2020

SHARING WITH LIVE MIGRATION ENERGY OPTIMIZATION TASK SCHEDULER FOR CLOUD COMPUTING DATACENTRES

Alshathri, Samah

UNIVERSITY OF
PLYMOUTH

SHARING WITH LIVE MIGRATION ENERGY OPTIMIZATION TASK SCHEDULER FOR CLOUD COMPUTING DATACENTRES

By

SAMAH ALSHATHRI

A thesis submitted to the University of Plymouth
In partial fulfilment for the degree of

DOCTOR OF PHILOSOPHY

School of Engineering, Computing and Mathematics

April 2020

I would like to dedicate this thesis to my amazing mother and kids who went with me through every step of my great scientific journey.

Copyright Statement

Abstract

Sharing with Live Migration Energy Optimization Scheduler for Cloud Computing Datacentres

Samah Alshathri

The use of cloud computing is expanding, and it is becoming the driver for innovation in all companies to serve their customers around the world. A big attention was drawn to the huge energy that was consumed within those datacentres recently neglecting the energy consumption in the rest of the cloud components. Therefore, the energy consumption should be reduced to minimize performance losses, achieve the target battery lifetime, satisfy performance requirements, minimize power consumption, minimize the CO2 emissions, maximize the profit, and maximize resource utilization.

Reducing power consumption in the cloud computing datacentres can be achieved by many ways such as managing or utilizing the resources, controlling redundancy, relocating datacentres, improvement of applications or dynamic voltage and frequency scaling. One of the most efficient ways to reduce power is to use a scheduling technique that will find the best task execution order based on the users demands and with the minimum execution time and cloud resources. It is quite a challenge in cloud environment to design an effective and an efficient task scheduling technique which is done based on the user requirements.

The scheduling process is not an easy task because within the datacentre there is dissimilar hardware with different capacities and, to improve the resource utilization, an efficient scheduling algorithm must be applied on the incoming tasks to achieve efficient computing resource allocating and power optimization. The scheduler must maintain the balance between the Quality of Service and fairness among the jobs so that the efficiency may be increased.

The aim of this project is to propose a novel method for optimizing energy usage in cloud computing environments that satisfy the Quality of Service (QoS) and the regulations of the Service Level Agreement (SLA). Applying a power- and resource-optimised scheduling algorithm will assist to control and improve the process of mapping between the datacentre servers and the incoming tasks and achieve the optimal deployment of the data centre resources to achieve good

computing efficiency, network load minimization and reducing the energy consumption in the datacentre.

This thesis explores cloud computing energy aware datacentre structures with diverse scheduling heuristics and propose a novel job scheduling technique with sharing and live migration based on file locality (SLM) aiming to maximize efficiency and save power consumed in the datacentre due to bandwidth usage utilization, minimizing the processing time and the system total make span. The propose SLM energy efficient scheduling strategy have four basic algorithms: 1) Job Classifier, 2) SLM job scheduler, 3) Dual fold VM virtualization and 4) VM threshold margins and consolidation. The SLM job classifier worked on categorising the incoming set of user requests to the datacentre in to two different queues based on these requests type and the source file needed to process them. The processing time of each job fluctuate based on the job type and the number of instructions for each job. The second algorithm, which is the SLM scheduler algorithm, dispatch jobs from both queues according to job arrival time and control the allocation process to the most appropriate and available VM based on job similarity according to a predefined synchronized job characteristic table (SJC). The SLM scheduler uses a replicated host's infrastructure to save the wasted idle hosts energy by maximizing the basic host's utilization as long as the system can deal with workflow while setting replicated hosts on off mode. The third SLM algorithm, the dual fold VM algorithm, divide the active VMs in to a top and low level slots to allocate similar jobs concurrently which maximize the host utilization at high workload and reduce the total make span. The VM threshold margins and consolidation algorithm set an upper and lower threshold margin as a trigger for VMs consolidation and load balancing process among running VMs, and deploy a continuous provisioning of overload and underutilize VMs detection scheme to maintain and control the system workload balance. The consolidation and load balancing is achieved by performing a series of dynamic live migrations which provides auto-scaling for the servers with in the datacentres.

This thesis begins with cloud computing overview then preview the conceptual cloud resources management strategies with classification of scheduling heuristics. Following this, a Competitive analysis of energy efficient scheduling algorithms and related work is presented. The novel SLM algorithm is proposed and evaluated using the CloudSim toolkit under number of scenarios, then the result compared to Particle Swarm Optimization algorithm (PSO) and Ant Colony Algorithm (ACO) shows a significant improvement in the energy usage readings

levels and total make span time which is the total time needed to finish processing all the tasks.

Author's Declaration

'At no time during the registration for the degree of Doctor of Philosophy has the author been registered for any other University award without prior agreement of the Doctoral College Quality Sub-Committee.

Work submitted for this research degree at the University of Plymouth has not formed part of any other degree either at the University of Plymouth or at another establishment. '

Relevant seminars and conferences were regularly attended at which work was often presented and several papers were published in the course of this research project.

Publications:

- Alshathri, S. (2016). Towards an Energy Optimization Framework for Cloud Computing Data Centres. Eleventh International Network Conference INC 2016 Proceedings, Frankfurt, Germany, 2016, pp.9-12.
  Cscan:
  https://www.researchgate.net/publication/330054162_Towards_Green_Cloud_Computing_an_Algorithmic_Approach_for_Energy_Minimization_in_Cloud_Data_Centers

- Alshathri, S. (2018). Contemporary Perception Of Task Scheduling Techniques In Cloud: A Review, 2nd European Conference on Electrical Engineering & Computer Science EECS 2018, Bern, Switzerland, pp. 201-205.
  DOI: https://doi.org/10.1109/EECS.2018.00045
  IEEE Xplore: https://ieeexplore.ieee.org/document/8910077/
- Alshathri, S.; Ghita, B.; Clarke, N. (2018). Sharing with Live Migration Energy Optimization Scheduler for Cloud Computing Data Centres. Future Internet 2018, Vol.10, pp.86.
  DIO: https://doi.org/10.3390/fi10090086
  MDPI: https://www.mdpi.com/1999-5903/10/9/86/pdf
- Alshathri, S. (2019). CloudSim and Comparative Study on Cloud Computing Simulation Platforms. Under review with the IEEE Internet Computing 2019.

Word count of main body of thesis: 42722 words.

Signed: 

Data: 4/10/2020

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Acknowledgement

PhD is once-in-a-lifetime opportunity and experience. It may feel like an eternity, but it passes by like a second and it teaches you a lot, and I am truly happy that I have had a chance to complete it. It would not have happened without all those people who helped me along the way. Firstly, I would like to express my sincere gratitude to my advisor Dr. Bogdan Ghita for the nonstop support of my PhD study and related research, for his patience, motivation, immense knowledge and giving me the access to the laboratory and research facilities. His guidance helped me in all the time of research and writing of this thesis. Also, I would like to express my gratitude to my other two PhD supervisors, Dr. Nathan Clarke and Dr. Samia All. Chelloug, for their insightful comments, answers and encouragement.

Last but not the least, I would like to thank my mother and my children for companion me though my visits and supporting me spiritually throughout writing this thesis and my life in general.

CHAPTER 1
INTRODUCTION

Chapter 1. Cloud Introduction

1.1 Introduction

With the fast emergence of the cloud concept, the interest of understanding this paradigm raised as an excessive need because it almost involves practically everything around us. The cloud computing terminology is not new but what's new about it is the evolution and progression of cloud computing approaches and policies. Cloud computing is one of the leading topics of the Information Technology sector and it is still the next phase in the evolution of computing where it was classified as one of the main five leading technologies used to improve the economics of IT investments [1].

Cloud Computing Technology (CCT) contribute profoundly in the operational efficiency of the IT platform through constructing the infrastructure and software solutions for the entire IT requirements via Internet. Recently, all shapes and sizes of companies begin to adapt to cloud computing to have outsourcing computations on-demand, achieve higher user's productivity and reach their business objectives and customer satisfaction with the least cost.

The cloud concept has emerged as a powerful mechanism for data storage by providing a suitable platform of datacentres and offered huge services to the Internet users and allowed organizations to focus on their work not their IT infrastructure. So, the decision of which deployment model of cloud computing should be embraced will be steered by the need of the sector and the comprehensive analysis of the business outcomes.

1.2 Cloud Integration and Adoption Challenges

The IDC Forecasts survey predicts that spending on public cloud will top $500 billion by 2020 [3]. The cloud deployment in our life is tremendously expanding

and that is becoming so obvious to be notes in the recent years where each cloud service provider serve millions of users on regular base around the globe, by renting an on demand resources on one physical cloud infrastructure such as Rack space [4] and Amazon's EC2 [5]. According to Cisco global cloud index, the number of existing datacentres in 2016 was 338 and it is expected to grow to 628 datacentre by the year of 2021[6], which raise the need for more researches to address the datacentres problems that associate with that fast and wide expansion of these datacentres such as the energy consumption, resource utilization, make span, cost. Each cloud service provider (CSP) faces several challenges while providing its services to the end users. On one hand, managing and mapping efficiently all the submitted tasks to the huge number of computing resources. On the other hand, scheduling tasks must be performed in a way that meets the quality of service (QoS) requirements and optimize the overall make span and cost. Above that, the large-scale data centres Infrastructure run thousands of servers and network devices to process the incoming tasks which consume a huge amount of energy.

Seeking for larger margins of revenues and minimizing the investment cost, CSP needs first to compact as many as Virtual machines in one physical server to effectively utilize the existing resources then use an effective and efficient task scheduler to reach that goal and maximize its profits.

The cloud datacentres (CDC) infrastructure in the United States of America consumed about 70 billion kilowatt per hour in 2016, which is sufficient for Washington State electricity consumption. By 2020, the data centre electricity consumption will increase to 140 billion kilowatt per hour with approximately 150 million metric tons of carbon emissions. Financially, between 30% and 50% of the CDC operational expenses will be spent on electricity bills to reach 13 billion dollars by 2020 [7], this will form one of the new age technology catastrophes if

the CSP did not apply innovative energy policies to improve energy consumption. Along with that comes the escalating rate of carbon emission footprint of the cloud datacentres, which was assessed by 50 million tons annually [8].

1.3 Motivation and Problem Statement

The hardware proficiency is not the only factor that effects the energy consumption within the datacentres, the used infrastructure resource management system have a powerful and major effect on the overall performance. Many of the cloud service providers concerns such as host consolidation, resources scalability, power saving polices, optimal energy efficient task scheduling etc. are challenging and are not exposed well by researchers to show their importance and the role they play to shape an efficient cloud environment. Energy consumption level is the cloud environment core concern, due to the annual growing size of data centres that cover miles of landscape with huge number of servers and resources [9]. Energy optimization task scheduling is one of the cloud challenges that needs extreme attention with much more excessive studying for the sake of achieving the anticipated energy efficient cloud environment. The following problems are explored:

- **Low Resource Utilization**

  Executing the maximum number of tasks using the minimum number of resources is one of the main factors to optimize the energy consumption. However, maximizing resource utilization or using its full capacity will not lead to minimize the energy consumption according to the SPEC power benchmark for server's utilization against performance. As shown in Table 1 and Figure 1, maximizing the utilization to 100 % still consume high energy and did not affect the rate of the growing average power number,

which need to set a resource provisioning plan to identify and assign resources to each user to meet the user requirement and with the use of the least possibly number of resources and executing task consolidation between virtual machines when reaching a specific CPU threshold peak [8].

Avrage Power (W)	24	37	55	63	79	84	89	105	109	115
Server Utlization %	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Performance to Power Ration (ssj_ops/W)	309	557	764	919	1041	1161	1251	1380	1494	1601

Table 1. Different server load power consumption

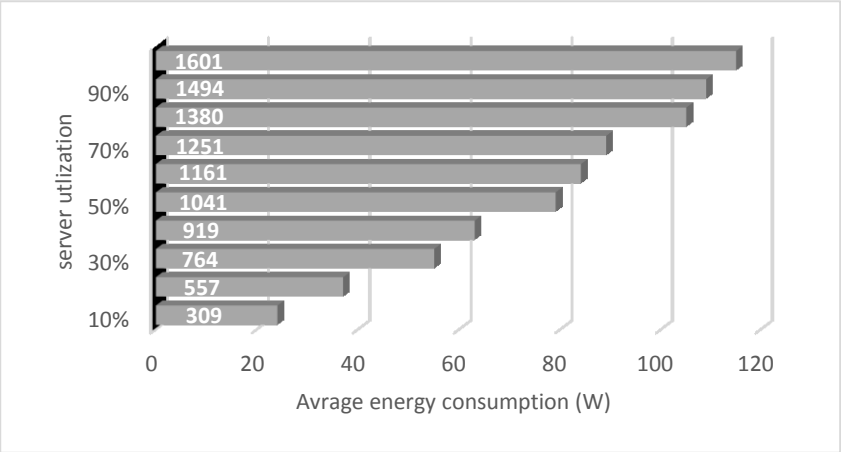


Figure1 . Server utilization VS energy consumption

- **Applying Sequential Job Processing**

Executing jobs sequentially without prioritizing increase the job waiting time and implies an extra overhead to the overall make span, which leads to low performance and inefficient datacentre with high energy consumption.

- **Uniform Load Distribution**

Usually maintaining a uniform and equal process distribution among servers becomes a serious problem and the resource allocation in cloud

DC become more difficult as the cloud infrastructure grow. For solving such a problem several heuristics are exposed to be used such as job consolidation which is one of the leading solutions for high resource utilization and preserving the minimum energy levels consumption [10].

