = max[q_i^j(t) - q_i(t)] + \eta_i^j(t) \tag{4.1}$$

where, $\eta_i^j(t)$ is the number of type k jobs routed to i^{th} nDC/cDC.

4.1.2 QoS Model

Ensuring SLA and providing high QoS is one of the important tasks of the cloud controller. In this direction, when the resources required $(\Re_r^{rq})$ to handle r^{th} incoming request exceeds the available resources $(\Re_s^{av})$ at s^{th} server, then a SLA violation $(\Im_s)$ occurs. The SLA violation $(\Im_s)$ experienced by s^{th} server is defined as below [103].

$$\Im_s = \frac{1}{S} \sum_{s=1}^{S} \frac{t_s^{th}}{t_s^{ac}} d_s^{mg} \tag{4.2}$$

where, t_s^{th} is the time for which threshold level of utilization is experienced by s^{th} server, t_s^{ac} denotes the total active time of s^{th} server, and d_s^{mg} denotes performance degradation value due to migration of resources.

In order to evaluate SLA violations, an SLA metric $(\Im_s^r)$ for r^{th} job assigned to s^{th} server at i^{th} nDC/cDC is defined as follows.

$$\Im_s^r = \sum_{i=1}^{I} \sum_{s=1}^{S} \frac{\Re_s^{rq}(t) - \Re_s^{al}(t)}{\Re_s^{rq}(t)} \tag{4.3}$$

where, $R_s^{al}(t)$ is server amount allocated by i^{th} nDC/cDC at time t.

In case a SLA violation occurs, then the CSPs has to bear a penalty $(\rho_s^{\Im})$ as given below [103].

$$\rho_s^{\Im} = \sum_{s=1}^{S} [\wp_s^{\Im} \ t_s^{\Im}] \tag{4.4}$$

where, $\wp_s^{\Im}$ is the unit cost for each SLA violation and $t_s^{\Im}$ is the duration for which SLA violation occurred at s^{th} processor of i^{th} nDC/cDC.

Now, in order to provision incoming user request using nDC/cDCs, one of the most important QoS parameter is latency. It is essential to provision cloud services to end users with lower latency. In this direction, latency associated for providing $\Re_s^{rq}(t)$ by s^{th} server at i^{th} nDC/cDC is defined as follows.

$$\ell = \tau_{pg} + \tau_{sr} + \tau_{qu} + \tau_{pr} + \tau^{jit} + \tau^{cac} \tag{4.5}$$

where, τ_{pg}, τ_{sr}, τ_{qu}, τ_{pr}, τ^{jit}, and τ^{cac} denotes propagation, serialization, queuing, processing, jitter, and caching delays, respectively.

In Eq. 4.5, propagation delay (τ_{pg}) is defined as below.

$$\tau_{pg} = \sum_{p} \frac{d_{sw_1,sw_2} \times \omega_{f,sw_1}(t) \times \omega_{f,w_s2}(t) \times a_{sw_1,sw_2}}{\tau_{md}(t)} \tag{4.6}$$

where, d_{sw_1,sw_2} is the distance between two switches, $\omega_{f,sw_1}(t)$ and $\omega_{f,sw_2}(t)$ are decision variables for two flows with respect to switches sw_1 and sw_2, respectively, a_{sw_1,sw_2} denotes binary variable for adjacency between two switches sw_1 and sw_2, and $\tau_{md}(t)$ is the propagation delay of the medium.

In Eq. 4.5, serialization delay (τ_{sr}) is defined as below.

$$\tau_{sr} = \sum_{i \in sw} \sum_{j \in p} \frac{P_{i,j}(t)}{B_{i,j} \times \Theta_{i,j}(t)} \tag{4.7}$$

where, $P_{i,j}(t)$ is the packet size, $B_{i,j}$ is the bandwidth j^{th} port at i^{th} switch, and $\Theta_{i,j}(t)$ denotes the occupancy ratio.

In Eq. 4.5, queuing delay (τ_{qu}) is defined as below.

$$\tau_{qu} = \sum_{i \in sw} \sum_{j \in p} \frac{|q(t)|}{B_{i,j} \times \Theta_{i,j}(t)} \tag{4.8}$$

where, $|q(t)|$ denotes ready queue.

In Eq. 4.5, processing delay (τ_{pr}) is defined as below.

$$\tau_{pr} = \sum_{i \in sw} \tau_{pr}^{i}(t) \times \sum_{f} (t_f^{en} - t_f^{st}) \times \omega_{f,s_i}(t) \tag{4.9}$$

where, $\tau_{pr}^{i}(t)$ denotes processing delay by i^{th} switch, t_f^{en} is end time of f^{th} flow, and t_f^{st} is start time of f^{th} flow.

Moreover, high response time is also one of the most desired requirements of cloud users. So, the response time (τ_i^{rs}) to deal with incoming job by i^{th} nDC/cDC is given as follows.

$$\tau_i^{rs} = \frac{1}{\nu_s \times \eta_s^j - q_i(t)} + \frac{1}{\nu_s} + \ell_{nt}\eta_i^j(t) \tag{4.10}$$

where, ν_s is the processing speed of s^{th} server, η_s^j denotes total number of servers used to handle j^{th} job, ℓ_{nt} is the delay associated with underlying networks.

At some moment of time, to meet the SLAs, some resources or requests need to be migrated. In such case, an additional migration delay (τ_i^{mg}) is incurred. Such a migration delay is defined as below.

$$\tau_i^{mg} = \ell_{nt}\gamma_i(t) \tag{4.11}$$

where, $\gamma_i(t)$ is the migration rate.

Jitter is an important parameter that measures difference in inter-arrival time of packets. So, to calculate overall latency, total jitter is also calculated as below.

$$\tau_i^{jt} = D_{p2p} + 2 \times n \times R_{rms} \tag{4.12}$$

where, D_{p2p} is the deterministic jitter, R is the random jitter, n depends on bit error rate required of the link.

For, edge devices or nDCs, the delay incurred (τ_i^{ed})is given as below.

$$\tau_i^{ed} = \frac{1}{\psi_i - \phi_i} \tag{4.13}$$

where, ψ_i denotes the service rate and ϕ_i denotes the arrival rate of jobs.

Moreover, in the case of edge devices, caching is widely adopted to store the contents. Now, at some period of time, if some of the contents are to be retrieved from the cache of edge devices, then some additional delay is incurred. So, this additional time required to fetch the contents from edge devices is also a part of the total time incurred for data transmission and is given as below.

$$\tau_i^{cac} = t_{lv} - t_{ass} \tag{4.14}$$

where, t_{lv} denotes leaving time at an edge device and t_{ass} denotes the association time at new device.

4.1.3 Energy model

Energy consumption of a DCs is one of the major contributor to the overall operational cost. Various entities such as-servers, network, cooling system, etc contribute towards energy consumption of a DC. Moreover, the energy consumption of a DCs directly depends upon the utilization of its hardware resources. So, the energy consumption of a i^{th} nDC/cDC is given as follows.

$$E_i = \sum_{s=1}^{S} E_i^s + E_i^n + E_i^c + E_i^{ad} \tag{4.15}$$

where, E_i^s), E_i^n, E_i^c, and E_i^{ad} denotes energy consumption of s^{th} server, network, colling system, and additional equipments at i^{th} DCs.

In Eq. 4.15, the energy consumed by s^{th} server of i^{th} DC depends on the level of utilization (U_i^s) of server. The energy consumed by s^{th} server of i^{th} nDC/cDC is defined as below.

$$E_i^s = E_{id}^s + (E_{mx}^s - E_{id}^s)\ U_i^s \tag{4.16}$$

where, E_{id}^s and E_{mx}^s denotes the energy consumed by idle s^{th} server and the maximum energy that s^{th} server can consume.

The level of utilization of s^{th} server depends on the resources utilized ($\Re_s^{ut}$) by

end users and maximum capacity ($\Re_s^{mx}$). The level of utilization of s^{th} server is given as follows.

$$U_i^s = \left(\frac{\Re_s^{ut}(t)}{\Re_s^{mx}} \right) \times 100 \tag{4.17}$$

Apart from energy consumed by servers, the network infrastructure consumes the second most energy in a DCs. The energy consumption of DCs depend on two parts, i.e., fixed part (E_{sw}^n) and dynamic part (E_p^n). The fixed part of energy consumption depends on the equipments such as-fan, chasis, etc and the dynamic part depends on the active ports. So, on the basis of these aspects, the energy consumption of network infrastructure in i^{th} nDC/cDC is defined as follows.

$$E_i^n = E_{sw}^n + E_p^n \tag{4.18}$$

$$E_i^n = E_{fx}^n \times \sum_{sw} (t_f^{en} - t_f^{st}) + E_{dy}^n \times \sum_{sw} (t_f^{en} - t_f^{st}) \tag{4.19}$$

where, E_{fx}^n is the energy consumed by fixed part and E_{dy}^n is the energy consumed by dynamic part.

Moreover, the energy consumed (E_i^{ed}) by i^{th} nDCs or edge devices for handling the incoming job requests (j_i^{ed}) is defined as below.

$$E_i^{ed} = P_i^{ed} \times t \tag{4.20}$$

$$E_i^{ed} = \left(a_i (j_i^{ed})^2 + b_i j_i^{ed} + c_i \right) \times t \tag{4.21}$$

where, $a_i > 0$ and $b_i, c_i \geq 0$ are the pre-determined parameters.

In the proposed work, the major objective is to sustain energy consumption of nDCs/cDCs using RES. For this purpose, two types of RES (solar panels and wind turbines) are considered. Now, the energy generated by RES (E_i^r) connected to i^{th} nDC/cDC is given as follows.

$$E_i^r = E_i^{sl} + E_i^{wn} \tag{4.22}$$

where, E_{dc}^{sl} and E_{dc}^{wn} denotes energy generated by solar panels and wind turbines.

In Eq. 4.22, the energy generated by solar panels (E_i^{sl}) connected to i^{th} nDC/cDC is given as below [169].

$$E_i^{sl} = \left(1 - L_{pv}^{tp}\right)\ \eta\ S_i^{sl}\ cos(\alpha)\ \text{Ŗ} \tag{4.23}$$

where, Ŗ denotes solar radiations, η demotes conversion efficiency of solar panel, α is the radiation angle of Sunlight, S_i^{sl} is the size of solar panel connected to i^{th} nDC/cDC, and L_{sl}^{tm} denotes loss due to temperature exedence of corridor.

In Eq. 4.22, the energy generated by wind turbines (E_i^{wn}) connected to i^{th} nDC/cDC is given as below [169].

$$E_i^{wn} = \frac{1}{2}\ \left[\text{Ç}\ \rho\ \hat{a}\ \upsilon^3\right]\ L_{wn}^{tp} \tag{4.24}$$

where, Ç denotes the rotar efficiency, ρ is the air density, $\hat{a}$ denotes the rotar swept area, υ denotes the wind speed, and L_{pv}^{tp} denotes loss due to temperature exedence of corridor.

Here, the primary source of energy used to power DCs is considered as RES. The DCs are provided energy from RES through a BESS. The BESS is used to handle the intermittency of RES. At some moment of time, the energy generated by RES can be higher than the energy required to power a DC. So, at such period of time, the excess energy (E_i^{ex}) is stored in BESS. At other moment of time, the energy generated by RES can be less than the energy required (E_i^{df}) to power a DC. At such period of time, the excess energy stored in BESS is utilized to power DCs. So, a BESS plays a critical role in handling the intermittency of a RES. Keeping in view all the above points, the energy stored in BESS (E_i^{bess}) at time t is given as below.

$$E_i^{bess}(t) = \left(E_i^{bess}(t-1) + E_i^r - E_i\right) L_i^{bess} \tag{4.25}$$

where, L_i^{bess} denotes the self-discharge rate of BESS.

In this direction, the excess energy stored at BESS (E_i^{ex}) is given as below.

$$E_i^{ex} = E_i^r - E_i \tag{4.26}$$

Similarly, the deficit energy drawn from BESS (E_i^{df}) is given as below.

$$E_i^{df} = E_i - E_i^r \tag{4.27}$$

If in case, the sufficient amount of energy is not available with BESS, then the energy is drawn from grid. The energy drawn from grid is given as follows.

$$E_i^{gr} = E_i - E_i^{bess} \tag{4.28}$$

4.1.4 Carbon Index

One of the major goals of this work is to reduce the carbon footprints related to DCs. PUE is one of the most important terms to indicate the energy efficiency of a nDC/cDC. In this regard, PUE of i^{th} nDC/cDC is defined as below.

$$PUE_i = \frac{P_i}{P_i^{it}} \tag{4.29}$$

where, P_i is the power consumption of i^{th} nDC/cDC and P_i^{it} is the power consumption related to IT infrastructure at i^{th} nDC/cDC.

Now, using PUE, a carbon index is defined as given below [125].

$$\varphi_i = \sum_{i,t} \left(PUE_i \times \sum_{i,s} \left(\varphi_i^{rt} \times \sum_{s} E_i^s \times t \right) \right) \tag{4.30}$$

where, φ_i^{rt} denotes the carbon footprint rate of i^{th} nDC/cDC.

4.2 Problem Formulation

For selecting an appropriate nDC/cDC to schedule incoming jobs, the utility function (to find payoff) of end user, nDC/CDC play an important role. In this direction, the utility functions of j^{th} job requested by end user is given as follows.

$$\mho_j = \left(\sum_{j=1}^{J} R_j\right) - \left(\sum_{j=1}^{J} P_i^j\right) \tag{4.31}$$

where, R_j is the revenue available with end users for availing resources for j^{th} job and P_i^k is the price charged by i^{th} nDC/cDC for allocating required resources to run j^{th} job to end users.

Similarly, the utility function of i^{th} nDC/cDC where the incoming job requests are scheduled is given as follows.

$$\mho_i = \left(\sum_{i,j} P_i^j\right) - \left(\sum_{i,j} C_i^j\right) \tag{4.32}$$

Now, the cost part in the above utility function comprises of costs incurred on computing (C_i^{cp}), communication network (C_i^{cm}), energy (C_i^{en}), migration (C_i^{mg}), and SLA violation penalty ($\rho_s^{\Im}$). So, the overall cost (C_i^j) for handling j^{th} job by i^{th} nDC/cDC is given as below.

$$C_i^j = C_i^{cp} + C_i^{cm} + C_i^{en} + C_i^{mg} + \rho_s^{\Im} \tag{4.33}$$

The cost related to the computing resources that are allocated to end users to handle j^{th} job is given as below.

$$C_i^{cp} = (\rho\varrho_i + \rho B_i + \rho\delta_i) \times t \tag{4.34}$$

where, $\wp$ is the price coefficient that varies as per different resources, ϱ_i, B_i and δ_i are the computing, bandwidth, and storage resources provisioned by i^{th} nDC/cDC.

The communication cost is directly dependent on the bandwidth requirement of the end user for handling j^{th} job and is given as follows.

$$C_i^{cm} = \sum_{i,j} b_i \eta_i^j(t) B_i \tag{4.35}$$

where, b_i denotes the bandwidth cost coefficient.

Now, to handle the j^{th} job by i^{th} nDC/cDC some amount of energy is consumed. The energy consumed by i^{th} nDC/cDC to handle j^{th} job is given as below.

$$C_i^{en} = \varepsilon E_i^j \tag{4.36}$$

where, ε is the price coefficient for energy per unit time.

An additional cost is incurred for migrations which is given as below.

$$C_i^{mg} = \sum_{i,j} b_i \aleph_i^j(t) B_i^{mg} \tag{4.37}$$

where, $\aleph_i^j$ denotes the number of jobs migrated and B_i^{mg} is the bandwidth requirement for migration.

Similarly, the utility function for network is given as below.

$$\mho_n = \frac{B_i \times \sigma_i^{av}}{(n_j + 1) \times \tau_i^{av}} d_{sw1,sw2} \tag{4.38}$$

where, B_i is the bandwidth required, σ_i^{av} and τ_i^{av} denotes the average anticipated throughput and delay of the network after including the new load.

Now, the overall utility comes out to be mapping of utility functions of users, network, and cDC/nDCs. The mapping ($\mho_{j,n,i}^{map}$) of the utilities is given as below.

$$\mho_{j,n,i} = \sum_{j,n,i} \begin{bmatrix} 1,1,1 & 1,2,1 & . & . & 1,n,1 \\ 1,1,2 & 1,2,2 & . & . & 1,n,2 \\ . & . & . & . & . \\ j,1,i & j,2,i & . & . & j,n,i \end{bmatrix}. \tag{4.39}$$

Now, in order to select an optimal pair form the above matrix, a decision variable (δ_{jni}, $\forall t$) is defined as below.

$$\delta_{jni} = \begin{cases} 1 & \text{for} \quad \mho_{jni} > \mho^*_{jni} \\ 0 & \text{for} \quad otherwise \end{cases} \tag{4.40}$$

where, jni^* is the set of all pair other than jni.

Using the above concept, the objective function is formulated as as below.

$$max \left[\sum_{n,i} (\mho_{1n_1i_1}) \delta_{1n_1i_1} + \mho_{1n_2i_2} \delta_{1n_2i_2} + ... + \mho_{1n_ni_n} \delta_{1n_ni_n} \right] \tag{4.41}$$

subject to the following constraints

$$\varphi_i < \varphi_i^{ds} \tag{4.42}$$

$$\delta_{ijk} \in [0, 1] \tag{4.43}$$

$$\mho_j(i) > \mho_j(i^*) \tag{4.44}$$

$$\mho_i(t) > \mho_i(t-1) \tag{4.45}$$

$$\mho_n(f) > \mho_n(f^*) \tag{4.46}$$

$$t_i^{rs} \leq t_{mx}^{rs} \tag{4.47}$$

$$0 < \sum_{j=1}^{J} q_i(t) A_i \leq \nu_s \eta_i^j(t) \tag{4.48}$$

where, φ_i^{thr} desired carbon index, $\mho_j(i^*)$ is the utility of j^{th} job with respect to all nDC/cDCs other than i, $\mho_n(f)$ is the utility of network with respect to all flow path f, $\mho_n(f^*)$ is the utility of network with respect to all flow paths other than f, and t_{mx}^{rs} denotes the maximum achievable response time.

4.3 Edge-based Energy Management Scheme

The proposed scheme work in edge cloud scenario in different phases. The schematic phase diagram of the proposed scheme is shown in Fig. 4.2.

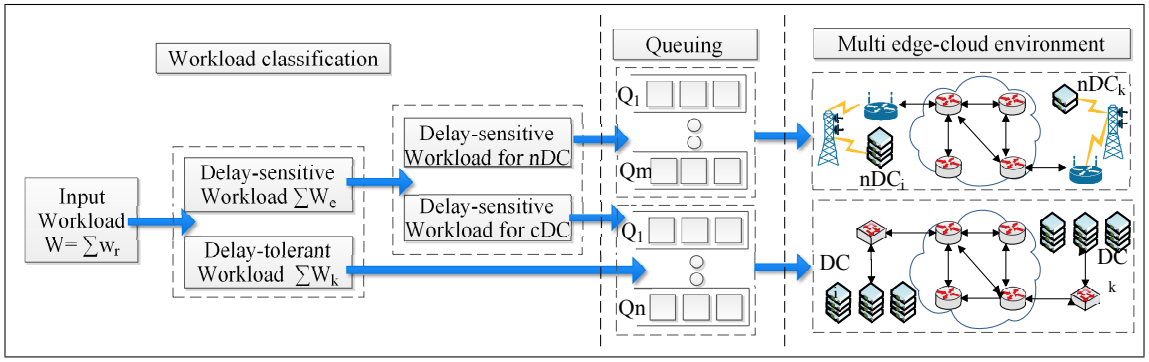


Figure 4.2: Workload slicing scheme

Initially, a SDN-based control scheme for fast forwarding is presented in order to provision low latency services to the end user. In the next section, a SVM-based classification scheme is designed to categorize the incoming user requests on the basis of latency and priority for provisioning at nDCs or cDCs. Finally, Stackelberg game based scheduling scheme is formulated to assign nDC/cDCs having sufficient amount of renewable energy for handling different categories of workload. The interactions taking place between nDCs and cDCs is sown in Fig. 4.3. Different parts of the proposed scheme are elaborated in the next sub-sections.

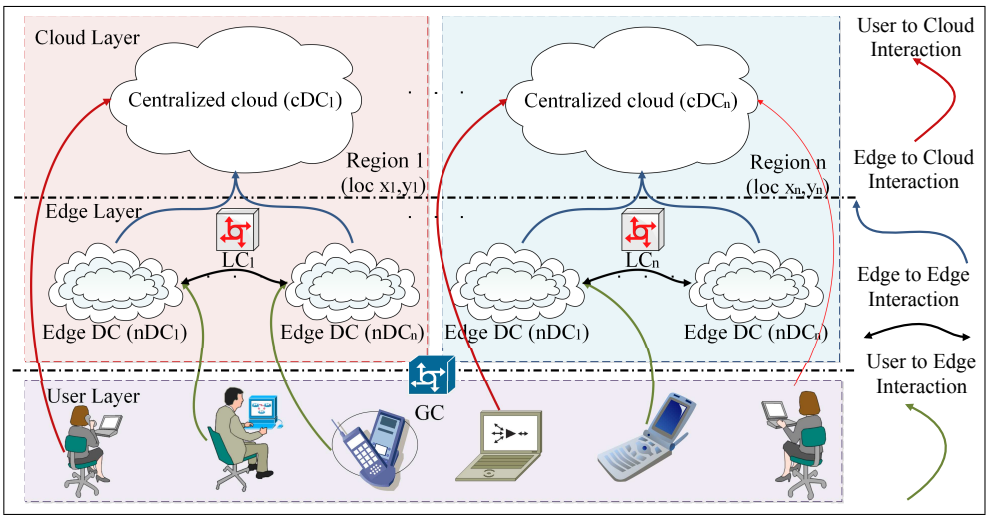


Figure 4.3: System model

4.3.1 SDN-based Control Framework

In order to handle millions of incoming users requests, the underlying network management should be effective. Moreover, the traditional hardware-based network approach is not suitable in edge-cloud environment. SDN is a programmable, flexible, and scalable network paradigm that can reconfigure its rules and policies according to various situations. Moreover, the decoupling of control and data planes help to reduce the complexity and improves the performance of the underlying networks. The concept of flow table management provides manifold benefits such as-flow consolidation, fast forwarding, etc. Such characteristics helps to increase QoS in terms of latency and response time and reduce energy consumption related to network infrastructure. So, an SDN-based control framework is presented along with a OF-switch load consolidation scheme to provide energy efficiency. Fig. 4.4 shows the proposed architecture of the SDN-based control framework. Open flow protocol is used as communication standard for the SDN-based control scheme [172].

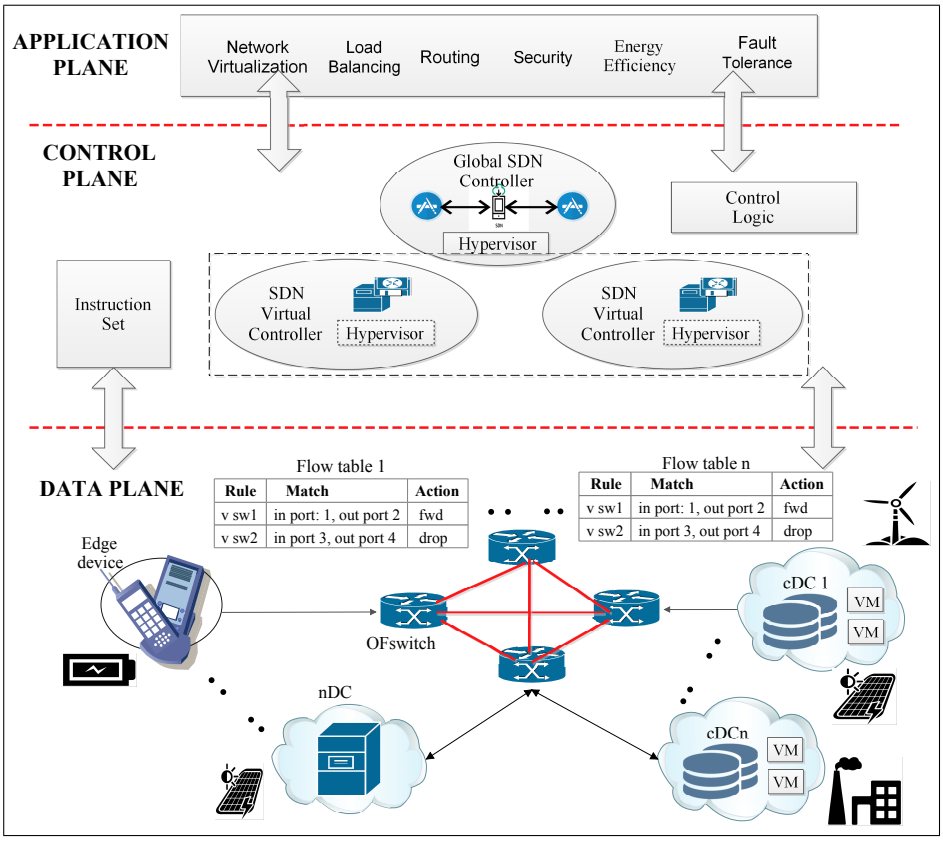


Figure 4.4: SDN-enabled control framework

For analysis of the proposed framework, different planes are discussed as below.

- **Data plane:** The data plane consists of forwarding devices such as-OFswitches, OFrouters, etc that connects end users to DCs. The data generated by various entities is routed through these forwarding devices on the basis of flow rules assigned by SDN controller. Moreover, the OF virtual switches are created as an instance of OF physical switches to virtualize the network resources. All the OFswitches stores their own flow table for fast forwarding of data. These flow tables act according to the forwarding decisions available in the instruction set provided by controller. In SDN architecture, the flow tables are small and consistent in order to simplify the flow table management. All the available flow tables are linked together using a pipeline. In oder to handle the data traffic flows across the communication channel which requires high bandwidth for large block of data transmission across different sites, the SDN-based control framework work efficiently. A load balancing rate (Υ) is used to control the data transmission across the communication channels as given below.

$$\Upsilon = \frac{1/j \times \sum_0^i L_i}{L_{max}} \tag{4.49}$$

  where, L_{max} is the maximum load a controller/switch can bear.

- **Control Plane:** The control plane acts as the central decision making plane where the SDN controller is located. SDN control is the brain of the SDN architecture. A centralized server houses the SDN controller that works on the basis of control logic provided by the control algorithm. The major task of the controller is to provide control decisions, commands, and instruction set to the forwarding devices for data flow from source to destination. The control logic used to define the instruction set may vary for different situations. The control algorithm reconfigures itself on the basis of feedback received from the application plane to provide better QoS. The control plane is responsible for the network policies and rules to govern the other two planes.

- **Application plane:** Different applications such as- network virtualization, load balancing, routing, security, fault tolerance, and energy efficiency are deployed at application plane. This plane is also known as management plane. It is also responsible for providing feedback to the control algorithm according to the QoS achieved by various applications.

It is highlighted in the existing proposals that network resources are one of the most energy consuming infrastructures after computing resources. So, in this direction, an energy efficient network load consolidation scheme is presented as below.

4.3.2 Energy-efficient network load consolidation scheme

In the proposed network load consolidation scheme, the major goal is to utilize the OF switches optimally in order to achieve energy efficiency. Generally, some of the switches are overloaded and others remain underloaded. So, the load on the overloaded and underloaded switches is consolidated to minimize the use of OF switches. The overall load is shifted to minimum number of switches thereby deactivating the idle switches to save energy. For this purpose, Fig. 4.5 shows the workflow of the proposed network load consolidation scheme.

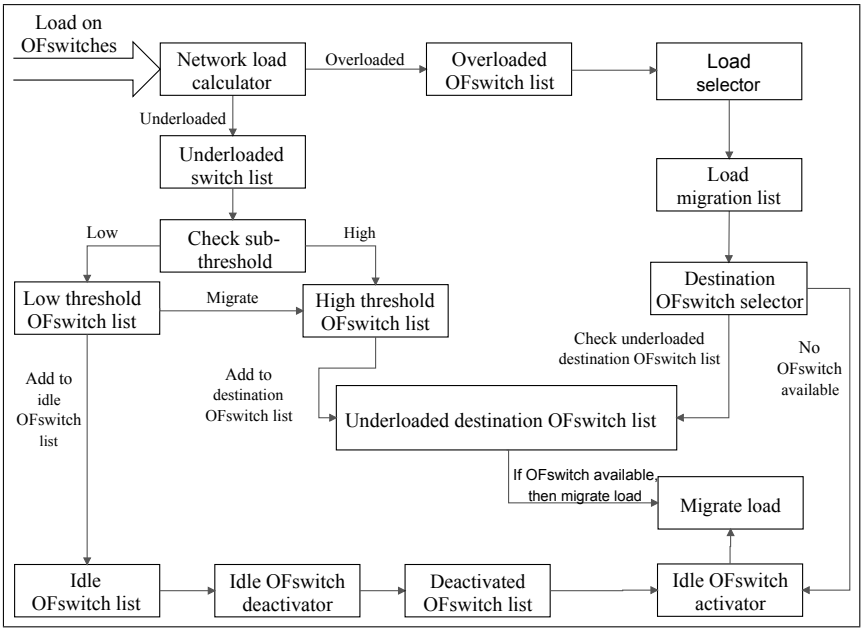


Figure 4.5: OFswitch load consolidation scheme

Different entities and steps related to the network load consolidation scheme are illustrated as below.

- **Network load calculator:** The incoming load on a running OFswitch is calculated in this phase. The load on a OFswitch is calculated as given below.

$$L_i^{sw}(t) = \frac{\sum_p \left(L_i^p\right)}{L_i^{cp}(t)} \times 100 \tag{4.50}$$

  where, L_i^p and L_i^{cp} denotes load on p^{th} port and total load capacity of the switch, respectively

  Using Eq. 4.50, the load at a OFswitch in current time slot is calculated. After this, the calculated load is compared with overload (OL^{thr}) and underload (UL^{thr}) thresholds. If a OFswitch has $L_i^{sw}(t)$ above OL^{thr}, then it is added to overload OFswitch list (OOL). Otherwise, the load on OFswitch is compared with UL^{thr}. If a OFswitch has $L_i^{sw}(t)$ below UL^{thr}, then it is added to underload OFswitch list (UOL).

- **Overload OFswitch list (OOL):** This list stores all OFswitches that has load above OL^{thr}.

- **Migration load selector:** Now, if a OFswitch is overloaded, then some of the load on the switch is migrated to any other switch in order to provide load balancing. For, this purpose, the load to be migrated is selected in this phase.

- **Migration load list:** This list includes the load to be migrated to another OFswitch.

- **Underload OFswitch list (UOL):** This list stores all OFswitches that has load below UL^{thr}.

- **Sub-threshold checker:** The load of OFswitches in UOL is compared with sub-threshold. If the load of OFswitches is lower than the sub-threshold value,

then such OFswitches are added to low threshold OFswitch list. All other OF switches are added to high threshold OFswitch list.

- **Low threshold OFswitch list:** This list stores all OFswitches that has load lower than sub-threshold value. All such load is migrated to OFswitches stored in high threshold OFswitch list and all these OFswitches are added to idle OFswitch list.

- **High threshold OFswitch list:** This list stores all OFswitches that has load higher than sub-threshold value. All load of OFswitches stored in low threshold OFswitch list is migrated to these OFswitches and all these OFswitches are added to underloaded destination OFswitch list.

- **Underloaded destination OFswitch list:** This list consists of underloaded OF switches that can be utilized to migrate the overload from migration list.

- **Idle OFswitch list:** There can be some OFswitches that becomes idle after performing load consolidation and balancing. Such OF switches are added to idle OF switch list.

- **Destination OFswitch selector:** This phase selects the destination OF switch where the load can be migrated.

- **Idle OFswitch deactivator:** In this module, the idle OF switches are deactivated in order to save energy.

- **Deactivated OFswitch list:** This list includes all OF switches that were deactivated as they were idle.

- **Idle OFswitch activator:** If in a case, an OFswitch is not available for migration in underloaded destination OFswitch list, then a OFswitch is activated from deactivated OFswitch list.

4.3.3 SVM-based workload classification scheme

The incoming user requests are classified on the basis of different parameters such as- resource requirements, delay-sensitivity, energy usage, and renewable energy available with BESS, etc [178]. Fig. 4.6 shows the model of workload classification.

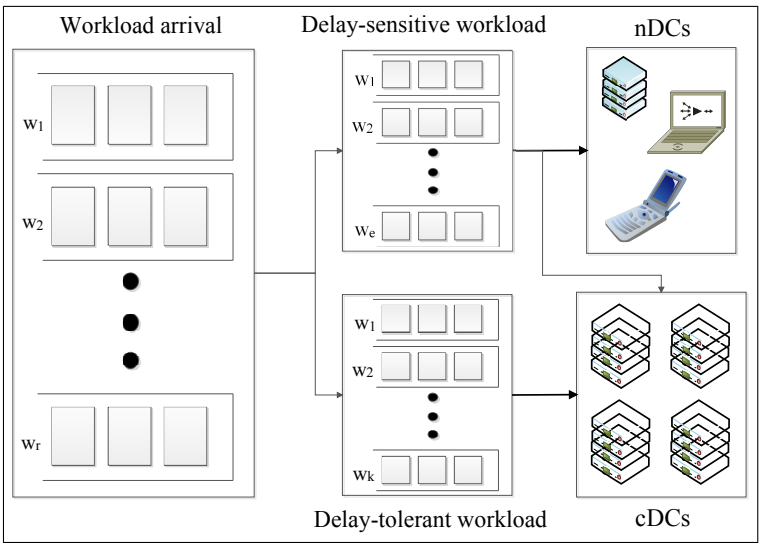


Figure 4.6: Workload classification

In this model, a SVM-based classification scheme is used to classify the incoming user requests. SVM is a supervised machine learning technique that is used to classify the data into distinct categories by minimizing the classification error using a decision function [179]. SVM is trained using training data and then it is used to predict the output class. Several classifiers are available for workload classification. However, SVM classifier has manifold advantages over other classifiers with respect to the current problem. Some of the advantages of SVM are listed as below:

- Kernel empowers SVM to choose the threshold that differentiates solvent and insolvent problems, that may not be linear, or even possess the same functional form for all types of data. This is due to the reason that its function is locally operated and non-parametric. So, they can operate with functional ratios, i.e., they have a non-monotone relation to the score and default probability or ones that are linearly independent and can act without working on each

non-monotone variable [180].

- In SVM, there is no need of supposition regarding the practical form of the transformation in order to make data linearly separable. The reason behind this aspect is that the kernel posses a non-linear transformation [180].
- If modeling parameters are chosen efficiently, then SVM generalizes the new samples effectively [180].
- SVM provides unique solution as compared to neural networks which provides multiple solutions based on local minima [180].

In this technique, a hyperplane is constructed to classify the data into distinct categories. Then, the hyperplane is extended to non-linear boundaries to maximize the margin of separation to achieve best classification results.

Given a training dataset of n points of form as follows:

$$(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n) \tag{4.51}$$

where, y_i is the class to which $\vec{x}_i$ belongs and are either 1 or -1.

Now, we want to maximize the hyperplane that separates the $\vec{x}_i$ for which $y_i = 1$ from any other group of points with $y_i = -1$. Such function is defined as distance between hyperplane and the nearest point ($\vec{x}_i$) from any of the groups. A hyperplane can be presented as given below.

$$\vec{w}.\vec{x} - b = 0 \tag{4.52}$$

where, w is normal vector to the hyperplane, and the parameter $\frac{b}{||\vec{w}||}$ indicates the offset of the hyperplane from the origin along the normal vector $\vec{w}$.

Now, in case the data is linearly separable, then two parallel hyperplanes can be used to separate two categories of data such that the distance between then is maximum. Such a region between two hyperplanes is known as 'margin'. The

halfway position between two hyperplanes is maximum margin hyperplane. Using this concept, the hyperplanes are give as below.

$$\vec{w}.\vec{x} - b = 1 \tag{4.53}$$

$$\vec{w}.\vec{x} - b = -1 \tag{4.54}$$

So, geometrically, the distance between two hyperplanes is given by $\frac{2}{||\vec{w}||}$. So, in such as case, in order to maximize the distance between two hyperplanes, $||\vec{w}||$ is minimized. Moreover, it is important to prevent data point falling on this margin. So, the optimization problem is defined as below.

$$Minimize||\vec{w}|| \tag{4.55}$$

$$s.t. \tag{4.56}$$

$$y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1, for\ i = 1, \ldots, n \tag{4.57}$$

Now, in case the data is not linearly separable, then the above problem is extended using following function.

$$\max\left(0, 1 - y_i(\vec{w} \cdot \vec{x}_i - b)\right) \tag{4.58}$$

Using the above function, the optimization problem reduces to

$$\left[\frac{1}{n}\sum_{i=1}^{n} \max\left(0, 1 - y_i(\vec{w} \cdot \vec{x}_i - b)\right)\right] + \lambda\|\vec{w}\|^2 \tag{4.59}$$

where, λ denotes the trade off between ensuring that the $\vec{x}_i$ lie on the correct side of the margin and increasing the margin-size.