- **High number of Idle Servers**

  The cloud service provider always provide huge number of active servers for their users around the clock, and the server usage go through non stable task arrival growing and shrinking based on the different timing of the day, week and vacations. Given that an idle server still consumes up to 70% of power [11], there is a good possibility to efficiently manage the state of the servers between active, idle, and off to preserve as much power as possible using an energy aware scheduler.

This thesis aims to propose a scheduling model that effectively manage to avoid the effect the previous cloud datacentres problem maximizing the servers utilization based on threshold margins of virtualized hosts, Parallel similar task processing under multiple constrains, applying VM consolidation with load balancer and changing the servers states between active and off mode.

1.4 Aims and Objectives

The aim of this research is to propose a novel scheduling algorithm that optimises resource utilization and reduces power consumption while balancing between Quality of Service (QoS) and fairness among the jobs so that the efficiency may be increased. The novel algorithm allocates resources based on an energy optimization method named Sharing with Live Migration (SLM), which selects and plans the usage of VMs based on the similarity between the tasks. The proposed

algorithm aims to minimize make span, power usage and costs across a datacentre infrastructure. The main objectives are as follows:

- Investigate the current solutions for optimizing the energy consumption within the cloud datacentres in regards to the energy model creation, provisioning, and execution.
- Compose a comprehensive review of state-of-the-art on energy efficient scheduling techniques and their impact on energy consumption in cloud computing datacentres.
- Investigate the diversity of current power optimization scheduling methods presenting their definitions, classifications and potential benefits and cons.
- Design a novel energy efficient job scheduling algorithm for cloud computing datacentres that classifies and optimize the dynamic workload allocation process and execution time based on workload characteristics.
- Design a method for partitioning active virtual machines and job distribution algorithm to provide concurrent execution for identical jobs based on file locality and flexible settings.
- Design a dynamic virtual machine consolidation algorithm with live migration triggered by threshold to detect and control the datacentre utilization levels, manage active server's state and prevent the performance degradation resulting from multiple VMs migration.
- Analyse the design job scheduling technique, VM allocation and consolidation policy in terms of energy consumption, migration number and make span of cloud datacentre.
- Evaluate the scheduling model of the cloud datacentre against two adapted algorithms (Ant Colony Optimization ACO and Swarm Optimization Algorithm) and provide its impact on the cloud performance

under several constrains to prevent any service level agreement (SLA) violation while maintaining the maximum QoS agreement.

1.5 Methodology

The research methodology of this research consists of the following steps:

1. Explore and investigate the most recently deployed energy efficient scheduling algorithm in cloud computing datacentres and evaluate existing technologies and their abilities to optimize the energy.
2. Design and develop an energy optimizing scheduling model based on the conducted analysis of existing types of scheduling techniques as well as an energy efficient job scheduler architecture to improve virtual machine consolidation, resource utilization and energy efficiency
3. Simulate the proposed energy efficient algorithm using the CloudSim simulation platform used for implementing and testing energy aware schedule model.
4. Evaluate the theoretical performance compared to other scheduling algorithms regarding the make span and energy consumptions parameters measurements.

1.6 Thesis Organization

This thesis has eight chapters. Following this introduction, chapter 2 describes the theoretical background of cloud computing by giving an overview and abstraction of the cloud architecture and basic functionality. Then, introducing the energy efficient cloud computing environment, current solutions and algorithms to optimize the energy consumption in cloud computing.

The first part of chapter 2 provides a detailed description and definition of the cloud concept. Section 2.3 presents the cloud benefits and drawbacks, specifying the cloud vendors and providers of this service. Furthermore, section 2.4 and 2.5 overview on cloud venders and their provided cloud service-oriented models and services types. In section 2.6, the cloud deployment challenges are defined in order to be able to evaluate existing technologies and related research, followed by resources management strategies for providing an energy efficient cloud infrastructure. Section 2.7 propose the security measurements deployed within the cloud environment of CSPs to prevent threat attacks on the cloud.

Chapter 3 propose a comprehensive analysis on the dynamic job scheduling techniques by first proposing the scheduling a taxonomy and concept in section 3.1 and 3.2. Section 3.4 state the current scheduling problem and introduce the main scheduling assessment factors which are used to verify the scheduling performance and rank the quality of deployed scheduling strategy within a cloud datacentre. Additionally, an overview of the existing scheduling algorithms and the current approaches are presented in section 3.5 followed by the scheduler amplifications and a comprehensive model aspects classification and analysis of adopted energy efficient schedulers denoting their limitations and weaknesses.

Chapter 4 describes the proposed SLM scheduling technique and analyses the designed structure, where section 4.2 present the novelty of the SLM scheduler and the scheduling technique model constrains and assumptions, while section 4.3 proposes the SLM model flowchart. Afterwards, section 4.4 and 4.5 propose the model and describes the SLM algorithm. In section 4.6 demonstrates the used model parameters while the user request model deployed with in the scheduler and the workload generation process are presented in 4.7. The deployed power

model used with in the datacentres and the overall power consumption with a detailed energy usage aggregation and constrains for each host in section 4.8.

Chapter 5 presents the scheduler amplifications and model aspects classification where section 5.2 present the SLM scheduler different jobs assortments and provisioning schemes of allocating VMs where section 5.3 present the job classifier algorithm policies used to manage the dynamic generated workload in the datacentre. Then, section 5.4 explain the host utilization and management, while the jobs VMs reservation and dispatch process is proposed in section 5.5 by demonstrating the scheme of dual-fold VMs structure virtualization to define and partition the CPU usage among concurrent jobs. In section 5.6 the cloudlet provisioning and allocation scheme Guided Backfilling Algorithm (GBA) used to preserve and allocate jobs to the dual fold VMs structure for providing synchronize job processing of the workload with in this scheduler. Then, section 5.7 propose the virtual machine assortments and provisioning strategies including the VM to host allocation scheme, the threshold margins setup policy and the applied hosts and VMs datacentre policy to reach the minimum VM to host allocation cost of the system. Section 5.8 demonstrate the SLM live migration scheme and its time aspects to satisfy the VMs consolidation and load balancing based on the SLM defined high and low threshold presented in section 5.7. Then, the SLM high and low utilization detection scheme is presented to reach the energy efficient state of the datacentre.

Chapter 6 discusses the SLM cloud simulation environment proposing in section 6.2 the adopted scheduler simulation requirements and the CloudSim tool that satisfy these requirements. In Section 6.3, the offered CloudSim architectural components for managing the SLM implementation was acknowledged with a detailed execution model and creation steps in section 6.4. Then, CloudSim

simulation components are demonstrated in section 6.5 illustrating the key scheduling managements classes used to manipulate the power consumption aiming for energy efficiency. The simulated SLM model assumptions and execution model are presented in section 6.6 and 6.7. In section 6.8 and 6.9, the SLM CloudSim experiment setup and simulation scenario was proposed including the creation phase, cloudlets generation and multiple runs simulation. Then, in section 6.10 the scheduler simulation results context are presented in regards to energy, utilization, make span and VMs migration numbers.

Chapter 7 analyse the environmental execution matrix and provide its impact on the cloud performance against adapted state-of-the-art algorithm for similar problem such ACO and PSO. The effectiveness of the SLM was presented in section 7.2 in regard to the VMs servers' utilization, migration number, energy consumption and make span, while section 7.3 present the impact of these three factors on the overall energy of the cloud datacentre.

Chapter 8 concludes the PhD thesis with a summary of the achievements of this work, an outline of the advantages of the proposed solution, a summary of the claims of novelty, a discussion of the limitations of the research, and potential avenues to pursue for future work.

CHAPTER 2
BACKGROUND AND FOUNDATIONS

Chapter 2. Background and Foundations

2.1 Introduction

The constant changes in computing and communications technology led to the need of on-demand access to shared computing resources to reduce cost and time. Cloud computing represents the preferred alternative for on-demand processing and storage where clients can save, retrieve, and share any measure of data in cloud [10], which delivers computing services to users as a pay-as-you-go manner through the Internet. Section 2.2 of this chapter presents an overview of the cloud and its associated benefits, while section 2.3 emphasises on the cloud concept, the pros and cons of cloud deployment in order to evaluate existing technologies and their abilities to avoid most of the cloud cons. Then section 2.4 introduces existing cloud vendors and their adopted model design models. In section 2.5, the architectural cloud considerations are presented and cloud deployment models are defined and classified to: Infrastructure-As-A-Service (IAAS), Platform-As-A-Service (PAAS) and Software-As-A-Service (SAAS). Section 2.6 present the cloud resources strategies that are used to manage assigning the system available resources to the incoming jobs, where section 2.6.1 define virtualization as one of the main concepts in resource management in cloud. Then the hypervisor abstraction is represented in 2.6.2 followed by the migration definition and deployment types in section 2.6.3. Section 2.6.4 propose the scheduling techniques deployments and its determination events as one of the main and vital resource management strategies in cloud datacentres. Section 2.7 point the cloud environment data confidentiality within the cloud datacentres and the applied security measurements.

2.2 Cloud Computing Abstraction

The use of cloud computing is tremendously growing, and it is becoming the driver for innovation in all companies to serve their customers around the world [12]. The centralized computing service concept goes back to the 1960s, when the mainframe time-sharing technology where used to deliver computing services over a network. The mainframe time-sharing mechanism utilized the computing resources successfully at that time and provided a satisfying performance to the user. The slow growing rate of the income Returned from the applications (ROI) and the low cost of the total cost ownership (TCO) with the lack of full control over the mainframe time-sharing applications led to the evolution of personal computers. Then the Internet became extensively adopted due to the huge grow in the computing business environment focusing on maximizing the income Returned and minimizing the total cost ownership, which raised the need of the client-server architecture became complex [13].

2.3 Pros and Cons of Cloud Deployment

Cloud computing datacentres make reaching any information or source possible from anywhere, eliminating the setup and installation step such that the user and the hardware may co-exist in different places. This comes beneficial for the users or the small companies that cannot effort to pay for the hardware, storage or resources as the big companies, where users don't need any extra arrangement used previously by their on premises computing solutions accept the internet connection to reach their cloud providers without the need of paying for licenses or worry about the installation and updating any resources [14]. The cloud environments are based on datacentres that are used to house computers and their associated components. Moreover, applications are run on servers in the

datacentre that consume considerable energy to host cloud applications. The cloud datacentres significant growth led to a high operational cost along with a huge impact on the environment due to the excessive power consumption. The rising expected electricity demand for datacentres to more than 66% over the period 2011–2035, and here comes the main motivation of this research [15], to throw the light on the above said and brings to the cloud computing environment an efficient energy saving model. There are three key power consuming factors in a datacentre which are servers, communications network and the cooling system. Idle server consumes about 70% of its peak power from the total power consumption with in the datacentre [16].

2.3.1 Cloud Concept

The concept of cloud has emerged as a powerful mechanism for data storage by providing a suitable platform of datacentres. Cloud computing is an online access to a shared hosted remote servers using the Internet and unlike preceding distributed computing such as grid computing, the cloud is available for the end users without the need to master the deployment details where users can easily manage data located far away from them. The cloud consists of either one or multiple data centres where each data centre is built using many storage units, servers, and communication infrastructure. This infrastructure Applications and services are offered to multiple end users through cloud Service providers CSPs using pay-as-use scheme as shown in Figure 2.

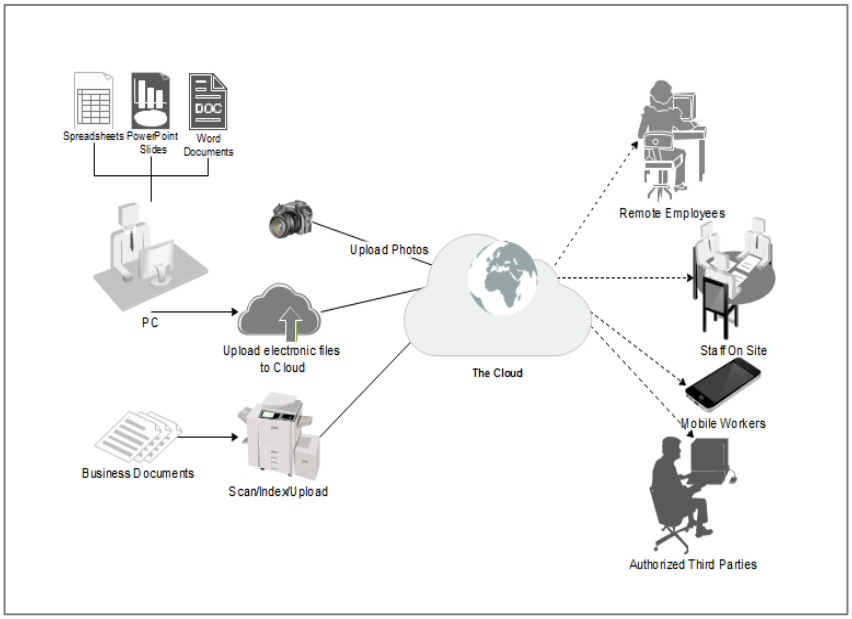


Figure 2. Cloud Infrastructure Applications and Services

National Institute of Standards and Technology (NIST) defines Cloud computing as follows: "Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This Cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models" [14].

Figure 3 show the cloud platform and how the shared resources that are offered by the internet service provider (ISP) are virtualized and controlled by the cloud management system. The management system consists of layers that offers a scheduling strategy, resource provisioning and execution manager.

The end users use a cloud provider application program interface APIs to access the CSPs services and develop applications from outside the cloud, which simplify the communications process between the users and cloud platform. The changing process from one CSP to another is not easy because each provider has a distinct APIs that is totally unique from the others. Along with APIs, the Media Server Mark-up Language MSML and Media Server Control Mark-up Language MSCML are used with either VoiceXML or Session Initiation Protocol SIP as media controlling standard to control the media communication such as videos and voices under the cloud [17].