Now, the above illustrated optimization problem can be defined as differentiable objective function by introducing a new variable (ζ) where, $\zeta_i = \max\left(0, 1 - y_i(w \cdot x_i + b)\right)$ for each $i \in \{1, \ldots, n\}$. The optimization problem can be written as a primal prob-

lem as follows.

$$Minimize \frac{1}{n}\sum_{i=1}^{n}\zeta_i + \lambda||w||^2 \tag{4.60}$$

$$\text{s.t. } y_i(w \cdot x_i + b) \geq 1 - \zeta_i \tag{4.61}$$

In order to construct a dual problem from above defined primal problem, it is solved using Lagrangian dual. The simplified problem obtained is given as below.

$$Maximize\ f(c_1 \ldots c_n) = \sum_{i=1}^{n} c_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i c_i (x_i \cdot x_j) y_j c_j, \tag{4.62}$$

$$\text{s.t. } \sum_{i=1}^{n} c_i y_i = 0, \text{ and } 0 \leq c_i \leq \frac{1}{2n\lambda} \text{ for all } i. \tag{4.63}$$

The variables c_i are given as below.

$$\vec{w} = \sum_{i=1}^{n} c_i y_i \vec{x}_i \tag{4.64}$$

The variable $c_i = 0$ if $\vec{x}_i$ lies on the correct side of the margin, and $0 < c_i < (2n\lambda)^{-1}$ if $\vec{x}_i$ lies on the boundary of the margin.

Before using the above mathematical model to maximize the margin between two hyperplanes, input data is normalized. In this step, the incoming data of user requests is normalized to give equal weightage to different parameters such as-computing resources, bandwidth, priority, delay-sensitivity, etc. By doing so, all the features are normalized in a range of [0,1]. To normalize the input data, a normalization function is given as below [181].

$$d^{'} = \alpha + \frac{(d - d_{min})(\beta - \alpha)}{d_{max} - d_{min}} \tag{4.65}$$

where, d is data value ($d \in D$), D is domain of d, $d^{'}$ is data value after normalization, d_{min} and d_{max} are minimum and maximum values in D, and α and β are minimum and maximum values in a specified output range.

In SVM, a kernel function is used to construct a decision boundary in order to classify the training set into two distinct parts. The selection of an appropriate kernel function is an important task in classification of data using SVM. Some of the popular kernel functions used to achieve a boundary function are illustrated as below [181–183].

$$k(x_i, x_j) = \begin{cases} (x_i \cdot x_j) & Linear\ kernel \\ (\gamma\ x_i \cdot x_j + b)^d & Polynomial\ kernel \\ exp(-\gamma|x_i - x_j|^2) & Radial\ basis\ function\ kernel \\ tanh(\gamma\ x_i \cdot x_j + b) & Sigmoid\ kernel \end{cases} \tag{4.66}$$

In this scheme, a non-linear Gaussian radial basis function (RBF) kernel is used. RBF is designed using a Gaussian function and is given as below.

$$k(x_i, x_j) = exp(-\gamma|x_i - x_j|^2) \tag{4.67}$$

where, x_i denotes support vector, x_j is the current value of data, and $\gamma = \frac{1}{2\sigma^2}$.

In the present scheme, two modeling parameters (Ç and γ) are chosen before using RBF kernel function. These modeling parameters directly effect the accuracy of the SVM classifier. So, it is very important to select optimal values of Ç and γ. Here, a library for support vector machine (LIBSVM) is used for optimizing the values of these modeling parameters based on grid search and cross validation. In cross validation, the training set is divided into k folds. Now, *k-1* subsets are randomly chosen in order to train the classifier and the remaining subsets are used for testing. After this process, the grid-search method is used to select the values of Ç and γ. Initially, the loose grid search along with cross-validation is performed for Ç and γ. After this, a finer grid search is performed with respect to the neighborhood pair which selects the best cross-validation accuracy in order to get the best values of Ç and γ. Finally, the classifier is trained using this pair and thereby achieves the best possible cross-validation accuracy [181].

The incoming user requests are classified on the basis of delay using Algorithm 6. In this algorithm, for all features (f_i), the dataset is normalized using Eq. 4.65. Once the dataset is normalized, it is stored in a list L_{nr} (line 1-3). Now, the RBF kernel function is applied on the normalized dataset using Eq. 4.67. After this step, the modeling parameters Ç and γ are selected. Using Ç and γ, the classifier is trained (line 4-6). Once the classifier is trained, then the SVMclassifier is ready to classify the normalized dataset (line 8). Now, using the SVMclassifier, the workload w is classified. If workload is delay-sensitive, then it is added to list (L_{ds}). Otherwise, the workload is delay-tolerant and it is added in list (L_{dt}) (line 9-17).

Algorithm 6 Workload classification algorithm

Input: Workload (w)
Output: L_{ds}, L_{dt}

1: **for** (i=1; i $\leq$ n; i++) **do** ▷ All features f_i
2: Normalize d using Eq. 4.65
3: Store $d \rightarrow L_{nr}$
4: Apply RBF using Eq. 4.67
5: Select modeling parameters Ç and γ
6: Train SVM $\rightarrow$ Ç and γ
7: **end for**
8: **return** SVMclassifier ▷ Classifier trained
9: **for** (i=1; i $\leq$ n; i++) **do**
10: Call SVMclassifier
11: Classify w
12: **if** $w \rightarrow ds$ **then**
13: Add $w \rightarrow L_{ds}$
14: **else**
15: Add $w \rightarrow L_{ds}$
16: **end if**
17: **end for**

4.3.4 Energy-efficient server consolidation scheme

In the server consolidation scheme, the workload on the physical hosts is consolidated to a minimum number of servers. By doing this, some of the over-utilized servers shed their load to the underutilized server. In this way, many underutilized servers becomes idle and are switched OFF in order to save energy. In this regard, Algorithm

7 shows the workflow of the proposed server consolidation scheme.

Algorithm 7 Server consolidation scheme

Input: Host S_i
Output: Destination host (S_i^{dst})

1: **for** (s=1; s $\leq$ S; s++) **do** ▷ $s \leftarrow$ Number of servers
2: Compute (U_i^s) using Eq. (4.17)
3: **if** ($U_i^s < U_{ul}^{thr}$) **then**
4: Add $S_i \rightarrow L_{ul}$
5: **if** S_i **then** $\leq U_{ul}^{sthr}$
6: Add in list L_{lul}
7: **else**
8: Add in list L_{hul}
9: **end if**
10: Migrate load on servers in $L_{lul} \rightarrow L_{hul}$
11: Server in list L_{lul} to idle mode
12: Deactivate idle servers
13: Add deactivated servers in L_{dact}
14: **else if** ($U_i^s > U_{ol}^{thr}$) **then**
15: Add $S_i \rightarrow L_{ol}$
16: Select $w \rightarrow$ migrated
17: Check for R_s^{rq} by w at S_i^{dst}
18: Map R_s^{rq} with R_s^{av} in $S_i \rightarrow L_{hul}$
19: **if** R_s^{rq} are available at $S_i \rightarrow L_{hul}$ **then**
20: Add and sort S_i in L_{dst}
21: Migrate load to S_i^{dst} in L_{dst}
22: **else**
23: Select S_i from L_{dact}
24: Activate selected S_i from L_{dact}
25: Migrate load on the selected S_i^{dst}
26: **end if**
27: **end if**
28: **end for**

Initially, the utilization (U_i^s) of all servers is computed using Eq. (4.17). If utilization of servers is more than the threshold utilization level, then such servers are added to underloaded server list (L_{ul}). Now, the servers in L_{ul} are compared with a sub-threshold value. If the utilization of servers is lower than the sub-threshold value, then such servers are added to L_{lul}. All other servers are added to L_{hul}. All the load on servers stored in list L_{lul} is migrated to servers stored in L_{hul}. These servers are now completely idle and are deactivated to save energy (line 1-13).

Now, in the other case, if utilization of servers is more than the threshold level

of utilization, then such servers are added to overloaded server list (L_{ol}). All the servers added in list L_{ol} consume excess energy and the extra load on these servers need to be migrated to another server. For this purpose, first the workload that can be migrated is selected. After this, the required resources to handle the selected workload are checked and then mapped with available resources ate each server stored in list L_{hul} (line 14-18). If the required resources are available at servers listed in L_{hul}, then such servers are sorted and accordingly the load is shed from overloaded servers to these selected servers. However, if the required resources are not available at any of the servers stored in L_{hul}, then the servers stored in list L_{dact} are activated and load is migrated to all such servers (line 19-28).

4.3.5 Two-stage game-theoretic workload scheduling scheme

In this scheme, the incoming workload classified using SVM is scheduled among n-DCs and c-DCs to achieve sustainability with respect to available renewable energy. This is done to achieve minimum carbon emissions associated with the energy consumption of DCs and easing the load on the prower grid. In order to schedule workload among various available nDCs and cDCs, the conditions mentioned in Table 4.1 may exist.

Table 4.1: Conditions for sustainable scheduling

Case No.	**Decision**	**Bandwidth**	**Computing resources**	**Energy (RES/BESS)**
Case 1	True	✓	✓	✓
Case 2	True	x (*)	✓	✓
Case 3	False	✓	x	✓
Case 4	True (*)	✓	✓	x (*)
Case 5	False	x	✓	✓
Case 6	False	✓	x	x
Case 7	False	x	x	✓
Case 8	True (*)	x (*)	✓	x (*)
Case 9	False	x	x	x

- (*) True only if virtual network resources are available (for bandwidth) and energy is drawn from grid (for energy).

In this table, Case 1 is always true, whereas, case 2 is true only if virtual network resources are available. Case 4 is also true in special condition, i.e, if energy is drawn from grid. Moreover, case 8 is true if both virtual network resources are available and energy can be drawn from grid. Rest all other cases are always false.

Now, to schedule user requests for cloud resources, a two-stage game is formulated. In this game, cloud scheduler act as first player and multiple nDCs and cDCs act as other players. For this purpose, Algorithm 8 is designed for the proposed two-stage game. The moves of the players in two-stage game are shown in Fig. 4.7.

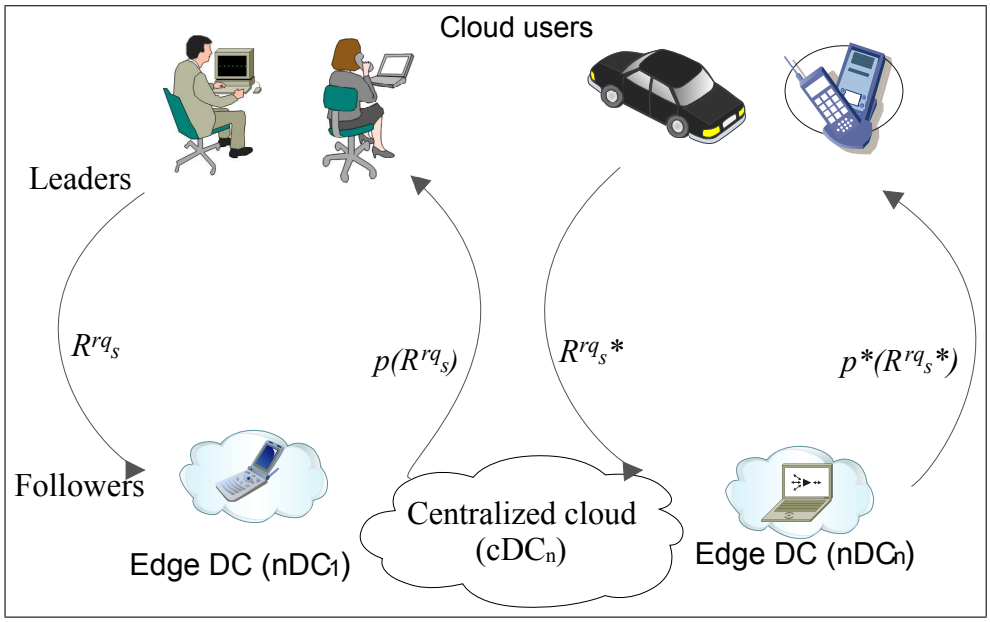


Figure 4.7: Schematic diagram for proposed two-stage game

In Algorithm 8, in Stage I, resources required to handle each workload w are compared with resources available with j servers of i nDCs/cDCs. If R_s^{rq} are available, then utility of such nDC/cDCs is computed using Eq. 4.32. If the utility shows a growth with respect to previous time instance, then price for such resources is calculated. Moreover, to achieve sustainability, the carbon index (φ_i) is also computed using Eq. 4.30. If φ_i is lower than the desired carbon index, then such nDC/cDCs are added to list L_1. After this ass all the other nDC/cDCs with carbon index lower than the desired carbon index in list L_{hci}.

Algorithm 8 Two-stage game for workload scheduling

Input: r, R_s^{rq}
Output: jni pair, destination nDC/cDC

for (i=1; i $\leq$ I; i++) **do** ▷ number of nDCs/cDCs
  for (s=1; s $\leq$ S; s++) **do** ▷ number of servers at each nDCs/cDCs
    for (w=1; w $\leq$ W; w++) **do** ▷ workload
      Check resources R_s^{rq} required ▷ Stage I
      if R_s^{rq} is available **then**
        Compute utility $\mho_i$ using Eq. 4.32
        if $\mho_i(t) > \mho_i(t-1)$ **then**
          Compute φ_i using Eq. 4.30
          if $\varphi_i < \varphi_i^{ds}$ **then**
            Add all such nDcs/cDCs to list L_1
          else
            Add nDC/cDCs with $\varphi_i > \varphi_i^{ds}$ in list L_{hci}
          end if
        end if
        Compute price P_i^j
      end if
      With P_i^j, compute $\mho_j$ using Eq. 4.31
      if $U_j(i) > U_j(i^*)$ **then**
        Add j^{th} nDC/cDC above all other nDC/cDCs in list L_2
      end if
    end for
  end for
end for
for (f=1; f $\leq$ F; f++) **do** ▷ number of flow paths
  Compute $\mho_n$ using Eq. 4.38
  if $U_n(f) > U_n(f^*)$ **then**
    Add f^{th} flow path above all other nDC/cDCs in list L_3
  elseSelect any other f
  end if
end for
Map all available jni pairs ▷ Stage II
Compute U_{jni} using Eq. 4.39
if $U_{jni} > U_{jni}^*$) **then**
  Set $\delta_{jni} == 1$
  Select jni pair
end if

However, if in case utility is not better for nDC/cDCs, then the concerned request cannot be served. Moreover, if required resources are not available at any server, then also such servers are not added in the list. Once the servers and nDC/cDCs are selected, the price (P_i^j) for resources to be charged by each of them is announced

(line 1-16). Now, on the basis of P_i^j, the end users computed their utility using Eq. 4.31. If the utility of j user for i^{th} nDC/cDC is more than the utility computed with respect to all other nDC/cDCs, then such nDC/cDC is added above others in list L_2 (line 17-23).

In Stage I, the preferable lists of nDC/cDCs (L_1) and user preference for each available DCs (L_2) are sorted. Now, the major task is to select the optimal flow path (f). The utility of network with respect to available flow paths is calculated using Eq. 4.38. Now, if the utility of network with respect to f^{th} flow path is more than the utility of network with respect to all other paths, then such flow path is added above others in list L_3 (line 24-30). Now, in Stage II, using all three utilities and lists, map all available pairs (jni). Compute the combined utility of each pair using Eq. 4.39. The pair that has higher utility than all other pairs is selected for each workload of end users. For such pair (jni), the value is decision variable is set to 1. Similarly, the next pair is selected (line 31-38). In this way, using this algorithm, the nDC/cDCs those are selected are having sufficient amount of renewable energy (on the basis of carbon index), to handle the workload of end users. Moreover, an optimal flow path is selected to ensure high latency services.

Summary of the Chapter: The Chapter 4 presents an efficient scheme for energy management for sustainability of Cloud DCs in Edge-Cloud Environment using SDN. The major goal of the proposed scheme is to classify the incoming work load on the basis of various parameters and schedule the delay sensitive workload to the edge devices. For this purpose, a support vector machine (SVM)-based workload classification approach is presented. Finally, a two-stage game for workload scheduling for sustainability of DCs is designed, which achieves energy efficiency and optimal utilization of network and computing resources using the designed consolidation schemes.

Chapter 5

Renewable Energy-aware Resource Allocation Scheme

In this chapter, an energy-aware resource allocation scheme is proposed using a Stackelberg game for energy management in cloud-based DCs[1]. For this purpose, a cloud controller is used to receive the requests of users which then distributes these requests among geo-distributed DCs in such a way that the energy consumption of DCs is sustained by RES. However, if energy consumption of DCs is not sustained by RES then the energy is drawn from the grid. The requests of users are routed to the DC which is offered lowest energy tariff from the grid. For this purpose, a Stackelberg game for energy trading is also proposed to select the grid offering lowest energy tariff to DCs.

5.1 System Model

The proposed system model is divided into three layers as shown in Fig. 5.1. The layers of proposed system model are discussed as below.

[1]The content of this chapter is taken from:

- G. S. Aujla, M. Singh, N. Kumar and A. Zomaya, "Stackelberg Game for Energy-aware Resource Allocation to Sustain Data Centers Using RES," IEEE Transactions on Cloud Computing, 2017. doi: 10.1109/TCC.2017.2715817

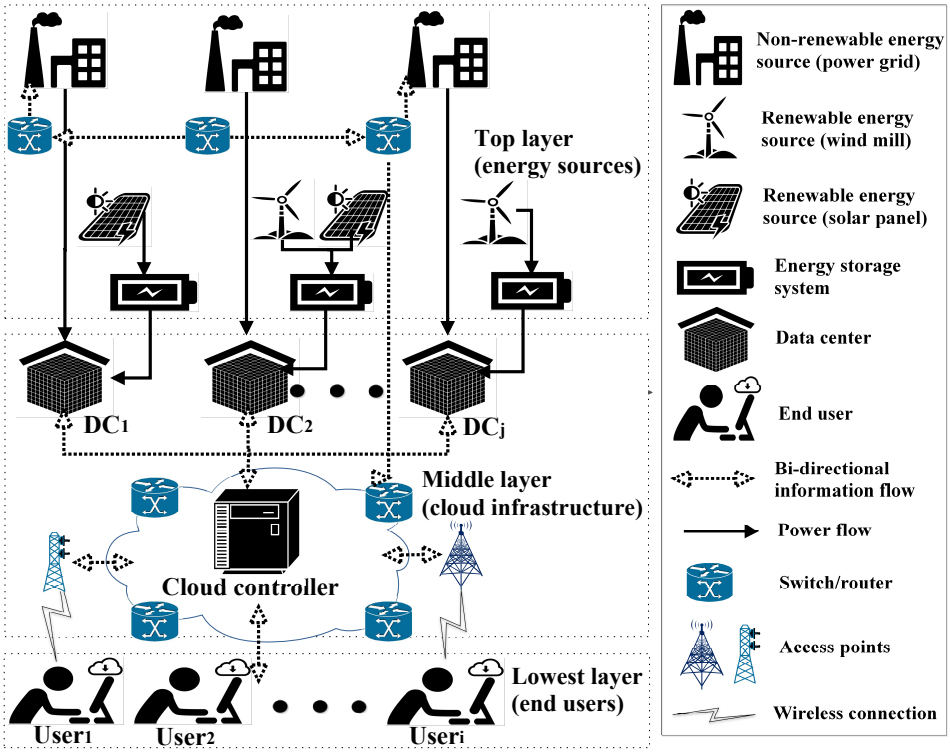


Figure 5.1: A three-layered system model

5.1.1 End user layer

The end user layer is the lowest layer of the proposed system. This layer consists of i end users who want to access computing resources such as network, storage, and servers. Such resources are provided to the end users by CSPs. The end users sends the requests for required resources to the CSPs. After receiving the requests, CSPs computes price (P_i) to be charged for the requested resources. This price is computed on the basis of time and quantity of resources to be accessed. After computing the price, CSPs announce it to the end users. If the price is acceptable to the end users, they pay the price and access the requested resources.

5.1.2 Cloud infrastructure layer

The cloud infrastructure layer is the middle layer of the system model. It consists of j geo-distributed DCs along with a cloud controller. The DCs are physical facilities which are equipped with information technology infrastructure such as servers, storage devices, and network devices. DCs act as backbone for CSPs and they provide

all CC operations [15]. To facilitate DCs and end users, cloud controller is used as a decision maker and resource allocation authority. Fig. 5.2 shows an interaction between end users and cloud infrastructure in the proposed system model.

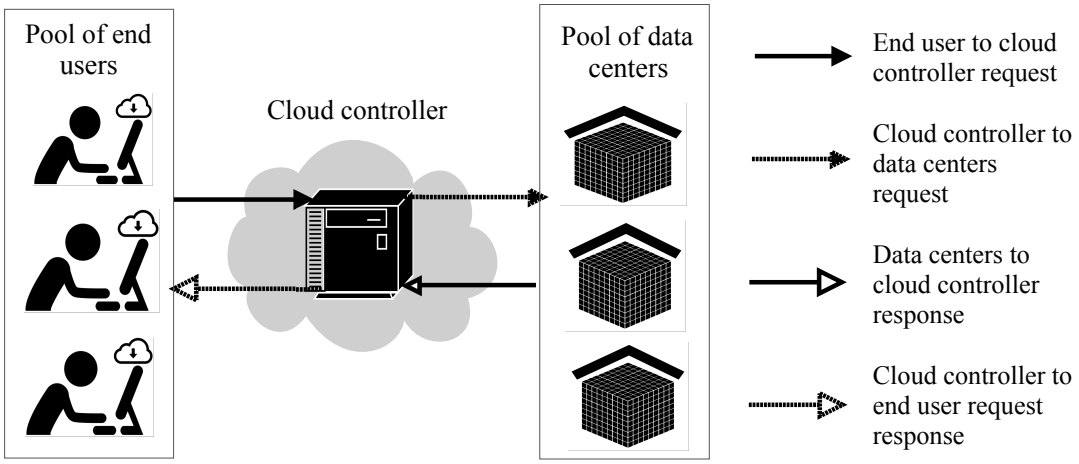


Figure 5.2: Interaction between middle layer and lowest layer

The controller receives the requests for accessing resources from end users and reply them with the price (P_i) to be charged for the requested resources. The cloud controller decides the price (P_i) of resources on the basis of various factors such as; (i) time and quantity of requested resources, (ii) available resources with DC, (iii) status of resources utilized (iv) energy available with storage units connected to DCs, (v) energy generated by RES, and (vi) energy price offered by grid. Using the above factors, cloud controller selects the DC which can serve the user request and decides the price to be charged accordingly. Once the users accept and pay the price, the requested resources are allocated to them for the given time period. The cloud controller allocates the users requests among geo-distributed DCs on the basis of resource availability with them and available energy with RES associated to them. This allocation is done in such a manner that the energy consumption of DCs is sustained by the energy produced by RES. Further, if the power consumption of DCs is not fulfilled by RES then, cloud controller trades for energy with SG on behalf of the DCs. After receiving the energy tariffs from SG, cloud controller selects the DCs which are offered minimum energy tariff by the grid. After the selection of such DCs, all the user requests are routed to such DCs.

5.1.3 Energy source layer

The topmost layer consists of various sources of energy. In this paper, two types of energy sources are considered: (i) RES (solar and wind energy) and (ii) grid source. The renewable sources are used as primary sources of energy for powering DCs. The grid energy is used only if the power demand of DCs is not fulfilled by RES. In such a case, cloud controller interacts with grid to trade for energy. Fig. 5.3 shows interaction between cloud controller and energy sources. The Solar energy is directly dependent on hours of sunshine during the day and wind energy is dependent on wind speed [127]. The hours of sunshine and wind speed are variable in nature. Hence, due to such variations, energy production by RES do not always align with the energy demand of DCs. At some period of time, energy production by RES may be more than the energy demand of DCs. So, in all such cases, energy need to be stored in some storage system so that it could be used at time when the generation is low or negligible. Hence, energy storage system (ESS) is used to charge and discharge energy from RES at a particular time.

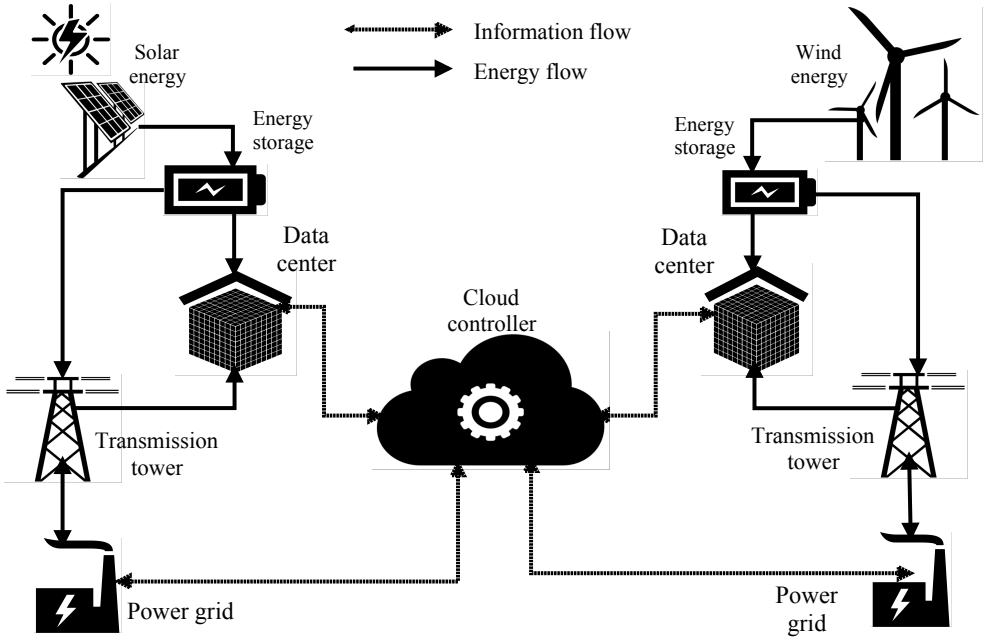


Figure 5.3: Interaction between middle layer and top layer

ESS comprise of batteries which are used to store the energy generated by RES. It supplies the energy to DCs as per their requirements. However, ESS also exhibits

loss due to self-discharge rate (L_{ess}) which is nearly 1-5 percent per hour [171]. A controller is used to manage the incoming and outgoing energy in ESS. The energy required by DC is drawn from ESS which receives it from various sources. The total energy available with ESS (E_j^{ess}) after managing the demand of DCs is given as below.

$$E_j^{ess}(t) = \left(E_j^{ess}(t-1) + E_j^{ren} - E_j\right) L_{ess} \tag{5.1}$$

where, E_j^{ren} is energy generated by RES, and E_j is the energy required by j^{th} DC.

Some aspects about solar and wind energy are discussed in the next segment.

- **Solar energy:** Solar energy is widely used RES as an alternative for traditional non-renewable energy sources. The radiant heat and light from Sun is converted into solar power using various methods. The photovoltaics (PV) panels are widely used to capture Sunlight. However, PV panels are effected by various factors such as: (i) conversion efficiency of PV panel, (ii) radiation angle of Sunlight on PV panels, and (iii) loss due to exedence of temperature of ESS corridor. The conversion efficiency (η) of PV panel is the ratio of solar energy received and amount of electric energy generated. Further, the radiant angle of sunlight (a = $cos\alpha$) on PV panel has a major impact on energy generation which depends on angular deviation (α) of sunlight. Moreover, energy loss due to exedence of temperature of corridor (L_{exe}) also occurs when weather is cold. Considering the aforementioned factors, energy (E_{solar}^{ren}) generated by capturing Sunlight using PV panels is given as below [184].

$$E_{solar}^{ren} = (1 - L_{exe})\ \eta\ S_{panel}\ cos(\alpha)\ \text{Ŗ} \tag{5.2}$$

  where, S_{panel} denotes panel size and Ŗ denote solar radiations.

- **Wind energy:** Wind is fast growing RES which uses wind turbines to capture the wind power [185]. Wind turbine is used to capture the kinetic energy of wind using a rotor with two or more blades which are mechanically hooked

up to an electric generator. The electric power (E^{ren}_{wind}) generated by a typical wind turbine is given as below [169].

$$E^{ren}_{wind} = \frac{1}{2} \left[C_p \ \rho \ A \ v^3\right] \tag{5.3}$$

where, C_p denotes power coefficient of the rotar or rotar efficiency, A denotes the area swept by rotar blades, ρ is the air density, and v is the wind speed. For a high-speed, two-blade wind turbine, maximum rotar efficiency that could be achieved is equal to 0.59 [169]. For practical consideration, maximum rotar efficiency is considered as 0.5. This generated energy is stored in an ESS where it may suffer loss due to exedence of temperature of corridor (L_{exe}). Using the above parameters, the electric power (E^{ren}_{wind}) generated is given as below

$$E^{ren}_{wind} = \frac{1}{4} \left[\rho \ A \ v^3\right] \ L_{exe} \tag{5.4}$$

5.2 Problem Formulation

In the system model, each DC is considered having k set of servers physically connected with l set of links between servers. Each server has m type of n resources, i.e., memory, storage, CPU associated with it. A set of i users request for resources from cloud controller at time t. The i^{th} end user requests for m^{th} type of resources ($\Re^{req}_{i,m}$) from cloud controller. The cloud controller checks for the available resources ($\Re^{avl}_{j,m}$) with j^{th} DC. If the requested resources are available with j^{th} DC, cloud controller computes the price (P_i) to be charged for the requested resources. Once price is accepted and paid by i^{th} user, cloud controller allocates requested resources from k^{th} server of j^{th} DC.

The required resources ($\Re^k_j(req)$) from k^{th} server of j^{th} DC are total resources

requested by all users as given below.

$$\Re_j^k(req) = \sum_{i,m} \Re_{i,m}^{req} \tag{5.5}$$

The load ($\Re_j^k(load)$) on k^{th} server of j^{th} DC is given as below.

$$\Re_j^k(load) = \sum_{i,m} \Re_{i,m}^{alloc} \tag{5.6}$$

where, $\Re_{i,m}^{alloc}$ is m^{th} type of resources allocated to i^{th} user.

The resources ($\Re_j^k(avl)$) with the k^{th} server of j^{th} DC that are available to be allocated are stated as below.

$$\Re_j^k(avl) = \Re_j^k - \sum_{k} \Re_j^k(load) \tag{5.7}$$

where, $\Re_j^k$ is the total capacity of k^{th} server of j^{th} DC.

The load ($\Re_j(load)$) on j^{th} DC is given as below.

$$\Re_j(load) = \sum_{j} \Re_j^k(load) \tag{5.8}$$

The resources ($\Re_j(avl)$) with j^{th} DC that are available to be allocated are stated as below.

$$\Re_j(avl) = \Re_j - \sum_{j} \Re_j(load) \tag{5.9}$$

The DCs deal with dynamic workloads, i.e., resource usage vary with respect to time. When the requested resources exceed the available capacity of resources, then a violation of SLA may exist between CSP and end user. In such a case, the CSP has to bear a penalty (C_j^{pen}) computed as below [186].

$$C_j^{pen} = C_j^{slav}\ T_j^{slav} \tag{5.10}$$

where, C_j^{slav} is the cost of SLA violation per unit time and T_j^{slav} is the time duration of SLA violation.

The SLA violations (SLA_j^v) are measured with respect to time during which j^{th} server is experiencing threshold utilization (T_j^{thr}), total time during which it is active (T_j^{act}), and performance degradation (D_j^{mig}) due to migration and is stated as below. [186].

$$SLA_j^v = \frac{1}{k} \sum_{j=1}^{J} \frac{T_j^{thr}}{T_j^{act}} D_j^{mig} \tag{5.11}$$

The energy consumption of DCs is decided on the basis of utilization of servers by the end users. The utilization of k^{th} server of j^{th} DC (U_j^k) after being provisioned to end users is specified as below.

$$U_j^k = (\frac{\Re_j^k(load)}{\Re_j^k}) \times 100 \tag{5.12}$$

Considering the utilization of servers, energy consumption (E_j^k) of k^{th} server of j^{th} DC is as given below.

$$E_j^k = E_{kj}^{idle} + (E_{kj}^{max} - E_{kj}^{idle})\ U_j^k \tag{5.13}$$

where, E_{kj}^{idle} is energy consumption of k^{th} server of j^{th} DC when it is idle and E_{kj}^{max} is maximum energy consumption of k^{th} server of j^{th} DC.

The total energy consumption of j^{th} DC is given as below.

$$E_j = \sum_k E_j^k + E_j^{cl} + E_j^{oth} \tag{5.14}$$

where, E_j^{cl} is the energy consumed for cooling of j^{th} DC and E_j^{oth} is the energy consumed on other activities at the j^{th} DC.

The energy consumption of k^{th} server of j^{th} DC is maximum when utilization of servers is maximum. Each server consumes about 60 percent of energy even when it is idle [21]. If k^{th} server of j^{th} DC is idle then it can be hibernated to save energy

and it can be activated whenever required.

The energy generated by RES (E_j^{ren}) at the j^{th} DC is given below.

$$E_j^{ren} = E_{solar}^{ren} + E_{wind}^{ren} \tag{5.15}$$

The energy generated by RES is stored in a ESS. For a DC to be self-sustainable, it must satisfy the condition given below.

$$E_j = E_j^{ren} \tag{5.16}$$

Due to intermittency, there may be a time when the energy demand of j^{th} DC is not fulfilled by RES. In such a case, j^{th} DC draws energy from grid after paying price (P_j^{grid}). So, energy consumption of j^{th} DC is given as below.

$$E_j = E_j^{ren} + E_j^{grid} \tag{5.17}$$

where, E_j^{grid} is the energy drawn by j^{th} DC from grid.

Keeping in view, all the aforementioned points, the main objective of the proposed scheme is to sustain the energy consumption of DCs for each time slot t using RES. Hence, the objective function is given as below.

$$E_{obj} = Minimize \sum_{t,j} \big(E_j(t) - E_j^{ren}(t)\big) \tag{5.18}$$

subject to following constraints

$$E_j^{ren}(t) \ > \ 0 : \ \forall j \tag{5.19}$$

$$E_j(t) \ > \ 0 : \ \forall j \tag{5.20}$$

If the energy demand of DCs is not fulfilled by RES then, it has to draw energy from grid. The objective function for this case is illustrated as below.

$$Minimize \sum_{t,j} E_j^{grid}(t) \tag{5.21}$$

subject to following constraints

$$E_j(t) > E_j^{ren}(t) : \forall j \tag{5.22}$$

$$E_j(t) > 0 : \forall j \tag{5.23}$$

$$E_j^{ren}(t) > 0 : \forall j \tag{5.24}$$

5.3 Game Formulation

In this paper, the proposed scheme works in two steps. In the first step, the objective of the proposed scheme is to sustain the DCs using energy generated from RES. For this purpose, an energy-aware resource allocation scheme is formulated using *multi-leader single-follower* Stackelberg game. In this scheme, various end users (multiple leaders) who want to access resources are distributedly located and do not cooperate with each other. The end users interact with cloud controller (single follower) which is responsible for announcing price and allocation of the resources. The cloud controller allocates the resources in such a way that energy consumption of DC is sustained using RES. For this purpose, it requires information about the available resources and energy with DCs. However, at some instance, due to intermittency of RES, the DCs have to be powered by grid. In such a situation, the second step comes into action. The cloud controller is responsible to select the DCs which are offered lowest energy tariff by grid and routes the end users requests to such DCs. To achieve this, a *single-leader multi-follower* Stackelberg game is formulated for energy trading. In this scheme, cloud controller (single leader) interacts with the different grids (multiple followers) to which each geo-distributed DC is connected.

A Stackelberg game is a special case of non-cooperative games. It is two-period hierarchical game with players classified as leaders and followers [187]. The leaders have a benefit to initiate the game and enforce their moves on followers. The followers observe the moves of leaders and revert with a best response. On the other hand, leader knows the above fact and have already played their best move [188]. Hence, by following Stackelberg game, both leaders and followers reach to an equilibrium state to maximize their payoffs. Stackelberg game has multiple advantages over other games such as, (1) suited for distributed environment, (2) addresses the economical aspects, (3) both players move in a sequential pattern, and (4) non-cooperative nature. In both the above discussed steps, the game proceeds individually in a distributed manner. Moreover, to study the economical benefits of both cases Stackelberg game is best suited for such an environment. Further, both the schemes moves in a sequential pattern similar to the trend followed by players in Stackelberg game.

The end users, cloud controller, and grid finalize their part of decisions on the basis of their payoffs. The terms price, cost, and revenue are considered while calculating the utility functions. Price refers to the amount charged to sell a product, cost is the amount incurred to manufacture the product, and revenue is the amount that producer receives after selling a product. The utility functions used to compute the payoffs for the proposed scheme are discussed as below.

Utility function of end users: A concave revenue function is considered which depicts the revenue available with i^{th} end user. The end users uses the revenue (R_i) to pay the price (P_i) to the cloud controller for accessing the required resources. The utility function ($\mho_i$) of i^{th} end user is the difference of revenue available with it and the price it has to pay for accessing the required resources. The utility function ($\mho_i$) of i^{th} end user is given as follows.

$$\mho_i = R_i - P_i \tag{5.25}$$

If the utility function ($\mho_i$) of i^{th} end user at present time slot is more than the utility function at previous time-slot then the end user accepts the price otherwise it rejects the price.

Utility function of cloud controller: The objective of cloud controller is to maximize revenue by selling the resources available with it at a price (P_i). The total amount of revenue (R_j^{cld}) generated by cloud controller depends on the price (P_i) charged from end users and is given as below.

$$R_j^{cld} = \sum_i P_i \tag{5.26}$$

The resources which are accessed by i^{th} end user after paying price (P_i) incurs some cost (C_j^{cld}). The total cost incurred on the resources allocated by cloud controller depends on maintenance cost (C_j^{mn}), migration cost (C_j^{mig}), SLA violation penalty, and energy consumption price (P_j^{grid}). An opportunity cost (C_j^{opor}) is also charged to compensate the end users who access resources at high price when DC is powered by gird. The total cost (C_j^{cld}) incurred on resources allocated to the end users is given as below.

$$C_j^{cld} = C_j^{mn} + P_j^{grid} + C_j^{pen} + C_j^{mig} + C_j^{opor} \tag{5.27}$$

The price (P_j^{grid}) incurred on purchasing energy (E_j^{grid}) from grid is an important part of total cost (C_j^{cld}). This component is accounted only when DCs draw energy from the grid. The price (P_j^{grid}) that cloud controller pays for energy (E_j^{grid}) drawn from grid is given as below.

$$P_j^{grid} = \wp_j^{grid}\ E_j^{grid} \tag{5.28}$$

where, $\wp_j^{grid}$ is the price coefficient, i.e., rate of energy.

The cloud controller is a decision making authority for resource allocation to end users as well as energy trading with grid. It finalizes the price (P_i) to be charged for resources provided to the end users. Further, it also decides the price (P_j^{grid})

to be paid for drawing energy from grid. Hence, two different utility functions are defined for both roles of cloud controller. For resource allocation to end users, utility function ($\mho_j^{cld}(1)$) of cloud controller for j^{th} DC is given below.

$$\mho_j^{cld}(1) = R_j^{cld} - C_j^{cld} \tag{5.29}$$

For energy trading with grid, utility function ($\mho_j^{cld}(2)$) of cloud controller for j^{th} DC considers only price to be paid for energy and is given below.

$$\mho_j^{cld}(2) = R_j^{cld} - P_j^{grid} \tag{5.30}$$

The cloud controller finalizes the price only when its utility function shows profit as compared to previous instance.

Utility function of grid: The objective of grid is to maximize revenue by selling available energy with it to DCs. The revenue (R_j^{grid}) generated by grid connected to j^{th} DC is based on the price (P_j^{grid}) which cloud controller pays for energy drawn from grid and is given as follows.

$$R_j^{grid} = \sum_j P_j^{grid} \tag{5.31}$$

The energy (E_j^{grid}) sold to j^{th} DC for a price (P_j^{grid}) incurs some cost (C_j^{grid}). This cost consists of generation cost (C_j^{gen}), operational cost (C_j^{opr}), and maintenance cost (C_j^{mnt}). The total cost (C_j^{grid}) incurred on energy supplied to j^{th} DC is defined below.

$$C_j^{grid} = C_j^{gen} + C_j^{opr} + C_j^{mnt} \tag{5.32}$$

The utility function ($\mho_j^{grid}$) of grid connected to j^{th} DC is the difference of revenue generated by selling energy and the cost incurred on generation and maintenance of

energy. The $\mho_j^{grid}$ of grid connected to j^{th} DC is given below.

$$\mho_j^{grid} = R_j^{grid} - C_j^{grid} \tag{5.33}$$

The grid finalizes the price if the utility function shows some profit as compared to previous instance.

The algorithms designed for the proposed resource allocation scheme are discussed as below.

5.3.1 Stackelberg game for energy aware resource allocation

A *Multi-leader single-follower* Stackelberg game is formulated for energy-aware resource allocation. The i number of end users act as *multi-leader* and cloud controller act as *single follower*. The major aspects related to resource allocation are SLA satisfaction and QoS guarantee while minimizing energy consumption of DC. For this purpose, Algorithm 9 has been designed to efficiently schedule the resources among the end users with QoS guarantee and SLA satisfaction. Generally, the QoS is computed in terms of SLAs with respect to the availability, delay, and response time.

The first move is for leaders, i.e., i end users who request the cloud controller for $\Re_{i,m}^{req}$ resources (line 1). The received requests are classified according to the type of workload into appropriate queue. Multi-level queuing scheme is used to schedule the user request on the basis of varying workloads. The different workloads are scheduled using different queues with different scheduling schemes. The workloads with high priority (P_{high}) are added to queue L_h and are scheduled using prioritized preemptive round robin (PPRR) scheme (line 2-35). The non-prioritized requests are added to an queue L_n and are scheduled using first come first serve (FCFS) scheme (line 36-37). After the incoming requests are added to appropriate queue, second move in the game is for followers, i.e., cloud controller. The cloud controller sort the DCs which could allocate the requested resources using Algorithm 10 and computes the price (P_i) for requested resources (line 38-40).

Algorithm 9 Algorithm for energy-aware resource allocation

Input: Resource requests by end users $\Re_{i,m}^{req}$, *TimeQuantumn*, *BurstTime*
Output: DC which is allocated the request of users, P_i, $\mho_i$

Resource requests ($\Re_{i,m}^{req}$) by i end users ▷ Leader move
Check the type of workload of all the requests
if $\Re_{i,m}^{req} = P_{high}$ **then**
  while true **do**
    Add $\Re_{i,m}^{req}$ to queue L_h and schedule requests using PPRR method.
    $\forall i$ give request number
    Queue $\leftarrow \Re_{i,m}^{req}$
    mutex==1
    TimeQuantum==3
    while Request in Queue **do**
      if mutex==1 **then**
        mutex==0
        for Requests in Queue **do**
          BurstTime=*BurstTime*-*TimeQuantumn*
          if *BurstTime* $\neq 0$ **then**
            allot resource $\leftarrow \Re_{i,m}^{req}$
            $\Re_{i,m}^{req} \leftarrow$ *updateQueue*
          end if
          for time==0; time $\leftarrow$ *BurstTime*; time++ **do**
            tempQueue $\leftarrow \Re_{i,m}^{req}$
            tempQueue.sort($\Re_{i,m}^{req}$)
            for Queue $\leftarrow \Re_{i,m}^{req}$ and *BurstTime* $\neq 0$ **do**
              BurstTime=*BurstTime*-1;
              allot resource $\leftarrow \Re_{i,m}^{req}$
              tempQueue $\leftarrow$ *updateQueue*
              mutex=1;
            end for
          end for
          $\Re_{i,m}^{req} = \Re_{i,m}^{req} \rightarrow next$
        end for
        mutex=1
      end if
    end while
  end while
else
  Add $\Re_{i,m}^{req}$ to queue L_n and schedule requests using FCFS scheme.
end if
for (i=1; i $\leq$ n; i++) **do** ▷ Follower move
  Cloud controller selects DCs having requested resources using Algorithm 2
  Cloud controller computes (P_i) for requested resources using Algorithm 2
  Cloud controller announce the price (P_i).
  End users calculate their utility function ($\mho_i$) using Eq. (5.25)
  if ($\mho_i t$) > $\mho_i(t-1)$) **then**
    Accept the price
  else
    Reject the price and wait for next announcement or quit
  end if
  End users who accepted the price, pay the price for resources ▷ Equilibrium
  Allocate the resources requests appropriately
end for

Once price is finalized, cloud controller announces the price to the requesting end users. Using the announced price, the end users calculate their utility function ($\mho_i$) using Eq. (5.25). The end users decides to accept or reject the price on the basis of utility function. If the price is accepted, end users pay the price to access the requested resources. However, if the user rejects the price then it has to wait for next announcement or quit (line 41-48). The cloud controller allocates the resources to end users who have paid the price (line 49-50).

5.3.2 Resource utilization and energy management

The efficient resource utilization and energy management by cloud controller is described in Algorithm 10. The cloud controller manages the utilization of the server and energy consumption of DCs by considering the requests from i end users. For all DCs, cloud controller checks for available resources (line 1-2). If the requested resources are available at DCs, then cloud controller checks for the utilization of servers (U_j^k). Otherwise, the requests are added in a waiting queue (line 3-7). If the utilization of servers is equal to utilization of idle server (U_{kj}^{idle}), then activate sleep mode and store the information in list M_1 (line 8-10). Otherwise, the utilization of servers is compared with utilization threshold of servers (U_{kj}^{thr}). Utilization threshold of a server is set to minimize SLA violations.

If the ($U_j^k < U_{kj}^{thr}$), then information of such servers is store in list M_2 and such servers are allocated whenever required. Further, if the list M_2 is empty then, servers listed in M_1 are activated and utilized for allocation (line 11-22). Similarly, cloud controller analyzes the energy consumption aspect of DCs. The energy generated by RES (E_j^{ren}) and energy consumption (E_j) are calculated using using Eq. (5.15) and Eq. (5.14), respectively (line 23-25). If ($E_j < E_j^{ren}$), then (R_j^{cld}) and (C_j^{cld}) are calculated using Eq. (5.26) and Eq. (5.27), respectively (line 26-28). In next step, utility function ($\mho_j^{cld}(1)$) is calculated using Eq. (5.29) (line 29). Using utility function, cloud controller decides on whether to announce the price or not (line 30-35).

Algorithm 10 Algorithm for DC selection

Input: $\Re_{i,m}^{req}$, $\Re_j(avl)$, U_j^{idle}, U_{kj}^{max}

Output: U_j^k, E_j^{ren}, E_j, R_j^{cld}, C_j^{cld}, $\mho_j^{cld}(1)$, P_i, E_j^{grid}, DCs

1: **for** (j=1; j ≤ n; j++) **do** ▷ $n \leftarrow$ Number of DCs
2: Check for available resources ($\Re_j(avl)$)
3: **if** ($\Re_{i,m}^{req} < \Re_j(avl)$) **then**
4: Calculate utilization of servers (U_j^k) using Eq. (5.12)
5: **else**
6: Add request in waiting queue (Q)
7: **end if**
8: **for** (k=1; k ≤ q; k++) **do** ▷ $q \leftarrow$ Number of servers
9: **if** ($U_j^k == U_{kj}^{idle}$) **then**
10: Shutdown k^{th} server and store in list M_1
11: **else if** ($U_j^k < U_{kj}^{thr}$) **then**
12: Store in list M_2
13: **else**
14: Add request in waiting queue (Q)
15: **end if**
16: **if** (M_2 is empty) **then**
17: Restart server listed in M_1 and utilize them to serve user requests
18: **else**
19: Utilize servers listed in M_2 to serve user requests
20: **end if**
21: **end for**
22: **end for**
23: **for** (j=1; j ≤ n; j++) **do**
24: Calculate E_j^{ren} using Eq. (5.15)
25: Calculate E_j using Eq. (5.14)
26: **while** ($E_j < E_j^{ren}$) **do**
27: Calculate revenue (R_j^{cld}) using Eq. (5.26)
28: Calculate cost (C_j^{cld}) incurred on requested resources using Eq. (**??**)
29: Calculate utility function ($\mho_j^{cld}(1)$) using Eq. (5.29)
30: **end while**
31: **if** ($\mho_j^{cld}(1)(t) > \mho_j^{cld}(1)(t-1)$) **then** ▷ (t) and (t-1) are time slots
32: Announce price P_i and available DCs
33: **else**
34: Add request in waiting queue (Q)
35: **end if**
36: **while** ($E_j > E_j^{ren}$) **do**
37: Calculate (E_j^{grid}) using using Eq. (5.17)
38: Get DCs arranged in ascending order of energy tariff using Algorithm 11
39: Announce price P_i on the basis of energy tariff and available DCs
40: **end while**
41: **end for**

Further, if ($E_j^{ren} < E_j$), then energy (E_j^{grid}) to be drawn from grid is calculated using using Eq. (5.17). In such a case, the cloud controller trades for energy with grids connected to respective DCs. Using Algorithm 11, DCs are arranged in ascending order of energy tariff offered by the respective grids. The price of resources is calculated by considering energy tariff of grid and announced along with available DCs (line 36-44).

5.3.3 Stackelberg game for energy trading

In step two of the proposed scheme, the energy is drawn from the grids connected to DCs. For this purpose, a *single-leader multi-follower* Stackelberg game for energy trading is formulated. The cloud controller acts as *single-leader* whereas grids act as *multi-followers*. Algorithm 11 is designed for efficient energy trading between cloud controller and grids.

Algorithm 11 Algorithm for energy trading with grid

Input: E_j^{ren}

Output: E_j, E_j^{grid}, P_j^{grid}, $\mho_j^{grid}$, $\mho_{j2}^{cld}$

1: **for** (j=1; j $\leq$ n; j++) **do** ▷ $n \leftarrow$ Number of DCs
2: Calculate E_j^{grid} using Eq. (**??**) and announce it to grid ▷ Leader move
3: For E_j^{grid} calculate price (P_j^{grid}) using Eq. (5.28) ▷ Follower move
4: With P_j^{grid}, calculate revenue (R_j^{grid}) using Eq. (5.31)
5: For E_j^{grid} calculate cost (C_j^{grid}) using Eq. (5.32)
6: For E_j^{grid} calculate utility function ($\mho_j^{grid}$) using Eq. (5.33)
7: **if** ($\mho_j^{grid}(t) > \mho_j^{grid}(t-1)$) **then** ▷ (t) and (t-1) are time slots
8: Announce price (P_j^{grid}) to cloud controller
9: **else**
10: Reject request or put in waiting list
11: **end if**
12: Calculate revenue of cloud controller (R_j^{cld}) using Eq. (5.26)
13: Using P_j^{grid} and R_j^{cld}, calculate utility function ($\mho_{j2}^{cld}$) using Eq. (5.30)
14: **if** $\mho_{j2}^{cld}(t) > \mho_{j2}^{cld}(t-1)$ **then**
15: Accept price (P_j^{grid})
16: **else**
17: Reject price
18: **end if**
19: Arrange DCs on the basis of P_j^{grid} in ascending order ▷ Equilibrium
20: **end for**

For all DCs, cloud controller calculates the energy required from grid (E_j^{grid}) using Eq. (5.17) and announce it to connected grids (line 1-2). The grids calculate price (P_j^{grid}), revenue (R_j^{grid}), and cost (C_j^{grid}) using Eqs. (5.28), (5.31), and (5.32), respectively (line 3-5). The grid calculates utility function ($\mho_j^{grid}$) using Eq. (5.33) and using it finalizes the price. Once the price is finalized, it is announced to cloud controller (line 6-11). After receiving the price, cloud controller calculates revenue (R_j^{cld}) using Eq. (5.26) (line 12). The utility function ($\mho_j^{cld}(2)$) is calculated using Eq. (5.30) (line 13). Using this utility function, cloud controller decides to accept or reject the price (line 14-18). Once price is accepted, DCs are arranged in ascending order on the basis of P_j^{grid} (line 19). The cloud controller selects DCs which are offered lowest energy tariff by grid and schedules user resource requests to it.

5.4 Existence of Nash equilibrium

Theorem 1: *The equilibrium strategies adopted by followers is the optimal response to the strategy followed by the leaders.*

Proof: Let S_i be the strategy set for i end users (where $i \in \mathbb{I}$) and S_{cc} be the strategy set for cloud controller acting on behalf of j DCs (where $j \in \mathbb{J}$). Each end user aims to maximize its payoff by accessing their required resources. Cloud controller aims to decide an optimal price of resources with respect to available energy with RES connected to each DC so as to maximize its utility. So, $R_{i,m}^{req^*} \in S_i$ is an equilibrium strategy for cloud controller if

$$\mho_i(R_{i,m}^{req^*}, P_i^*(R_{i,m}^{req^*}))(t) \geq \mho_i(R_{i,m}^{req^*}, P_i^*(R_{i,m}^{req^*}))(t-1) \tag{5.34}$$

where, P_i^* is the optimal response of cloud controller.

Theorem 2: *A unique Nash equilibrium exists between multiple end users and cloud controller, and thereby a unique Stackelberg equilibrium.*

Proof: A Nash equilibrium for the proposed stackelberg game for energy-aware

resource allocation scheme exists for each end user and cloud controller. The end users announce their requirements to cloud controller which in turn announce the price, which exists at Nash equilibrium point. The user side and cloud controller side analysis for proof of existence of Nash equilibrium is given as below.

For a given user request announced by user i, cloud controller sets an optimal price (P_i) with respect to revenue (R_j^{cld}) generated by CSP and cost incurred (C_j^{cld}). The utility of cloud controller is given by Eq. 5.29. The optimization problem for cloud controller is formulated as follows.

$$\mho_j^{cld} = Maxmize \sum_{t,j} \left(R_j^{cld} - C_j^{cld}\right) \tag{5.35}$$

$$s.t.\ R_j^{cld} - C_j^{cld} > 0 \tag{5.36}$$

$$R_j^{cld} > 0 \tag{5.37}$$

$$C_j^{cld} > 0 \tag{5.38}$$

Karush-Kuhn-Tucker (KKT) conditions are applied to achieve Nash equilibrium [189] [174] as given below.

$$\begin{aligned} F(obj) + \bigtriangledown_n \lambda_{j,1}(t) \left(R_j^{cld} - C_j^{cld}\right) + \bigtriangledown_n \lambda_{j,2}(t) \left(R_j^{cld}\right) \\ + \bigtriangledown_n \lambda_{j,3}(t) \left(C_j^{cld}\right) \\ \bigtriangledown_n \lambda_{j,2}(t) \left(R_j^{cld}\right) > 0, \\ \bigtriangledown_n \lambda_{j,3}(t) \left(C_j^{cld}\right) > 0 \end{aligned} \tag{5.39}$$

where, $\lambda_{j,1}(t)$, $\lambda_{j,2}(t)$, and $\lambda_{j,3}(t)$ are non-negative Lagrangian constants.

The overall utility function ($\hat{\mho}_j^{cld}$) of all the DCs is given as below.

$$\hat{\mho}_j^{cld} = \sum \mho_j^{cld)} \tag{5.40}$$

Now, considering the overall utility function ($\hat{\mho}_j^{cld}$), the following statement is derived.

$$\bigtriangledown_n \hat{\mho}_j^{cld} - \bigtriangledown_n \lambda_{j,1}(t)\left(R_j^{cld} - C_j^{cld}\right) - \bigtriangledown_n \lambda_{j,2}(t)\left(R_j^{cld}\right) - \bigtriangledown_n \lambda_{j,2}(t)\left(C_j^{cld}\right) = 0 \tag{5.41}$$

Hence, the Jacobian matrix of overall utility function ($\hat{\mho}_j^{cld}$) is given as follows.

$$J\hat{\mho}_j^{cld} = \begin{bmatrix} \mho_1^{cld} & 0 & . & . & 0 \\ 0 & \mho_2^{cld} & . & . & 0 \\ . & . & . & . & . \\ . & . & . & . & . \\ 0 & 0 & . & . & \mho_j^{cld} \end{bmatrix}. \tag{5.42}$$

$J\hat{\mho}_j^{cld}$ is a positive diagonal matrix as the value of utility can not be negative. Hence, there exists a generalized Nash equilibrium.

Summary of the Chapter: The Chapter 5 presents an energy-aware resource allocation scheme is proposed using a Stackelberg game for energy management in cloud-based DCs. For this purpose, a cloud controller is used to receive the requests of users which then distributes these requests among geo-distributed DCs in such a way that the energy consumption of DCs is sustained by RES. However, if energy consumption of DCs is not sustained by RES then the energy is drawn from the grid. The requests of users are routed to the DC which is offered lowest energy tariff from the grid. For this purpose, a Stackelberg game for energy trading is also proposed to select the grid offering lowest energy tariff to DCs.

Chapter 6

CoaaS-based Job Classification and Scheduling Scheme

In this chapter, a renewable energy-aware multi-indexed job classification and scheduling scheme using *CoaaS* model for data centers sustainability is proposed[1]. In the proposed scheme, incoming workloads from different devices are transferred to the DC which has sufficient amount of renewable energy available with it. For this purpose, a renewable energy-based host selection and container consolidation scheme is also designed.

6.1 System Model

A geo-distributed cloud environment having i DCs connected to RES and grid with ESS is considered. Fig. 6.1 depicts the two hierarchical controllers, i.e., global and local deployed in the proposed model. The global controller selects DCs where the incoming jobs can be scheduled in such a way that energy consumption is sustained using RES. The status of all the jobs being executed at a specific DC is regularly

[1]The content of this chapter is taken from:

- G. S. Aujla, N. Kumar, S. Garg, K. Kaur, R. Ranjan and S. K. Garg, "Renewable Energy-based Multi-Indexed Job Classification and Container Management Scheme for Sustainability of Cloud Data Centers," IEEE Transactions on Industrial Informatics, 2018, doi: 10.1109/TII.2018.2800693.

updated by this controller. The local controllers handles the allocation of j containers ($\mathbb{C}$) deployed over p servers. Some of the important preliminaries about the model are illustrated as below.

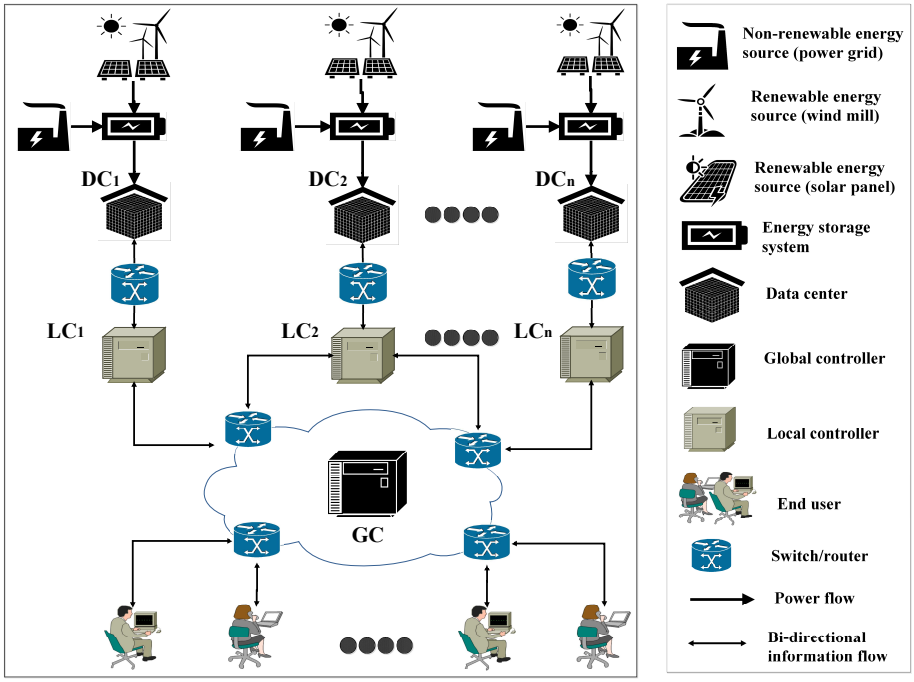


Figure 6.1: System architecture for geo-distributed DCs

Fig. 6.2 shows the proposed *CoaaS* model deployed directly over the infrastructure layer using a docker engine.

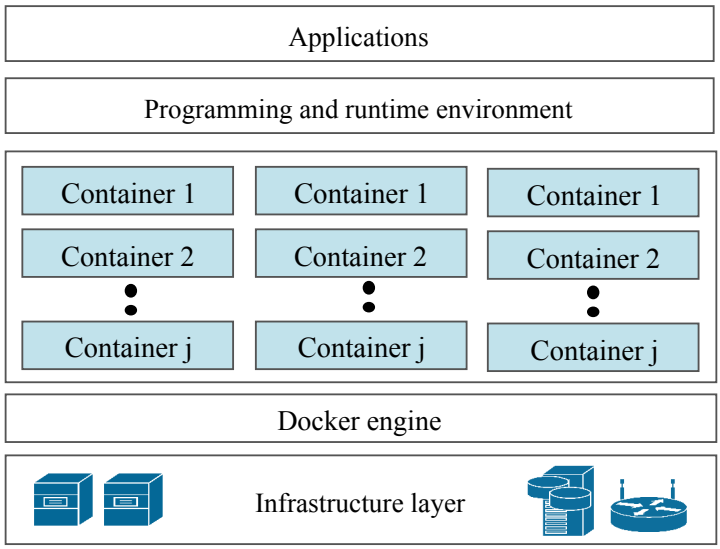


Figure 6.2: CoaaS model

The docker contains all the components required for runtime task execution such as configuration files and databases. It enables an additional layer of abstraction and virtualization. It is used for Linux OS and leverages the resource isolation feature of the underlying kernel. This, in turn, permits containers to be independently executed within the kernel instance, thereby eliminating the need to initiate and maintain VMs [108].

6.1.1 Workload model

Let us consider an incoming workload (ω) that consists of m number of κ type of jobs. This workload is classified and added to different queues for scheduling. The global controller decides the DC to which the job can be routed. At time t, the type κ jobs are modeled using poisson distribution with an arrival rate, $a(t)$. The type κ jobs scheduled at i^{th} DC follow the queues dynamics [129] as given below.

$$Q_i^{\kappa}(t+1) = max[Q_i^{\kappa}(t) - Q_i(t)] + m_i^{\kappa}(t) \quad \text{s.t.} \;\; 0 < \sum_{\kappa=1}^{m} Q_i(t)\tilde{c_{\kappa}} \leq s_p\delta_p \tag{6.1}$$

where, $m_i^{\kappa}(t)$ is the number of type κ jobs routed to i^{th} DC, $\tilde{c_{\kappa}}$ is the required computing resources to handle type κ jobs, s_p is the total number of servers allocated and δ_p is the processing speed of each server.

6.1.2 QoS Model

SLA violation occurs if the available capacity of resources at a DC is less than the resources required to process the workload. The SLA violations (SLA_p^v) of p^{th} server of i^{th} DC are given as below [103].

$$SLA_p^v = \frac{1}{p}\sum_{p=1}^{P}\frac{t_p^{thr}}{t_p^{act}}D_p^{mg} \tag{6.2}$$

where, t_p^{thr} is the time for which p^{th} server has threshold utilization level, t_p^{act} is total active time, and D_p^{mg} in the performance degradation due to migration.

Using SLA violations, a SLA metric for j^{th} container is defined as below [109].

$$SLA_j^v = \sum_{p=1}^{P} \sum_{j=1}^{J} \sum_{v=1}^{V} \frac{s_{rq}(\mathsf{C}_j, t_v) - s_{al}(\mathsf{C}_j t_v)}{s_{rq}(\mathsf{C}_j t_v)} \tag{6.3}$$

where, $s_{rq}(\mathsf{C}_j, t_v)$ is server amount requested by j^{th} container on p^{th} server at time t_v and $s_{al}(\mathsf{C}_j, t_v)$ is server amount allocated by j^{th} container at time t_v.

In order to ensure SLAs, at some occasions, container migrations are required. The delay occurred due to these migration (d_j^{mg}) for j^{th} container is given as below.

$$d_j^{mg} = d_{nt} \varphi_j(t) \tag{6.4}$$

where, d_{nt} is delay offered by underlying network and φ_j is the migration rate.

6.1.3 Energy consumption model

The energy consumption of i^{th} DC is given as below.

$$E_i = \sum_{p} E_i^p + E_i^{net} + E_i^c \tag{6.5}$$

where, E_i^p, E_i^{net}, and E_i^c are energy consumed by processors, network, and cooling, respectively.

Energy consumption of a server has linear relation with its CPU load. So, the energy consumption of p^{th} server at i^{th} DC is given as below.

$$E_i^p = E_{idl}^p + (E_{max}^p - E_{idl}^p)\, U_i^p \tag{6.6}$$

where, E_{idl}^p is the energy consumed by idle p^{th} server, E_{max}^p is the maximum energy that p^{th} server can consume, and U_i^p is the level of utilization of p^{th} server.

The level of utilization of p^{th} server at i^{th} DC depends on the amount of resources

consumed (R^p) at time t and maximum capacity of processor (R^p_{mx}) and is given as.

$$U_i^p = \left(\frac{R^p(t)}{R^p_{mx}}\right) \times 100 \tag{6.7}$$

The level of utilization of j^{th} container running on p^{th} server is given as below.

$$U_p^j = \left(\frac{R^j(t)}{R^j_{mx}}\right) \times 100 \tag{6.8}$$

where, $R^j(t)$ denotes the resources consumed by j^{th} container at time t and R^j_{mx} is the maximum capacity of the container.

The energy consumption related to network infrastructure depends on; 1) fixed part (E^{nt}_{sw}): switches, fan, chasis, etc and 2) dynamic part (E^{nt}_{pr}): active ports. So, the network related energy consumption at i^{th} DC is given as below.

$$E_i^{net} = E^{nt}_{sw} + E^{nt}_{pr} \tag{6.9}$$

The Eq. (6.9) can be extended on the basis of incoming flow passing through the ports and the time for which the flow is active. The extended representation is given as below.

$$E_i^{net} = E^{nt}_{fx} \times \sum_{sw}(t^{en}_f - t^{st}_f) + E^{nt}_{dy} \times \sum_{sw}(t^{en}_f - t^{st}_f) \tag{6.10}$$

where, t^{en}_f and t^{st}_f denotes the end and start time of f^{th} flow and E^{nt}_{fx} and E^{nt}_{dy} refers to the energy consumed by fixed part and dynamic part, respectively.

6.1.4 Energy generation model

In this paper, two types of sources of energy (renewable and grid) are considered to handle the energy demand of DCs. Renewable energy is utilized as a primary source of energy and grid energy is utilized only in worst case. The energy generated (E^r_i)

by RES is given as below.

$$E_i^r = E_i^{pv} + E_i^{wn} \tag{6.11}$$

where, E_i^{pv} is the energy generated by PV panels and E_i^{wn} is the energy generated by wind turbines.

The energy (E_i^{pv}) generated by PV panels is given as below [184].

$$E_i^{pv} = \left(1 - L_{pv}^{tp}\right)\ \eta\ S_i^{pv}\ cos(\alpha)\ \text{ɣ} \tag{6.12}$$

where, solar radiations (ɣ), conversion efficiency of panel (η), radiation angle of Sunlight ($cos\ \alpha$), size of panel (S_i^{pv}), and temperature exedence loss (L^{tp}).

The energy (E_i^{wn}) generated by a typical wind turbine is given as below [169].

$$E_i^{wn} = \frac{1}{2}\ \left[\text{Ç}\ \rho\ \hat{a}\ \nu^3\right]\ L^{tp} \tag{6.13}$$

where, Ç is the rotar efficiency, ρ is the air density, $\hat{a}$ is the area swept by rotar blades, and ν is the wind speed.

The energy generated by RES is stored in ESS connected to the DCs. The energy stored (E_i^{ess}) in ESS deployed at i^{th} DC is given as below.

$$E_i^{ess}(t) = (E_i^{ess}(t-1) + E_i^r - E_i)\,\iota_i^{ess} \tag{6.14}$$

where, ι_i^{ess} is the ESS's self-discharge rate.

6.1.5 Problem Formulation

Now, various cases are considered for defining the objective function.

- **Case 1:** If ($E_i \leq E_i^r$), then the excess energy generated by RES is stored in ESS and is used by DCs in case of deficit. The excess energy (E_i^{ex}) stored in ESS is given as below.

$$E_i^{ex} = E_i^r - E_i \tag{6.15}$$

- **Case 2:** If $(E_i > E_i^r)$, then the energy deficit of DCs is fulfilled from excess energy (E_{ex}) stored in ESS. The energy deficit (E_i^{df}) is given as below.

$$E_i^{df} = E_i - E_i^r \tag{6.16}$$

In this case, there may be a sub-case in which energy deficit (E_i^{df}) is not fulfilled by excess energy (E_i^{ex}) stored in ESS. In such a case, the energy is drawn from grid. The energy drawn from grid (E_i^{gr}) is given as below.

$$E_i^{gr} = E_i^{df} - E_i^{ex} \tag{6.17}$$

However, in such a case, the workload is assigned to another DCs having sufficient amount of energy generated by RES.

Keeping in view of the above discussed cases and other related aspects, the objective function of the proposed scheme is illustrated as below.

$$E_{obj} = Minimize \sum_{t,i} (E_i(t) - E_i^r(t)) \tag{6.18}$$

subject to following constraints

$$E_i^r(t) \ > \ 0 : \ \forall i \tag{6.19}$$

$$E_i(t) \ > \ 0 : \ \forall j \tag{6.20}$$

$$t_i^{rs} \leq t_{mx}^{rs} \tag{6.21}$$

$$0 < E_i^{ex} \leq E_{ess}^{mx} \tag{6.22}$$

where, $E_i(t)$, t_{mx}^{rs} and E_{ess}^{mx} denotes response time, maximum desirable response time and maximum storage capacity of ESS, respectively.

6.2 CoaaS-based Workload Management Scheme

The propose scheme works in different steps, discussed in the subsequent Sections.

6.2.1 Multi-indexed Classification and Scheduling Scheme

In this section, a multi-indexed workload classification and scheduling scheme is presented. The incoming workload is classified into three categories; 1) high priority, 2) high resource requirement, and 3) best effort. Initially, the controller classifies ω using priority index, load index, and availability index. The high priority workload is indexed using priority index and the high resource requirement workload is classified into load index. The best effort workload is indexed into availability index.

A priority index ($\wp$) is used to classify the incoming job request requiring high priority are as defined as below.

$$\wp = P(\frac{c_j}{m}) = \frac{P(c_j)P(\frac{m}{c_j})}{P(m)} \tag{6.23}$$

where, $P(c_j)$ and $P(\omega)$ depicts probabilities of j^{th} containers available capacity and incoming job requests respectively.

Apart from priority, load index (ℓ) is defined to classify ω requiring high computing resources. Hence, ℓ is defined on the basis of load ($\imath_i$) at i^{th} DC as below.

$$\ell = P(\frac{\imath_m}{\imath_i}) = \frac{P(\imath_m)P(\frac{\imath_i}{\imath_m})}{P(\imath_i)} \tag{6.24}$$

where, $P(\imath_m)$ and $P(\imath_i)$ denote m^{th} job and i^{th} DC load probabilities, respectively.

Finally, the workload not classified in $\wp$ or ℓ is indexed using availability index (α) as defined below.

$$\alpha = P(\frac{\Re_{rq}}{\Re_{av}}) = \frac{P(\Re_{rq})P(\frac{\Re_{av}}{\Re_{rq}})}{P(\Re_{av})} \tag{6.25}$$

where $P(\Re_{rq})$ and $P(\Re_{av})$ represents the probabilities of requested resources and available resources respectively.

Algorithm 12 Classification and Scheduling of Workloads

Input: ω, available i DC list (aDCL)
Output: allocated DC_i

for (ω=1; $\omega \leq$ m; ω++) **do** ▷ $m \leftarrow$ incoming workloads
  $\omega \leftarrow Q_i(t)$ ▷ Add to queue
  call.classifier(ω)
  if $\omega \leftarrow \wp$ **then**
    $Q_{i1} \leftarrow \omega$
    call.PPRR(Q_{i1}) ▷ Initiate PPRR
    while $\omega \leftarrow Q_{i1}$ **do**
      $\forall\ \omega$
      mutex==1
      TimeQuantum==3
      if mutex==1 **then**
        mutex==0
        for Requests in Queue **do**
          BurstTime=BurstTime-TimeQuantumn
          if *BurstTime* $\neq$ 0 **then**
            allot resource $\leftarrow \Re_{rq}$
            $\Re_{rq} \leftarrow$ *updateQueue*
          end if
          for time==0; time $\leftarrow$ *BurstTime*; time++ **do**
            tempQueue $\leftarrow \Re_{rq}$
            tempQueue.sort($\Re_{rq}$)
            for Queue $\leftarrow \Re_{rq}$ and *BurstTime* $\neq$ 0 **do**
              BurstTime=BurstTime-1;
              allot resource $\leftarrow \Re_{rq}$
              tempQueue $\leftarrow$ *updateQueue*
              mutex=1;
            end for
          end for
          $\Re_{rq} = \Re_{req} \rightarrow next$
        end for
        mutex=1
      end if
    end while
    if $\omega \leftarrow \ell$ **then**
      $Q_{i2} \leftarrow \omega$
      call.PPRR(Q_{i2}) ▷ Initiate PPRR
      while $\omega \leftarrow Q_{i2}$ **do**
        Repeat Step 8-32
      end while
      if $\omega \leftarrow \alpha$ **then**
        $Q_{i3} \leftarrow \omega$
        call.PPRR(Q_{i3}) ▷ Initiate PPRR
        while $\omega \leftarrow Q_{i3}$ **do**
          Repeat Step 8-32
        end while
      end if
    end if
  else
    $Q_{i4} \leftarrow \omega$
    call.FCFS(Q_{i4})
  end if
  aDCL.allocate(DC_i) $\leftarrow \forall\omega$
  aDCL $\leftarrow$ update(ω status)
end for
while (Q_{in}) **do**
  $Q_{i1} \leftarrow$ *updatepriorityqueue*
  while (Q_j) **do**
    $Q_{i2} \leftarrow$ *updateloadqueue*
    while (Q_k) **do**
      $Q_{i3} \leftarrow$ *updateavailabilityqueue*
    end while
  end while
end while
while $\forall(Q_{in})$ **do**
  updateoutgoingquery $\leftarrow Q_{in}$
end while

Once ω is classified, then it is scheduled using Algorithm 12. In this algorithm, ω is added to a $Q_i(t)$ at time t. For all elements in the queues, classifier is used to segregate ω on the basis of $\wp$, ℓ, and α (line 1-3). If ω is classified in $\wp$, then it is added to a new queue (Q_{i1}) and thereby scheduled using PPRR scheme (line 4-33). If ω is classified in ℓ, then it is added to a new queue (Q_{i2}) and it is also scheduled using PPRR scheme (line 34-39). Moreover, if ω is classified in α, then it is added to a new queue (Q_{i3}) and scheduled using PPRR scheme (line 40-45). In this way, ω is classified differently on the basis of various indexes and added to the different queues. All queues are scheduled separately using PPRR scheme.

However, the workloads which are not classified in any of the aforementioned queues are added to queue (Q_{i4}) and are scheduled using FCFS (line 48-51). Now, each workload is allocated to an appropriate DC listed in available DC list (aDCL). Once the DC are allocated, then the aDCL status is updated (line 52-54). Finally, the respective queues are updated accordingly (55-66).

6.2.2 Renewable Energy-aware Host Selection Scheme

Now, Algorithm 13 is presented for renewable energy-aware host selection. The objective of this algorithm is to utilize the resources optimally and to select the container, server, and DC in such a way that energy consumption is sustained using energy generated by RES. In this algorithm, the subsequent steps are performed for all the servers deployed in available DCs.