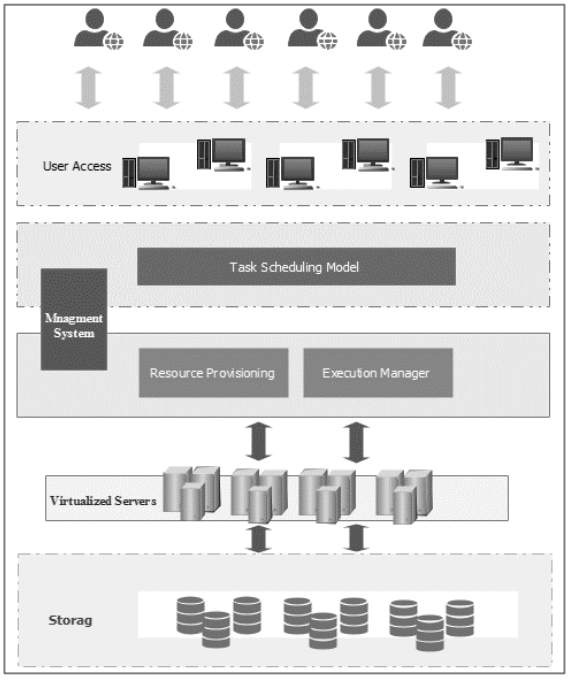


Figure 3. Cloud Computing Platform

2.3.2 Benefits of Cloud Computing

Companies that need to construct their own distinctiveness and characteristic must shift to cloud environment to start dealing with the future challenges and have access to a larger market. The Cloud has opened the world of computing to a range of uses and offered the ease of use to its users, which helped them to concentrate on their business without worrying about the IT issues. The cost

element was not the only reason behind the massive growing demand of the cloud computing industry, and the huge benefits can say that this revolution is here to stay for a long time [18]. One of the main benefits of the cloud is that it provides access to users from everywhere to a desirable flexible environment to work in with no need for any software installations or extra cost, where the huge escalation of business competitions raised the need of installing the latest software and applications to keep up-to-date with the leading technology. The cloud provides the access to the latest apps and infrastructure to its users without having to invest in installations or worry about the maintenance and updates which happens on regular bases automatically and easily. Rapid implementation is another attractive advantage where cloud technology allows a quick deployment and software integration for cloud services. On the other hand, the cloud infrastructure offers a central management of resources which makes the maintenance abilities easy for a giant IT services with less headache on the IT team. Then, a reliable backup and recovery platform is another significant benefit offered by the cloud providers to its users since all the data is stored in the cloud infrastructure which is much reliable than keeping and saving the data on in-house IT infrastructure. Above that, most of the cloud service provider's guarantees assistants all days around the clock with 99.99% availability under the Service Level Agreement without worrying about the server's maintenance or failures [19]. The elastic and on demand basis existence of infrastructure is another benefit of the cloud that provides almost unlimited storage capability and resources to the users with reduced costs of deployment of services, so the cloud makes the IT expenses affordable at economical rates. Also, financially using the cloud services have lower personal training cost because it does not demand an extensive training due to its ease of use where it can easily be adapted and deployed. Cloud involve the lowest learning curve on hardware and software

issues such as Gmail and Google Docs. Finally, we say that the biggest and great benefit of using the cloud is the location Independence, because it can be log on through any device that have access to the internet, so the user don't only have the choice of which device to use, but also where to access the service from.

2.4 Vendors and cloud model advocacy

Cloud computing has a variety of design models and the cloud services offered by different vendors varies based on what model design they are using. The most used models are Infrastructure as a Service IaaS, Platform as a Service PaaS and Software as a Service SaaS [18]. Based on these models, Amazon and Google succeeded in defining the standards for cloud computing and building the largest public cloud. Amazon Web Services (AWS), is one of the well-known platforms in the IaaS service model areas by Amazon, where it offers many virtual servers based creation and management services using any operating systems. Simple Queue Service (SQS), Elastic Cloud Computing EC2 and Simple Storage Service S3 are some of the most used AWS web services for creating, using, and deleting on demand virtual servers and computing capacity. On the other hand Google App Engine (GAE) was the pioneer in PaaS cloud services based on sandbox technology as a Google cloud infrastructure. GAE provides all the necessary requirements for building and running Web applications using automated cloud computing tools with no configuration requirement. While Amazon and Google dominated the IaaS and the PaaS models, Microsoft's used the SaaS model in the well-known Microsoft Live Mesh Platform to provide software services such as live desktop, file sharing and storage. A mesh operating environment (MOE) client software have to be install in order to start adding devices to the platform but it doesn't support all the OS as in most of the cloud platforms [20].

2.5 Architectural Considerations

Cloud computing designers can deliver the cloud practices and benefits through the cloud infrastructure to the end users. This infrastructure consists of several layers to allow and facilitates the use of cloud application and services through an ordinary PC or mobile devices. The architectural components mainly based on a combination of a front-end, which is formed by the cloud-user, browser and the internet connection, and a back-end represented by the unseen complex subdivisions formed by software, applications and storage structure to satisfy the user need and demands. The datacentre encapsulates the physical machines called hosts (memory, cores, capacity, and storage) which can be virtualized later to meet the processing demands. Every datacentre has a set of policies to control its components and a broker which acts as an intermediary between the end user and the cloud where it submits service requests from anywhere in the world to the cloud [21]. Multiple virtual machines created on one physical machine provide a multiple tasks processing on a single device to meet the datacentre need for resource utilization. Generally most of the cloud environments are constructed using multiple datacentres for covering the escalating customer's needs, and with that huge infrastructure a cloud coordinator is used to be responsible for the communication between multiple data centres and their brokers supervising the internal state of the data centre itself for load balancing decisions and sustaining the regulation with delivering a reliable service [22].

Explicitly as in Figure 4, users and cloud brokers submit processes to the cloud datacentre for processing where the role of Service Level Agreement Resource Allocator SLA-RA start to serve as a mediator and assign users/brokers to the available resources in the cloud datacentre. The SLA-RA invokes a service request examiner, which is a utility computing system that is responsible for admission control, and admission control mechanism to validate and accept or reject the incoming process depending on the quality of service agreement QoS requirements and the availability of resources. To successfully manage and accomplish the service resource allocation process, two thing must be provided to the resources allocator before the processing phase start : 1) the current states of the approved resource provided from the Virtual Machine Monitor mechanism VMM, 2) the processing workload provided from the Service Request Monitor Mechanism (SRM). One of the basic management requirements at resource allocation time, cloud price decision, will be specified based on either resource value, the time of submission or charge rate. According to a list of all the VMs with their dedicated resources, a dispatcher will start the execution of process [23].

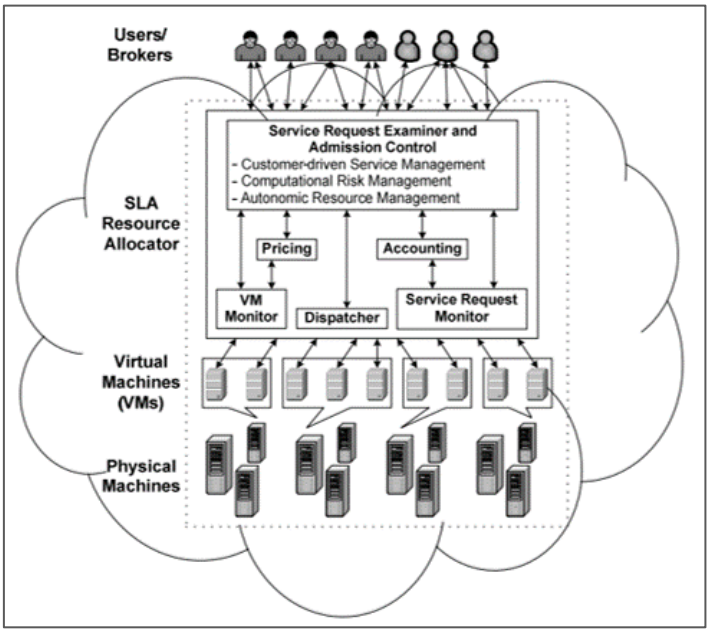


Figure4 . Cloud Architectural Consideration

Cloud Exchange (CEx) represents the connection point between the cloud platform users and coordinators with brokers where CEx is responsible for storing the recent status of the cloud deployment levels in the market and usage prices providing the users/brokers with best matching offers from CSP to their services. Meta-Negotiation Middleware (MNM) is the negotiation infrastructure used to establish the cloud exchange connection using a meta-negotiation document that specifies the conditions and the user's prerequisites for the establishment as in figure 5 and 6 [24].

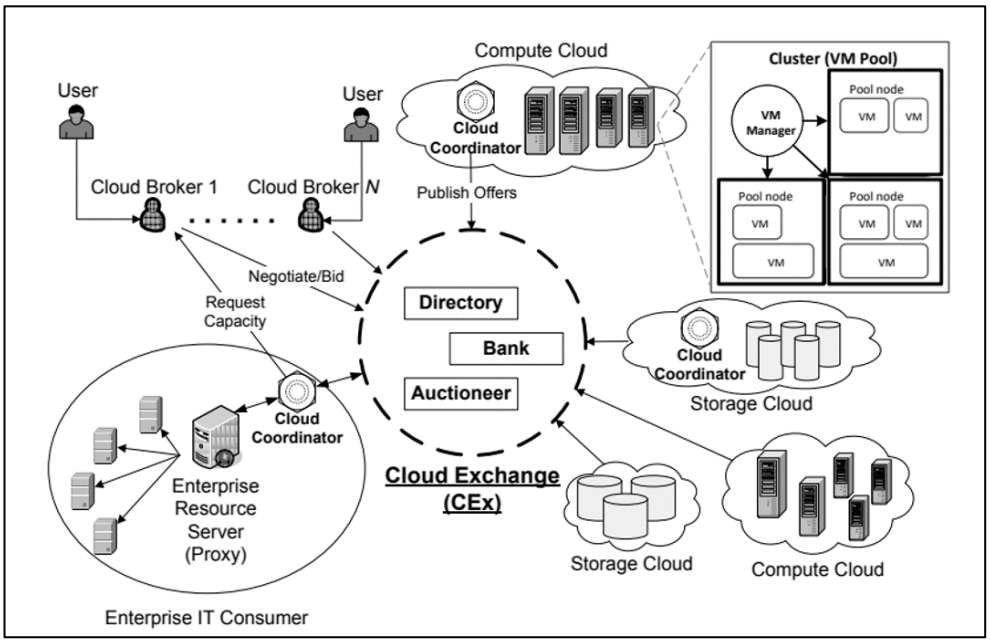


Figure5 . Cloud elements [24]

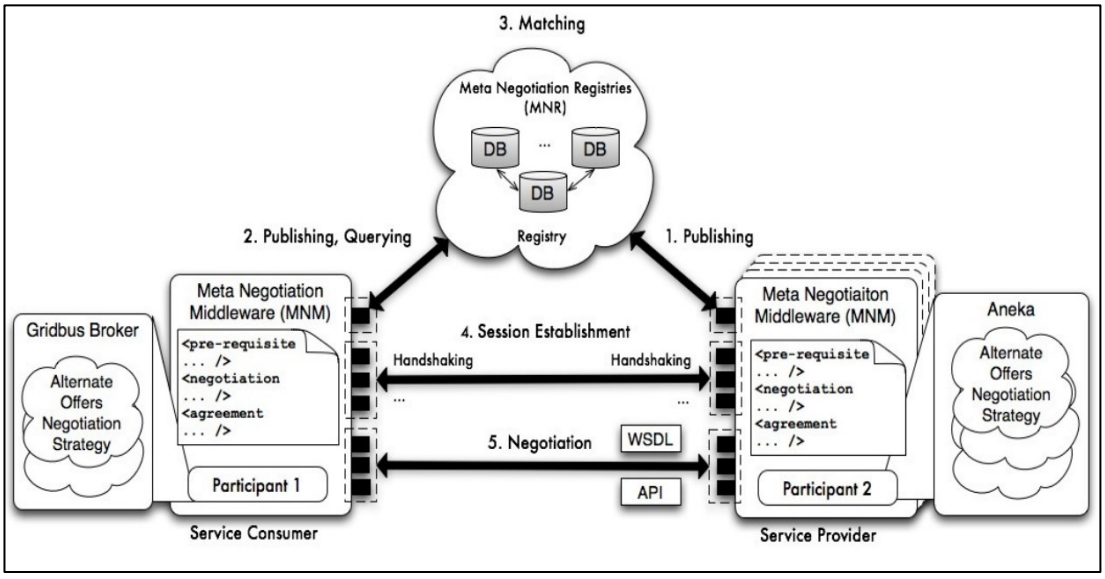


Figure6 . Meta-negotiations architecture [24]

Cloud deployment models define the user accessibility rights to the cloud elements, where the cloud models are classified to public, private, community and hybrid cloud. A public cloud is most identifiable cloud computing model that provides services to multiple customers hence cloud infrastructure is offered to the public users and enables consumers to get the services with very little financial cost. In contrast, a private cloud is a platform created and built for the use of a specific organization or a precise user to have the ultimate control, security, and quality of service of the data. Finally, a hybrid cloud is a combination of two or more of the previously defined clouds types and bound them together as a unit to obtain the best of private and public clouds. The main challenge for the hybrid cloud is the capability to sustain the communication among the different types of cloud models it is made of besides dealing with the different forms of security platforms measures to these models. To form a strong a trust-worthy relationship between end users and the cloud services, a predefined agreement was represented, confirmed and applied under the name of the service level agreement SLA. This agreement provides the user with a clear view to the cloud environment, security, management policies and service monitoring.