Initially, $\Re_{av}(p,i)$ available with p^{th} server of i^{th} DC are checked. If ($\Re_{rq}(i) \leq \Re_{av}(p,i)$), then level of utilization (U_i^p) of each server is computed using Eq. (6.7). If (U_i^p) is equal to (U_{pi}^{id}), then the server is idle and so it is shutdown and added to list M_1 (1-8). Otherwise, for each active container in p^{th} server of i^{th} DC, the level of utilization (U_p^j) is computed using Eq. (6.8). If (U_p^j) is equal to (U_{jp}^{id}), then it is having no workload and can be deactivated and stored in list M_2 (line 9-14). Rest all the containers are active and are stored in list M_3 (line 10-19).

Algorithm 13 Renewable energy-aware host selection

Input: $\Re_{rq}(i)$, $\Re_{av}(p,i)$, , U_i^{mx}
Output: $aDCL$, $\complement_j$

for (i=1; i $\leq$ I; i++) **do** ▷ $I \leftarrow$ Number of DCs
  for (p=1; p $\leq$ P; p++) **do** ▷ $P \leftarrow$ Number of servers
    Check available resources ($\Re_{av}(p,i)$)
    if ($\Re_{rq}(i) < \Re_{av}(j,i)$) **then**
      Compute (U_i^p) using Eq. (6.7)
      if ($U_i^p == U_{pi}^{id}$) **then**
        Shutdown $\leftarrow k^{th}$ server (S_k)
        Store $\leftarrow M_1$
      else
        for (j=1; j $\leq$ J; j++) **do** ▷ $J \leftarrow$ Number of container
          Compute (U_p^j) using Eq. (6.8)
          if ($U_p^j == U_{jp}^{id}$) **then**
            Shutdown $\leftarrow j^{th}$ container ($\complement_j$)
            Store $\leftarrow M_2$
          else
            Store $\leftarrow M_3$
          end if
        end for
      end if
    end if
    if (M_3 is empty) **then**
      Activate container ($\complement_j$) $\leftarrow M_2$
      Assign jobs $\leftarrow M_2$
    else
      Activate new servers $\leftarrow M_1$
      Deploy container ($\complement_j$) $\leftarrow$ server in M_1
    end if
  end for
  Calculate E_i using Eq. (6.5)
  Calculate E_i^r using Eq. (6.11)
  if ($E_i \leq E_i^r$) **then**
    Add DCs in $aDCL_1$
    Store E_i^{ex} calculated using Eq. (6.15) in ESS
  else
    Draw E_i^{df} calculated Eq. (6.16) is drawn from E_i^{ex} stored in ESS
    Add DCs in $aDCL_2$
    if ($E_i^{df} < E_i^{ex}$) **then**
      Draw energy (E_i^{gr}) using Eq. (6.17) from grid
    end if
  end if
  Sort selected DCs in descending order in aDCL
end for

Initially, all the container for p^{th} server of i^{th} DC are allocated from list M_3. Now, if in case list M_3 is empty, then first the deactivated containers in list M_2 are activated and utilized. But, if still the demand is not fulfilled, then the servers in list M_1 are restarted and containers are deployed (line 20-28).

Once, the resource availability is confirmed at DCs, then the renewable energy available at each DC is compared with energy demand of DCs. For this purpose, E_i of i^{th} DC is calculated using Eq. (6.5) and energy generated by RES is calculated using Eq. (6.11) (29-30). Now, if $(E_i \leq E_i^r)$, then such DCs are added in $aDCL_1$. In this case, E_{ex} generated by RES is calculated using Eq. 6.15 and stored in ESS (line 31-33). However, if $(E_i > E_i^r)$, then E_{ex} stored in ESS is used to manage E_{df} calculated using Eq. 6.16. Such DCs are added in $aDCL_2$. But, if still the energy requirement is not fulfilled, then E_i^{gr} calculated using Eq. 6.17 is drawn from grid (line 34-40). DCs which are having sufficient renewable energy resources are sorted and added to aDCL and are selected to schedule various workloads by global controller (line 41).

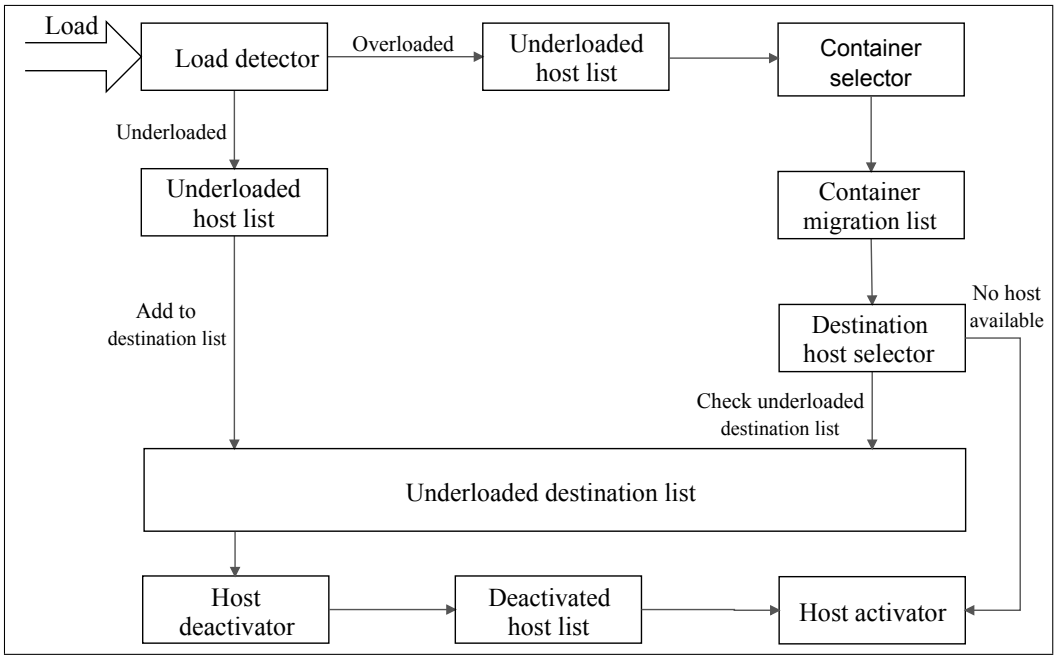


Figure 6.3: Flowchart of *CoaaS* consolidation scheme

6.2.3 Container Consolidation and Migration Scheme

In container consolidation scheme, active load on a running host is consolidated on minimum number of servers to run the applications. This consolidation is performed to minimize energy consumption of DCs. Fig. 6.3 shows the flow diagram of the consolidation scheme. Various steps of consolidation scheme are discussed as below.

- **Load detector (LD):** The current load of a running host is detected and classified as overloaded or underloaded on the basis of OL/UL threshold.
- **Underload host list (UHL):** Underloaded hosts are added to UHL.
- **Underloaded destination list (UDL):** The underloaded hosts are added to this list. If any of the host in this list is used to migrate the containers from overloaded hosts, then such hosts remain active. The host that are not utilized are sent to host deactivator.
- **Host de-activator (HD):** Un-utilized underloaded hosts are deactivated.
- **Deactivated host list (DHL):** All the underloaded hosts which are deactivated are added to this list.
- **Overload host list (OHL):** Hosts classified as overloaded are added to OHL.
- **Container selector (CS):** If a running host is overloaded, then the active containers to be migrated are selected.
- **Container migration list (CML):** It contains the containers identified for migration from a overloaded host to another underloaded or new hosts.
- **Destination host selector (DHS):** In this step, a destination hosts where the containers in migration list can be shifted are selected from UDL. However, if no host is available in the list, then a new host is activated from DHL.
- **Host activator (HA):** If DHS is not able to find any underloaded host, then a host is activated from DHL.

Algorithm 14 shows working of the consolidation scheme.

Algorithm 14 Container consolidation scheme

Input: Host S_k, C_j, R_{rq}
Output: Destination host (S_k^{des})

for (p=1; p $\leq$ P; p++) **do** ▷ $i \leftarrow$ Number of servers
    Compute (U_i^p) using Eq. (6.7)
    if ($U_i^p < U_{ul}^{thr}$) **then**
        Add $S_k \rightarrow$ UHL
        Add $S_k \rightarrow$ UDL
    else if ($U_i^p > U_{ol}^{thr}$) **then**
        Add $S_k \rightarrow$ OHL
        Select C_j
        Add $\mathsf{C}_j \rightarrow$ CML
        Pass control $\rightarrow$ DHS
        Check $\rightarrow$ UDL
        Check $\rightarrow R_{rq}$
        if S_k^{des} in UDL **then**
            Migrate $\mathsf{C}_j \rightarrow S_k^{des}$
            if $\alpha_{jd_ad_b} == 0$ **then**
                Update C_j
            else
                No update required
            end if
        else
            Activate $S_k^{new} \rightarrow$ DHL
        end if
    else
        Host is neither overloaded nor underloaded
    end if
    Deactivate unallocated hosts in UDL
    Add deactivated hosts $\rightarrow$ DHL
end for

For synchronization of containers located with in the same or different machines with respect to data updates, a variable $\alpha_{jd_ad_b}$ is considered. Here, d_a is the data located on the host server and d_b denotes the data located server where C_j is to be migrated. All the machines synchronize with the controller and a task is completed only when all the data updates are done. The following condition needs to be satisfied.

$$\sum_{i=1} i <= n = n \times \alpha_{jd_ad_b} \tag{6.26}$$

Summary of the Chapter: The Chapter 6 presents a renewable energy-aware multi-indexed job classification and scheduling scheme using *CoaaS* model for data centers sustainability. In the proposed scheme, incoming workloads from different devices are transferred to the DC which has sufficient amount of renewable energy available with it. For this purpose, a renewable energy-based host selection and container consolidation scheme is also designed.

Chapter 7

Results and Discussion

The proposed schemes were evaluated using a simulated environment with respect to various performance metrics. This chapter presents the results obtained after evaluation of all the proposed schemes with respect to various performance metrics.

7.1 EV-based Energy Management Scheme

In this section, the first proposed scheme and algorithms are simulated using the workload traces for a single DC located at Mohali (Punjab), India having 100 heterogeneous servers. The heterogeneous physical servers considered for the purpose of evaluation are categorized as, HP ProLiant DL380 G7 X5690 Hexacore 3.46 GHz (180 W), and HP ProLiant DL80 Gen9 Hexacore E5-2630 v3 (220 W). Each heterogeneous server considered in the proposed evaluation setup has different configuration and energy consumption properties. The proposed scheme is not limited to a single DC, type of servers, input parameters, and hardware configuration as per requirement. The interactive workload traces considered for evaluation are rescaled from real traffic from Wikipedia [140]. These workload traces are having variable arrival rate. To evaluate the proposed scheme, solar radiations [165] and wind speed [166] at DC are considered for 12-hours. Various other input parameters and constants considered for calculations and evaluation are given in Table 7.1.

Table 7.1: Pre-defined input parameters

Parameter	Value	Parameter	Value	Parameter	Value
η	0.7	L_{pv}^{temp}	0.25	SoC_{thr}	40%
S_{dc}^{pv}	3271 m^2	L_{wn}^{temp}	0.25	SoC^{mx}	100%
$cos(\alpha)$	0.07	L_{dc}^{esu}	0.5	D_{dis}^{max}	75%
ρ (av)	1.146732	E_{mx}^{esu}	50 kWh	D_{dis}^{min}	10%
Ç	0.5	E^{rat}	16 & 12	β	10
$\hat{a}$	1800 m^2	ζ $(op\ time)$	10	ζ $(p\ time)$	20

The interactive workload traces having variable arrival rate as shown in Fig. 7.1 are rescaled from real traffic from Wikipedia [140]. The considered workload arrival rate is normalized for the sake of evaluation purposes. As evident from the figure, for initial few hours (1300 hrs to 2000 hrs), the arrival rate shows growth, but after this is shows a minor downward curve.

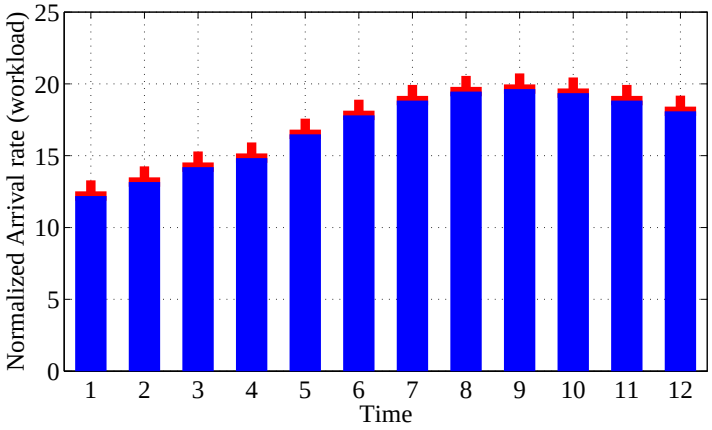


Figure 7.1: Workload arrival rate

Using the proposed scheme, the energy consumption of DC is shown in Fig. 7.2. The energy consumption of DCs is compared for two cases; 1) DCs with SDN-based control scheme, and 2) DCs with traditional network setup. The result obtained clearly shows that the energy consumption of DC is less when the proposed SDN-based energy-aware flow scheduling scheme is used. The major reason for the above result is the optimal use of network infrastructure. The use of SDN-based model results in almost 11.38% lesser energy consumption by DC.

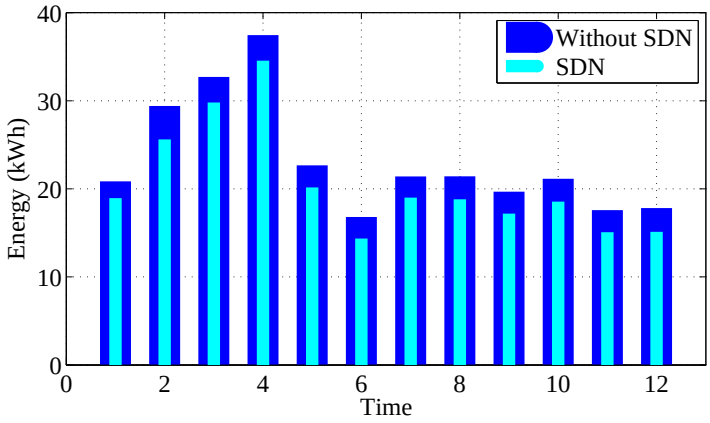
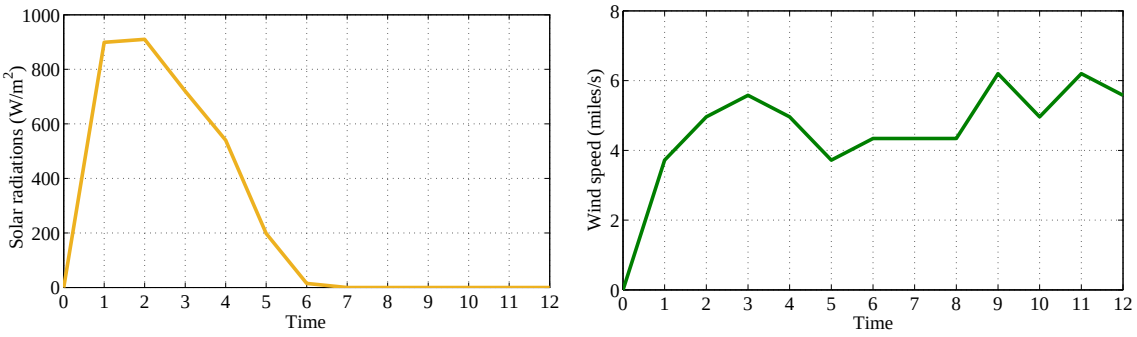


Figure 7.2: Energy consumption of DC

7.1.1 Energy generation by RES

The objective of this work is to sustain the energy consumption of DCs using RES. For this purpose, the energy generated by each DC is computed using weather traces. The DC is connected to in-house renewable energy generation sources, i.e, PV panels and wind turbines. For this purpose, solar radiations [165] and wind speed [166] at DC are considered for 12-hours as shown in Fig. 7.3(a) and Fig. 7.3(b).



(a) Solar radiations (b) Wind speed

Figure 7.3: Solar radiations and wind speed

The energy generated by PV panels and wind turbines deployed at DC is shown in Fig. 7.4. The energy generation by PV panels is high from 1300 hrs to 1400 hrs. After 1400 hrs, the energy generation goes down steeply and it becomes nil after 1800 hrs. However, the energy generation by wind turbines is variable throughout the time frame. The wind energy is almost negligible at 1300 hrs and 1700 hrs. The

wind energy is maximum from 2100 hrs to 2400 hrs. It is clearly evident that the energy generation by both RES is highly intermittent during 12 hours.

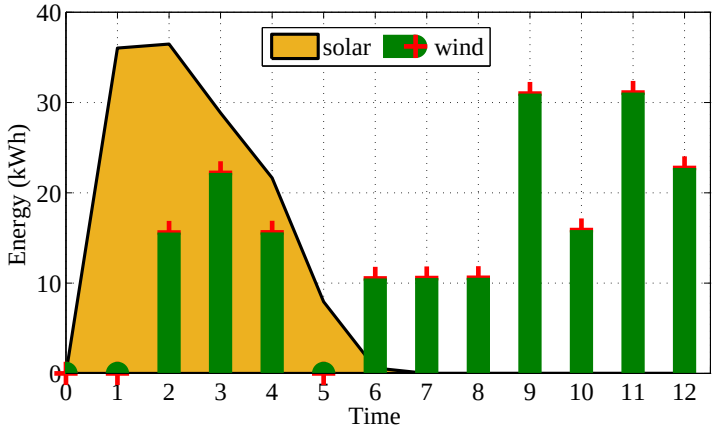


Figure 7.4: Energy generation from RES

7.1.2 Analysis of energy management scheme

Due to intermittent and variable nature of RES, the energy required by DC is different from the energy generation patterns. Fig. 7.5 shows the mapping of energy required by DC at each time slot with respect to energy generated by RES. The result shows that the energy generation by RES is in excess till 1600 hrs. After 1600 hrs, the energy generated by RES is not sufficient to power the DC. In this direction, Fig. 7.5 shows that the energy generation and requirement shows similar mismatch patterns for further time slots also.

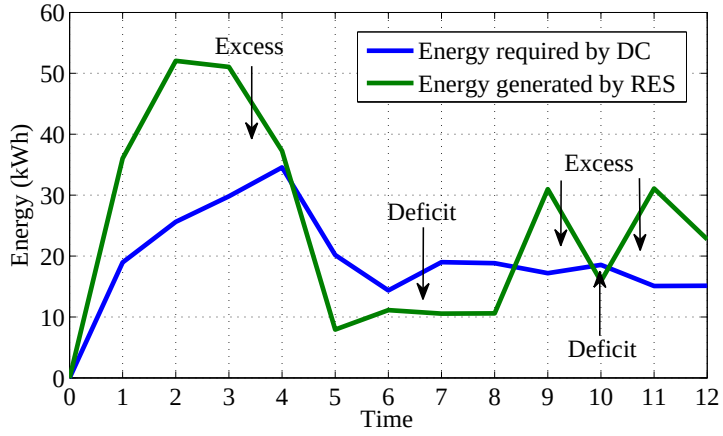


Figure 7.5: Mapping of DC energy consumption and energy generated by RES

For the above reason, the proposed energy management scheme is used to supply sufficient amount of energy required to power the DC using ESU, EVs, and power grid. Fig. 7.6 shows the excess and deficit in energy to power the DC. It is evident from the figure that the green colored bars shows time slots with excess energy and the red colored bars shows the time slots with energy deficit.

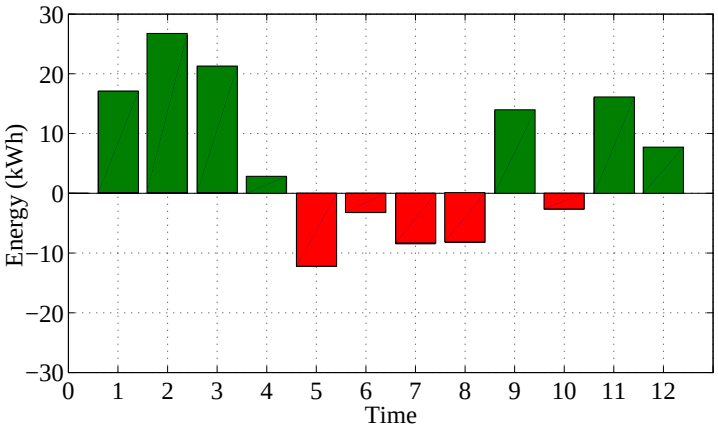
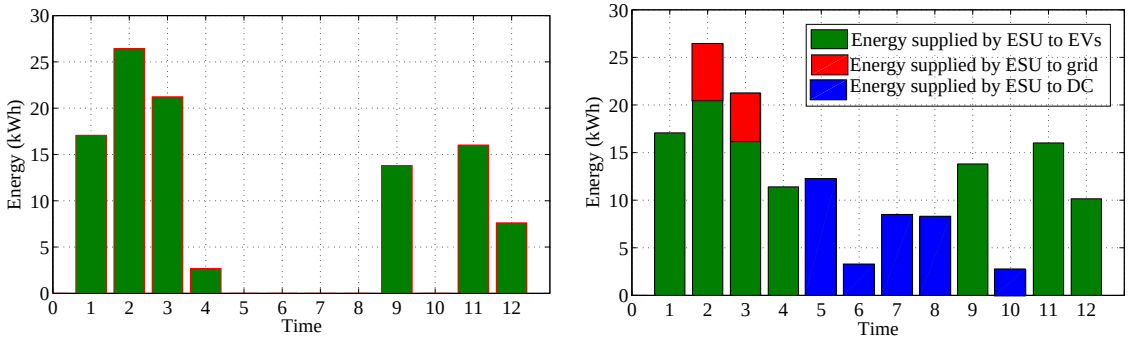


Figure 7.6: Excess and deficit of energy

To map the energy requirement of the DC, the excess energy generated by RES is supplied to ESU. The excess energy drawn by ESU from RES to maintain its maximum capacity is shown in Fig. 7.7(a). Once the ESU reaches its maximum capacity, the excess energy is either supplied to EVs or grid. The ESU supplies the stored energy to power DC at times when there is deficit of generation by RES. The energy supplied by ESU to various sources is shown in Fig. 7.7(b).

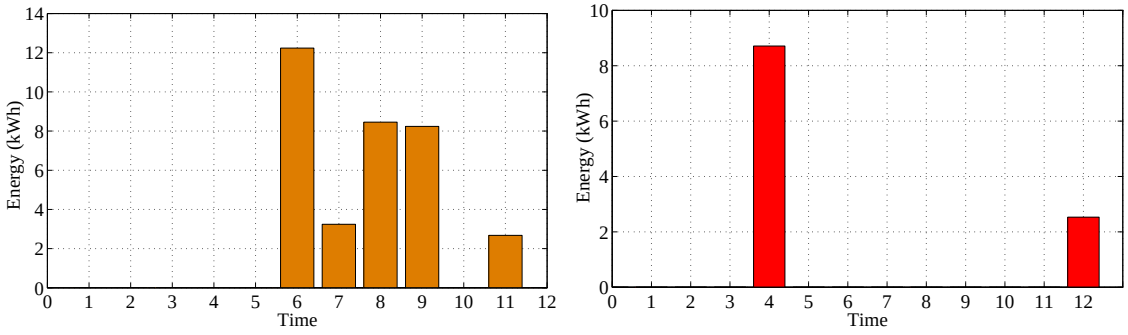


(a) Energy drawn by ESU from RES

(b) Energy supplied by ESU

Figure 7.7: Inflow and outflow of energy with respect to ESU

Further, whenever the ESU is below the maximum capacity then, it draws energy from EVs and grid. The energy supplied to grid is taken back from it whenever it is required. The result shows that energy is supplied to grid at 1200 hrs and 2300 hrs. This energy supplied to grid is taken back by ESU at 1400 hrs and 2400 hrs when it is in deficit to maintain its maximum capacity. The energy drawn by ESU from EVs and grid is shown in Fig. 7.8(a) and 7.8(b), respectively.



(a) Energy drawn by ESU from EVs (b) Energy drawn by ESU from grid

Figure 7.8: Energy drawn by ESU from EVs and grid

The EVs join the reward point scheme to participate in the energy management scheme for sustainability of DC using RES. The participating EVs supply the available energy to ESU whenever required. The ESU return the energy drawn from EVs when they leave the DC. The excess energy supplied to EVs above the energy drawn from it, reflects revenue. The price of excess energy supplied to EVs is computed using energy trading scheme based on Stakelberg game. Fig. 7.9 shows the revenue generated by DC using energy trading scheme. The proposed scheme supplies excess energy to EVs and gains revenue in addition to its objective of sustainability. It is evident from results that the proposed energy management scheme successfully sustains the energy consumption of DC using RES. However, to manage the intermittency of RES, the proposed scheme utilize the charging and discharging capability of EVs. Lastly, for the worst case scenario, power grid is also used as a partner to cope with the intermittency of RES.

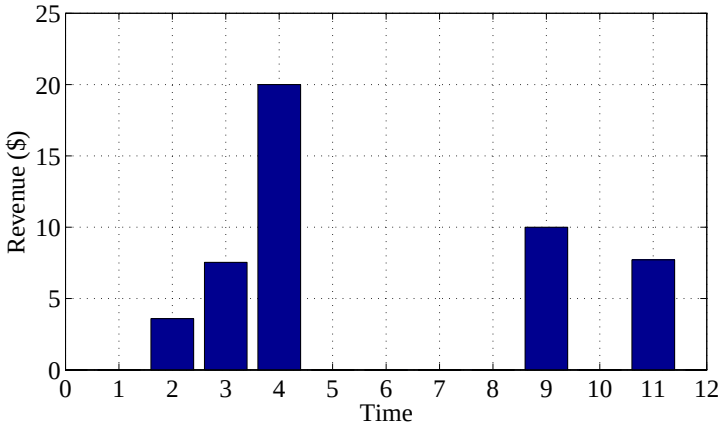
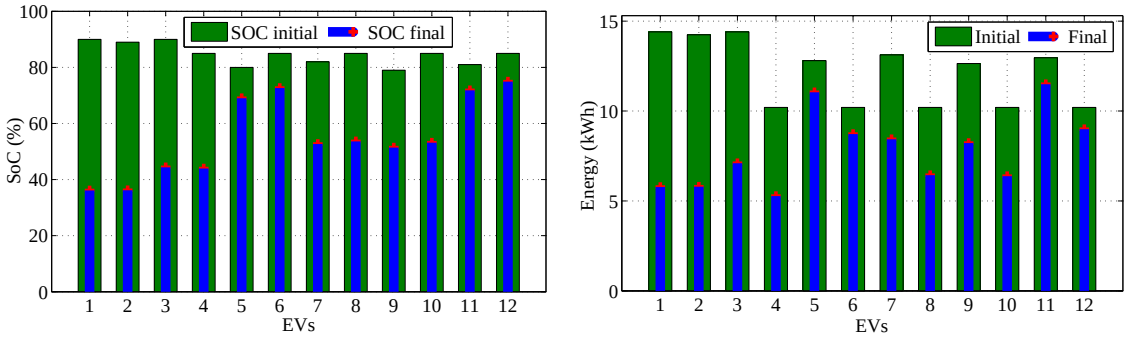


Figure 7.9: Revenue generated

7.1.3 Analysis of EVs participation in reward point scheme

EVs play a vital role in managing the intermittency of RES for sustaining the energy consumption of DC using RES. The charging and discharging capability of EVs is availed to handle the intermittency of RES. The EVs discharge the energy available with it to ESU when the energy generation by RES is in deficit. Fig. 7.10(a) shows the initial SoC and final SoC of participating EVs. The initial SoC is high and reaches at a low level once it discharge the required energy to ESU. Fig. 7.10(b) shows the initial and final energy level of participating EVs.



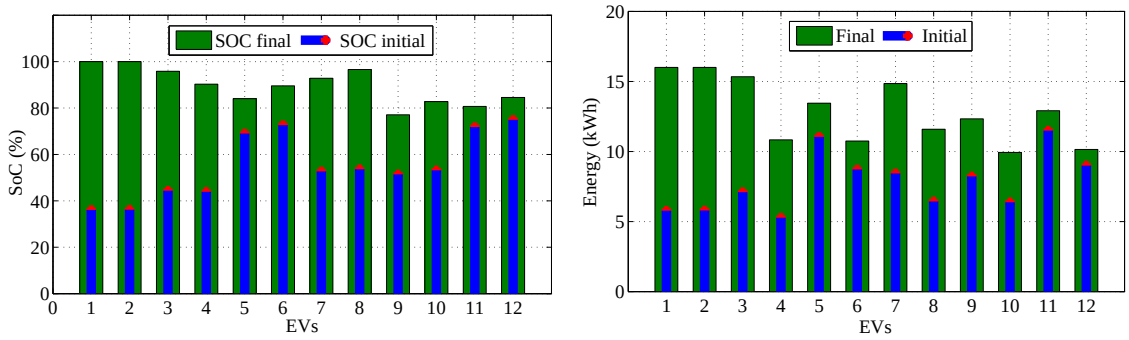
(a) SoC level during EV discharging

(b) Energy level during EV discharging

Figure 7.10: Analysis of EV discharging scheme

After the discharging process, the EVs are supplied the energy whenever they require. The minimum energy to be supplied to EVs must meet the threshold level. Generally, the energy drawn from EVs is limited to the threshold SoC. However,

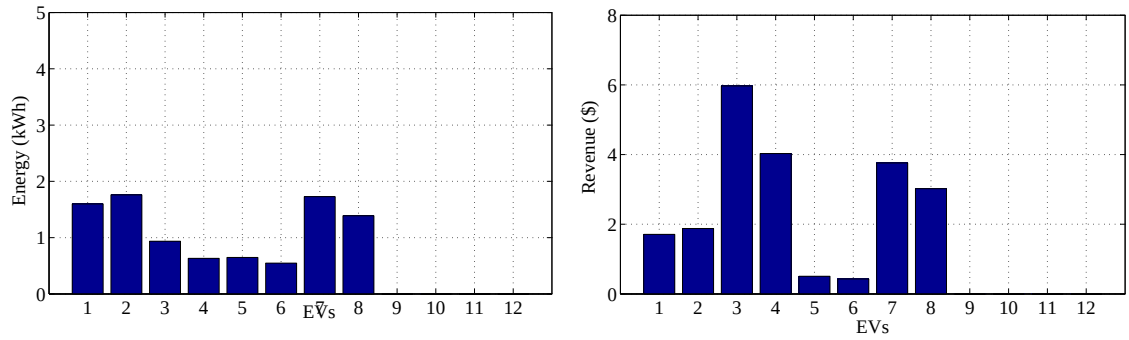
here, the energy drawn from EVs is even lower than threshold limit at some time slots. The initial SoC of EVs when they charge back the supplied energy is shown in Fig. 7.11(a). The final SoC attained after energy is supplied to participating EVs is shown in Fig. 7.11(a). Now, the energy supplied to EVs is more than the energy supplied by them to support the proposed energy management scheme. Fig. 7.11(b) shows the initial and final level of energy of EVs in the charging process.



(a) SoC level during EV charging (b) Energy level during EV charging

Figure 7.11: Analysis of EV charging scheme

The excess energy supplied to EVs is charged according to the price computed using energy trading scheme. The participating EVs that are supplied excess energy above the initial committed level are shown in Fig. 7.12(a). The corresponding revenue generated by supplying excess energy to EVs is shown in Fig. 7.12(b).



(a) Excess energy supplied to EVs (b) Revenue generated

Figure 7.12: Trading of excess energy supplied to EVs

Hence, after analysis of the above results, it is evident that the EVs drawn the required energy from ESU and they also supply the excess energy available with them

to ESU whenever required. To summarize this, the energy supplied and drawn from various participating EVs is shown in Fig. 7.13. Moreover, the proposed scheme not only supports the intermittency of RES, but also generates additional revenue for DCs by selling energy to EVs and grid.

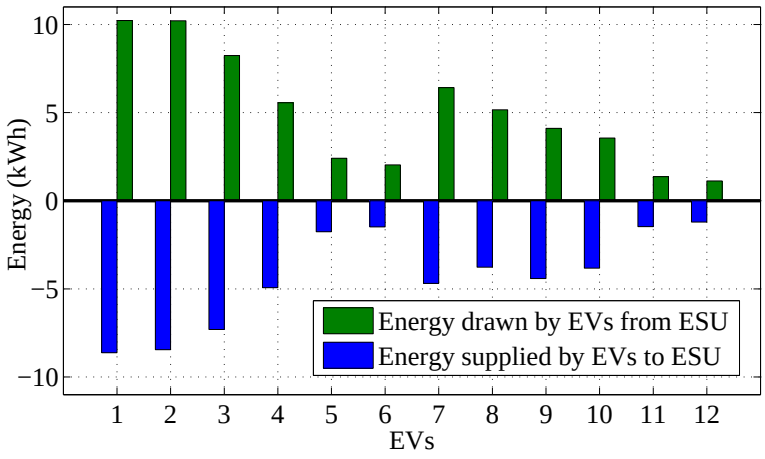


Figure 7.13: Energy drawn and supplied by EVs with respect to ESU

The EVs that participate in the reward point scheme are given reward points with respect to the energy drawn from them. The reward points are computed on the basis of initial, final, and threshold SoCs of the EVs. The SoC threshold for the proposed scheme is fixed at 40% for evaluation. The variation of reward points gained by EVs with respect to initial and final SoC is shown in Fig. 7.14.

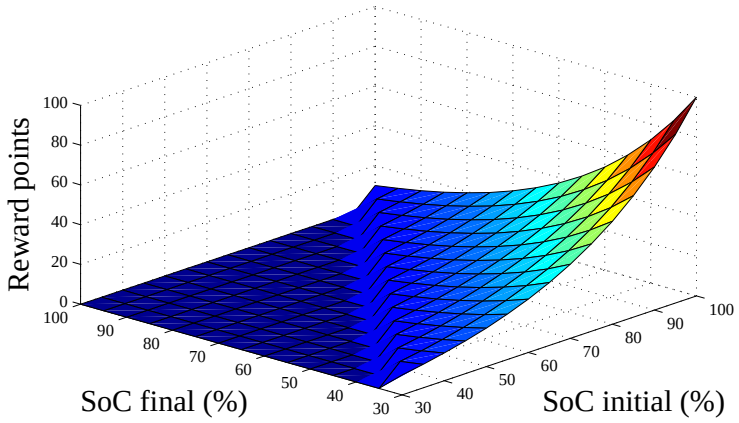


Figure 7.14: Variation of reward points with respect to initial and final SoC of EVs

7.1.4 Analysis of energy trading scheme for EVs

The proposed single-leader multi-follower Stackelberg game for energy trading is designed to decide an optimal price for excess energy supplied to EVs. The initial and final SoC of the EVs that drew energy from ESU using the proposed energy trading scheme are shown in Fig. 7.15.

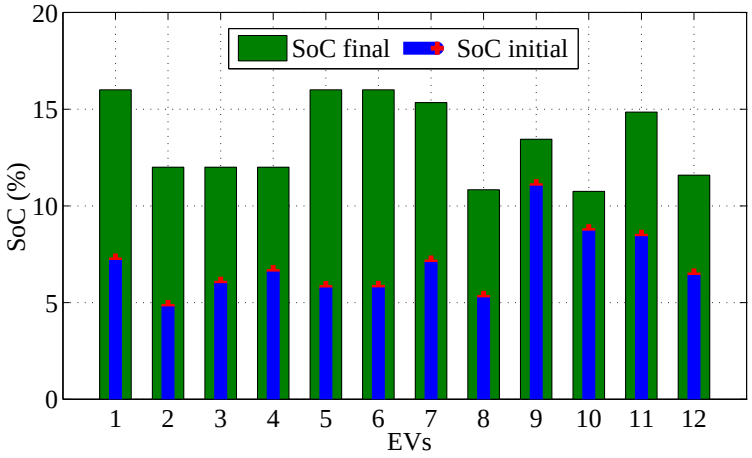


Figure 7.15: SoC level during energy trading

The energy drawn by EVs from ESU is shown in Fig. 7.16. The price of energy is computed on the basis of peak and off-peak time. The peak and off-peak time is decided on the basis of energy available at ESU and energy required by EVs. Fig. 7.16 shows that the EV3 and EV4 charge energy from ESU when it is peak time. The price charged in peak time is more than the price charged in off-peak time.

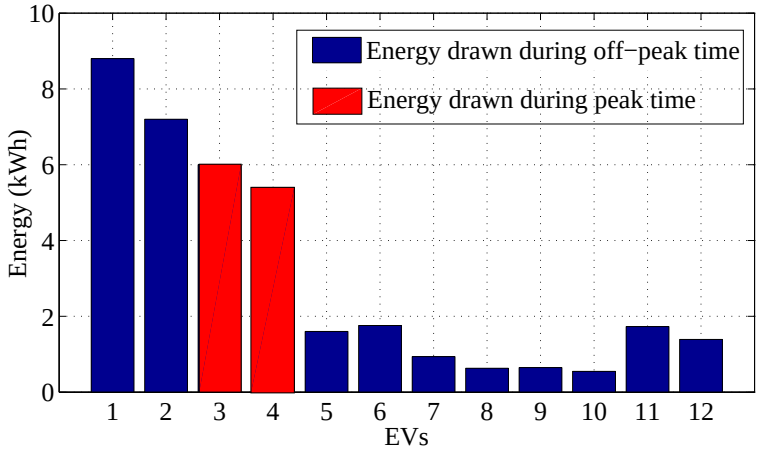


Figure 7.16: Energy drawn by non-reward point EVs

The revenue generated after supplying energy to EVs using the price computed by proposed Stackelberg game for energy trading is shown in Fig. 7.17. The high towers shown for EV3 and EV4 clearly depict the peak hours in the proposed scheme.

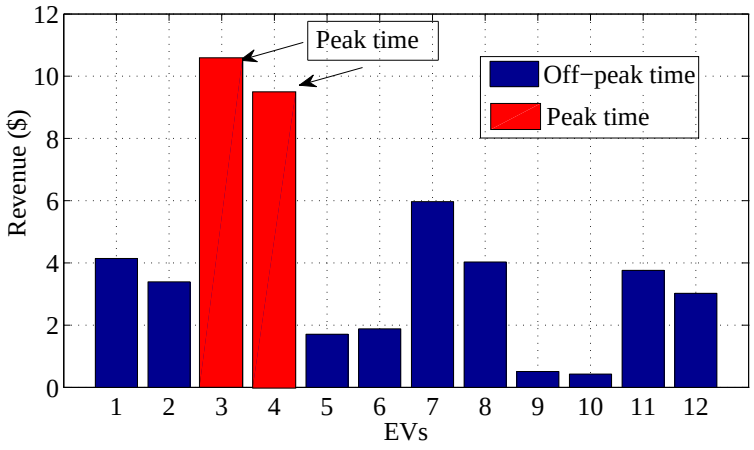


Figure 7.17: Revenue generated from non-reward point EVs

7.1.5 Existence of Nash equilibrium

The existence of Nash equilibrium is proved practically while deciding the optimal price of energy drawn from ESU by EVs using single-leader multi-follower Stackelberg game for energy trading. Table 7.2 shows the proof of existence of Nash equilibrium when 4 EVs request for energy from ESU. The single DC and multi followers reaches the equilibrium stage with respect to their utility functions. EV1 an CS shows the existence of Nash equilibrium and so EV1 is selected from the available EVs to charge the required energy from ESU.

Table 7.2: Existence of Nash equilibrium with respect to utility function

CS	**EV 1**	**EV 2**	**EV 3**	**EV 4**
EV 1	0.133/0.128	0.133/0.082	0.133/0.123	0.131/0.118
EV 2	0.067/0.128	0.067/0.082	0.067/0.123	0.067/0.118
EV 3	0.082/0.128	0.082/0.082	0.082/0.123	0.082/0.118
EV 4	0.034/0.128	0.034/0.082	0.034/0.123	0.034/0.118

7.2 Edge-based Energy Management Scheme

In this section, the second proposed scheme is evaluated in a simulated environment using a workload trace of 500 jobs extracted from Google workload traces [167,190]. For this purpose, three scenarios (case 1) only edge DCs and edge devices, (case 2) only cloud DCs, (case 3) edge-cloud environment, and (case 4) edge-cloud environment with server consolidation scheme are compared. For all these cases, using realistic weather traces, the mapping of energy generated by RES and energy consumption is done to assess the impact of proposed scheme on sustainability. For this purpose, using realistic weather traces [165] [166], the energy generated by RES (solar panels and wind turbines) is computed for a 12-hr scenario.

Table 7.3 shows the different parameters related to RES that are considered for calculations. Moreover, the impact of SDN and OFswitch consolidation scheme is compared with traditional networks. Finally, the role of different values of OL/UL threshold for server and network consolidation schemes is assessed. In this direction using three cases (1) 80, (2) 90, and (3) 100 for OL threshold and (1) 40, (2) 50, and (3) 60 for UL threshold are analyzed for both the consolidation schemes.

Table 7.3: Pre-defined input parameters

Parameter	Value	Parameter	Value
η	0.7	L_i^{bess}	0.8
Ç	0.5	L_{pv}^{tp}	0.25 & 0.75
$cos(\alpha)$	0.07	a	216 m^2
S_i^{sl}	3271 m^2	ρ	1.1357

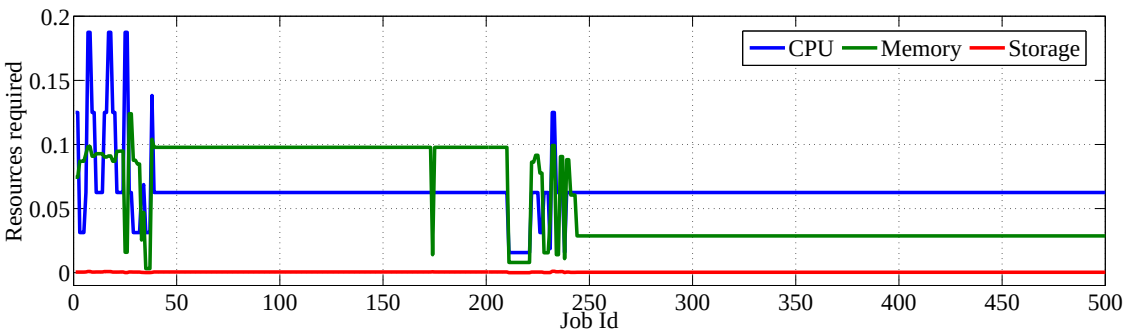


Figure 7.18: Resources required for handling incoming jobs

For the considered workload trace, various computing resources required to execute them. Fig. 7.18 shows the different resources such as-CPU, memory, and storage that are required to handle each incoming job. The arrival rate of each job is shown in Fig. 7.19.

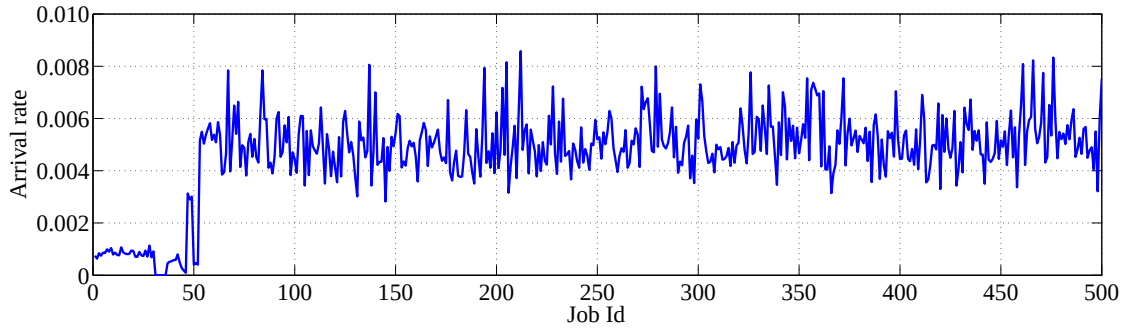


Figure 7.19: Job arrival rate

In this evaluation, the priority of each jobs is set using resource requirements and actual resource capacity of the cDC/nDCs. Apart from the resources, one of the most important parameters for classification of job requests is the level of delay-sensitivity expected for each incoming job. On the basis ot these parameters, Fig. 7.20 shows the priority of each incoming jobs.

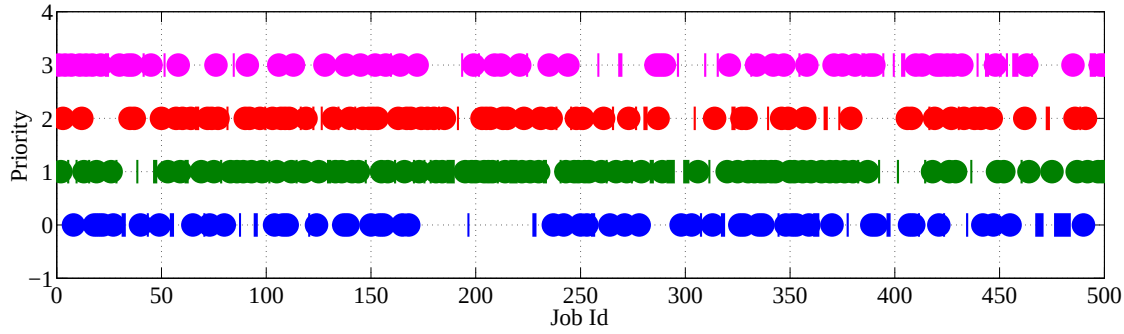


Figure 7.20: Priority of each job

Using this priority index, the SVM classifies all the jobs among cDCs and nDCs. After classification, these jobs are scheduled among cDCs or nDCs using the two stage game. The jobs that requires quick response time are routed towards nDCs. However, the jobs that required high computing resources are scheduled among cDCs. Fig. 7.21 shows the scheduling of the classified jobs among cDCs and nDCs.

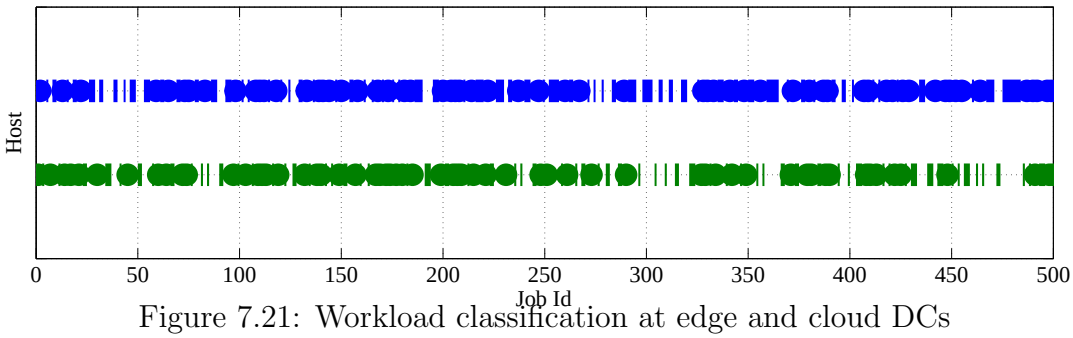


Figure 7.21: Workload classification at edge and cloud DCs

Now, the cDCs and nDCs consume some amount of energy to to execute different scheduled jobs smoothly. The energy consumption for each job depends on the resource requirement and the active time period for each job. Using these aspects the the energy consumed by cDCs and nDCs is shown in Fig. 7.22.

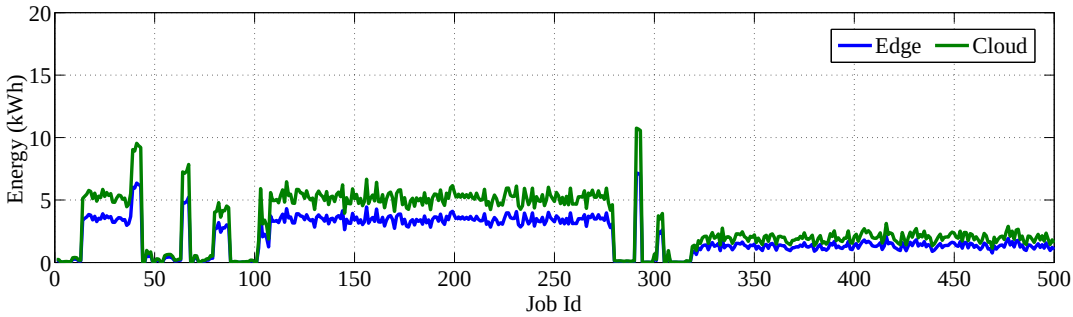


Figure 7.22: Energy consumed for handling each job

The energy consumed by edge-cloud environment is compared with case 1 and 2. Fig. 7.23 shows that the energy consumed by edge-cloud environment is lower than the energy consumed in case 1 and 2. From these results, it is evident that the proposed classification and scheduling scheme helps to minimize the energy consumption of each job by distributing them effectively among different nDC/cDCs.

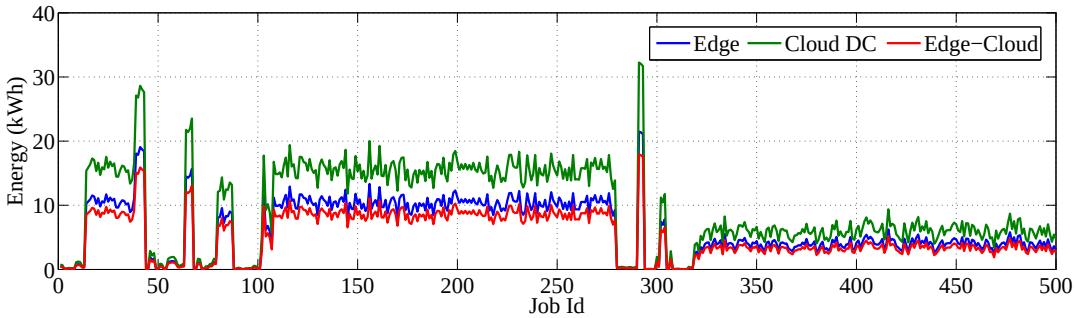


Figure 7.23: Comparison of energy consumption

For deep analysis of the proposed scheme, 4 cDCs and 5 nDCs are considered for evaluation purpose. In this regard, the scheduling of various classified job among these cDCs and nDCs is shown in Fig. 7.24.

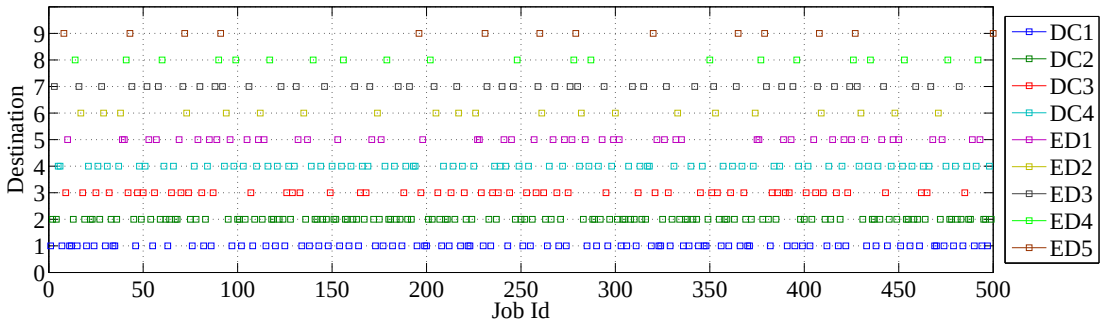


Figure 7.24: Destination selected using two stage game

The major goal of the proposed scheme is to achieve sustainability, i.e, mapping the energy consumption with energy generated by RES. To analyze the impact of proposed scheme on sustainability, a 12-hr scenario is presented. In this regard, the energy generated by solar panels and wind turbines is depicted in Fig. 7.25. The energy generated by RES is variable and goes high at some time slots and lower at other. Fig. 7.25 shows that energy generation by solar panels increases from 0100 hrs to 0800 hrs. After 0800 hrs, the energy generation by solar panels shows step drop. The energy generated by wind turbines is variable throughout the scenario.

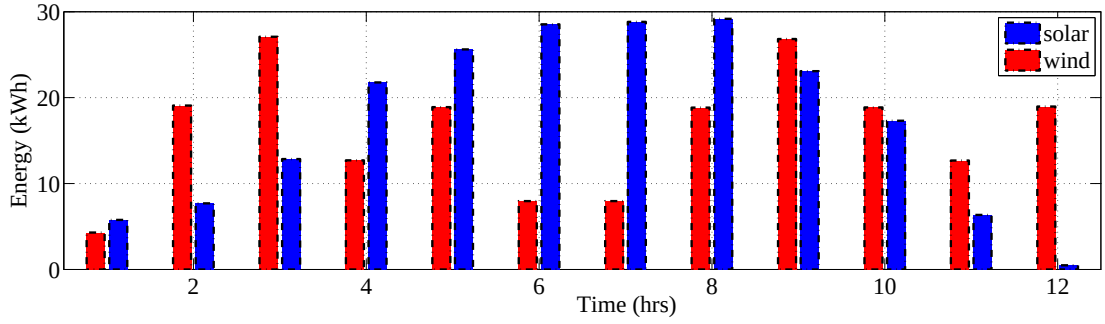


Figure 7.25: Energy generated by RES

The energy generated by RES is mapped with energy consumed to execute different jobs using nDCs. In this regard, Fig. 7.26 shows the mapping of energy generated by RES with the energy consumed by edge devices to handle all jobs.

The mapping shows that the energy generated by RES is in excess for just a short period of time (0720 hrs to 0950 hrs) only. For rest of the time period, energy deficit is witnessed at nDCs. Now, to overcome this energy deficit, the energy needs to be drawn from power grid which adds to the revenue burden and load on the grid also.

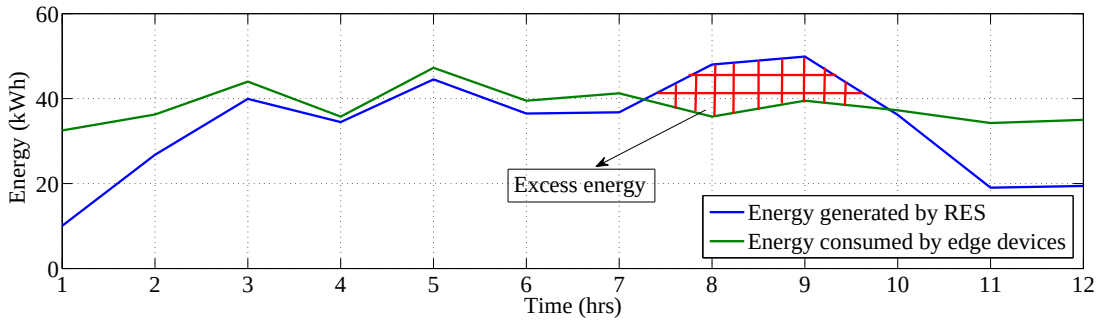


Figure 7.26: Mapping of renewable energy and energy consumed by edge devices

Similarly, Fig. 7.27 shows the mapping of energy generated by RES an the energy consumed by cloud DCs to handle different jobs. It is evident that time period from 0725 hrs to 0940 hrs witness excess generation. For rest of the time period DCs are in energy deficit. Here also, for mot of the period energy needs to be drawn from the power grid which adds to the revenue of the DCs.

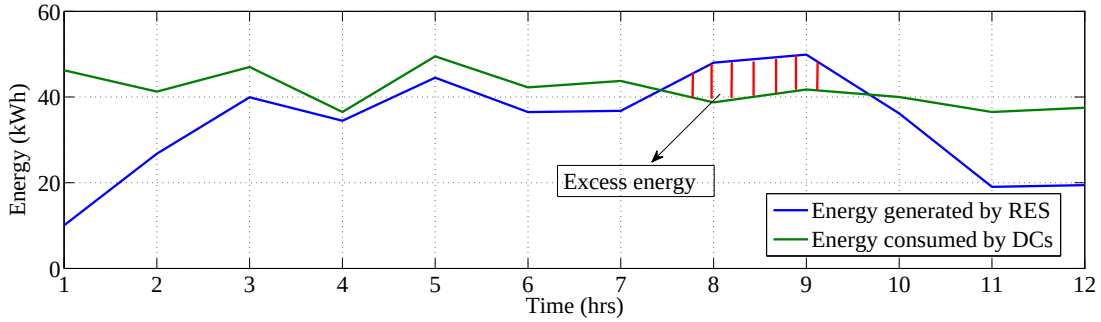


Figure 7.27: Mapping of renewable energy and and energy consumed by DCs

The mapping of energy generated by RES and energy consumed by edge-cloud environment is shown in Fig. 7.28. The mapping shows that energy deficit is witnessed for 3 hrs (0100hrs to 0230hrs and 1020 hrs to 1200 hrs). For rest of the time period, the energy generated by RES is more than the energy demand of the considered edge-cloud environment. Therefore, it is evident form the results obtained that

the proposed environment helps to sustain the energy demand of nDCs and cDCs in edge-cloud environment using RES.

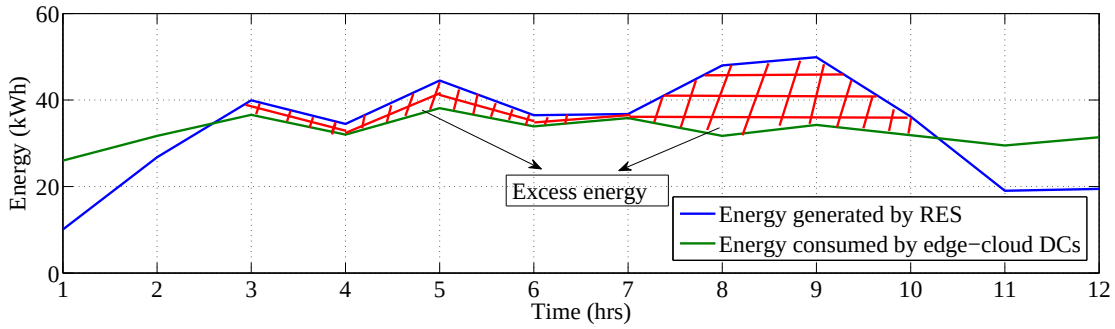


Figure 7.28: Mapping of renewable energy and energy consumed by edge-cloud DCs

Now, if the server and network consolidation schemes are used in the proposed edge-cloud environment, then the results show a great extent of reduction in the energy consumption. In this regard, Fig. 7.29 shows the energy saved by using server and network consolidation schemes. The energy saved using server consolidation scheme is higher than the energy saved using OFswitch consolidation scheme.

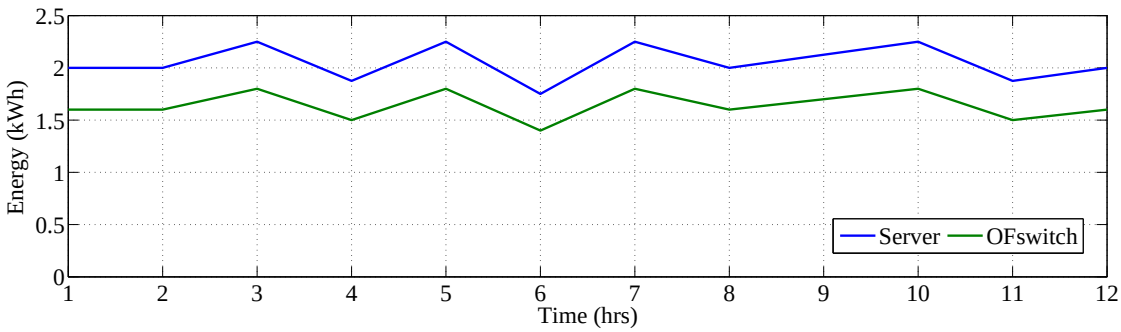


Figure 7.29: Energy saved

Therefore, using the proposed server and OFswitch consolidation schemes in the edge-cloud environment, the energy deficit goes down from 3 hrs to 2.5 hrs as compared to previous scenario. Fig. 7.30 shows the mapping of energy generated by RES with energy demand in edge-cloud environment using server consolidation scheme. The excess energy is maintained for almost 90 percent time period. Hence, it is quite evident from the above discussed results that the proposed scheme work well in order to achieve energy sustainability using RES for edge-cloud environment.

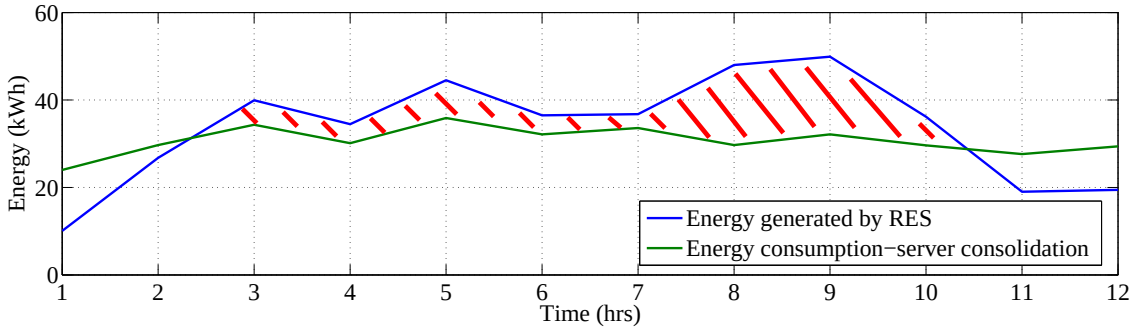


Figure 7.30: Mapping for edge-cloud DCs using consolidation scheme

Apart from energy sustainability, the proposed scheme proves its effectiveness with respect to SLA violations also. Fig. 7.31 shows that the SLA violations incurred for proposed scheme are far less than the case 1 and 2.

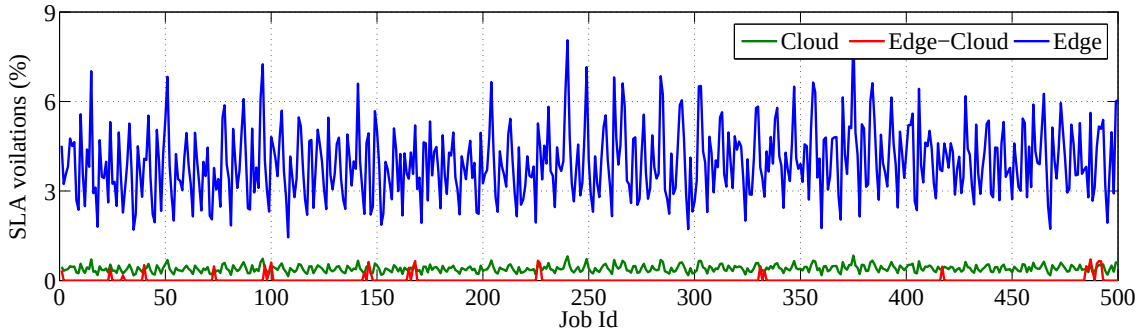


Figure 7.31: SLA violations

7.2.1 Analysis of SDN-based framework

In the presented edge-cloud environment, SDN-based control framework is used as to handle the control flow (flow management of incoming workload) of the underlying network. Therefore, an analysis of the impact of SDN-based framework on the proposed edge-cloud environment may help to reach different conclusions. In this direction, the results obtained suggest that the use of SDN in edge-cloud environment plays an important role in sustainability and QoS in terms of delay and bandwidth utilization. In this regard, Fig. 7.32 shows the bandwidth requirement for handling the considered workload.

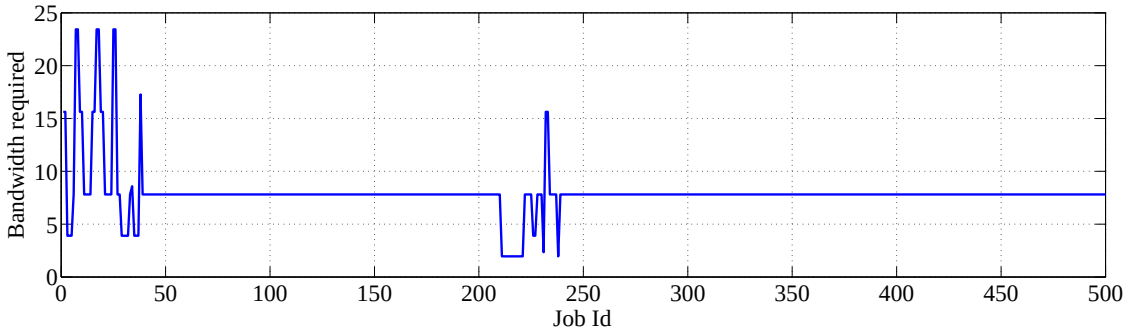


Figure 7.32: Bandwidth required

Moreover, SDN plays a significant role in handling the job migrations by ensuring lower migration delay and cost. In this regard, Fig. 7.33 shows the migration delay incurred which far less than the traditional networks. This is due to the dynamic selection of flow path according to the requirement.

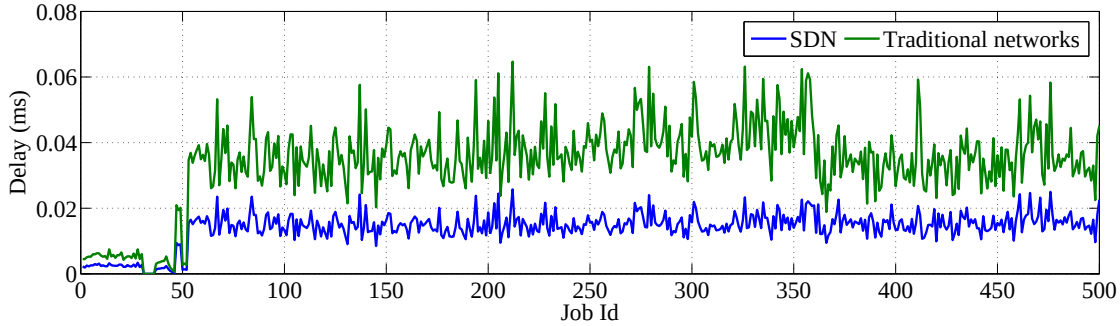


Figure 7.33: Migration delay

Furthermore, the migration cost (Fig. 7.34) incurred is also lower as compared to traditional networks. This is due to the lower migrations occurring in the edge-cloud scenario. Also, the delay and energy consumed for the job migrations is lower as compared to traditional networks which accounts to lower migration costs.

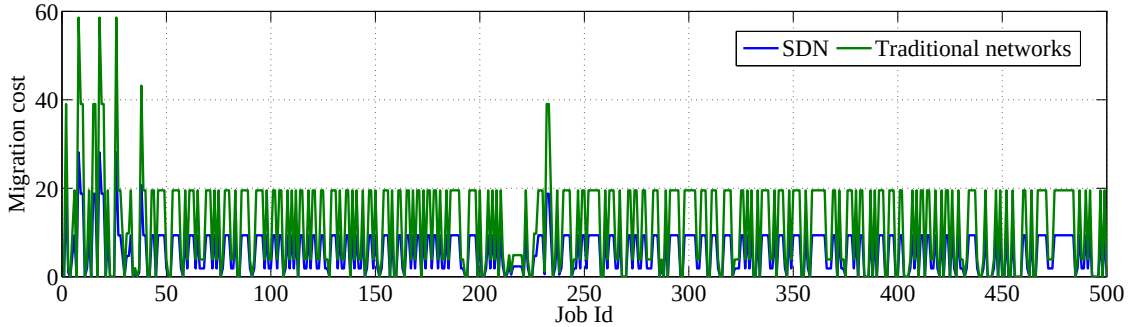


Figure 7.34: Migration cost

Now, the energy consumption of network infrastructure using SDN is analyzed and compared with the traditional networks. The results obtained suggest that the energy consumed by network devices using SDN is lower than the traditional networks. Fig. 7.35 shows the energy consumed by network infrastructure. However, if the proposed OFswitch consolidation scheme is used then the energy consumption reduced to further extent which is evident from the Fig. 7.29.

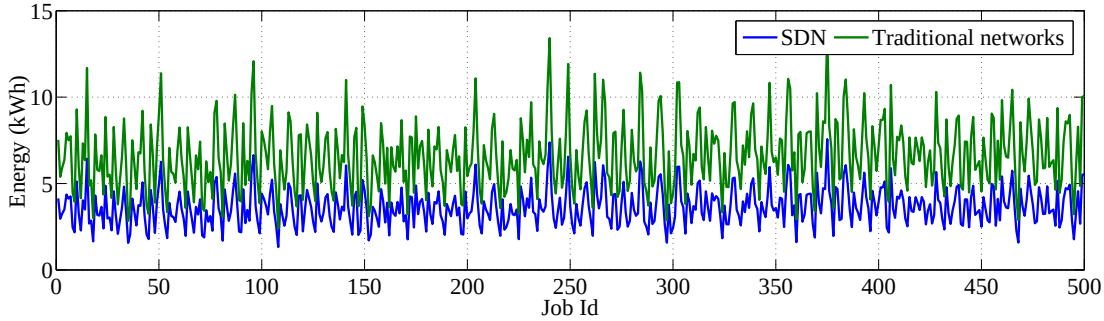


Figure 7.35: Network energy consumption

Finally, the impact of the data rate is compared with the performance of the proposed scheme. Fig. 7.36 shows the variation of latency with respect to different data rates. It is evident from the results that the SDN-based control framework leads to low variation in latency as compared to the other schemes.

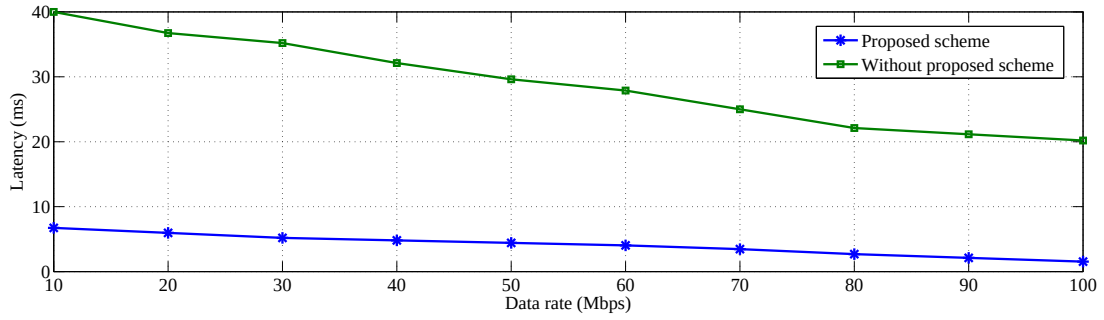


Figure 7.36: Variation of latency with respect to different data rates

7.2.2 Impact of OL/UL threshold on consolidation schemes

The performance of server and OFswitch consolidation scheme depend on the OL and UL thresholds. A significant effect of OL and UL threshold value is witnessed on the energy consumption, SLA violations, and migration rate. The OL thresholds

are considered as 80, 90, and 100 percents and the UL thresholds are opted as 40, 50, and 60 percents for comparison purposes. The analysis of energy consumption, SLA violations, and migration rate with respect to different OL/UL thresholds is discussed in details as below.

- **Variation of OL/UL on server consolidation:**

  In the performed analysis, the energy consumption is minimum for OL threshold of 90 percent and UL threshold of 60 percent. Fig. 7.37 shows the variation of energy consumption with respect to different OL thresholds, i.e., 80, 90, and 100 percents. The 80 percent threshold ends up in resource wastage and 100 percent threshold results in higher SLA violation and thereby increases the migration rate that adds to an increase in the energy consumption.

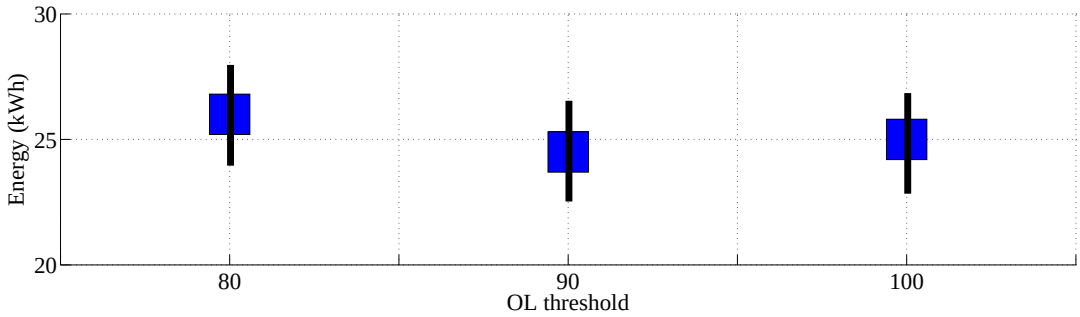


Figure 7.37: Variation of server energy consumption wrt OL threshold

Fig. 7.38 shows the variation of energy consumption with respect to different OL thresholds, i.e., 40, 50, and 60 percents. The lower UL thresholds result in higher resource wastage.

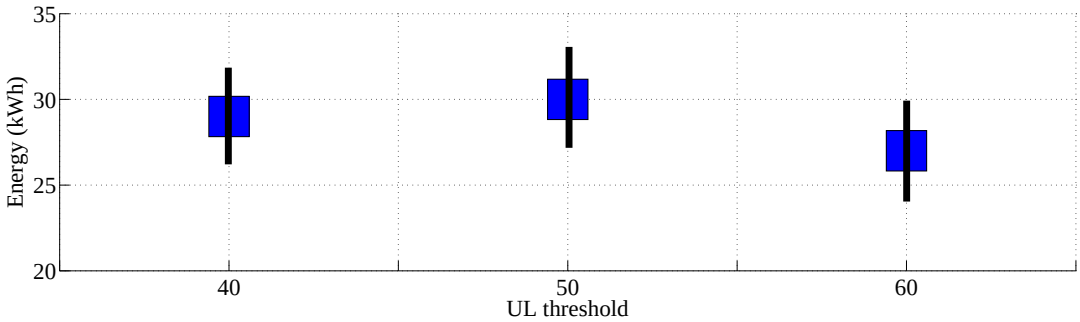


Figure 7.38: Variation of server energy consumption wrt OL threshold

The variation of SLA violations with respect to change in OL and UL thresholds is shown in Figs. 7.39 and 7.40. The results depict that 80 percent OL threshold and 50 percent UL threshold shows minimum SLA violations. This is due to the fact that higher OL threshold reduces the available resources and lower UL threshold ends up in resource wastage. So, it is evident that an appropriate value of OL/UL threshold maximizes the performance.