2.5.1 Cloud Service oriented Models

Choosing a specific cloud service model depend on three things: the target industry, the requirements of the cloud service provider and the customers' type. The cloud providers play different roles based on the design model adopted in their infrastructure and they deliver services at different layers of the cloud hence the service provided depend on the model design. According to NIST Cloud Computing Reference Architecture the deployment

models are classified in to: Infrastructure-As-A-Service (IAAS), Platform-As-A-Service (PAAS) and Software-As-A-Service (SAAS). The potential befits along with the company's IT infrastructure capability must be considered before electing and deploying most suitable service model [25]. The difference between the housed IT computing and the cloud services models is illustrated in Figure 7, where the existing architectural layers owned by the end users in a private network differs from the other three cloud services models layers while Figure 8 show the layers of cloud Services models.

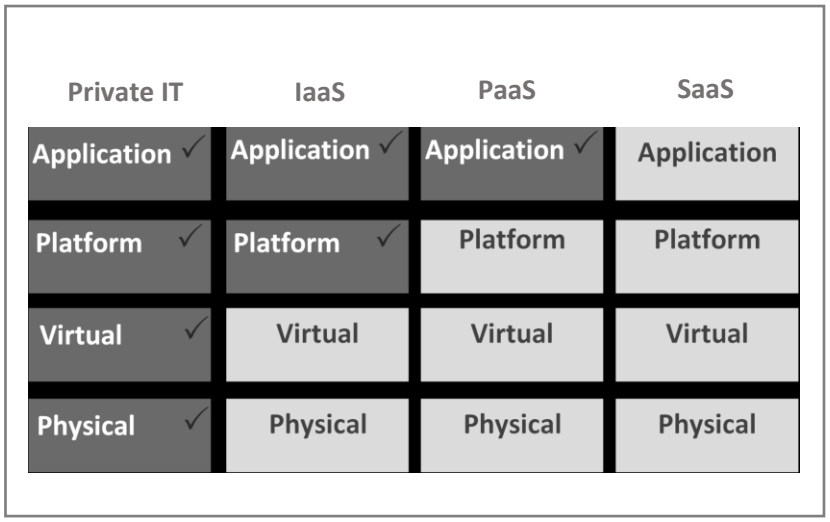


Figure7 . User ownership of cloud architectural layers in different models

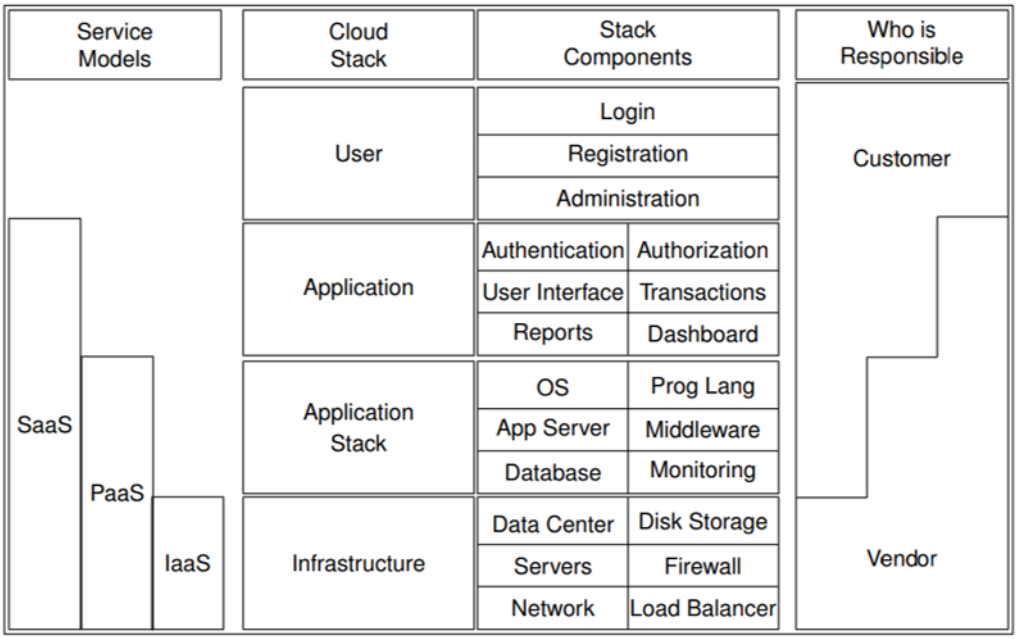


Figure8 . Cloud services acquisition layers

2.5.1.1 Infrastructure-As-A-Service (IAAS)

The maturity and the size of the business or company IT usually is the motivating key for any to shift for IaaS service model, where transforming to this model reduce the Capital Expense (CapEx) for Small and Medium Businesses (SMBs) and empower its customers with unlimited IT access. On the other hand, for large organization it offers conceptual datacentres geographical expansion with minimum administrator-to-server ratios, maximum control of IT assets and hosts utilization. IaaS considered as the lowest layer of main three existing cloud computing layers, where the CSP own the application and infrastructure then offers the virtual hardware, storage, resources and network on the cloud infrastructure. Which minimize the company's IT infrastructure acquisition and operational high cost which eliminate the complexity of deploying. In this model, the end user has the capability of dynamic virtual server creation and management based on need and the ability to configure the settings and implementation of the software and programming environment such as Amazon

EC2, Rackspace and GoGrid. On the other hand, cloud provider is responsible for the installation, storing and upgrading applications with full guarantees on the processing aspects and security [22]. The following IaaS benefits set motivation to organization to move for the IaaS significant capabilities:

- The IaaS can easily identify the system jam points and job loss ratio within business processes.
- Large margins of resources utilization improvement and real time system provisioning with integrated management.
- Dynamic operational strategies suitable for quick organization adoption with the alternating market strategies.

2.5.1.2 Platform-As-A-Service (PAAS)

PaaS delivers an integrated platform to create, run and supervise conventional applications via automated set of software and regularization. As the second layer of cloud computing, PaaS provides development environment so that the user can build the service and develop their application needed using a programming languages suite and development tools. The work of load required from the developer's team in this service model is reduced due to the existence of ready stack of software to serve in development process of all the required users' applications. The users assigning for PaaS service cannot have a full control over the service because of the existence of a software layer between the user and the hardware offered by the provider which is not eventually considered to be a negative thing because this abstraction eliminate the rule that says that the user have to be an expert to deal with this kind of service. Microsoft Azure and Google App Engine deployed the PaaS cloud service model according to the standardization and existing platform automation provided by this service which offer to efficient, and elastic load management. Regarding the workload

physiognomies, PaaS provide the ability for dynamic modifications to the business load priorities under the regulations of service level agreements (SLAs). The following feature distinguish this service among the SaaS and IaaS [26]:

- Uncomplicated administration system structure with workload consolidation and standardized pattern-based load distribution.
- Fixed load conveniences offering cohesive development platform with up scaled provisioning.
- Adopting DevOps, containers, micro-services as associative application technologies for the IT and software staff integration and cooperation.
- SLA based workload optimization and prioritization according to the business requirement.
- Running applications with ready provision infrastructure and SLA enforcement which eliminate the necessity of experienced team presence.
- The capability of economical and quick new application development is the key incentive for large business to consider PaaS subscriptions and deployment.

2.5.1.3 Software-As-A-Service (SAAS)

Small and medium business (SMBs) with minimum IT investments can easily transform to adopting SaaS services where it offer the final layer of the cloud services .It is the model in which the user can access a finished applications from a variety of devices hosted by the provider through a web browser without the need of any software instillation, paying for license or updates, data loss or storage space neither maintenance expenses such as Google Docs, Gmail and Cisco WebEx. In this model the user can reuse and customize the offered services however have no privilege to change the applications design or control the servers, storage or network. The end users are not notified here about the

data location or how is it managed, which decrease the level of reliability for customers on these services [22]. Based on the SaaS level customization, this service is classified in to two conceptual headings: (1) Horizontal SaaS offerings (HSO) that offers archetypal layers of SaaS that match the need of wide range of common applications enough to fulfil the need of public organization without infrastructural customization. (2) Sector-specific offerings (SSO) which conduct devoted version of SaaS to specific sectors with distinctive application constrains. The SaaS vary from other oriented classified models of services by several main features as the following:

- SaaS utilize diminutive and scalable package solutions at the deployment phase which maximize the vendor's ability to adopt effortlessly and faster with current states of the market.
- Subscribing on SaaS services minimizes the CapEx of establishment and delivered the service with the minimum level of configuration control on public internet.
- The CSP is responsible of the Data backup, security aspects and solutions without involving the costumer of the cloud.
- SaaS offers cost flexibility management to organization and based on service acquirement according to the subscribed numbers of costumers.
- The model upgrades are directly controlled and initiated by the CSP division, verified then deployed invisibly from the user's awareness.

Table 2 present a comparison representation of the different existing cloud platforms illustrating some system examples on each.

Table 2. Cloud architectural platforms comparison [24]

Property\ System	**Amazon** Elastic Compute Cloud (EC2)	**Google** App Engine	**Microsoft** Live Mesh	**Sun** Network.com (Sun Grid)	**GRIDS** Lab Aneka
Focus	Infrastructure	Platform	Infrastructure	Infrastructure	Software Platform for enterprise Clouds
Service Type	Compute, Storage (Amazon S3)	Web application	Storage	Compute	Compute
Virtualization	OS Level running on a Xen hypervisor	Application container	OS level	Job management system (Sun Grid Engine)	Resource Manager and Scheduler
Dynamic Negotiation of QoS Parameters	None	None	None	None	SLA-based Resource Reservation on Aneka side
User Access Interface	Amazon EC2 Command-line Tools	Web-based Administration Console	Web-based Live Desktop and any devices with Live Mesh installed	Job submission scripts, Sun Grid Web portal	Workbench, Web-based portal
Web APIs	Yes	Yes	Unknown	Yes	Yes
Value-added Service Providers	Yes	No	No	Yes	No
Programming Framework	Customizable Linux-based Amazon Machine Image (AMI)	Python	Not applicable	Solaris OS, Java, C, C++, FORTRAN	APIs supporting different programming models in C# and other .Net supported languages

2.6 Cloud Resource Management Strategies

Datacentres were struggling constantly from the increased cost of hardware infrastructure, low agility and resource with difficulties in system recovery. Racks of IT equipment's form the basic structure of the cloud datacentres and in the early stages of cloud datacentres, a management system was the only method used to monitor the status of the cloud usage levels, where at the users request arrival, a resource management system (RMS) checks the availability status of the requested resource then constantly keeps monitoring the allocated resource for the sake of system optimization [27] [28]. Recently, several innovative technologies and techniques were presented and adopted by multiples cloud

vendors to provide a better cloud infrastructural construction, deployment and management.

2.6.1 Virtualization

Virtualization is the creation of a logical number of virtual machines that runs on the same physical machine. It plays a decisive role in cloud and it is the heart of the cloud computing industry where it is a way to reach the ultimate and efficient use of resources storage and it provides datacentre deployment of new methods with reliable management for data to achieve green computing through the best utilization of data centre resources. The main challenge facing cloud computing datacentres is to optimize the total operating cost while maintaining the desired Quality-Of-Service (QoS) standards and one of the key elements to achieve that is scheduling. Scheduling connects the users' application with the computing resource based on their requirements and it is one of the most effective approaches that affects the performance of the cloud computing datacentres, but still is quite a challenge in cloud environment to be implemented and applied effectively.

The motivation behind the establishment of virtual machines is to process different tasks that cannot be performed on a single host. Each VM (virtual machine) acts as a one physical server with its own Random Access Memory (RAM), Central Processing Unit (CPU), Network Interface Controller (NIC) and hardware disk. The benefits of creating virtual machines in the datacentres are to reduce infrastructure costs, save energy, faster server provisioning, improve recovery, and isolate applications. The testing and development phase also can be much easier more efficient with existence of the virtualization [12].

Virtualization was introduced as an innovative technology to be one of leading fundamental concepts in cloud for confronting the resource and cost inefficiency,

which provide a software layer as an abstraction layer between the application and hardware layer as in Figure 9.

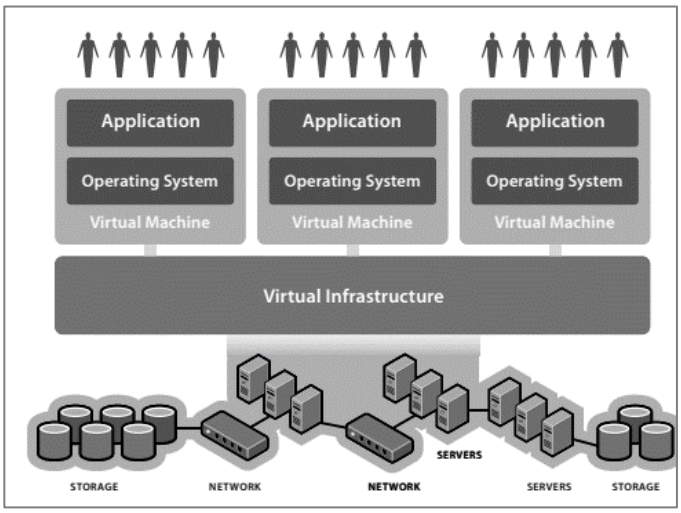


Figure 9. Virtual infrastructure [29]

As a concept, virtualization refers to the process of dividing a single hardware to multiple virtual versions allowing various application to share one physical machine.

Each virtual produced machine has its own memory, CPU, NIC and disk to satisfy all the expected operational requirements which deliver a hardware-independence, high scalability, and great performance in the cloud architecture regarding to the resource decision independency resulting from the VMs operator Isolation [29]. This server portioning procedure take a place in the Instruction Set Architecture (ISA) which considered to be the interface between the software and the hardware at the application layer of the system. VMs is sustained by the capability to transfer, move or copy to any hardware platform based on the server's load states to adopt with best server utilization status. Based on the virtualized area and motivation, virtualization can be performed to any of the architectural layers wither if it is hardware, data, network or applications.

2.6.2 Hypervisor Abstraction

Cloud computing is transforming from rational resource organizing to hyper-visioning industry structure, which will diminish the existence of private desktops operating system market in the future. The transformation progression from a single physical host to virtual partitions is accomplished through a thin layer of software called hypervisor as in Figure 10 (such as Citrix xenServer, Virtual Machine Monitor (VMM), VMware, Microsoft's Hyper-V) which assembles in second all the needed resources for creating a virtual machine VM such as virtual CPU, virtual disk, virtual NIC. The Hypervisor increases the CPU utilization through running various operating systems OS on one CPU [30].

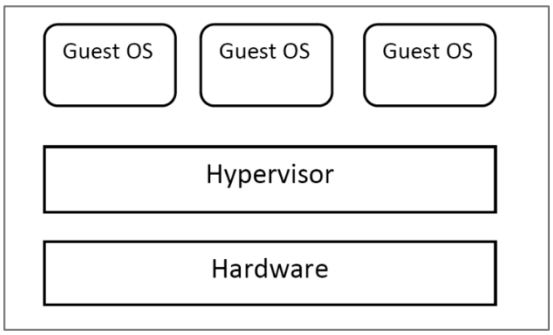


Figure 10. Virtual Machine Diagram

The basic virtualization structure consists of four layers as in Figure 11, the application layer as the first top layer, which involves varying requested user's applications. Then comes the operating system layer followed by the virtualization layer. The virtualization layer compromises of VMM as a hypervisor, which manage multiple operating systems to work on the same machine, along with the local resource manager to control the robustness and scalability among on and off devices.

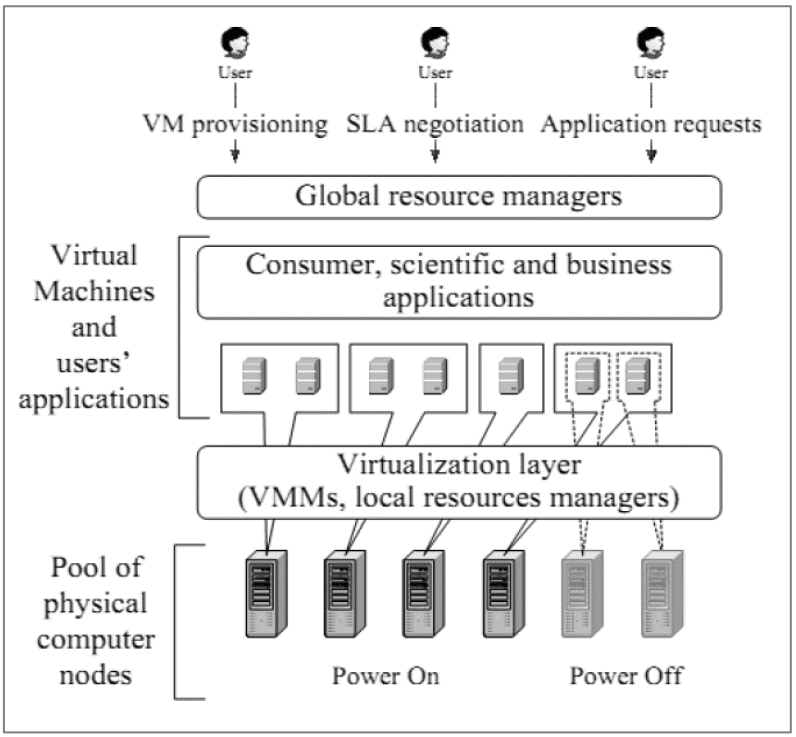


Figure 11. System structure

2.6.3 Live and Offline VM Migration

In general, the most common parameter considered by a cloud provider to facilitate auto-scaling is the threshold. The threshold parameter defines the aggregate value calculated based on current performance metric values, at which point auto-scaling of resources is triggered to. It is an essential need in cloud computing infrastructure to manage resources dynamically regarding the upper and lower threshold and therefore the VM migration is becoming increasingly used in cloud computing. Live VM migration is basically transferring VM from one physical server to another to grant workload balancing, hardware maintenance, high availability services and consolidated management [31]. The modified data pages during the migration phase time are saved and named as dirty pages. This migration needs a continues offered access to the existing host and this access known as the network attached storage (NAS) which allows instant dirty pages copying from the VMs memory at the transfer time. The main three steps to initiate

the live migration are pre-copy, pre-copy termination and stop and copy phase as shown in Figure 12. The first phase is called the pre-migration brownout, where the brownout is a partial and temporary reduction in total system capacity. In this phase, pages are copied from the basic host to destination on different rotations while the VM is still running on the source. The time of the first phase usually depend on the memory size of the destination and the rate of the coping process of the pages. Next, at the second phase, a stop condition must be applied to finalize the pre-coping at the pre-copy termination phase which cloud be either reaching a limited number of rotations or that the target memory had been transferred completely. This phase is called blackout phase because it cause a complete interruption and pauses the VM where now it is not running neither at the source nor the distention. Then, the post migration brownout phase start where the final stop and copy phase ends then the VM starts running in the destination but the source VM is not totally suspended and still provide services when needed until the network catgut the new target VM [32]. Live migration contributes to power saving, as VM migrations can minimize the number of running physical machines by moving tasks to underutilized machines (consolidation) and moving processes from overloaded to less loaded servers (load balancing). Live VM migration Maintain the load balancing in the data centre along with server Consolidation which allows the maintenance of the virtual machine offline whenever that is needed or for upgrading without the need for shutting down the whole system. Applying VM migration in data centre leads for more possibility of turning off more hosts to have resource availability and better energy efficiency using server consolidation. While live migration is performed with without stopping the service, the offline migration is accomplished by stopping the whole process before the job transmission which cost more downtime to the overall processing time.

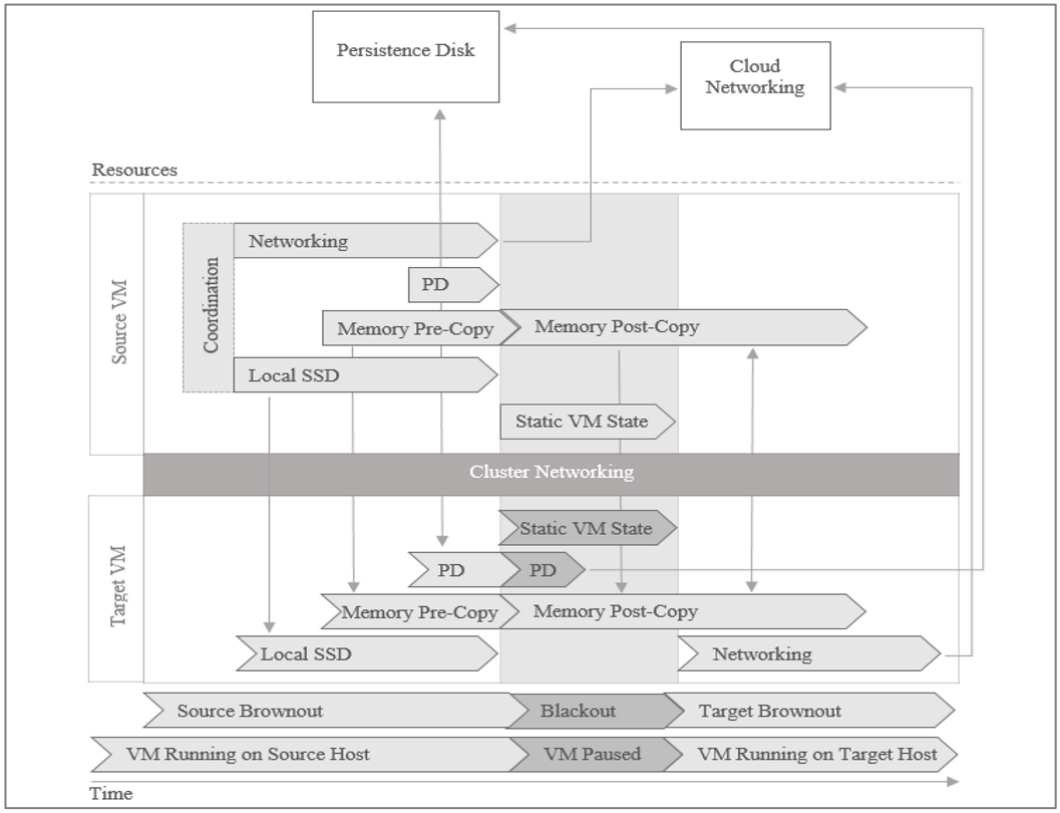


Figure12 . Live Migration Phases

2.6.4 Scheduling Techniques Deployment

Resource and data sharing in cloud environment is the core of cloud evolution where data are distributed and located in different datacentres' which raised the need for efficient resources management and utilization with optimal datacentre performance. This process is challenging because it involves resources heterogeneity, buffering and matching processes. To provide a trusted packet delivery without data delays or losses, the cloud requires a good scheduling at VMs, tasks or resources levels. Scheduling is the process of mapping jobs to existing resources for job execution [33] and to achieve the scheduling process in efficient way, the mapping phase must carry out to the most suitable resource for achieving the optimal cloud performance. As in Figure 13, generally scheduling is accomplished in three main phases [34] where it starts with resource detection phase and during this phase the resources list is inspected by

the CSP to gather the needed information about the existing resources and their availability status. The second phase of scheduling is resource identifying and electing, based on the user and job restrictions the needed resource is chosen for the job execution process. The identifying phase specifies and effects the total cloud performance for that reason it is the critical and critical phase in scheduling. The final phase is job submission and execution, the resource in this phase is available for the job to start its processing [35]. Given that task scheduling is one of the critical determinants in energy consumption for a cloud datacentre, innovative green task scheduling algorithms are designed and established to reduce energy consumption by determining the optimal processing speed to execute all tasks before a deadline [36].

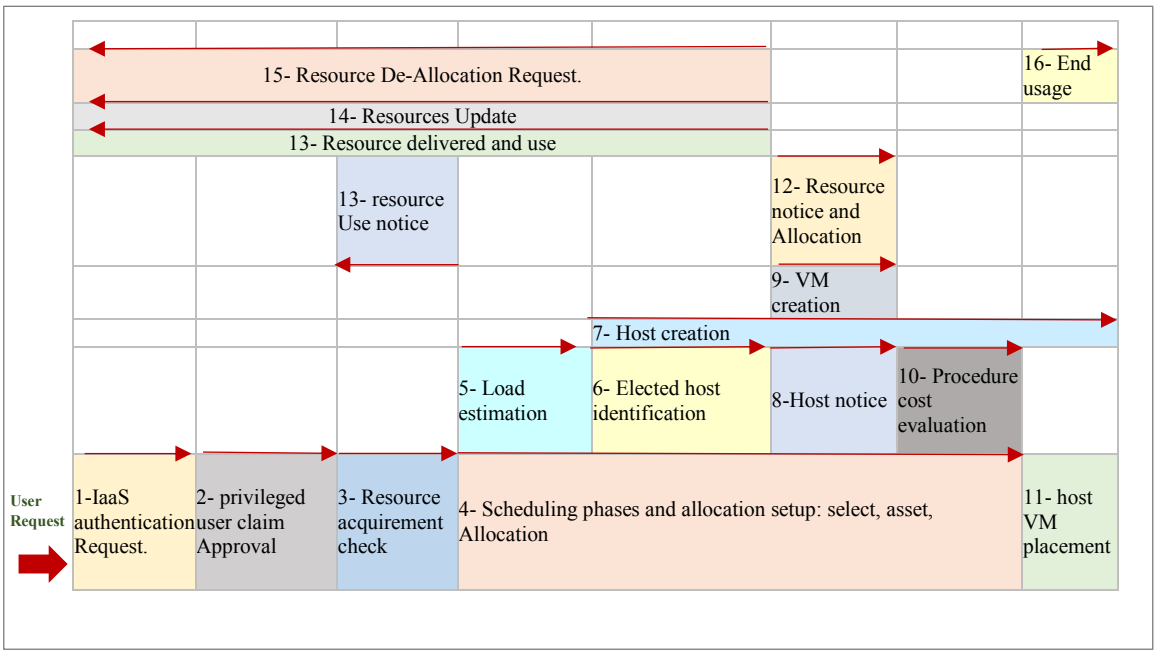


Figure 13. Scheduling Determination Events

2.7 Security measurements in cloud environment

Cloud computing represents the preferred alternative for on-demand computation and storage where clients can save, retrieve, and share any measure of data in

the cloud. CSP locate multiple user's private data on a single host which means that there is a possibility that data cloud be viewed by other users sharing the host. Hence cloud users and CSP always have concerns about their data concealment and privacy, since it is saved in datacentres located across the countries and shared among customers, which form a challenge to the data integrity and security aspects of switches, firewalls, API, hypervisors, DNS etc.. The need for securing the cloud environment raise from multiple authorized internet access, distributed computing, virtualization levels and shared resources. Strong security, integrity and data confidentiality measurements must be applied by providers to prevent the unauthorized access and outside or inside malicious attacks. It is not always the CSP obligation to keep a secure cloud premises, a third-party auditor TPA leased by the CSP take the role of verifying the stored data and deliver an audit report to the user. Users are responsible of checking security measurements in advance with the CSP such as Data segregation, provider viability and regulatory compliance.