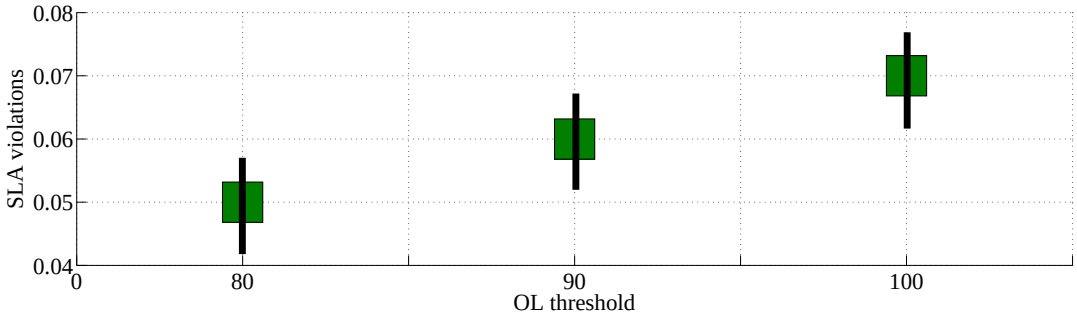


Figure 7.39: Variation of server SLA violations wrt OL threshold

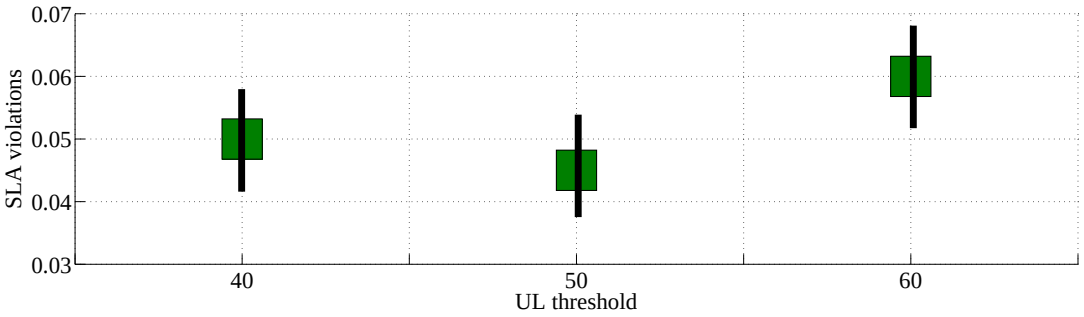


Figure 7.40: Variation of server SLA violations wrt UL threshold

Similarly, 100 percent and 40 percent OL and UL thresholds show lowest migration rates. The variation of migration rate with respect to change in OL and UL thresholds is shown in Figs. 7.41 and 7.42. After deep analysis of the above discussed results, it is concluded that 90 percent OL threshold and 40 percent UL threshold is best suited combination with respect to all three parameters. However, the OL/UL threshold can be opted according to the job requirements, desirable QoS, and SLA requirements. However, optimal OL/UL thresholds is necessary for the benefit of any designed solution.

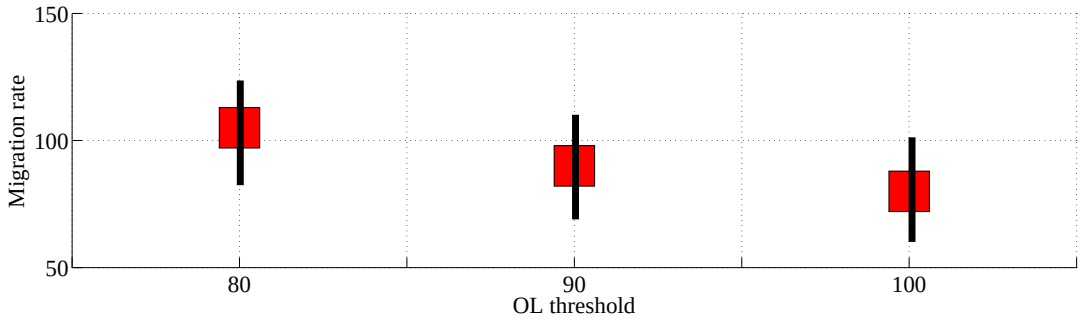


Figure 7.41: Variation of server migration rate wrt OL threshold

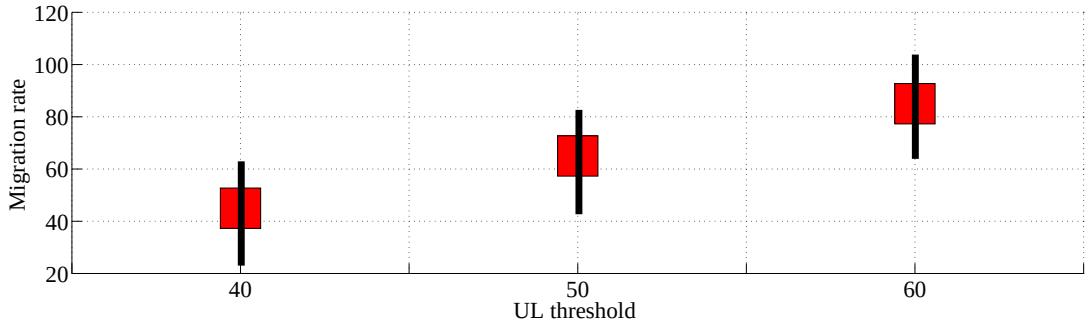


Figure 7.42: Variation of server migration rate wrt UL threshold

- **Variation of OL/UL on OFswitch consolidation:**

  In case of OFswitch consolidation scheme, the energy consumption for 90 percent and 40 percent OL and UL threshold respectively is lowest amongst other values of OL/UL thresholds. The variation of energy consumption with respect to change in OL thresholds is shown in Fig. 7.43. As discussed in the server consolidation scheme, the 80 percent OL threshold ends up in resource wastage and 100 percent OL threshold results in higher SLA violation and thereby increases the migration rate that adds up to the energy consumption.

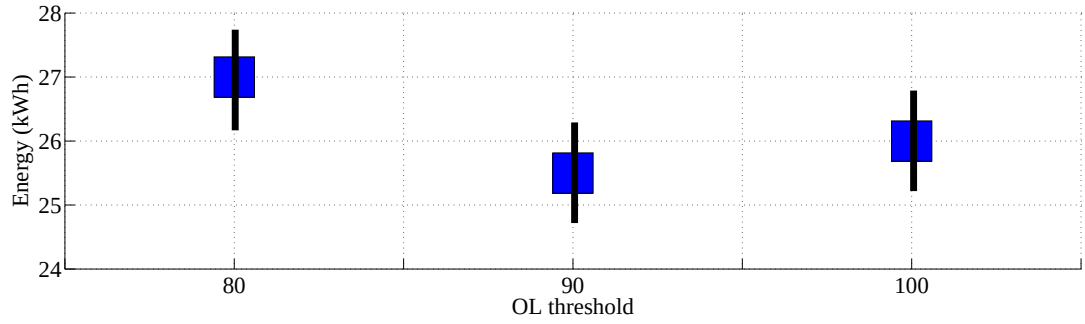


Figure 7.43: Variation of switch energy consumption wrt OL threshold

The variation of energy consumption with respect to change in UL thresholds is shown in Fig. 7.44. Similarly, 40 percent UL ends up in minimum resource usage, thereby resulting in minimum energy consumption. Therefore, it is evident from the above discussion that the appropriate value of OL/UL thresholds is necessary for achieving minimum energy consumption for handling different incoming jobs in edge-cloud environment.

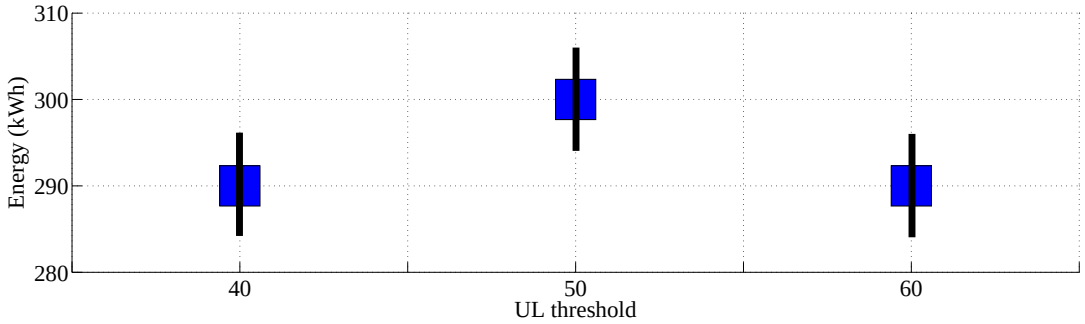


Figure 7.44: Variation of switch energy consumption wrt UL threshold

Similarly, the migration rate is also analyzed and the results show that 100 percent and 40 percent OL and UL thresholds respectively shows lowest migration rate. The variation of migration rate with respect to change in OL thresholds is shown in Fig. 7.45.

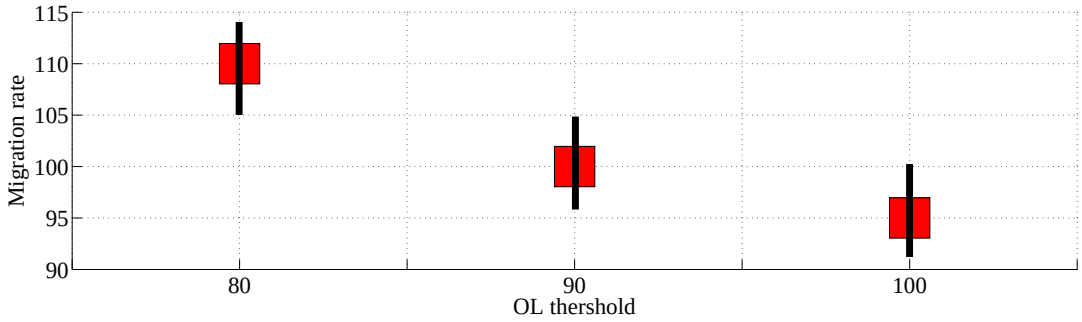


Figure 7.45: Variation of switch migration rate wrt OL threshold

The variation of migration rate with respect to change in UL thresholds is shown in Fig. 7.46. However, after deep analysis of the above discussed results, it is concluded that 90 percent OL threshold and 40 percent UL threshold is best suited combination for OFswitch consolidation scheme with respect to both the parameters. It is evident from the results obtained that the

OL/UL threshold can be opted according to the job requirements, desirable QoS, and SLA requirements. However, optimal OL/UL thresholds is necessary for the benefit of any designed solution. Therefore, it is necessary to decide the OL/UL threshold in advance before handling the jobs. It may be possible that different jobs may have different OL/Ul thresholds as per their resource requirements and SLA agreement.

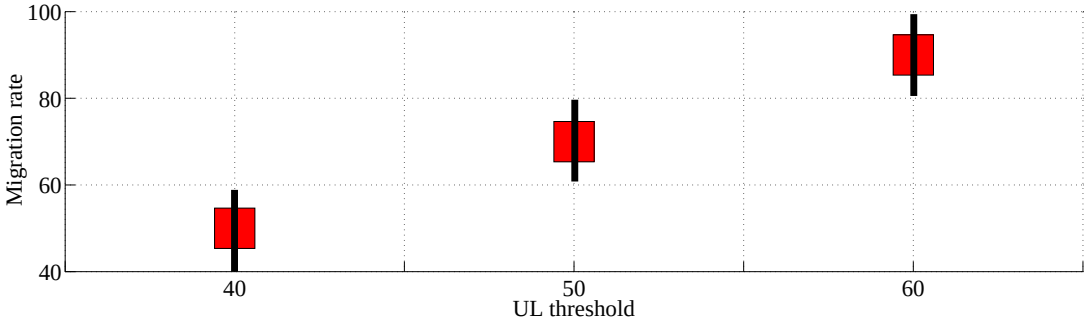


Figure 7.46: Variation of switch migration rate wrt UL threshold

7.2.3 Accuracy of SVM classifier

The accuracy of SVM effects the performance of the classifier. Hence, it becomes quiet necessary to analyze the variation of accuracy of the SVM classifier with respect to different parameters. The accuracy of the SVM classifier depends on the selection of optimal values of Ç and γ. The analysis of accuracy of SVM classifier with respect to Ç and γ is shown in Fig. 7.47.

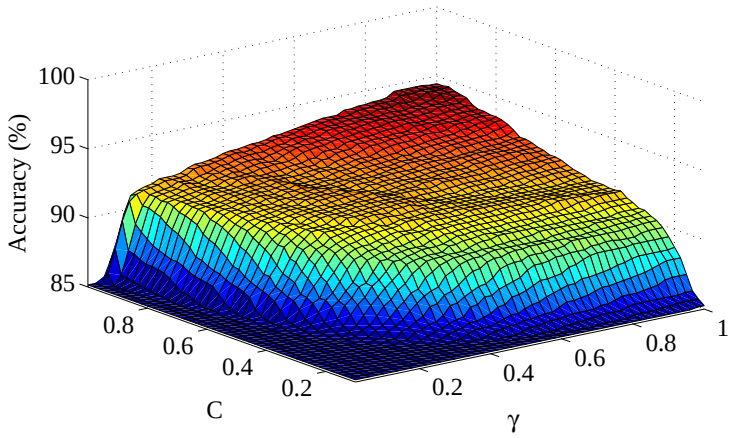


Figure 7.47: Accuracy of SVM with respect to Ç and γ

7.2.4 Variation of carbon index

Lastly, the variation of carbon index with respect to carbon rares and PUE of cDCs and nDCs is also analyzed. The analysis of carbon index with respect to carbon rate and PUE is shown in Fig. 7.48.

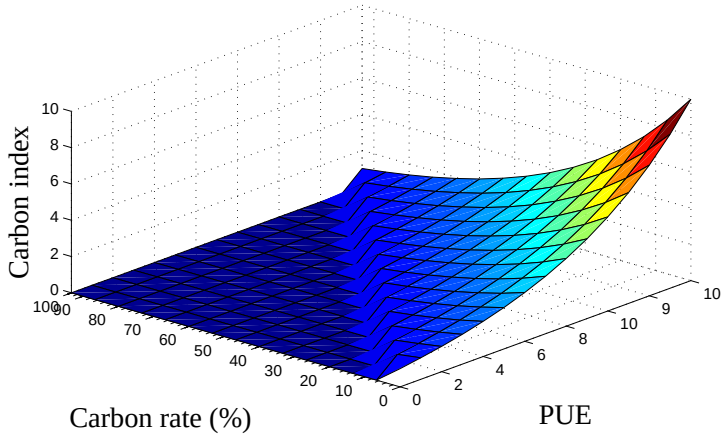


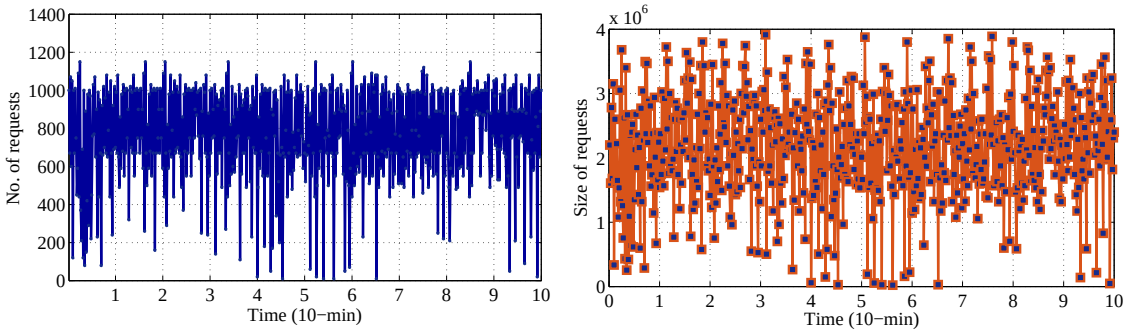
Figure 7.48: Variation of carbon index with respect to carbon rates and PUE

7.3 Energy-aware Resource Allocation Scheme

In this section, the proposed scheme and algorithms are evaluated using workload traces released by Google [167] in a simulated environment for three geo-distributed DCs. For gathering weather traces for solar radiations and wind speed, it is assumed that these DCs are geo-located in India: Mohali (Punjab), Chennai (Tamil Naidu), and Jaiselmer (Rajasthan). Each DC is having 20 servers which are allocated on the basis of utilization level. However, a case of 6 DCs having 120 servers is also considered to evaluate the scalability of the proposed scheme for sustainability. The heterogeneous physical servers considered for evaluation of the proposed scheme are categorized as, HP ProLiant DL380 G7 X5690 Hexacore 3.46 GHz (190 W) and HP ProLiant DL380p G8 E5-2697 v2 2.7 GHz 12-core (210W). The proposed scheme is not limited to any special type of servers or hardware configuration but can be extended according to the requirements. Each server has different hardware configuration and energy consumption profile.

For evaluating the proposed scheme and algorithms precisely, the results have been divided into three time-frames: 10-minute, 24-hour, and 1-year. The 10-minute time frame (1500 hrs to 1510 hrs on 9^{th} May 2015) is considered to get a deeper look into the working of the proposed scheme. The other two time-frames are used to depict the effectiveness of proposed scheme for a larger time scenario. The proposed scheme is compared with a resource-aware round robin scheduling policy-based scheme depicted as "without proposed scheme". The major objective of this work is to manage the energy required by DCs using energy generated by RES. To achieve this objective, the energy consumption of DCs is minimized through an optimal resource allocation and utilization. The following segments depicts the effectiveness of the proposed scheme for different time frames.

Cloud controller receives the user requests for services and then distributes the requests among the three DCs. Fig. 7.49(a) shows the number of requests received at cloud controller. The inter-job arrival is modeled using Poisson distribution. Fig. 7.49(b) shows the size of each received service request for the 10-min time-frame. The number of user request received are of varying type, workload, and size.



(a) No. of requests (b) Size of requests

Figure 7.49: Profile of considered workload traces

Cloud controller selects the available servers from different DCs as per the type of requests, size of request, and workload. The cloud controller divides the incoming requests into two queues. The high-priority requests are scheduled using PPRR scheme and low-priority requests are scheduled using FCFS scheme. Cloud controller decides the price to charged for resources and announce it to the requesting

users. Once the user accepts the price, the resources are allocated to them. Fig. 7.50 shows the number of servers provisioned to serve the service requests. The result clearly depict that by using the proposed resource allocation scheme, number of servers provisioned to serve the user request are reduced by 30.58 percent.

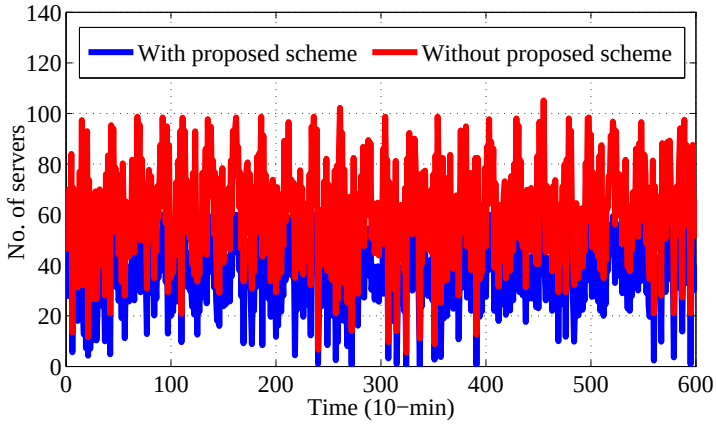


Figure 7.50: Number of servers provisioned

The above shown performance is achieved by utilizing the servers to their optimal level of utilization. The achieved level of utilization of servers provisioned is shown in Fig. 7.51. The optimal utilization of servers solves the problem of under-utilization or over-utilization of servers.

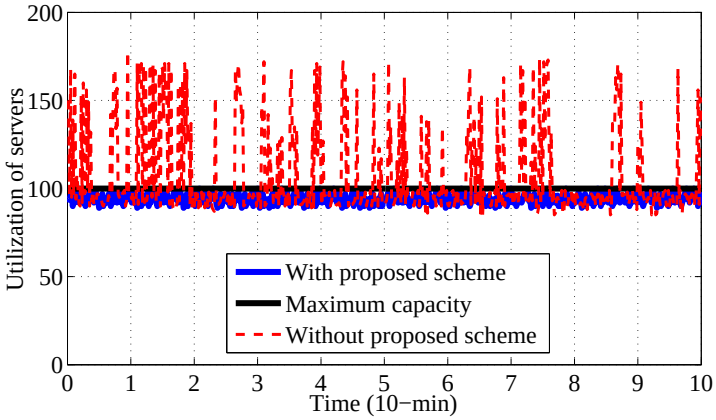


Figure 7.51: Utilization of servers

Fig. 7.51 shows that when the proposed resource utilization scheme is not used then, the most of the servers are not properly utilized. Hence, number of requests require additional resources at DCs as shown in Fig. 7.52. Hence, in such a case, the use of additional resources to meet the end user requests increases the energy con-

sumption of DCs. Moreover, if additional resources are not available at concerned DC, then such request is migrated to another DC, which generates additional migration delay and cost.

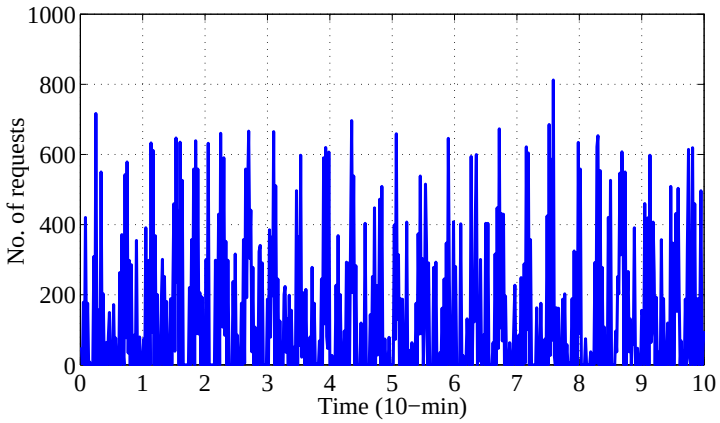


Figure 7.52: Requests allocated additional resources (without proposed scheme)

Using the proposed scheme, the optimal utilization of resources is achieved by reducing the number of servers provisioned for job requests. The un-provisioned servers, i.e., idle servers are shifted to energy saving mode as shown in Fig. 7.53.

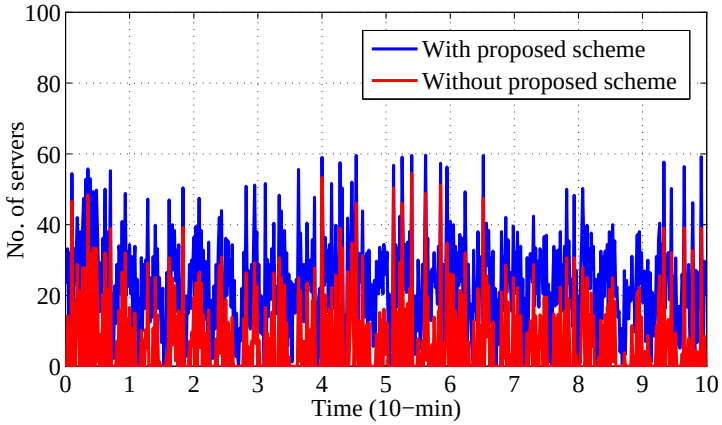


Figure 7.53: Number of servers shifted to energy saving mode

The energy consumption of DCs depends on servers provisioned and their utilization. Hence, with the reduction in number of servers allocated, the energy consumption of DCs also decreases. The energy required by DCs to serve the received user requests is shown in Fig. 7.54. The results clearly show a 22.81 percent decrease in the energy consumption of DCs. The results depicts that the energy savings achieved are enough to run for many hours.

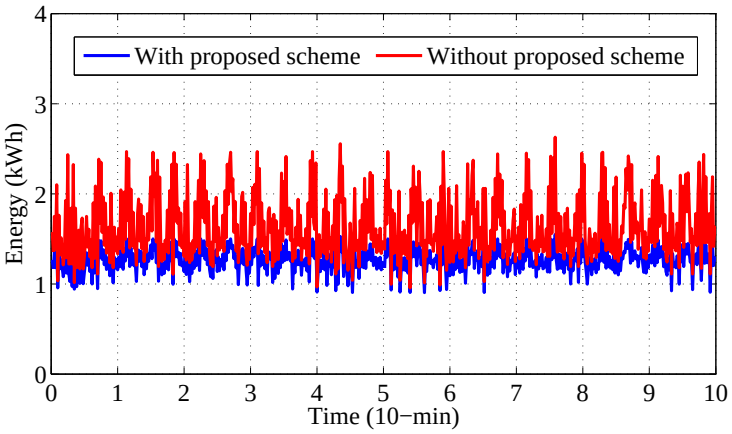
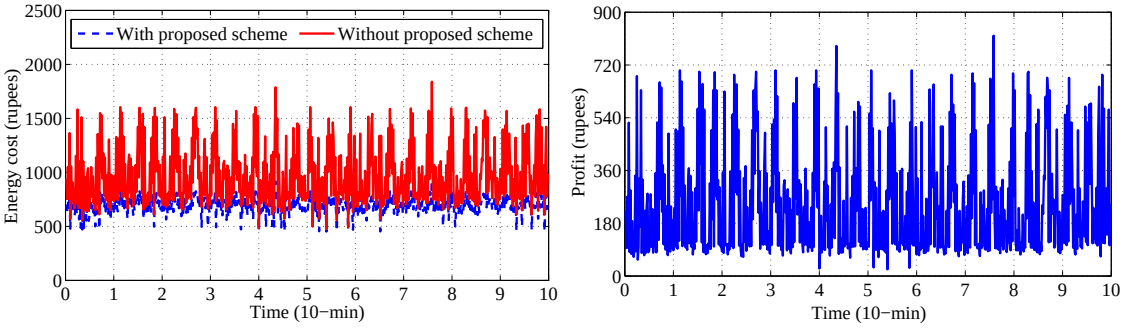


Figure 7.54: Energy demand of DCs

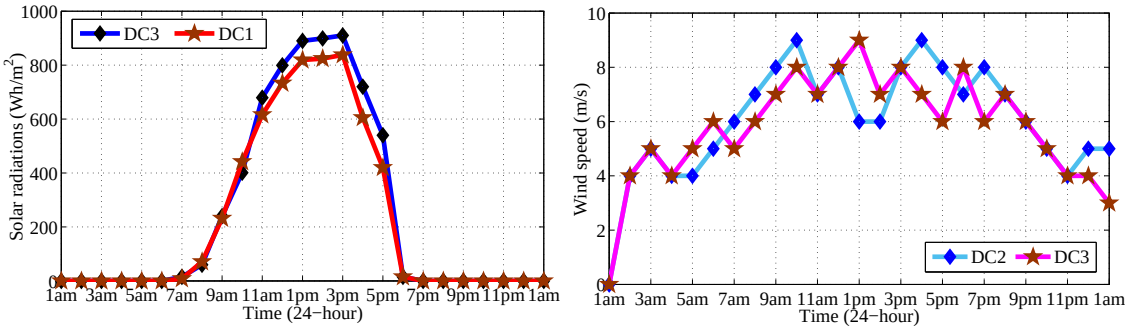
The cost incurred on energy is directly dependent on energy consumption of DCs. With reduction in energy consumption, the cost incurred on energy for DCs also reduces. Fig. 7.55(a) shows a 26.33 percent reduction in energy cost incurred for DCs whereas Fig. 7.55(b) shows the profit of DCs.



(a) Cost incurred on energy generation by RES (b) Total saving on energy consumption

Figure 7.55: Economics savings using proposed scheme

Now, the major objective of this work is to manage the energy consumption of DCs using RES. For this purpose, the realistic weather traces [165] [166] are considered. The DC located at Mohali (DC1) is connected to PV panels whereas the DC located at Chennai (DC2) is connected to wind energy source. The DC located at Jaisalmer (DC3) is connected to both solar and wind energy sources. To evaluate the proposed scheme, solar radiations [165] at DC1 and DC2 and wind speed [166] at DC2 and DC3 are considered for 24-hours (9^{th} May 2015) as shown in Fig. 7.56(a) and Fig. 7.56(b).



(a) Solar radiations (b) Wind speed

Figure 7.56: RES profile with respect to solar radiations and wind speed

The input parameters and constants considered for RES are given in Table 7.4.

Table 7.4: Pre-defined input parameters

Parameter	Value	Parameter	Value
η	0.7	L_{ess}	0.8
C_p	0.5	L_{exe}	0.25 & 0.75
a	0.07	A	216 m^2
S_{panel}	3271 m^2	k	20
U_{kj}^{thr}	0.95	C_j^{opor}	0.1

The energy generated by PV panels connected to DC1 and DC2 increases from 0800 hrs to 1600 hrs as shown in Fig. 7.57. The maximum solar radiations are received during this period of time. Energy generated by wind sources at DC2 and DC3 is also shown in Fig. 7.57. The energy generation is maximum from 0600 hrs to 2200 hrs. For rest of the day, the energy generation is almost negligible.

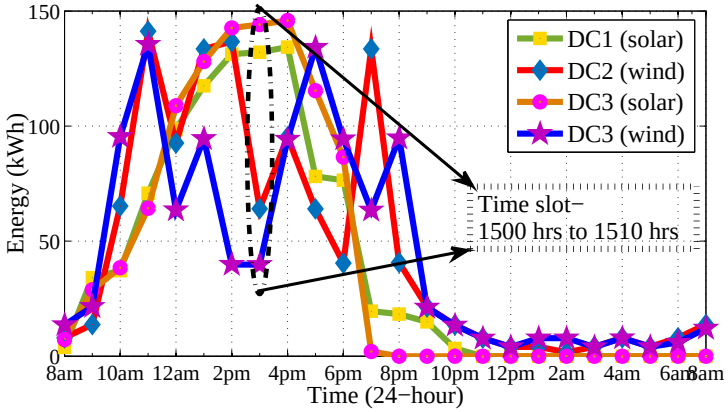


Figure 7.57: Energy generated by RES

The results highlighted in Fig. 7.57 shows that energy generated by RES connected to all DCs for 10-min time-frame (1500 hrs to 1510 hrs on 9^{th} May 2015) is

more than the energy demand of each DC as given in Fig. 7.54. Hence, the objective of managing the energy demand of DCs by RES is effectively achieved.

To achieve the above objective, the user requests received at cloud controller are allocated to available servers among all three DCs as per renewable energy available with them. The DC3 generates maximum amount of renewable energy in the current time-frame followed by DC2. Hence, DC3 is allocated highest number of user requests. The above results clearly show that the energy demand of DCs is completely met by RES using the proposed resource allocation scheme. Fig. 7.58 shows the distribution of user requests among DCs.

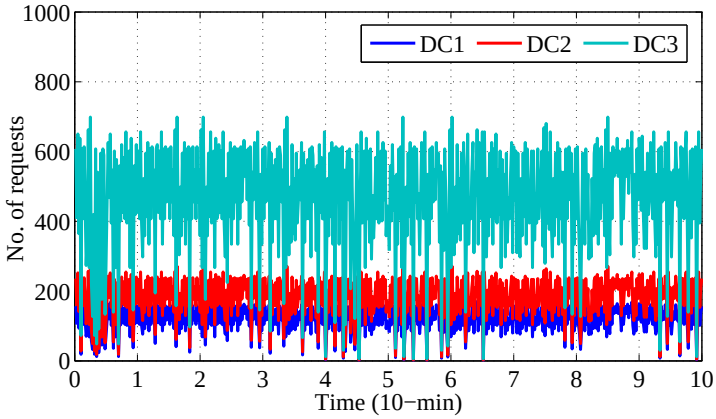


Figure 7.58: Distribution of request among DCs

Fig. 7.59 shows the energy drawn by DCs from all sources of generation. The results depict that the DCs need to draw 30 percent energy from grid, if the resources are not utilized effectively. In this way, the additional expenditure on drawing energy from grid is reduced.

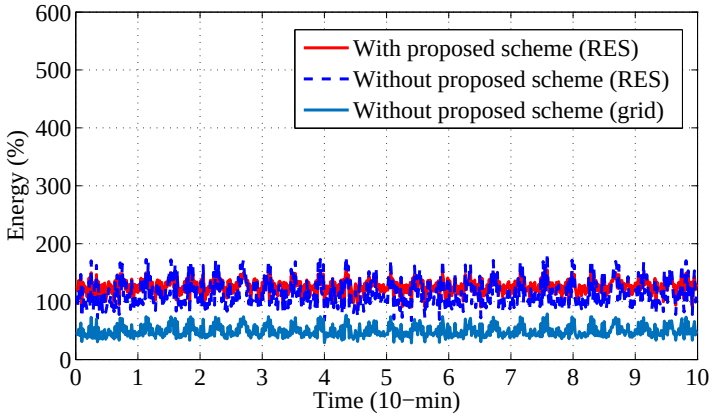


Figure 7.59: Energy drawn from various sources

After analyzing the 10-min time-frame, the proposed scheme is evaluated using 24-hour time-frame for 9^{th} May 2015 using data traces. The results show that the energy demand of all the three DCs is borne efficiently using RES as shown in Fig. 7.60. The resource allocation scheme reduces the number of servers provisioned to serve the user requests for 24-hour time-frame also. Hence, the energy consumption of DCs also reduces, thereby managing the energy demand of DCs using RES.

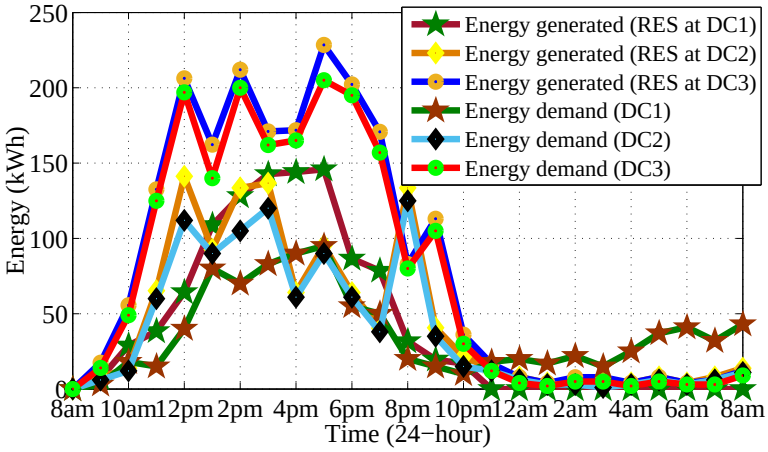


Figure 7.60: Mapping of generation and demand of energy

The DCs consume the energy stored in ESS which is generated by RES. If the energy stored at ESS is more than the energy consumption of DCs, then it is used at later times when there is energy deficit. The results show that from 0800 hrs to 2200 hrs, the demand is borne by energy generated using RES. During this period, RES generates energy more than the demand. However, after 2200 hrs the energy generated by RES is almost negligible. Hence, in such time the excess energy stored in ESS is used to manage the energy demand of DCs. However, at such times the users are provided with DC options according to the excess stored energy available with them. If a DC is not having enough stored energy then, such DC is not considered for user requests till the time it generates the abundant energy to serve the future demands. In the present scenario, user requests were scheduled among all the three DCs in such a way that DC1 and DC3 could store abundant amount of energy to met the demand at time when there is no energy generation.

The cost-effectiveness of the proposed scheme is also compared with price offered if energy is drawn from grid. Fig. 7.61 shows that the energy cost incurred for RES is far less than the energy price offered by grid. The cost considered for RES includes capital, maintenance and installation costs. However, an opportunity cost is also considered in the proposed scheme. The cost incurred on energy consumption of DCs leads the overall expenditure.

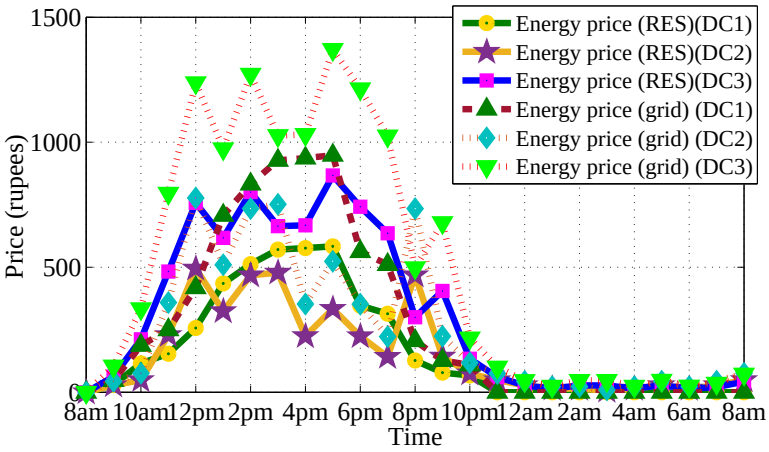


Figure 7.61: Comparison of energy price of RES with grid

Hence, reduction in energy cost leads to growth in overall profit. The profit margin per hour for each DC is shown in Fig. 7.62. The results show that the energy generation is maximum at DC3, thereby making it a major profit maker with respect to other DCs. The DC1 shows profit even at time when it does not generate energy and uses the energy stored at ESS.

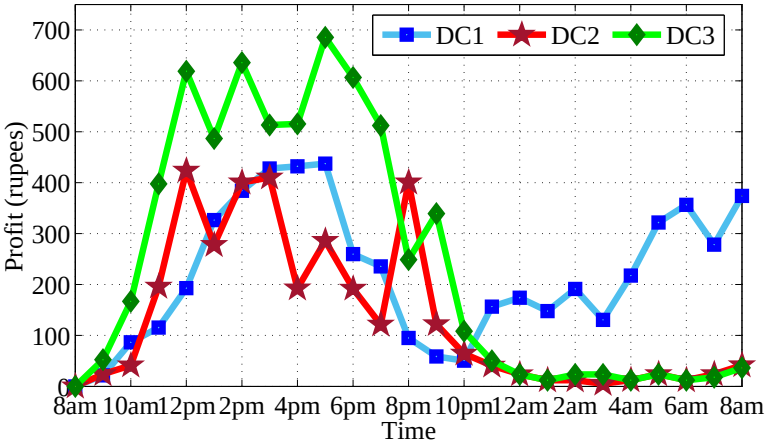


Figure 7.62: Profit gained by DCs

The increase in profit margin of all three DCs increases utility of cloud controller. Fig. 7.63 shows utility function of cloud controller with a linear growth while serving the user request efficiently. The profit gain occurs not only at DC level but at user level also. With lower expenditure on energy, DCs offer lower price for resources to users. Fig. **??** shows that utility of end users maintain itself close to the highest value, even with an increase in number of requests.

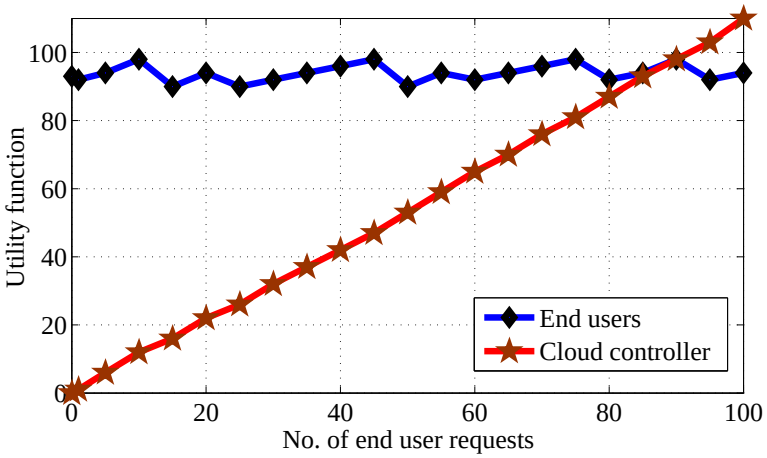


Figure 7.63: Utility function

Finally, the proposed scheme is evaluated for one year time-frame. The results show that reduction in energy consumption of DCs helps to sustain them using RES. The yearly break-up of energy management of DCs with respect to RES, ESS, and grid is shown in Fig. 7.64. However, due to the intermittent nature, energy generation by RES is reduced during some period of the year.

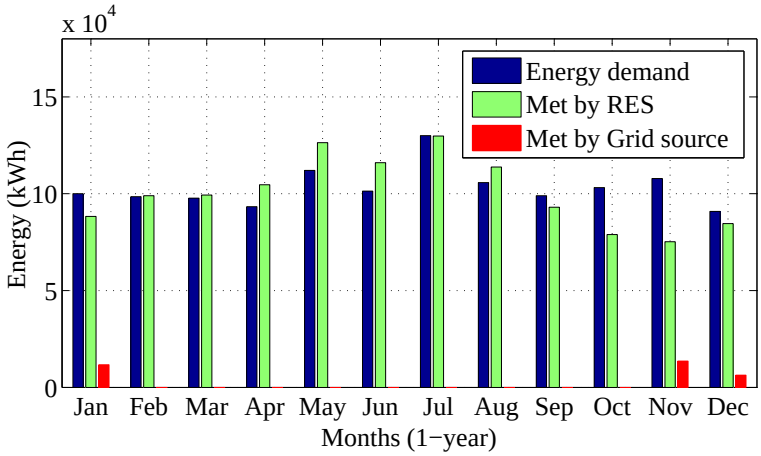


Figure 7.64: Energy drawn from various resources

Hence, in such period of time the energy is drawn from grid. With cold weather and foggy days at DC1 from January to March and from October to December, it shows negligible solar radiations. Furthermore, the variation in wind speed during some months lead to reduction in energy generation at DC2. In such times, the excess stored energy at DCs is used to manage the demand. In present scenario due to reduction in energy generation, the DCs considered have to draw energy from grid in the months of January, November, and December. During these months, some amount of energy is drawn from grid connected to DC2 which offers minimum energy tariff. Using the proposed energy trading scheme, DC2 is selected for serving the user requests as it is offered minimum energy tariff by grid [191].

7.3.1 Scalability evaluation of the proposed scheme

In this subsection, the proposed scheme has been evaluated for scalability and effectiveness in large-scale environment. For this purpose, total 120 heterogeneous servers have been considered for 6 DCs located at different locations. The results obtained clearly depict the effectiveness and scalability of the proposed scheme. Fig. 7.65 shows the energy required by DCs to serve the received user requests. It is evident from the results that a 20.81 percent decrease in energy consumption of DCs is achieved by using the proposed scheme.

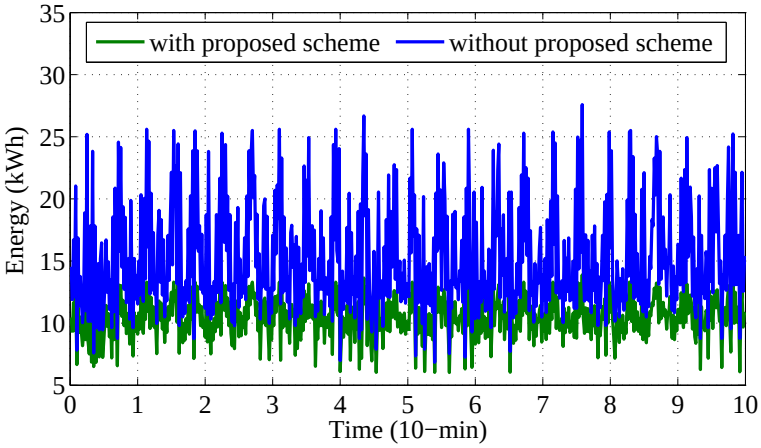


Figure 7.65: Energy demand of DCs

Fig. 7.66 shows a 24.63 percent reduction in energy cost incurred for DCs.

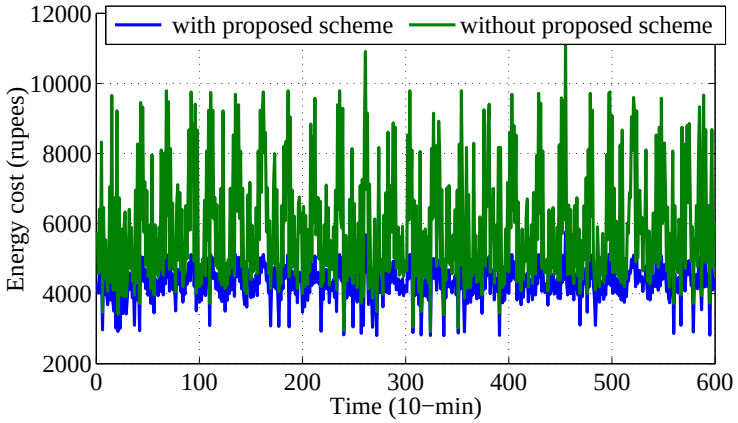
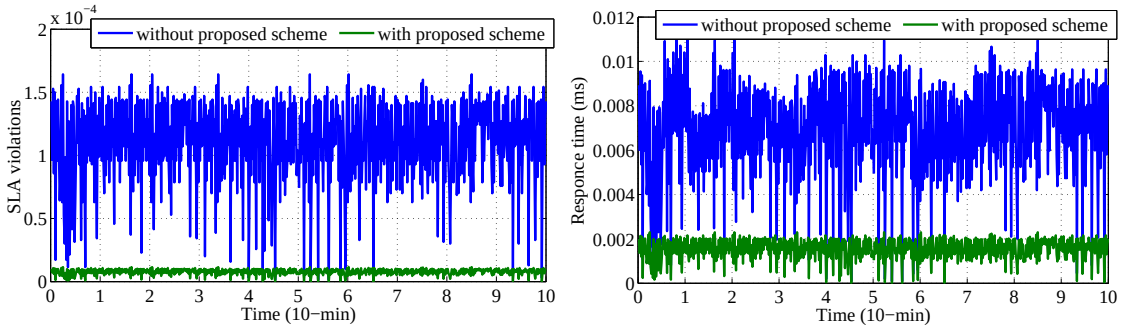


Figure 7.66: Cost savings

Further, the SLA violations incurred using the proposed scheme are shown in Fig. 7.67(a). The response time for serving the user requests is shown in Fig. 7.67(b). All the above achieved results clearly show that the proposed scheme is better suitable for sustaining the energy consumption of DCs using RES.



(a) SLA violations (b) Response time

Figure 7.67: QoS parameters

7.4 CoaaS-based Job Management Scheme

In this section, the proposed scheme is evaluated using extensive simulations for 3 geo-distributed DCs having 200 heterogeneous servers using Google workload traces [190]. In simulation setup, a startup and deployment delay of 0.5 seconds and 100 seconds respectively is considered for each container. Table 7.5 shows the server and container configurations. Network bandwidth is assumed to be 1 GB/s and 300 KB/s for servers and containers.

Table 7.5: Server and container Configurations

Server Configuration					Container Configuration		
Server type	CPU	Memory (GB)	E^p_{id} (kW)	E^p_{mx} (kW)	Container Type	CPU MIPS	Memory (GB)
1	4 cores	64	100	150	1	4658	128
2	8 cores	128	120	200	2	9320	256
3	16 core	256	150	250	3	18636	512

7.4.1 Sustainability of DCs

The impact of proposed scheme on sustainability of DCs is evaluated on the basis of two scenario: 1) 12 hrs and 2) 24 hrs, which are discussed as below.

- **12 hrs Scenario:** In this scenario, the proposed scheme is compared with Random Host Selection (RHS) and First Fit Host Selection (FFHS) schemes. The renewable energy traces used in the 12 hrs scenario are scaled up version of [129]. Using the consolidation scheme, the energy consumption of DCs reduces to a great extent. Fig. 7.68 shows the energy consumption of DCs.

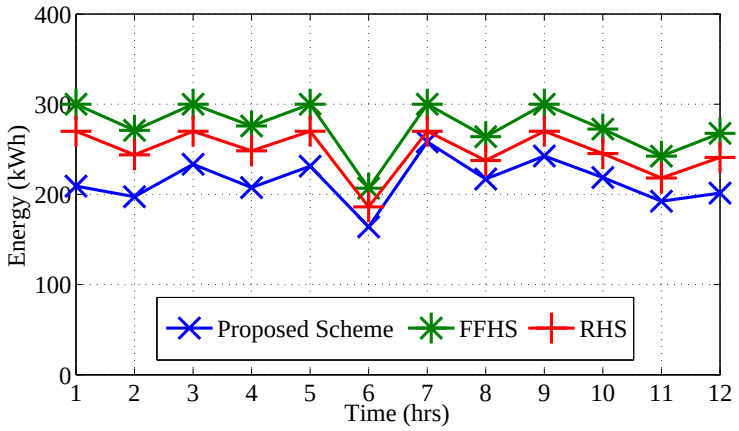


Figure 7.68: Energy consumption

FFHS scheme consumes highest amount of energy to handle the incoming workload. The proposed scheme helps to utilize DC resources in an optimal manner which results in maximum utilization of resources. By doing so, a large number of servers are deactivated to save energy. The proposed scheme saves 15 percent and 28 percent energy as compared to RHS and FFHS. Energy generated by RES (solar and wind) is shown in Fig. 7.69. Using the energy generated by RES, the demand of DCs is fulfilled.

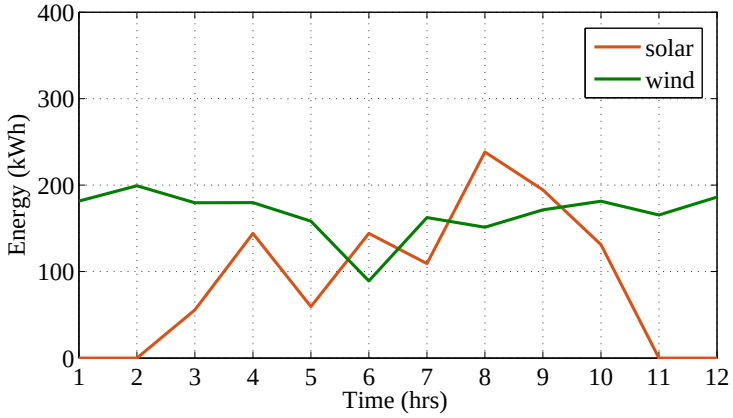


Figure 7.69: Energy generation

At initial time slot, energy generated by solar panels is low when no Sunlight is available, so it creates a deficit of energy at DCs. Such energy deficit is fulfilled by drawing energy from grid. This is because initially, the ESS is also having no energy stored in it. Fig. 7.70 shows the mapping of energy consumption of DCs and energy generation by RES.

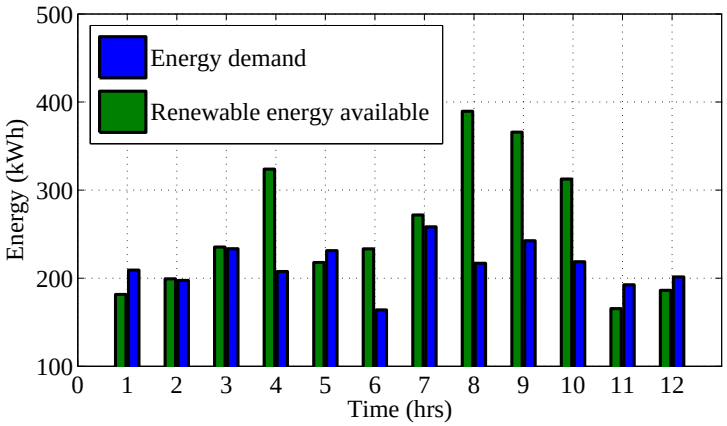


Figure 7.70: Mapping of Energy consumption and generation

The results show that 80 percent of the scenario witnessed excess energy generation that is stored in ESS and used to manage the energy deficit. Fig. 7.70 shows that only two occurrences of energy deficit exists (at 0500 hrs and 1100 hrs to 1200 hrs). At 0500 hrs, the energy stored in ESS is used to power DCs. Similarly, at other time slot with energy deficit, the required energy is supplied by ESS. Fig. 7.71 shows the cases of energy deficit and excess with respect to the mapping of consumption and generation of energy at DCs.

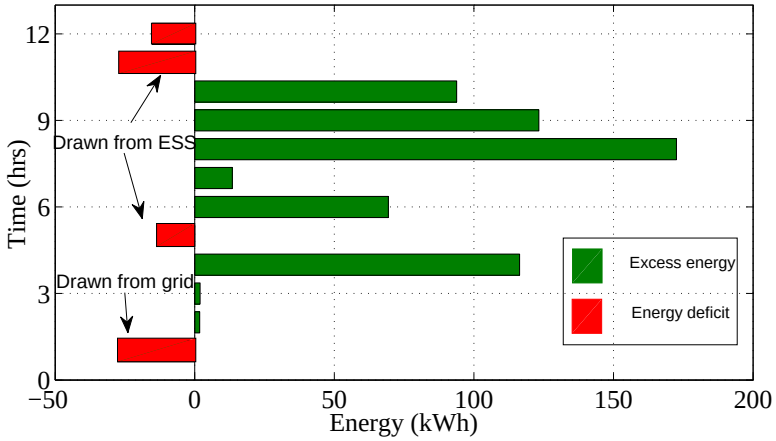


Figure 7.71: Excess and deficit of energy

- **24 hrs Scenario:** In this scenario, the proposed scheme is compared with resource allocation scheme presented in chapter 5. The renewable energy traces used in the 24 hrs scenario are scaled up version of [129]. Fig. 7.72 shows the energy consumed by the DCs using proposed scheme. It depicts that the energy consumed using proposed scheme is 10.55 percent lower in contrast to the other scheme proposed in chapter 5. Therefore, it is evident that containers are beneficial to improve the energy efficiency.

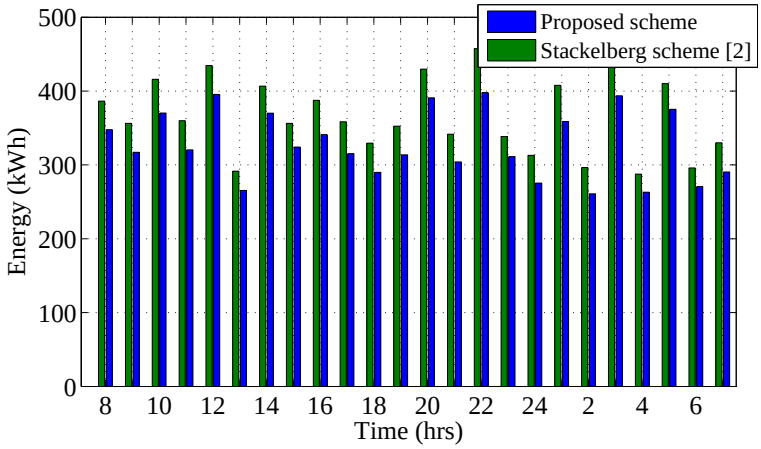


Figure 7.72: Energy consumption

Fig. 7.73 shows the energy generated by RES (solar and wind). It is evident form the figure that the renewable energy available is high in initial hours (0800 hrs to 2000 hrs). However, the renewable energy availability is lower in the later hours (2001 hrs to 0759 hrs).

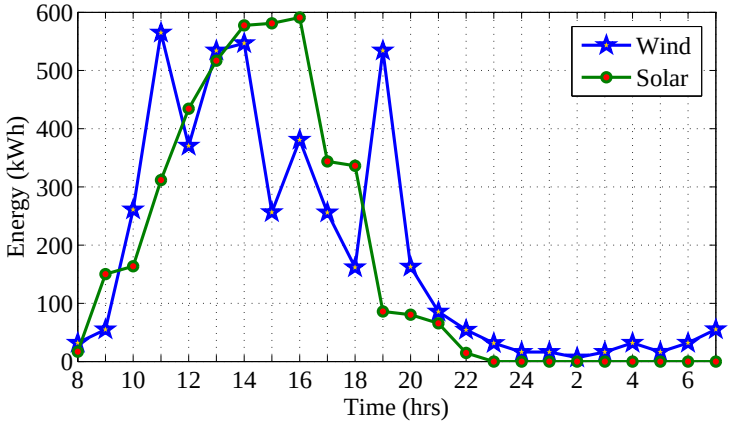


Figure 7.73: Energy generation

However, using the proposed scheme, the energy consumption of DCs is effectively sustained with the available renewable energy using ESS. The excess energy available in initial hours is stored in the ESS. The energy stored in ESS is used to power DCs in later hours when generation is at lower end. Fig. 7.74 shows the cases of energy deficit and excess with respect to the mapping of consumption and generation of energy at DCs.

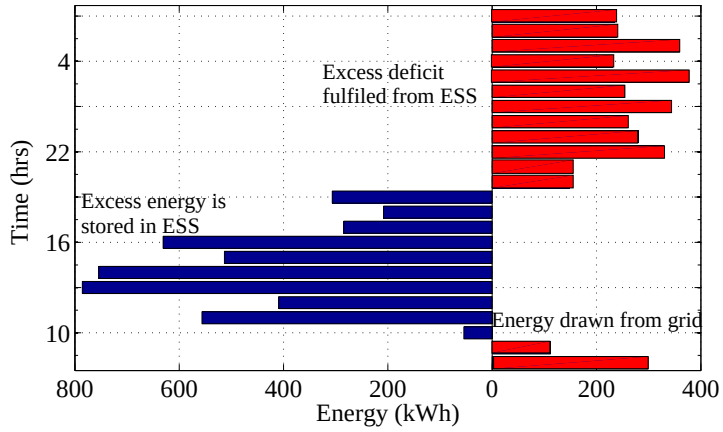


Figure 7.74: Mapping of energy consumption and generation

7.4.2 Overhead

Although the *CoaaS* model exhibits negligible startup, migration overheads. However, there is some overhead due to the network flow. Fig. 7.75 shows the overhead generated using the proposed scheme. It is evident that the proposed scheme exhibits lower overhead in contrast to other schemes.

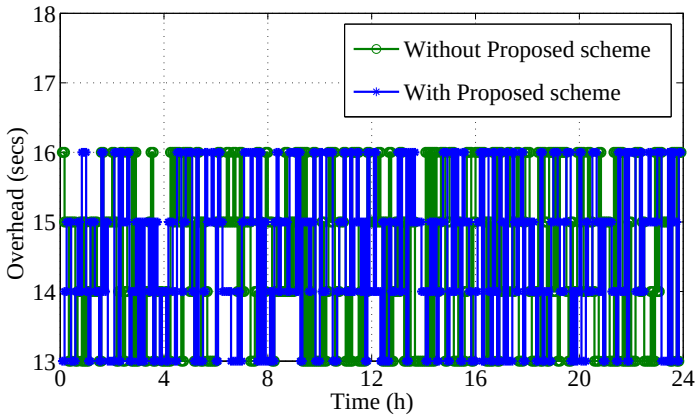


Figure 7.75: Overhead

7.4.3 Impact of Overload(OL)/Underload(UL) on DCs

The above discussion clearly shows that the container consolation scheme helps to reduce the energy consumption of DCs. By doing so, it becomes easy for the global controller to sustain energy consumption of DCs using RES. However, the OL and UL threshold value has a significant impact on the energy consumption, SLA violation, and container migration rate. In this sub-section, the impact of change in OL and UL threshold is analyzed. For this purpose, the proposed scheme (A), RHS (B), and FFHS (C) are compared. The OL threshold is considered as 80 percent, 90 percent, and 100 percent. Similarly, UL threshold is considered as 40 percent, 50 percent, and 60 percent.

- **Variation of OL:** In this analysis, energy consumption of proposed scheme (A) is lower than other two schemes (B and C). Fig 7.76 shows that the energy consumption for proposed scheme for 80 percent and 90 percent OL threshold is lower than 100 percent OL threshold.

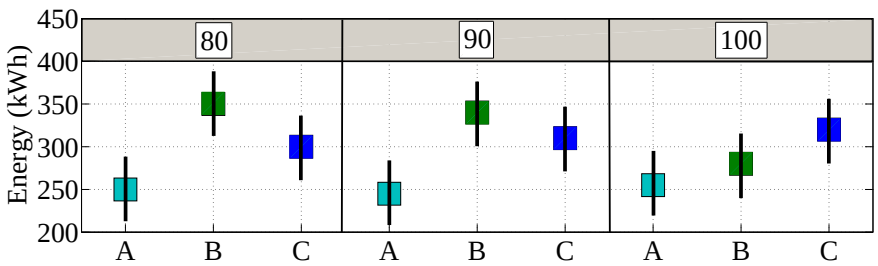


Figure 7.76: Energy consumption vs OL threshold

Moreover, the SLA violations of the proposed scheme is lower than 0.05 for 80 percent and 90 percent OL threshold as shown in Fig 7.77. But, it shows an increase in SLA violations when OL threshold is considered as 100 percent. So, it is evident that, OL threshold is not suitable at 100 percent.

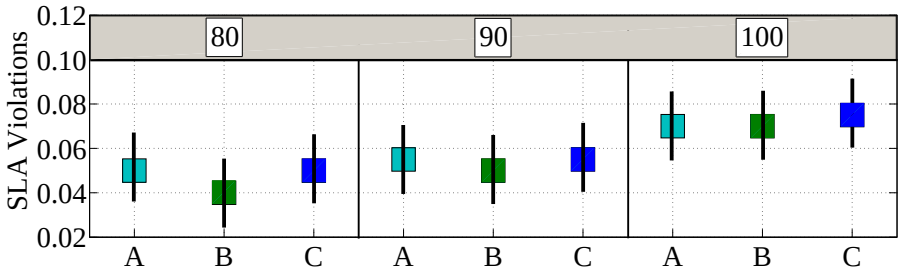


Figure 7.77: SLA violations vs OL threshold

Fig. 7.78 shows that by increasing the OL threshold, the migration rate for all three scheme decreases. This is because less number of host are identified as overloaded. Moreover, the decrease in container migration rate results in lower number of new container creation.

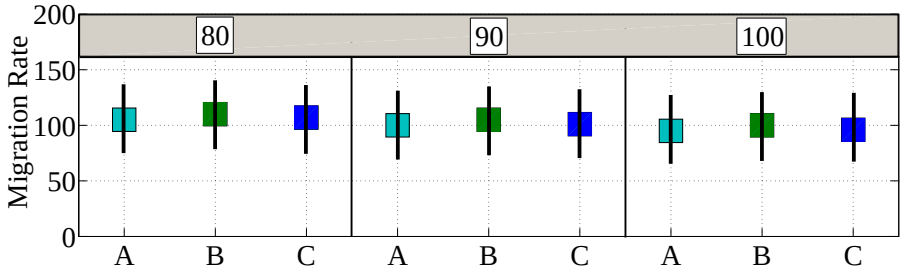


Figure 7.78: Container migration rate vs OL threshold

The above discussion shows that the proposed scheme consumes less energy and ensures viable SLAs. Moreover, 80 percent and 90 percent OL thresholds prove to be viable and 100 percent OL thresholds perform worst for all schemes.

- **Variation of UL:** In the analysis of UL variation, energy consumption of proposed scheme (A) is lower than other two schemes (B and C). Fig 7.79 shows that the energy consumption for proposed scheme for 40 percent and 60 percent UL threshold is lower than 50 percent OL threshold.

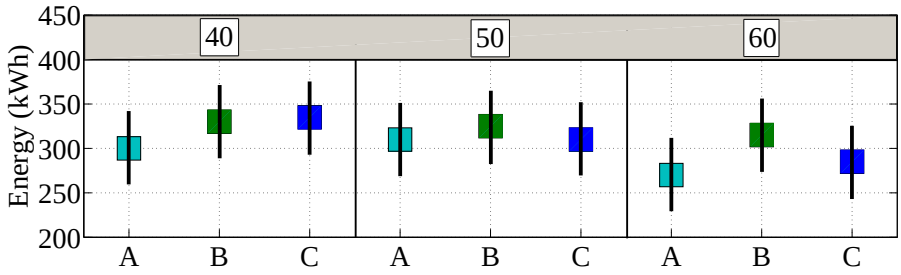


Figure 7.79: Energy consumption vs UL threshold

After energy consumption, the SLA violations witnessed for handling the incoming jobs are analyzed with respect to UL threshold variations. The results show that he SLA violations of the proposed scheme are below 0.05 for all UL threshold as evident from Fig 7.80.

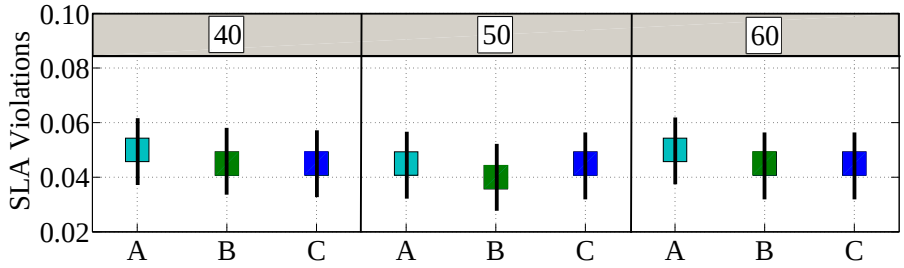


Figure 7.80: SLA violations vs UL threshold

Finally, Fig. 7.81 shows that by increasing the UL threshold, the migration rate for all three scheme increases. This is because more number of host are identified as underloaded. The above discussion shows that the proposed scheme consumes less energy and ensures SLAs.

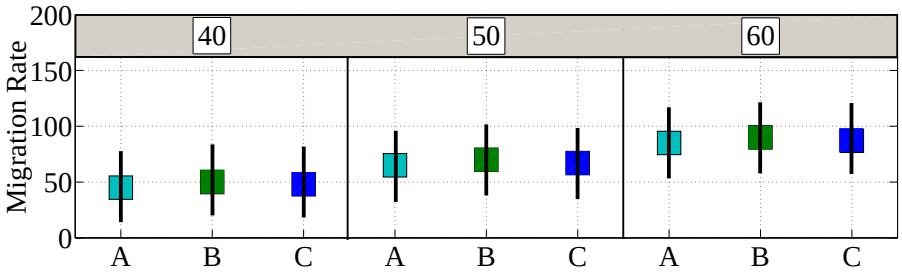


Figure 7.81: Container migration rate vs UL threshold

Summary of the Chapter: The Chapter 7 presents the evaluation results of all the four techniques designed for sustainable DCs. In all the four techniques DCs are powered by solar and wind energy. *Technique 1* is evaluated using interactive workload traces from Wikipedia. *Technique 2,3 and 4* are tested using a Google workload traces. All the four technique performs better than its other variants in terms of energy consumption, revenue generated, reward points gained, energy cost, energy savings, SLA violations, migration delay, migration cost, response time, container migration rate and overhead.

Chapter 8

Conclusion and Future Scope

The major purpose of this work was to sustain the energy consumption of DC using RES. However, due to intermittent nature of RES, it becomes a challenging task. For this reason, four techniques in different environments are designed.

In the first technique, an SDN-based energy management scheme for sustainability of cloud DC using RES has been proposed. To cope up with the intermittent nature of RES, the unique charging-discharging property of EVs has been utilized to support the energy consumption of DC. In the second technique, an SDN-based and renewable-aware efficient scheme for DC sustainability in edge-cloud environment has been designed. In this technique, SVM classifies the incoming jobs on the basis of priority and delay-sensitivity. The classified jobs were routed using an optimal flow path to different cDCs or nDCs having sufficient amount of renewable energy. For this purpose, a two stage workload scheduling scheme has been presented for sustainability of DCs using RES while ensuring lower SLA violations. In the third technique, a multi-leader multi-follower Stackelberg game has been designed for renewable energy-aware resource allocation. To effectuate the paramount intention of DC sustainability, an optimal utilization of cloud resources was achieved by aligning them with energy consumption of DCs. Finally, a *CoaaS* model has been deployed in a geo-distributed cloud environment. To achieve the quintessential objective of sustainability, a renewable energy-aware multi-indexed job classification and scheduling

approach has been designed. Using a global controller, the proposed scheme allocates the incoming workload to those DCs which have sufficient amount of renewable energy to handle the incoming jobs.

The proposed techniques were validated using realistic workload collected from Wikipedia and Google traces. For evaluation purpose, the realistic weather traces were considered for different locations in India. The evaluation results obtained with respect to different performance metrics prove the superiority of the proposed techniques in contrast to various existing stat-of-the-art schemes. For the first technique, the results show that EVs effectively support the energy consumption of DCs when the energy generation by RES was negligible. Moreover, the desired SLAs were preserved while sustaining the energy consumption of DCs using RES. The results obtained for the second technique depict that the edge-cloud environment plays an important role toward energy sustainability of its physical infrastructure of DCs using RES. Moreover, the server and OFswitch consolidation schemes help to reduce the energy deficit and save energy for the time periods when the energy generation by RES is low. The proposed technique demonstrate lower SLA violations, migration delay and operational cost. The cost incurred on energy reduces which in turn increases the overall profit of CSPs. Using the game-theoretic technique, the optimal utilization of resources help to reduce the number of servers provisioned, which in turn minimizes the energy consumption of DCs. For the designed game, the existence of Nash equilibrium proves that the objective of designing sustainable DCs using RES is achieved. Finally, the *CoaaS* model proves to be more energy efficient and quicker in contrast to other existing variants. The proposed container migration and consolidation scheme reduces the energy consumption of DCs along with guaranteeing minimum SLA violations and migration delay.

Therefore, after analysis of the results obtained for all the four techniques, it is evident that the objective of this thesis, i.e., sustaining energy consumption of DCs using RES is successfully achieved. The amalgamation of novel energy efficient techniques and RES minimizes the energy consumption of DCs which in turn end up

to be a cost effective solution for DC sustainability. Apart from the main objective, the proposed solution reduces the operational cost of DCs which in turn leads to increase in the revenue.

8.1 Future Scope

In future following works would be explored.

- Thermal storage could be beneficial for sustainable DCs as it is less susceptible to self discharge loss. Moreover, thermal storage can amalgamate with the heating and colling process of DCs to provide further reduction in energy consumption, carbon footprints and operational costs.
- Controller placement problem and virtual SDN networks can provide higher energy efficiency along with improved performance in edge-cloud environment.
- Security is one of the important aspects that need to be analyzed in the designed scenarios. Due to the involvement of edge devices and programmable SDN, this aspect needs more attention in the future.

Bibliography

[1] S. Bera, T. Ojha, S. Misra, and M. S. Obaidat, "Cloud-based optimal energy forecasting for enabling green smart grid communication," in *IEEE Global Communications Conference (GLOBECOM)*, 2015, pp. 1–6.

[2] A. Narayan and S. Rao, "Power-aware cloud metering," *IEEE Transactions on Services Computing*, vol. 7, no. 3, pp. 440–451, Jul 2014.

[3] K. Li, C. Liu, K. Li, and A. Zomaya, "A framework of price bidding configurations for resource usage in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, 2015, doi: 10.1109/TPDS.2015.2495120.

[4] T. Qiu, R. Qiao, M. Han, A. K. Sangaiah, and I. Lee, "A lifetime-enhanced data collecting scheme for the internet of things," *IEEE Communications Magazine*, vol. 55, no. 11, pp. 132–137, 2017.

[5] K. P. Sharma and T. P. Sharma, "rDFD: Reactive distributed fault detection in wireless sensor networks," *Wireless Networks*, vol. 23, no. 4, pp. 1145–1160, 2017.

[6] B. Daly. (2016, Mar.) After a decade of dominance, whats next for AWS? 451 Research. [Accessed on: Nov. 2016]. [Online]. Available: https://blogs.the451group.com/techdeals/infrastructure-software/after-a-decade-of-dominance-whats-next-for-aws/

[7] J. Shuja, K. Bilal, S. Madani, M. Othman, R. Ranjan, P. Balaji, and S. Khan, "Survey of techniques and architectures for designing energy-efficient data centers," *IEEE Systems Journal*, 2014, doi: 10.1109/JSYST.2014.2315823.

[8] M. Dayarathna, Y. Wen, and R. Fan, "Data center energy consumption modeling: A survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 732–794, 2016.

[9] K. Kaur, T. Dhand, N. Kumar, and S. Zeadally, "Container-as-a-service at the edge: Trade-off between energy efficiency and service availability at fog nano data centers," *IEEE Wireless Communications*, vol. 24, no. 3, pp. 48–56, 2017.

[10] Cisco global cloud index: Forecast and methodology, 2015-2020 white paper. Cisco PublicCISCOmanual. [Accessed on: March 2017]. [Online]. Available: http://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.pdf

[11] N. Kumar, S. Zeadally, N. Chilamkurti, and A. Vinel, "Performance analysis of bayesian coalition game-based energy-aware virtual machine migration in vehicular mobile cloud," *IEEE Network*, vol. 29, no. 2, pp. 62–69, Mar 2015.

[12] K. Wang, Y. Wang, X. Hu, Y. Sun, D.-J. Deng, A. Vinel, and Y. Zhang, "Wireless big data computing in smart grid," *IEEE Wireless Communications*, vol. 24, no. 2, pp. 58–64, 2017.

[13] L. Wu, B. Chen, K. R. Choo, and D. He, "Efficient and secure searchable encryption protocol for cloud-based internet of things," *Journal of Parallel Distributed Computing*, vol. 111, pp. 152–161, 2018.

[14] D. Evans, "The internet of things: How the next evolution of the internet is changing everything," Cisco Internet Business Solutions Group, Apr. 2011, [Accessed on: March 2017].

[15] J. Whitney and P. Delforge, "Data center efficiency assessment–scaling up energy efficiency across the data center industry: Evaluating key drivers and barriers," *Rep. IP NRDC and Anthesis*, pp. 14–08, 2014.

[16] *Green data centers*, accessed on: Feb 2016. [Online]. Available: http://www.greendatacenternews.org/articles/796756/india-the-next-green-data-center-hotspot-by-doug-m

[17] T. Mastelic and I. Brandic, "Recent trends in energy-efficient cloud computing," *IEEE Cloud Computing*, vol. 2, no. 1, pp. 40–47, Jan 2015.

[18] B. Walsh, *The Surprisingly Large Energy Footprint of the Digital Economy*. [Online]. Available: http://science.time.com/2013/08/14/power-drain-the-digital-cloud-is-using-more-energy-than-you-think/

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

[20] N. Scenario and M. East, "World energy outlook 2014 factsheet," *International Energy Agency Paris*, 2015.

[21] D. Paul, W. Zhong, and S. Bose, "Demand response in data centers through energy-efficient scheduling and simple incentivization," *IEEE Systems Journal*, 2015, doi: 10.1109/JSYST.2015.2476357.

[22] B. Xu, Y. Chen, J. R. Carrión, J. Loo, and A. Vinel, "Energy-aware power control in energy cooperation aided millimeter wave cellular networks with renewable energy resources," *IEEE Access*, vol. 5, pp. 432–442, 2017.

[23] A. K. Sangaiah, A. Abraham, P. Siarry, and M. Sheng, "Intelligent decision support systems for sustainable computing," in *Intelligent Decision Support Systems for Sustainable Computing*. Springer, 2017, pp. 1–6.

[24] G. e-Sustainability Initiative *et al.*, "Gesi smarter 2020: the role of ICT in driving a sustainable future," *Global e-Sustainability Initiative, Brussels, Belgium*, 2012.

[25] N. D. Han, Y. Chung, and M. Jo, "Green data centers for cloud-assisted mobile ad hoc networks in 5G," *IEEE Network*, vol. 29, no. 2, pp. 70–76, March 2015.

[26] C. G. C. Index, "Cisco global cloud index 2013-2018," *Cisco Systems Inc. San Jose*, 2016, [Accessed on: March 2017].

[27] S. Wang, X. Huang, Y. Liu, and R. Yu, "Cachinmobile: An energy-efficient users caching scheme for fog computing," in *IEEE/CIC International Conference on Communications in China (ICCC)*, Jul 2016, pp. 1–6.

[28] J. Shuja, A. Gani, S. Shamshirband, R. W. Ahmad, and K. Bilal, "Sustainable cloud data centers: a survey of enabling techniques and technologies," *Renewable and Sustainable Energy Reviews*, vol. 62, pp. 195–214, 2016.