The main trusted establishment connection in cloud services are the secure shell security keys SSH. This identity management (IDM) mechanism is used to authenticate users and services to be shared among all VMs located on a single server which cause a threat on the saved data which led to necessity for using the proof of retrieve ability POR tests as a way to perform the data integrity a provable data possession PDP [37]. On the hardware level attacks, spoofing on the LAN segment using the address resolution Protocol (ARP) can have accesses on restricted data to unauthorized users. Corruption of data resulting from cloud system malware is another possibility on software level. As internal datacentres security measurements for the design of the access structure, encryption algorithms are applied for maximizing the privacy and data security such as the Cipher text-policy ABE CP-ABE, fully homomorphic encryption FHE,

key-policy ABE KP-ABE and the attribute-based encryption ABE. Data protection laws and regulations are part of the security fences in the cloud which provide Compliances requirements for customer's data that cloud users should be aware of as the federal information security management Act FISMA [38].

2.8 Conclusions

The rise of cloud computing datacentres offered pay as you go accessible data to end users with reliable services using virtualized computing utility, resources and storage. The cloud prominent existing and deployed architectures are categorized in to public, private and hybrid to serve different objectives for diverse varieties of customers. The architectural design provides flexible set of different service models with diverse levels of structure manipulations for system integration named as software as service, platform as service and infrastructure as service. These deployment models offer different architecture exposure providing a wide infrastructure options for cloud vendors and providers. IaaS is the lowest infrastructural layer which offers conditional permits of access to different computing resources such as servers, storage. The intermediate layer of the cloud infrastructure is PaaS, this platform Deliver network programming network access to the cloud datacentres environment. On the other hand, SaaS is the uppermost layer of services platform that verifies users and validate their access based on requirements to several provided services by the CSP.

In a way to guarantee the optimal utilization of the datacentre and have a substantial effect on the cloud performance, a management strategy of the cloud resources and storage are applied on different datacentre layers. This chapter covers many aspects in energy efficient cloud and resource allocations strategies. Many strategies had been designed to optimize the energy

consumption levels and most of these techniques was based on three main solutions. First, virtualization which eliminate the challenge of providing sufficient utility and resources to the user using the existing structure without the need of raising the expenses charged by the CSP, where virtualization deliver ultimate and efficient use of resources storage with new datacentre method deployment for reliable management for data. Then, virtual machine migration on the other hand consolidate existing host utilization within the datacentre to maintain stabilized load balance among VMs. Another strategical cloud management element is adopting the optimal scheduling scheme, which is one of the crucial stages in the system that plays a significant role in the overall performance by distributing the system load on processors to maximize the utilization and minimize the total tasks execution time.

CHAPTER 3

COMPETITIVE ANALYSIS OF DYNAMIC JOB SCHEDULING TECHNIQUES

Chapter3. Competitive Analysis of Dynamic and energy efficient Job Scheduling Techniques

3.1 Introduction

Aside from cloud benefits, cloud computing datacentres are facing structural problems, where high power consumption and balancing between minimizing energy usage and the delivered performance is one of the most significant challenges faced by cloud service providers. As the fourth concept in cloud resource management, scheduling techniques are defined and presented in section 3.2.1, then scheduling process main phases and execution events are proposed. Section 3.2.2 identifies the cloud energy efficient structure and declares the scheduling conception and its assessment matrix. Then section 3.3 proposes the problem statement regarding job scheduling, afterwards the energy efficient scheduling techniques related work are presented indicating their improvements and limitations in section 3.4. Afterwards, the scheduling heuristics are classified in to six basic criteria and presented with recently conducted scheduler's examples on each type indicating to each algorithms failures and successes in providing energy optimizing datacentres.

3.2 A taxonomy of energy efficient job scheduling

Datacentres typically run all the year 24/7 with usually 10 KW/m^2 power density to satisfy several businesses daily operation processing needs, and this number is growing annually by 12% Over 90%. The electricity used by cloud computing datacentres is consumed by the server, storage, and network plus the cooling system [39], and the way to minimize the energy consumption within the cloud datacentres is to start managing the power of one or all these elements.

3.2.1 Scheduling Conception

Different scheduling models as in Figure 14 have been proposed and studied comprehensively, but most of the existing schedulers do not cover the latest and most recent cloud environment weaknesses, or works well with one aspect raising or leaving another downside unsolved which raised the need for more efficient and innovative scheduling models.

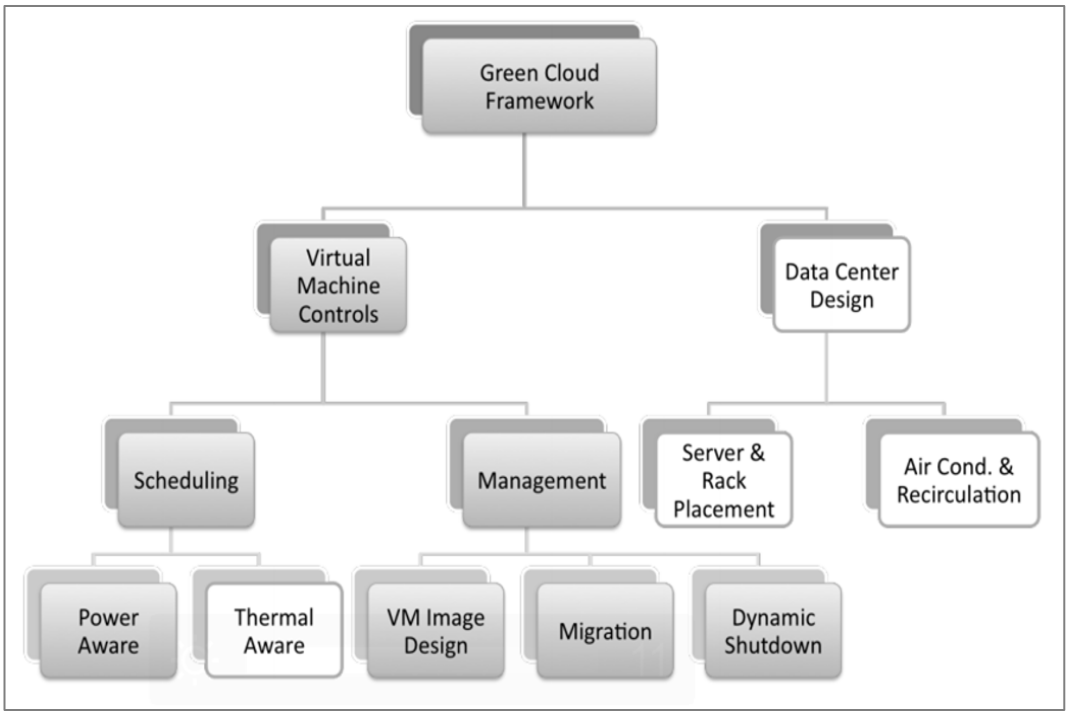


Figure 14. Green cloud scheduling models

Scheduling is a wide concept and key element in cloud computing with a prodigious effect on the superiority of cloud service, focusing on all computing resources. A scheduler could classify and analyse computing resources reasonably so it could reduce the execution time of tasks [40]. Optimizing the efficiency of the scheduling mechanism in a cloud environment can increase the performance of the server and associated resources and maximize the processes managing proficiency [41] [42].

Job scheduling works on mapping users request to the most appropriate resources efficiently and effectively using one or more strategy [43]. Scheduling

algorithm can control the overall processing time, total energy consumption and the datacentre performance of the cloud [33]. The deployment of an energy efficient job scheduler maximizes the numbers of accepted jobs by the CSP and helps to maintain the SLA with reliable resource provisioning to reduce the ratio of VMs migrations between servers and the processing cost rate [44]. Power-aware and thermal-aware systems are main cloud characteristics considered at the green computing scheduling phase, while the first model aims to minimize the energy consumption the second tend to reduce the temperature in the hardware structure and maximizing the energy levels within the datacentre.

3.2.2 Energy efficient Scheduling Assessment Factors

The cloud is mainly used to provide the user with a Quality of Service (QoS) and applying a job scheduler within the cloud datacentre should eventually lead to reach that purpose. There are many factors and different objective functions that can be used and directed to increase the job scheduling efficiency and improve its performance with in the datacentre, and based on these factors the job scheduler performance is evaluated and prioritized among other scheduling technique applied in the datacentre and evaluate its performance. The measurements of selecting these factors and objectives fluctuate based on the end user and the service provider of the cloud and these factors are proposed and defined in Table.3 [45] [46].

Table 3. Scheduling Techniques Measurements Factors

Parameter		Definition
user	**Execution Time (ET)**	It is the time from submitting a task to the cloud until it executed.
	Response Time (RT)	The time that the system starts responding to the submitted task. *RT=Service Time-Wait Time*
	Execution Cost	It is the over-all cost of the resources used at the task execution.
	Make span (MS)	It is the total time to finish all the tasks scheduling. *MS= Completion Time – Start of processing*
	Reliability	It means that the user should receive continuous service without any kind of failures.
	Scalability	The system power to develop itself due to the growing demand or the increasing of the data.
	Fairness	It means the tasks equality in sharing the CPU time.
	Job Rejection Ratio	It is the ration of the overall rejected tasks to the total submitted number of tasks.
	User Satisfaction Level	It is the satisfaction of the user on the resources like storage and computation.
	Budget constrains	The cost limitation for processing all tasks.
CSP	**Priority**	Give advantage to specific jobs to use available resources earlier than other submitted job based on AR, ET or service deadline.
	Energy efficiency	Use technology solutions or platforms tools to minimize the energy production within the datacentre.
	Predictability	The stability of the job response time qualifies the system performance to be judged as a predictable cloud.
	Resource utilization	The system should use the resources in the most optimal possible way to minimize the utility cost and guarantee the maximum revenue.

3.3 Problem statement

Providing an eco-friendly scheduling algorithm for large-scale data centres is the purpose of this research project. Existing cloud servers suffer from overloading and high utilization due to high workload and insufficient number of resources which cause cloud performance degradation along with energy levels elevation. On the other hand, mostly used scheduling techniques by companies are limited

in use and their deployment does not satisfy the eco-friendly environment need. A cording to the energy efficient mostly used bin packing scheduling algorithms such as first fit decreasing scheduling algorithm, incoming cloudlets placement go through the following diagram flow starting from submitting the task to the data centre until the cloudlets are fully processed as in Figure 15 [47], where each new added job to the system will be dynamically assigned to the first host a new available time slot for processing. Then, jobs are executed one after another regardless any job ranking where a new placement allocation take place and a new time slot reservation in the next direct available VM at job arrival, which will cause inefficient use of resources and storage which maximize the total make span of the system. Since in IaaS, several jobs of different types and weight coexist in the same environment and share resources and storage simultaneously, the need to derive a conceptual workload management and scheduling strategies is raised to handle different job types and prioritize incoming workload unlike previously cloud deployed strategies.

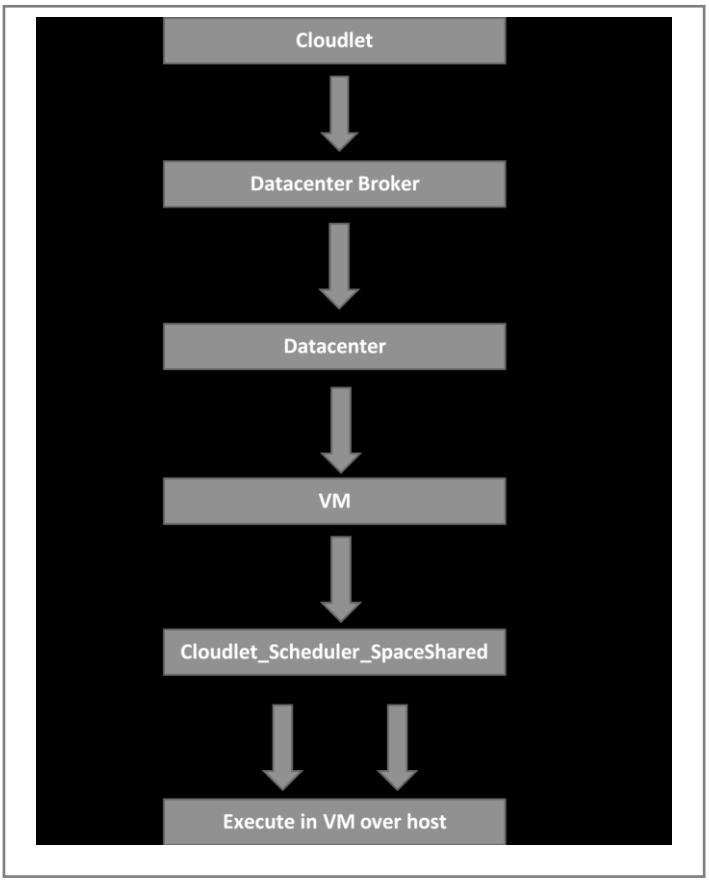


Figure 15. Job execution flow chart

3.4 Related work

Job allocation and scheduling complications have been comprehensively studied and efficient algorithms have been proposed to minimize the energy consumption, make span, cost, and deliver the best quality of service (QoS). There have been a considerable amount of research papers conducted using task scheduling strategies in cloud datacentres and managed successfully reducing the total energy consumption.