[29] D.-J. Deng, S.-Y. Lien, C.-C. Lin, S.-C. Hung, and W.-B. Chen, "Latency control in software-defined mobile-edge vehicular networking," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 87–93, 2017.

[30] *OpenFlow*, [Accessed on: Nov 2016]. [Online]. Available: https://www.opennetworking.org/

[31] B. R. Al-Kaseem and H. S. Al-Raweshidy, "SD-NFV as an energy efficient approach for M2M networks using cloud-based 6LoWPAN testbed," *IEEE Internet of Things Journal*, 2017, doi: 10.1109/JIOT.2017.2704921.

[32] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 27–51, 2015.

[33] M. Erol-Kantarci and H. Mouftah, "Energy-efficient information and communication infrastructures in the smart grid: A survey on interactions and open

issues," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 179–197, 2015.

[34] *12 green data centers worth emulating*, Accessed on: Feb 2016. [Online]. Available: https://www.greenbiz.com/article/12-green-data-centers-worth-emulating-apple-verne

[35] S. Roy, A. Rudra, and A. Verma, "An energy complexity model for algorithms," in *4th conference on Innovations in Theoretical Computer Science*. ACM, 2013, pp. 283–304.

[36] R. Jain, D. Molnar, and Z. Ramzan, "Towards understanding algorithmic factors affecting energy consumption: switching complexity, randomness, and preliminary experiments," in *Joint workshop on Foundations of mobile computing*, 2005, pp. 70–79.

[37] B. M. Tudor and Y. M. Teo, "On understanding the energy consumption of arm-based multicore servers," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, no. 1. ACM, 2013, pp. 267–278.

[38] R. Ge, X. Feng, and K. W. Cameron, "Modeling and evaluating energy-performance efficiency of parallel processing on multicore based power aware systems," in *IEEE International Symposium on Parallel & Distributed Processing*, 2009, pp. 1–8.

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

[40] Z. Wang, N. Tolia, and C. Bash, "Opportunities and challenges to unify workload, power, and cooling management in data centers," in *Proceedings of the Fifth International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks*. ACM, 2010, pp. 1–6.

[41] H. Li, G. Casale, and T. Ellahi, "Sla-driven planning and optimization of enterprise applications," in *Proceedings of the first joint WOSP/SIPEW international conference on Performance engineering*. ACM, 2010, pp. 117–128.

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

[43] Y. Yao, L. Huang, A. Sharma, L. Golubchik, and M. Neely, "Data centers power reduction: A two time scale approach for delay tolerant workloads," in *Proceedings IEEE INFOCOM*. IEEE, 2012, pp. 1431–1439.

[44] Y. Tian, C. Lin, and K. Li, "Managing performance and power consumption tradeoff for multiple heterogeneous servers in cloud computing," *Cluster Computing*, vol. 17, no. 3, pp. 943–955, 2014.

[45] D. Shin, J. Kim, N. Chang, J. Choi, S. W. Chung, and E.-Y. Chung, "Energy-optimal dynamic thermal management for green computing," in *Proceedings of the International Conference on Computer-Aided Design*. ACM, 2009, pp. 652–657.

[46] R. Bertran, M. Gonzalez, X. Martorell, N. Navarro, and E. Ayguade, "A systematic methodology to generate decomposable and responsive power models for cmps," *IEEE Transactions on Computers*, vol. 62, no. 7, pp. 1289–1302, 2013.

[47] W. L. Bircher and L. K. John, "Complete system power estimation: A trickle-down approach based on performance events," in *IEEE International Symposium on Performance Analysis of Systems & Software, 2007.*, 2007, pp. 158–168.

[48] K. Li, "Optimal configuration of a multicore server processor for managing the power and performance tradeoff," *The Journal of Supercomputing*, vol. 61, no. 1, pp. 189–214, 2012.

[49] J. Lin, H. Zheng, Z. Zhu, E. Gorbatov, H. David, and Z. Zhang, "Software thermal management of dram memory for multicore systems," *ACM SIGMETRICS Performance Evaluation Review*, vol. 36, no. 1, pp. 337–348, 2008.

[50] J. H. Ahn, N. P. Jouppi, C. Kozyrakis, J. Leverich, and R. S. Schreiber, "Improving system energy efficiency with memory rank subsetting," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 9, no. 1, p. 4, 2012.

[51] K. T. Malladi, I. Shaeffer, L. Gopalakrishnan, D. Lo, B. C. Lee, and M. Horowitz, "Rethinking DRAM power modes for energy proportionality," in *45th Annual IEEE/ACM International Symposium on Microarchitecture*, 2012, pp. 131–142.

[52] J. Lin, H. Zheng, Z. Zhu, H. David, and Z. Zhang, *Thermal modeling and management of DRAM memory systems.* ACM, 2007, vol. 35, no. 2.

[53] A. W. Lewis, N.-F. Tzeng, and S. Ghosh, "Runtime energy consumption estimation for server workloads based on chaotic time-series approximation," *ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 9, no. 3, p. 15, 2012.

[54] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya, "Virtual machine power metering and provisioning," in *1st ACM symposium on Cloud computing*, 2010, pp. 39–50.

[55] Y. Zhang, S. Gurumurthi, and M. R. Stan, "SODA: Sensitivity based optimization of disk architecture," in *44th ACM/IEEE Design Automation Conference*, 2007, pp. 865–870.

[56] A. Hylick, R. Sohan, A. Rice, and B. Jones, "An analysis of hard drive energy consumption," in *IEEE International Symposium on Modeling, Analysis and Simulation of Computers and Telecommunication Systems*, 2008, pp. 1–10.

[57] J. Park, S. Yoo, S. Lee, and C. Park, "Power modeling of solid state disk for dynamic power management policy design in embedded systems," *Software Technologies for Embedded and Ubiquitous Systems*, pp. 24–35, 2009.

[58] V. Mohan, T. Bunker, L. Grupp, S. Gurumurthi, M. R. Stan, and S. Swanson, "Modeling power consumption of nand flash memories using flashpower," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 7, pp. 1031–1044, 2013.

[59] M. Poess and R. Othayoth Nambiar, "A power consumption analysis of decision support systems," in *First joint WOSP/SIPEW international conference on Performance engineering.* ACM, 2010, pp. 147–152.

[60] A. Gandhi, M. Harchol-Balter, and I. Adan, "Server farms with setup costs," *Performance Evaluation*, vol. 67, no. 11, pp. 1123–1138, 2010.

[61] I. Mitrani, "Trading power consumption against performance by reserving blocks of servers," in *Computer Performance Engineering.* Springer, 2012, pp. 1–15.

[62] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, "Cutting the electric bill for internet-scale systems," in *ACM SIGCOMM computer communication review*, vol. 39, no. 4, 2009, pp. 123–134.

[63] D. Chernicoff, *The shortcut guide to data center energy efficiency.* Realtimepublishers. com, 2009, [Accessed on: March 2017].

[64] M. K. Patterson, S. W. Poole, C.-H. Hsu, D. Maxwell, W. Tschudi, H. Coles, D. J. Martinez, and N. Bates, "TUE, a new energy-efficiency metric applied at ornls jaguar," in *International Supercomputing Conference*, 2013, pp. 372–382.

[65] M. S. Obaidat, A. Anpalagan, and I. Woungang, *Handbook of green information and communication systems*. Academic press, 2012.

[66] L. H. Sego, A. Márquez, A. Rawson, T. Cader, K. Fox, W. I. Gustafson Jr, and C. J. Mundy, "Implementing the data center energy productivity metric," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 8, no. 4, p. 30, 2012.

[67] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "Elastictree: Saving energy in data center networks." in *NSDI*, vol. 10, 2010, pp. 249–264.

[68] I. Widjaja, A. Walid, Y. Luo, Y. Xu, and H. J. Chao, "Small versus large: Switch sizing in topology design of energy-efficient data centers," in *IEEE/ACM 21st International Symposium on Quality of Service (IWQoS)*, 2013, pp. 1–6.

[69] D. Li, Y. Shang, and C. Chen, "Software defined green data center network with exclusive routing," in *IEEE INFOCOM*, 2014, pp. 1743–1751.

[70] A. Vishwanath Member, K. Hinton, R. Ayre, and R. Tucker, "Modeling energy consumption in high-capacity routers and switches," *Selected Areas in Communications, IEEE Journal on*, vol. 32, no. 8, pp. 1524–1532, 2014.

[71] M. Pedram, "Energy-efficient datacenters," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 10, pp. 1465–1484, 2012.

[72] H. Hlavacs, G. Da Costa, and J.-M. Pierson, "Energy consumption of residential and professional switches," in *IEEE International Conference on Computational Science and Engineering*, vol. 1, 2009, pp. 240–246.

[73] J. Ahn and H.-S. Park, "Measurement and modeling the power consumption of router interface," in *16th IEEE International Conference on Advanced Communication Technology (ICACT)*, 2014, pp. 860–863.

[74] R. Basmadjian, H. De Meer, R. Lent, and G. Giuliani, "Cloud computing and its interest in saving energy: the use case of a private cloud," *Journal of Cloud Computing: Advances, Systems and Applications*, vol. 1, no. 1, p. 5, 2012.

[75] C. Kachris and I. Tomkos, "Power consumption evaluation of hybrid WDM PON networks for data centers," in *16th IEEE European Conference on Networks and Optical Communications (NOC)*, 2011, pp. 118–121.

[76] ——, "Power consumption evaluation of all-optical data center networks," *Cluster Computing*, vol. 16, no. 3, pp. 611–623, 2013.

[77] W. Van Heddeghem, F. Idzikowski, W. Vereecken, D. Colle, M. Pickavet, and P. Demeester, "Power consumption modeling in optical multilayer networks," *Photonic Network Communications*, vol. 24, no. 2, pp. 86–102, 2012.

[78] D. Bouley and W. Torell, "Containerized power and cooling modules for data centers," *Schneider Electric White Paper*, 2012.

[79] M. E. M. Diouri, O. Glück, J.-C. Mignot, and L. Lefèvre, "Energy estimation for mpi broadcasting algorithms in large scale hpc systems," in *20th European MPI Users' Group Meeting*. ACM, 2013, pp. 111–116.

[80] T. Li and L. K. John, "Run-time modeling and estimation of operating system power consumption," *ACM SIGMETRICS Performance Evaluation Review*, vol. 31, no. 1, pp. 160–171, 2003.

[81] J. D. Davis, S. Rivoire, M. Goldszmidt, and E. K. Ardestani, "Chaos: Composable highly accurate os-based power models," in *IEEE International Symposium on Workload Characterization (IISWC)*, 2012, pp. 153–163.

[82] N. Kim, J. Cho, and E. Seo, "Energy-based accounting and scheduling of virtual machines in a cloud system," in *IEEE/ACM International Conference on Green Computing and Communications (GreenCom), 2011.* IEEE, 2011, pp. 176–181.

[83] H. Liu, H. Jin, C.-Z. Xu, and X. Liao, "Performance and energy modeling for live migration of virtual machines," *Cluster computing*, vol. 16, no. 2, pp. 249–264, 2013.

[84] X. Fang, D. Yang, and G. Xue, "Evolving smart grid information management cloudward: A cloud optimization perspective," *IEEE Transactions on Smart Grid*, vol. 4, no. 1, pp. 111–119, Mar 2013.

[85] L. Rao, X. Liu, L. Xie, and W. Liu, "Coordinated energy cost management of distributed internet data centers in smart grid," *IEEE Transactions on Smart Grid*, vol. 3, no. 1, pp. 50–58, Mar 2012.

[86] J. Li, Z. Bao, and Z. Li, "Modeling demand response capability by internet data centers processing batch computing jobs," *IEEE Transactions on Smart Grid*, vol. 6, no. 2, pp. 737–747, Mar 2015.

[87] M. Ghasemi-Gol, Y. Wang, and M. Pedram, "An optimization framework for data centers to minimize electric bill under day-ahead dynamic energy prices while providing regulation services," in *International Green Computing Conference (IGCC)*, Nov 2014, pp. 1–9.

[88] H. Wang, J. Huang, X. Lin, and H. Mohsenian-Rad, "Proactive demand response for data centers: A win-win solution," *IEEE Transactions on Smart Grid*, vol. 3, no. 3, pp. 1584–1596, 2015.

[89] N. Tran, D. Tran, S. Ren, Z. Han, E. Huh, and C. Hong, "How geo-distributed data centers do demand response: A game-theoretic approach," *IEEE Transactions on Smart Grid*, vol. 7, no. 2, pp. 937–947, 2015.

[90] X. Qiu, H. Li, C. Wu, Z. Li, and F. C. M. Lau, "Cost-minimizing dynamic migration of content distribution services into hybrid clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 12, pp. 3330–3345, Dec 2015.

[91] A. Forestiero, C. Mastroianni, M. Meo, G. Papuzzo, and M. Sheikhalishahi, "Hierarchical approach for efficient workload management in geo-distributed data centers," *IEEE Transactions on Green Communications and Networking*, vol. 1, no. 1, pp. 97–111, Mar 2017.

[92] P. Wang, L. Rao, X. Liu, and Y. Qi, "D-Pro: Dynamic data center operations with demand-responsive enerctricity prices in smart grid," *IEEE Transactions on Smart Grid*, vol. 3, no. 4, pp. 1743–1754, Dec 2012.

[93] C. Jin, J. Tang, and P. Ghosh, "Optimizing electric vehicle charging with energy storage in the electricity market," *IEEE Transactions on Smart Grid*, vol. 4, no. 1, pp. 311–320, Mar 2013.

[94] L. Rao, X. Liu, L. Xie, and Z. Pang, "Hedging against uncertainty: A tale of internet data center operations under smart grid environment," *IEEE Transactions on Smart Grid*, vol. 2, no. 3, pp. 555–563, Sep 2011.

[95] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment," *IEEE Network*, vol. 29, no. 2, pp. 56–61, Mar 2015.

[96] S. Wang, A. Zhou, C. H. Hsu, X. Xiao, and F. Yang, "Provision of data-intensive services through energy- and qos-aware virtual machine placement in national cloud data centers," *IEEE Transactions on Emerging Topics in Computing*, 2015, doi: 10.1109/TETC.2015.2508383.

[97] H. Goudarzi and M. Pedram, "Energy-efficient virtual machine replication and placement in a cloud computing system," in *5th IEEE International Conference on Cloud Computing (CLOUD)*, June 2012, pp. 750–757.

[98] M. Islam, S. Ren, H. Mahmud, and G. Quan, "Online energy budgeting for cost minimization in virtualized data center," *IEEE Transactions on Services Computing*, 2015, doi: 10.1109/TSC.2015.2390231.

[99] L. Gu, D. Zeng, S. Guo, and B. Ye, "Joint optimization of vm placement and request distribution for electricity cost cut in geo-distributed data centers," in *International Conference on Computing, Networking and Communications (ICNC)*, Feb 2015, pp. 717–721.

[100] A. Beloglazov and R. Buyya, "Energy efficient resource management in virtualized cloud data centers," in *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*, May 2010, pp. 826–831.

[101] Y. J. Chiang, Y. C. Ouyang, and C. H. . Hsu, "An efficient green control algorithm in cloud computing for cost optimization," *IEEE Transactions on Cloud Computing*, vol. 3, no. 2, pp. 145–155, Apr 2015.

[102] E. Feller, C. Rohr, D. Margery, and C. Morin, "Energy management in iaas clouds: a holistic approach," in *IEEE 5th International Conference on Cloud Computing*. IEEE, 2012, pp. 204–212.

[103] 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.

[104] Y. Wang and X. Wang, "Virtual batching: Request batching for server energy conservation in virtualized data centers," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 8, pp. 1695–1705, Aug 2013.

[105] 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.

[106] A. Khosravi, S. K. Garg, and R. Buyya, "Energy and carbon-efficient placement of virtual machines in distributed cloud data centers," in *European Conference on Parallel Processing*, 2013, pp. 317–328.

[107] R. Morabito, "Power consumption of virtualization technologies: an empirical investigation," in *IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*, 2015, pp. 522–527.

[108] K. Kaur, T. Dhand, N. Kumar, and S. Zeadally, "Container-as-a-service at the edge: Trade-off between energy efficiency and service availability at fog nano data centers," *IEEE Wireless Communications*, vol. 24, no. 3, pp. 48–56, 2017.

[109] S. F. Piraghaj, A. V. Dastjerdi, R. N. Calheiros, and R. Buyya, "Efficient virtual machine sizing for hosting containers as a service," in *IEEE World Congress on Services*, Jun 2015, pp. 31–38.

[110] X. Dai, Y. Wang, J. M. Wang, and B. Bensaou, "Energy-efficient planning of QoS-constrained virtual-cluster embedding in data centres," in *IEEE 4th International Conference on Cloud Networking*, Oct 2015, pp. 267–272.

[111] J. O. Gutierrez-Garcia and A. Ramirez-Nafarrate, "Collaborative agents for distributed load management in cloud data centers using live migration of virtual machines," *IEEE Transactions on Services Computing*, vol. 8, no. 6, pp. 916–929, Nov 2015.

[112] S. F. Piraghaj, A. V. Dastjerdi, R. N. Calheiros, and R. Buyya, "A framework and algorithm for energy efficient container consolidation in cloud data cen-

ters," in *IEEE International Conference on Data Science and Data Intensive Systems*, Dec 2015, pp. 368–375.

[113] ——, "Efficient virtual machine sizing for hosting containers as a service (services 2015)," in *IEEE World Congress on Services (SERVICES)*, 2015, pp. 31–38.

[114] ——, "Containercloudsim: An environment for modeling and simulation of containers in cloud data centers," *Software: Practice and Experience*, vol. 47, no. 4, pp. 505–521, 2017.

[115] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing towards balanced delay and power consumption," 2012.

[116] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, and R. S. Tucker, "Fog computing may help to save energy in cloud computing," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 5, pp. 1728–1739, May 2016.

[117] R. Kaewpuang, S. Chaisiri, D. Niyato, B. S. Lee, and P. Wang, "Cooperative virtual machine management in smart grid environment," *IEEE Transactions on Services Computing*, vol. 7, no. 4, pp. 545–560, Oct 2014.

[118] K. K. Nguyen, M. Cheriet, M. Lemay, M. Savoie, and B. Ho, "Powering a data center network via renewable energy: A green testbed," *IEEE Internet Computing*, vol. 17, no. 1, pp. 40–49, Jan 2013.

[119] L. Yu, T. Jiang, and Y. Cao, "Energy cost minimization for distributed internet data centers in smart microgrids considering power outages," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 1, pp. 120–130, Jan 2015.

[120] D. Paul, W.-D. Zhong, and S. K. Bose, "Energy efficiency aware load distribution and electricity cost volatility control for cloud service providers," *Journal of Network and Computer Applications*, vol. 59, pp. 185 – 197, 2016.

[121] S. Chen, S. Irving, and L. Peng, "Operational cost optimization for cloud computing data centers using renewable energy," *IEEE Systems Journal*, vol. 10, no. 4, pp. 1447–1458, Dec 2016.

[122] Y. Wang, X. Lin, and M. Pedram, "A stackelberg game-based optimization framework of the smart grid with distributed PV power generations and data centers," *IEEE Transactions on Energy Conversion*, vol. 29, no. 4, pp. 978–987, Dec 2014.

[123] L. Sharifi, F. Freitag, and L. Veiga, "Envisioning cloud of energy," in *IEEE International Conference on Smart Grid Communications (SmartGridComm)*, Nov 2015, pp. 229–234.

[124] K. K. Nguyen and M. Cheriet, "Environment-aware virtual slice provisioning in green cloud environment," *IEEE Transactions on Services Computing*, vol. 8, no. 3, pp. 507–519, May 2015.

[125] A. Khosravi, S. K. Garg, and R. Buyya, "Energy and carbon-efficient placement of virtual machines in distributed cloud data centers," in *European Conference on Parallel Processing*, 2013, pp. 317–328.

[126] A. Mondal, S. Misra, and M. S. Obaidat, "Distributed home energy management system with storage in smart grid using game theory," *IEEE Systems Journal*, vol. 11, no. 3, pp. 1857–1866, 2017.

[127] E. Terciyanli, T. Demirci, D. Kucuk, M. Sarac, I. Cadirci, and M. Ermis, "Enhanced nationwide wind-electric power monitoring and forecast system," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 2, pp. 1171–1184, May 2014.

[128] T. Chen, X. Wang, and G. B. Giannakis, "Cooling-aware energy and workload management in data centers via stochastic optimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 2, pp. 402–415, Mar 2016.

[129] H. Huang, S. Guo, J. Wu, and J. Li, "Green datapath for TCAM-based software-defined networks," *IEEE Communications Magazine*, vol. 54, no. 11, pp. 194–201, Nov 2016.

[130] F. Milano and O. Hersent, "Optimal load management with inclusion of electric vehicles and distributed energy resources," *IEEE Transactions on Smart Grid*, vol. 5, no. 2, pp. 662–672, Mar 2014.

[131] K. Kaur, A. Dua, A. Jindal, N. Kumar, M. Singh, and A. Vinel, "A novel resource reservation scheme for mobile phevs in V2G environment using game theoretical approach," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 12, pp. 5653–5666, Dec 2015.

[132] R. Rana, K. Kaur, M. Singh, N. Kumar, and P. Kumar, "Integrated fleet of electric vehicles as a frequency regulation agent in the grid," in *IEEE International Transportation Electrification Conference (ITEC)*, Aug 2015, pp. 1–7.

[133] J. Druitt and W.-G. Früh, "Simulation of demand management and grid balancing with electric vehicles," *Journal of Power Sources*, vol. 216, pp. 104–116, 2012.

[134] M. Caramanis and J. M. Foster, "Management of electric vehicle charging to mitigate renewable generation intermittency and distribution network congestion," in *48th IEEE Conference on Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference.*, Dec 2009, pp. 4717–4722.

[135] R. Freire, J. Delgado, J. Santos, and A. Almeida, "Integration of renewable energy generation with ev charging strategies to optimize grid load balancing,"

in *IEEE Annual Conference on Intelligent Transportation Systems*, 2010, pp. 392–396.

[136] A. Y. Saber and G. K. Venayagamoorthy, "Plug-in vehicles and renewable energy sources for cost and emission reductions," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 4, pp. 1229–1238, Apr 2011.

[137] Y. Guo, Y. Gong, Y. Fang, P. P. Khargonekar, and X. Geng, "Energy and network aware workload management for sustainable data centers with thermal storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2030–2042, Aug 2014.

[138] T. C. Patil and S. P. Duttagupta, "Hybrid self - sustainable green power generation system for powering green data center," *IEEE International Conference on Control, Instrumentation, Energy and Communication (CIEC)*, pp. 331–334, Jan 2014.

[139] T. Chen, X. Wang, and G. B. Giannakis, "Energy and workload management for data centers in renewable-integrated power grid," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Dec 2015, pp. 513–517.

[140] T. Chen, Y. Zhang, X. Wang, and G. B. Giannakis, "Robust workload and energy management for sustainable data centers," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 651–664, Mar 2016.

[141] M. Polverini, A. Cianfrani, S. Ren, and A. V. Vasilakos, "Thermal-aware scheduling of batch jobs in geographically distributed data centers," *IEEE Transactions on Cloud Computing*, vol. 2, no. 1, pp. 71–84, Jan 2014.

[142] C. Gu, Z. Li, C. Liu, and H. Huang, "Planning for green cloud data centers using sustainable energy," in *IEEE Symposium on Computers and Communication (ISCC)*, 2016, pp. 804–809.

[143] G. Callou, P. Maciel, E. Tavares, E. Sousa, B. Silva, J. Figueiredo, C. Araujo, F. Magnani, and F. Neves, "Sustainability and dependability evaluation on data center architectures," in *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2011, pp. 398–403.

[144] J. He, X. Deng, D. Wu, Y. Wen, and D. Wu, "Socially-responsible load scheduling algorithms for sustainable data centers over smart grid," in *IEEE Third International Conference on Smart Grid Communications (SmartGridComm)*, 2012, pp. 406–411.

[145] T. Chen, Y. Zhang, X. Wang, and G. B. Giannakis, "Robust geographical load balancing for sustainable data centers," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 3526–3530.

[146] Z. Abbasi and S. K. Gupta, "Holistic management of sustainable geo-distributed data centers," in *IEEE 22nd International Conference on High Performance Computing (HiPC)*, 2015, pp. 426–435.

[147] L. J. Klein, S. Bermudez, H.-D. Wehle, S. Barabasi, and H. F. Hamann, "Sustainable data centers powered by renewable energy," in *28th Annual IEEE Semiconductor Thermal Measurement and Management Symposium (SEMI-THERM)*. IEEE, 2012, pp. 362–367.

[148] A. N. Toosi, C. Qu, M. D. de Assunção, and R. Buyya, "Renewable-aware geographical load balancing of web applications for sustainable data centers," *Journal of Network and Computer Applications*, vol. 83, pp. 155–168, 2017.

[149] B. Wang, Z. Qi, R. Ma, H. Guan, and A. V. Vasilakos, "A survey on data center networking for cloud computing," *Computer Networks*, vol. 91, pp. 528–547, 2015.

[150] C. Kachris, K. Kanonakis, and I. Tomkos, "Optical interconnection networks in data centers: recent trends and future challenges," *IEEE Communications Magazine*, vol. 51, no. 9, pp. 39–45, 2013.

[151] J. Son and R. Buyya, "A taxonomy of SDN-enabled cloud computing," *ACM computing surveys*, vol. 1, no. 1, pp. 1–31, Mar 2017.

[152] R. Tu, X. Wang, and Y. Yang, "Energy-saving model for SDN data centers," *The Journal of Supercomputing*, vol. 70, no. 3, pp. 1477–1495, 2014.

[153] G. Xu, B. Dai, B. Huang, and J. Yang, "Bandwidth-aware energy efficient routing with SDN in data center networks," in *IEEE 17th International Conference on High Performance Computing and Communications*, Aug 2015, pp. 766–771.

[154] G. Xu, B. Dai, B. Huang, J. Yang, and S. Wen, "Bandwidth-aware energy efficient flow scheduling with SDN in data center networks," *Future Generation Computer Systems*, vol. 68, pp. 163–174, 2017.

[155] S.-H. Wang, P. P.-W. Huang, C. H.-P. Wen, and L.-C. Wang, "EQVMP: Energy-efficient and QoS-aware virtual machine placement for software defined datacenter networks," in *IEEE International Conference on Information Networking (ICOIN)*, 2014, pp. 220–225.

[156] J. Son, A. V. Dastjerdi, R. Calheiros, and R. Buyya, "SLA-aware and energy-efficient dynamic overbooking in SDN-based cloud data centers," *IEEE Transactions on Sustainable Computing*, 2017.

[157] P. Habibi, M. Mokhtari, and M. Sabaei, "QRVE: QoS-aware routing and energy-efficient VM placement for software-defined datacenter networks," in *8th IEEE International Symposium on Telecommunications (IST)*, 2016, pp. 533–539.

[158] P. Borylo, A. Lason, J. Rzasa, A. Szymanski, and A. Jajszczyk, "Anycast routing for carbon footprint reduction in WDM hybrid power networks with data centers," in *IEEE International Conference on Communications (ICC)*, 2014, pp. 3714–3720.

[159] ——, "Energy-aware fog and cloud interplay supported by wide area software defined networking," in *IEEE International Conference on Communications (ICC)*, May 2016, pp. 1–7.

[160] R. Tu, X. Wang, J. Zhao, Y. Yang, L. Shi, and T. Wolf, "Design of a load-balancing middlebox based on SDN for data centers," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Apr 2015, pp. 480–485.

[161] A. Sadasivarao, S. Syed, P. Pan, C. Liou, I. Monga, C. Guok, and A. Lake, "Bursting data between data centers: Case for transport SDN," in *IEEE 21st Annual Symposium on High-Performance Interconnects (HOTI)*, 2013, pp. 87–90.

[162] D. Tuncer, M. Charalambides, S. Clayman, and G. Pavlou, "Adaptive resource management and control in software defined networks," *IEEE Transactions on Network and Service Management*, vol. 12, no. 1, pp. 18–33, Mar 2015.

[163] M. Wang, M. Ismail, X. Shen, E. Serpedin, and K. Qaraqe, "Spatial and temporal online charging/discharging coordination for mobile PEVs," *IEEE Wireless Communications*, vol. 22, no. 1, pp. 112–121, Feb 2015.

[164] 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.

[165] *http://mnre.gov.in/sec/solar-assmnt.htm*, accessed on: Jan 2016.

[166] *http://www.windfinder.com/contact/weatherdata.htm*, accessed on: Jan 2016.

[167] J. Wilkes, "More Google cluster data," Google research blog, Nov. 2011, posted at http://googleresearch.blogspot.com/2011/11/more-google-cluster-data.html.

[168] D. Kruger, C. Buschmann, and S. Fischer, "Solar powered sensor network design and experimentation," in *6th International Symposium on Wireless Communication Systems*, Sep 2009, pp. 11–15.

[169] R. P. Mukund, "Wind and solar power systems," *CRC press*, 1999.

[170] K. Kaur, A. Dua, A. Jindal, N. Kumar, M. Singh, and A. Vinel, "A novel resource reservation scheme for mobile phevs in v2g environment using game theoretical approach," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 12, pp. 5653–5666, Dec 2015.

[171] R. Pecen and A. Nayir, "Design and implementation of a 12 kW wind-solar distributed power and instrumentation system as an educational testbed for electrical engineering technology students," in *Proceedings of the International Symposium Modern Electric Power Systems*, 2010, pp. 1–6.

[172] K. J. Kerpez, J. M. Cioffi, G. Ginis, M. Goldburg, S. Galli, and P. Silverman, "Software-defined access networks," *IEEE Communications Magazine*, vol. 52, no. 9, pp. 152–159, Sep 2014.

[173] *PlugShare-Electric Vehicle Charging Network.*, [Accessed on: Nov 2016]. [Online]. Available: http://www.plugshare.com

[174] H. Yang, X. Xie, and A. V. Vasilakos, "Noncooperative and cooperative optimization of electric vehicle charging under demand uncertainty: A robust stackelberg game," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 3, pp. 1043–1058, Mar 2016.

[175] H. Zhang, M. Bennis, L. A. DaSilva, and Z. Han, "Multi-leader multi-follower stackelberg game among wi-fi, small cell and macrocell networks," in *IEEE Global Communications Conference*, Dec 2014, pp. 4520–4524.

[176] C.-C. Lin, D.-J. Deng, W.-Y. Liu, and L. Chen, "Peak load shifting in the internet of energy with energy trading among end-users," *IEEE Access*, vol. 5, pp. 1967–1976, 2017.

[177] X. Liang, X. Li, R. Lu, X. Lin, and X. Shen, "UDP: Usage-based dynamic pricing with privacy preservation for smart grid," *IEEE Transactions on Smart Grid*, vol. 4, no. 1, pp. 141–150, Mar 2013.

[178] S. Garg and S. Batra, "Fuzzified cuckoo based clustering technique for network anomaly detection," *Computers & Electrical Engineering*, 2017, doi: 10.1016/j.compeleceng.2017.07.008.

[179] M. Gupta, S. S. Bharti, and S. Agarwal, "Support vector machine based gender identification using voiced speech frames," in *Fourth IEEE International Conference on Parallel, Distributed and Grid Computing (PDGC)*, 2016, pp. 737–741.

[180] S. Karamizadeh, S. M. Abdullah, M. Halimi, J. Shayan, and M. j. Rajabi, "Advantage and drawback of support vector machine functionality," in *International Conference on Computer, Communications, and Control Technology*, Sep 2014, pp. 63–65.

[181] A. Jindal, A. Dua, K. Kaur, M. Singh, N. Kumar, and S. Mishra, "Decision tree and SVM-based data analytics for theft detection in smart grid," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 3, pp. 1005–1016, Jun 2016.

[182] A. Jindal, N. Kumar, and M. Singh, "A data analytical approach using support vector machine for demand response management in smart grid," in *IEEE Power and Energy Society General Meeting (PESGM)*, Jul 2016, pp. 1–5.

[183] S. Garg and S. Batra, "A novel ensembled technique for anomaly detection," *International Journal of Communication Systems*, vol. 30, no. 11, 2017, dOI: 10.1002/dac.3248. [Online]. Available: http://dx.doi.org/10.1002/dac.3248

[184] D. Kruger, C. Buschmann, and S. Fischer, "Solar powered sensor network design and experimentation," in *6th International Symposium on Wireless Communication Systems*, Sep 2009, pp. 11–15.

[185] M. Ozkan and P. Karagoz, "A novel wind power forecast model: Statistical hybrid wind power forecast technique (SHWIP)," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 2, pp. 375–387, Apr 2015.

[186] 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, Sep 2012.

[187] P. Y. Nie and P. A. Zhang, "A note on stackelberg games," in *Chinese Control and Decision Conference*, Jul 2008, pp. 1201–1203.

[188] J. Lee, J. Guo, J. K. Choi, and M. Zukerman, "Distributed energy trading in microgrids: A game-theoretic model and its equilibrium analysis," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 6, pp. 3524–3533, June 2015.

[189] A. Mondal and S. Misra, "Game-theoretic energy trading network topology control for electric vehicles in mobile smart grid," *IET Networks*, vol. 4, no. 4, pp. 220–228, 2015.

[190] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google cluster-usage traces: format + schema," Google Inc., Mountain View, CA, USA, Technical Report, Nov. 2011, revised 2012.03.20. Posted at http://code.google.com/p/googleclusterdata/wiki/TraceVersion2.

[191] "Electricity rates", [Accessed: Jan 2016]. [Online]. Available: http://www.bijlibachao.com/news/domestic-electricity-it-tariff-slabs-and-rates-for-all-states-in-india-in-2015.html

LIST OF PUBLICATIONS

Journal Publications (SCI/SCIE):

1. G. S. Aujla and N. Kumar, "SDN-based energy management scheme for sustainability of data centers: An analysis on renewable energy sources and electric vehicles participation", Journal of Parallel and Distributed Computing, Vol. 117, July 2018, pp. 228-245. - **IF: 1.930**

2. Gagangeet Singh Aujla and Neeraj Kumar, "MEnSuS: An Efficient Scheme for Energy Management with Sustainability of Cloud Data Centers in Edge-Cloud Environment," *Future Generation Computer Systems*. 2017, doi:10.1016/j.futu re.2017.09. 066. - **IF: 3.993**

3. Gagangeet Singh Aujla, Neeraj Kumar, Albert Y. Zomaya, Rajiv Ranjan, "Optimal Decision Making for Big Data Processing at Edge-Cloud Environment: An SDN Perspective," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 2, pp. 778-789, 2018. - **IF: 6.764**

4. Gagangeet Singh Aujla, Neeraj Kumar, Sahil Garg, Kuljeet Kaur, Rajiv Ranjan and Sourabh Kumar Garg, "Renewable Energy-based Multi-Indexed Job Classification and Container Management Scheme for Sustainability of Cloud Data Centers," IEEE Transactions on Industrial Informatics, 2018, doi: 10.1109 /TII.2018.2800693. - **IF: 6.764**

5. Gagangeet Singh Aujla, Mukesh Singh, Neeraj Kumar and Albert Zomaya. "Stackelberg game for energy-aware resource allocation to sustain data centers using RES." *IEEE Transactions on Cloud Computing*, 2017, doi:10.1109@TCC .2017.2715817.

6. G. S. Aujla, N. Kumar, M. Singh and A. Y. Zomaya, "Energy Trading with Dynamic Pricing for Electric Vehicles in a Smart City Environment", Journal of Parallel and Distributed Computing, 2018. - **IF: 1.930**

Magazine Publications (SCI/SCIE):

1. Gagangeet Singh Aujla, Rajat Chaudhary, Neeraj Kumar, Joel J. P. C. Rodrigues, and Alexey Vinel, "Data Offloading in 5G-Enabled Software-Defined Vehicular Networks: A Stackelberg-Game-Based Approach," *IEEE Communication Magazine*, vol. 55, no. 8, pp. 100-108, Aug. 2018.- **IF: 10.235**

2. Gagangeet Singh Aujla, Rajat Chaudhary, Neeraj Kumar, Ashok Kumar Das and Joel J. P. C. Rodrigues, "SecSVA: Secure Storage, Verification, and Auditing of Big Data in the Cloud Environment," *IEEE Communication Magazine*, vol. 56, no. 1, pp. 78-85, Jan. 2018.- **IF: 10.235**

3. Rajat Chaudhary, Gagangeet Singh Aujla, Neeraj Kumar, Ashok Kumar Das and Joel J. P. C. Rodrigues, "Optimized Big Data Management across Multi-Cloud Data Centers: Software Defined Networks based Analysis," *IEEE Communication Magazine*, vol. 56, no. 2, Feb. 2018.- **IF: 10.235**

Conference Publications (outside India):

1. Gagangeet Singh Aujla, Anish Jindal, Neeraj Kumar and Mukesh Singh. "SDN-based data center energy management system using RES and electric vehicles." In Proc. of *IEEE Globecom*, 4-8 December 2016, Washington, USA.

2. Gagangeet Singh Aujla, Rajat Chaudhary, Neeraj Kumar, Ravinder Kumar and Joel J. P. C. Rodrigues. "An Ensembled Scheme for QoS-aware Traffic Flow Management in Software Defined Networks." In Proc. of *IEEE ICC*, 20-24 May 2018, Kansas City, USA.

3. Anish Jindal, Gagangeet Singh Aujla, Neeraj Kumar, and Sudip Misra. "Sustainable Smart Energy Cyber-Physical System: Can Electric Vehicles Suffice Its Needs?" In Proc. of *IEEE ICC*, 20-24 May 2018, Kansas City, USA.