Almezeini et al [48] propose a novel task-scheduling algorithm designed using the Lion Optimization Algorithm (LOA) for a cloud environment. The scheduler was implemented to imitate the lion instinct reactions of lions by tracing the optimal lion behaviour that have the best fitness value represented by the jobs make span, which is the maximum completion time for the tasks. The scheduler is represented using lions where each lion identifies his optimal solution that is

updated on regular base to minimize the total execution time for the cloud datacentre. The scheduler minimizes the make span and the degree of imbalance sustaining high performance and resource utilization compared to the Genetic Algorithm (GA) and Particle Swarm Optimization (PSO) scheduling techniques. Razaque et al. [49] proposed an efficient task-scheduling algorithm based on the available bandwidth where each VM has a maximum rate of data transfer known as VM bandwidth and jobs in this proposed algorithm are apportioned then assigned to VMs according to the VM bandwidth using nonlinear programming model. The number of jobs in each VM is quantified and have a maximum execution time limit to all the assigned jobs while the bandwidth is above zero. This algorithm eliminates the job waiting time factor which minimize the make span and improve the cloud efficiency. Amjad et al. [50] proposed an Adaptive Genetic Algorithm (AGA) for real-time job scheduling as a new version of the GA algorithm. The AGA deployed a range of appropriate mutation and crossover techniques to prioritize jobs over VMs to achieve qualitative scheduling solutions. The main drawback of the AGA is that it involves a large computational time but it is sufficient for escaping local minimums with high accuracy and best performance compared to the GA. N.Moganarangan et al. [51] presented a new Hybrid algorithm for reducing energy consumption and make span using voltage scaling factor by merging the advantages of ant colony optimization ACO and cuckoo search algorithm CSA. In this algorithm the energy status maintains a certain level while increasing the number of processors with less job execution time compared to the ACO. Deepika et al [52] proposed an algorithm that Categorize jobs based on their deadline and cost restrictions then assign theses jobs to different priority queue. Regarding the resource selection the scheduler select VM with the lowest turnaround time for each individual task. The algorithm is more cost efficient and fair compared with consecutive job scheduling with

simple constraints-based task scheduling algorithm. D.I Esa et al [53] proposed a novel job scheduling mechanism for optimizing the job execution time through performing multiple iterations of a firefly algorithm where each round of job to resource assignment represents a firefly. The random job assignment iterations are accomplished in order to find the ultimate execution time scheduling plane with best fitness value. The evaluation showed this technique outperformed the FCFS algorithm and delivered better job execution time. Komarasamy et al [54] proposed a new scheduling technique called Content-based Federated Job Scheduling (CFJS) algorithm in the cloud computing, to execute the deadline and non-deadline-based jobs concurrently in a VM to eliminate the system delay. A higher priority of processing was offered to the deadline-based jobs then assigned to the most suitable VM to reduce the job waiting time. On the other hand, non-deadline-based jobs were assigned to the idle VMs for better resource utilization. The scheduler minimizes the waiting time of tasks, maximize the resource utilization and throughput. Paper [55] proposed the generalized priority algorithm GPA that assigns priority to tasks based on task size where the highest size task has highest priority. In the GPA, VMs selection is based on their MIPS values where the VM with the highest MIPS is given the top priority to be selected. The scheduler evaluation results indicate a better performance and efficient job execution against FCFS and SJF.

In [56], authors proposed a dynamic voltage and frequency scaling (DVFS) enabled energy efficient workflow task scheduling algorithm (DEWTS). The DVFS use randomly generated DAGs workflows and the heterogeneous earliest finish time HEFT algorithm for the job scheduling order estimation, which is used to calculate the deadline balances and make span by dividing the parallel tasks in workflows then executes jobs at appropriate time to reduce the power consumption. Without affecting the make span or the jobs reliance restrictions,

jobs are distributed to idle slots based on the voltage scaling algorithm, Evaluation tests showed that the power consumption was minimized by 46.5% in parallel tasks execution.

The priority task scheduler from [57] is a green energy efficient that efficiently uses the dynamic voltage frequency scaling (DVFS) technique to allocate proper resources to jobs according to requirements of jobs under the Service Level Agreement (SLA). This technique classified jobs based on length, deadline and age to ascend job in the priority queue if it has smaller deadline compared to the existing job deadline in that priority queue. This scheduler improved the resource utilization and reduced the power consumption of the datacentre hosts by 23% in comparison to the dynamic Voltage Scaling technique [58]. An offline scheduling strategy for intensive applications is proposed in [59] where data are represented in binary tree to reduce the energy consumption by lowering the SLA rate. Using data correlation clustering algorithm and Task Requirement Degree TRD calculations, the proposed scheduling strategy decrease network traffic, improve the utilization of servers and reduce the energy consumption due to decreasing the number of active hosts in the cloud datacentre.

In [60], the authors introduce an energy aware scheduling strategy with DVFS integration in multicore processors, which works with two different execution mode. To set the execution speed and the order of job an optimal scheduling used for batch mode and heuristic algorithm for online jobs. The scheduler improved the energy consumption by 27% for batch mode applications, and 70% in the online mode applications. The proposed new virtual machine scheduler in [61] specify an optimal performance power ratio to schedule the virtual machines by selecting first the highest performance power virtual machine (VM) with in the deadline constraint. The algorithm divides the scheduling to multiple periods to efficiently allocate VMs to hosts then reconfigure the resources allocation to lower

the level of energy. This scheduling algorithm improved the processing capacity by 8% and up to 20% energy saving. Pavithra and Ranjana in [62] proposed an energy efficient resource provisioning framework with dynamic virtual machine placement using energy aware load balancer in cloud. The presented technique uses weighted first com first served to reduce the number of overloaded hosts and minimize the energy by obtaining measurable improvements in resource utilization, cost and execution time of cloud computing environment.

The proposed a scheduling scheme in [63] utilize the resources and save the energy consumption by optimizing the VM allocation and using a consolidation policy to running servers. Results show energy consumption improvement compared the DVFS and Energy aware Scheduling algorithm using Workload aware Consolidation Technique (ESWCT).

The proposed efficient server first scheduling in [64] used the response time in respect of the number of active servers to reduce energy consumption within the data centre. About 70% of the energy is saved in comparison of random selection-based strategy.

The previously stated job scheduling algorithms consider the single job processing and managed jobs independently in order to provide the best performance or energy efficiency. While the idea of elevating the Scheduler performance of similar jobs concurrent processing was neglected. Also, previous schedulers worked on one parameter of job scheduling while the algorithm must provide and relay on many factors to provide fairness to users when providing services and deliver the ultimate performance with the best reduced cost and power consumption. Intel's Cloud Computing 2015 Vision stresses the need for dynamic resource scheduling approaches to increase the power efficiency of datacentres by shutting down idle servers [65] and using a good job scheduler can reduce both the power consumption of the utilized resources and the

processing time of an application [66]. Table 4 shows a comparison of energy efficient scheduler highlighting their limitation and improvements.

Following from the limitations of the above studies, this thesis focuses on designing a novel scheduler to minimize the total processing and communication costs for a set of tasks to be executed on virtual machines aiming to reduce the overall energy consumption.

Table 4. Comparison of energy efficient scheduler

References	Strategy	Scheduling Goal	Simulation Tool	Improvements	Limitation
[48]	•Used the LOA algorithm •Maximize the completion time	•Make span optimization •Energy consumption	CloudSim toolkit	•Minimize the make span •Low degree of imbalance •High performance •Resource utilization	•Insignificant progress for small number of cloudlets
[49]	•Applied bandwidth convenience •Jobs are apportioning	•Cloud efficiency •Make span minimization	Java	•Eliminate waiting time factor •Minimize the make span •Improve cloud efficiency	•Has a maximum execution time limit • limited number of jobs in each VM
[50]	•Used mutation and crossover technique •Prioritize jobs over VMs	•Energy efficiency •Server load	Developed their own simulation toolkit in C++	•Eliminate local minimums •High accuracy •High performance	•large computational time
[51]	•Used voltage scaling factor •Utilized and merged ACO and CSA	•Sustainable energy consumption •Make span •Server load	Developed their own simulation toolkit	•Minimize execution time •Energy optimization for high load	•Maximizing the job load with steady energy level
[52]	•Applied deadline and cost restrictions •Use different priority queue	•Increases utilization of resources •Energy consumption	CloudSim toolkit	•Cost efficient •Fairness among jobs	•Complex constrains
[53]	•Used firefly algorithm •Best fitness value	•Job execution time optimizing •Energy consumption	CloudSim toolkit	•Minimize job execution time	•Used multiple iterations
[54]	•Concurrently deadline and non-deadline jobs execution •Deploy in the parallel VM architecture	•Resources utilization rate •Energy efficiency •Throughput •SLA violations	CloudSim toolkit	•Eliminate the system delay •Better resource utilization •Minimize the SLA violations and avoid system starvation	•Suitable only for deferent deadline job types •Different job constraints
[55]	•Generalized Priority algorithm •Assigns priority to tasks based on task size	•Efficient job execution •Energy efficiency	CloudSim toolkit	•High performance •Efficient job execution	•Queue waiting time •Conducted for fluctuating number of VMs and workload

References	Strategy	Scheduling Goal	Simulation Tool	Improvements	Limitation
[56]	•Use DAGs workflows and HEFT algorithm to calculate the deadline balances and make span •Apply Dynamic voltage scaling algorithm to distribute jobs in the idle slots.	•Traffic load •Energy consumption	Developed their own simulation toolkit	•Power consumption was minimized by 46.5% •Balance the scheduling performance	•Not considering system reliability •Need extra measurement to be applied in real systems
[57]	•Apply DVFS technique to allocate jobs. •Classified jobs based on length, deadline and age to ascend job in the priority queue.	•Improve resource utilization •Optimize power consumption •SLA violation	Developed their own simulation toolkit	•Reduced the energy of the datacentre hosts by 23%. •Increase resource utilization	•Did not achieved load balancing
[58]	•Data are represented in binary tree •Apply correlation clustering algorithm and TRD calculations	•Network traffic minimization •Energy optimization Lower the active number of hosts	CloudSim toolkit	•Improve server utilization •Reduce energy consumption	•Drop the Service level agreement (SLA) rate
[59]	•Set the execution speed and the order of job •Used optimal scheduling for batch mode and heuristic algorithm	•Total energy consumption optimization •low time complexity	Developed their own simulation toolkit	•Improved the energy consumption by 27% for batch mode •70% energy improvement in the online mode.	•works with two different execution mode
[60]	•Select the first highest performance VM with in a deadline constraint •Divide the scheduling to multiple periods	•Energy consumption optimization	Developed their own simulation toolkit	•Improved the processing capacity by 8% •20% energy saving	NA
[61]	•Use weighted first com first served. •Applied energy aware load balancer	•Optimize the energy •Reduce the number of overloaded hosts.	CloudSim toolkit	•Minimize the energy •Efficient resource utilization, cost and execution time	•Static VM placement consume more execution time
[62]	•Optimize the VM allocation and using a consolidation policy to running servers. •Used ESWCT.	• Energy consumption optimization	CloudSim toolkit	•Energy consumption improvement	NA

3.5 Scheduling Heuristics Classification

Job scheduling is one of the stages in the system that plays a significant role in the overall performance. The goal of scheduling algorithms is distributing the load of the system on processors to maximize the utilization and minimize the total tasks execution time. Scheduling techniques could be classified based on six criteria: Urgency (immediate vs batch), priority (pre-emptive vs non-pre-emptive), distribution (centralized vs decentralized), cooperation (independent vs workflow), prior knowledge (heuristic vs meta-heuristic), and flexibility (static vs dynamic) [15].

The immediate (online) mode heuristic scheduling is used in real-time services as an event triggered (ET) systems based on predefined rules which provide flexibility to the system. The ET scheduling all jobs parameters are not fully known and scheduling will take place at the job arrival time with no waiting based on a prescheduling test that meets all jobs priority and deadlines constrains. Comparative ratio CR is used in ET scheduling as measurements analysis number to compare the online scheduler's performance based on best optimal solution ratio with minimum ratio value set as 1. The CR for a list of J jobs calculated as:

$$CR = \frac{NUMBER\ OF\ used\ BINs}{OPTIMUM\ B_J} \qquad (1)$$

Where bins are containers of fixed capacity used for packing different tasks while B_j is the optimum needed number of bins for serving the list of Jobs [67]. Four types of immediate scheduler are presented with their advantages and drawbacks in Table 5.

Batch Mode Heuristic Scheduling Algorithms (BMHA) also known as the offline scheduling used in time triggered (TT) systems where all activities take

place in progression with time. The jobs parameters must be known in advance so that when all jobs arrive at the same time, they will be collected for processing at fixed intervals of execution. The TT scheduling uses a scheduling table with a list of the jobs and their system activities which grantees a fair job execution process. This scheduling involves a comprehensive understanding of cloud environment and the system requirements to have the ability to create the execution tables based on suitable solutions that satisfies the requirements. Table 5 demonstrate some of the offline scheduling [68] [69].

Table 5. Online and offline scheduling techniques examples

Scheduling Technique type	Scheduling technique	Description	Advantages	Disadvantages
Online Scheduling	**First fit (FF)**	This algorithm assigns the first task in the queue to any available idle server with competitive ration 1.7 while a task is assigned to a new server once there are not enough resources in the available servers	• Ease of implementation due to the allocation simplicity	• Maximize the memory external fragmentation and have high level address space consumption • Does not provide efficient packing for late large weight jobs
	Best fit (BF)	The scheduler here chooses the server where the task best fits in with the least remaining resource after the task is assigned to it	• Relatively simple • Offers space block availability.	• Implies a method for free partitions which maximize the time cost and suitable for small job allocation
	Worst fit (WF)	This algorithm attempts to choose the server that has the largest amount of remaining resources.	• Avoids tiny external fragmentations • fast allocation mechanism	• Work on medium size job allocation and tends to leave large free blocks. • Open new bins directly without fitting.
	Random fit (RF)	The scheduler assigns the task randomly to any available server with enough resources	• Time efficient with no allocation and deallocation strategies applied	• limited number of jobs per time unit

Offline Scheduling	**First fit decreasing (FFD)**	This algorithm sorts all tasks in descending order based on metric stated by the algorithm packing task in the first suitable bin, then packed in the first fitting bin.	• Provide good solutions but not the best.	• Works only for greedy strategy • The least tight value additive constant has conflict.
	Best fit decreasing (BFD)	This algorithm sorts all tasks in descending order based on the least amount of remaining resource then push the task in the fullest bin where it fits the most	• Result with the optimal performance ratio for BP Problems	• Requires large computational cost • Need parallel techniques

Pre-emptive and non-pre-emptive scheduling are another classification of job scheduling algorithm. Pre-emptive scheduling allocates jobs for a restricted time permitting task processing to be interrupted and moved to another resource based on priority, where a high priority jobs are admitted to a ready queue and low priority jobs may starve processing. This allocation offers resources to jobs for a limited time allowing the process interruption which provide a level of system flexibility and improve response time but results with starvation for lower priority jobs. Shortest remaining time first (SRTF) and Pre-emptive priority-based job scheduling algorithm in green cloud (PPJSGC) are examples on pre-emptive scheduling [70]. In the case of Shortest remaining time first, the task with the lowest remaining time until completion is selected to start processing, while for Pre-emptive priority-based job scheduling algorithm in green cloud, jobs are assigned based on the best fit as per their energy requirements and server frequency availability, which is executed by the DVFS Controller [71]. On the other hand, non-pre-emptive scheduling will not transfer or pause the CPU cycle of a job once the job is at the processing phase. At long job processing, the second consecutive job may starve due to the long burst time which eliminate the flexibility of this scheduling. Non pre-emptive scheduling is a fair scheduling with high throughput and low interrupt latency time considering its simple design. A couple of techniques that are typical examples on this scheduling are improved shortest job first scheduling algorithm to decrease starvation and Priority. The first

technique is an enhancement Shortest Job First scheduling where the scheduler assigns the processes in order of task arrival as shortest job first with a possibility to change the priority to diminish the waiting time of task. The second technique is Priority scheduling, where this scheduling assigns each task with a priority level and allocate it based on this level, so that tasks with higher priority are processed first [72].

Centralized (distributed) and decentralized scheduling are other examples of scheduling models that are frequently deployed in cloud and have substantial effects on the cloud services performance and security aspects. In centralized scheduling, jobs in the system are scheduled using a unified scheduler which enhances the speed of processing and ease the complex monitoring procedure of the system resources. The easy implementation of the centralized scheduling reduces the unregistered VMs cases but on the other hand increase the implication risk of the system failure due to the high tolerance of job processing management fault. A high level of resource scheduling and operation managements are handled by a single server in the eucalyptus cloud controller CLC through a management user console or API requests command-line interfaces.

The Priority job Scheduling Strategy for Heterogeneous multi-datacentres is one of the examples on the centralized scheduling where in this scheduling the strategy is constructed to maximize the benefits of resources utilization and make span .the priority job scheduling is based on three main factors: (1) job size which is chosen to provide fast resource releasing after ranking and processing short jobs,(2) job deadline that is used mainly as a key element to allocate the processes before acceding its predefined deadline premises avoiding the release of processing SLA violation, (3) job age to amend the maximum execution time

of the process after it suffers from long waiting time when its ranked as low priority [73]. Resilient cloud datacentres are based on decentralized scheduling which have no central control and a cluster controller keeps the state of scheduling for its respective cluster, which makes the resource scheduling more efficient but less control over scheduling process. The techniques are presented in Table 6 as examples on this scheduling.

Table 6. Decentralizes scheduling examples

Scheduling Technique Type	Scheduling Technique	Description	Advantages	Disadvantages
Decentralizes scheduling	Load balancing task scheduling algorithm based on feedback mechanism	The algorithm uses the weighted random strategy, overload assessment and feedback to submit tasks first to the best resources with the greatest performance after assuring that it is not to overload [74].	• Improve the cloud efficiency • Reduce the average response time of jobs	• The data centre suffers from lower throughput • Lower finish times
	De-centralized dynamic task scheduling using hill climbing algorithm	It is a dynamic scheduling algorithm that uses hill climbing algorithm aiming to reduce the completion time of jobs and improve the throughput and resources utilization [75].	• Achieved load balance • Reduces jobs completion time of tasks	• Require scheduling computation for resources • Involves constant provisioning

The following presented and used classification of scheduling models are Independent and workflow schemes. Independent scheduling applies jobs execution based on arrival and the order given by the priority list once they become free. Workflow scheduling in cloud becomes an important research topic and it is one of the prominent issues in this domain. In this type of scheduling, tasks are dependent on each other where a task can start its execution until all its preceding tasks are already finished. Workflow scheduling is described by a Directed Acyclic Graph (DAG), in which each task is represented by a node and the flow by edges. The main advantage of

this scheduling is to reduce the make span and expand the utilization of resources. The following techniques in Table 7 are examples on both scheduling techniques [76].

Table 7. Independent Vs Workflow Scheduling Techniques

Scheduling Technique Type	Scheduling Technique	Description	Advantages	Disadvantages
Independent Scheduling	Period ACO_based scheduling Algorithm (PACO)	PACO uses ant colony optimization algorithm with the first proposed scheduling period strategy and the improvement of pheromone intensity update strategy, which optimize the make span and load balance [77].	• Good performance in make span due to periodic strategy employment compared to Min-Min algorithm • Achieved Load balanced cloud cluster	• Does not work on job dependence
	Self-adaptive ant colony optimization (SAACO)	This algorithm uses the swarm optimization (PSO) to change the parameters of Ant colony optimization (ACO) to be self-adaptive. This algorithm improves the total make span and the load balance [78].	• Updated pheromones and calculation • Better performance than PACO both in make span and load balance.	• Limited to and useful for solving construction project and time-cost optimization problems
Workflow Scheduling	Improved particle swarm optimization (IPSO)	The IPSO is used to minimize the total cost of assigning tasks on available resources. Total cost values are obtained by varying the communication cost between the resources, task dependency cost values, and the execution cost of compute resources [79].	• Improved convergence accuracy and fewer cost function evaluations compared to generic algorithm (GA)	• Did not control the power loss problem
	Positive And Reactive (PRS) scheduling algorithm	This Uncertainty-Aware Real-Time Workflow Scheduling algorithm combines the proactive and the reactive scheduling methods, to achieve cheaper computational overhead [80].	Make good trade-offs compared to SHEFT and RTC among: • Cost • Resource utilization • Deviation.	• Only needed for processes with priorities

	Cost effective genetic algorithm for workflow scheduling in cloud under deadline constraint (CEGA)	It is a novel scheduler based on genetic algorithm (GA) for encoding, population initialization, crossover, and mutation operators of the Genetic Algorithm [81].	Minimize the : • Finishing time • Cost	Dose not consider : • Virtual machines performance • variation acquisition delay

Job scheduling approaches can be further categorized as a heuristic scheduling algorithms and Meta-Heuristic based job Scheduling. Heuristic are extracted from intuitions and usually get the easy and quick solution but not the best one. The performance of heuristic-based algorithms profoundly relied on the success of the heuristics [82] [83]. Unlike heuristic meta-heuristic scheduling methods use a guided-random-search-based process for solution searching. Metaheuristic methods always have much higher computational cost than heuristic but can obtain better performance in terms of schedule quality [84]. Table 8 show different examples on the heuristic and meta-heuristic scheduling.

Table 8. Heuristic and meta-heuristic scheduling techniques examples

Scheduling Technique type	Scheduling technique	Description
Heuristic Scheduling	The heterogeneous earliest finish time (HEFT)	This algorithm assigns priorities based on the earliest start time of each task and it minimizes the task's start time by allocating a task to the processor.
	Max- Min scheduling	Max-Min algorithm selects a task from the tasks list that have the maximum completion time on a resource that can execute it within a shorter period
	Enhanced Max-Min	It is an enhanced Max-Min algorithm where the scheduler assigns a task from the tasks list to resource based on the average time of job execution instead of highest completion time.
	Genetic scheduling algorithms (GA)	In GA, a chromosome is used to represent each possible solution and a random population is taken and used as an initial data.
	Ant colony optimization scheduling algorithms (ACO)	This algorithm imitates the ant's lifestyle where number of artificial ants' exchange information through a communication

Meta-Heuristic Scheduling		scheme to help in creating solution for optimization problems
	Particle swarm optimization scheduling algorithms (PSO)	It is an intelligent algorithm based on the social behaviour of animals such as a flock of birds searching for a food source or a school of fish defending themselves from a predator. It is practical for function optimization problems. İn PSO, the solutions are named particles and each particle will have a fitness value that will be defined by a fitness function to have trajectory based on its best position and the position of the best particle of the whole population.
	League championship algorithms (LCA)	It is inspired by the contests of sport teams in the league sport. A league schedule is designed periodically for individuals to play in pairs and the win or loss result depends on the fitness value of a team.

In Static job scheduling, as in table 9 all the existing resources and the data of the tasks are available in advance by the time of task is scheduled and there are no resource failures. Patel et al. [85] presented the Enhanced load balanced Min-Min ELBMM for Static Meta Scheduling in Cloud Computing, where this scheduling is based on applying two stages of allocation to process the job. Min-Min strategy is performed as first step to selects the task with minimum completion time then as a second step pick the heaviest load job to assign it to the most underutilized server with the appropriate resource. This scheme assigns jobs to servers with the minimum completion time to effectively utilizes resource and deliver better make span but the limitation here is that it maximizes the response time for short jobs because it gives priority to long ones.

To deal with the fluctuating loads in the cloud datacentre, dynamic job scheduling techniques are deployed to deliver a QoS assurance to satisfy the SLA by dynamic scheduling over time to manage the different workload. This scheduling is more flexible than static scheduling but cause more overhead on the system.

Table 9. Static and Dynamic scheduling techniques

Scheduling Technique Type	Scheduling Technique	Description
Static Scheduling	Enhanced load balanced Min-Min (ELBMM)	This scheduling is based on Min-Min approach where it selects the task with maximum completion time and assigns it to appropriate resource which effectively utilizes resource and deliver better make span [85].
Dynamic Scheduling	Dynamic resource scheduling method based on fuzzy control theory	This algorithm predicts user resource requirement using the Second moving average method then the relationships between resource availability and the resource requirements are determined [86].
	A delay-based dynamic scheduling algorithm for bag-of-task workflows with stochastic task execution times	This algorithm calculates the actual task execution times using the summation of task execution time expectation and standard deviation. It Minimize the cloud resource renting cost by the deployment of both a bag-based delay scheduling strategy and a single-type based virtual machine interval renting method [47].
	Self-adaptive layered sleep-based method for security dynamic scheduling	This algorithm improves the resource utilization, accuracy and efficiency of security resources scheduling by combining three different models which are decision-making tree, top-down analytical and self-adaptive filtering [87].

3.6 Conclusions

As the cloud industry is becoming a major resource and energy consumer, a high efficiency cloud computing datacentres are needed to guarantee a green computing future, and to meet the industrial needs scheduling techniques are designed using the most optimal approach for utilizing the available system resources. In this chapter, the scheduling is identified as a concept presenting its determination events and amassment factors. The heuristic Scheduling algorithms are classified into different categories varying based on their final goal to serve the end users requests, then evaluated in regard to their overall effect on energy. After conducting a comprehensive literature survey on existing schedulers, it is found that the recent existing job schedulers doesn't consider the

job's characteristics neither its requirements while the main considered factor is mostly the response time and waiting time minimization.

On the other hand, authors have not consider parallelism in multiple job processing and managed jobs independently which implied extra overhead on the servers and increased the waiting of the incoming workload. On the other hand, the idea of elevating the Scheduler performance of similar jobs concurrent processing was neglected. Also, previous schedulers worked on one parameter and a single factor of job scheduling such as improving the system utilization, load balancing and active host state management while the algorithm must rely on the accumulative effect of many factors to provide fairness to users when providing services and deliver the ultimate performance with the best reduced cost and power consumption.

CHAPTER 4

THE PROPOSED SHARING WITH LIVE MIGRATION (SLM) MODEL

Chapter 4. The Proposed Sharing with Live Migration (SLM) Model

4.1 Introduction

Cloud datacentres are usually involved with deadline obligations to accomplish a chain of jobs processing before pre-specified deadlines. An optimized mechanism must be developed to increase the cloud computing job processing mechanism and at the same time maintain the maximum profit margins of using the cloud architectural model. The users of cloud services are multiplying every day due to the vast spreading of the cloud environments which create a challenge for the developers to schedule and provision efficiently the cloud components and utilize resources. With that the management of power consumption in cloud datacentres may lead to some improvements in the energy consumptions levels. Refining the energy optimization of cloud computing raised the need of the optimal solutions and researches in this area. Applying an innovative idea of scheduling algorithms will help to govern and optimize the mapping process time between the datacentre servers with the incoming tasks. In this research we present a novel algorithm for a cloud computing environment that could allocate resources based on energy optimization methods called Sharing with Live Migration Model (SLM).

This chapter describe the proposed structure of a sharing with live migration scheduler where Section 4.2 present the system model starting from the job model to the deployed power model. Then a detailed demonstration of the used SLM scheduling technique specifically the model algorithm and flowchart. The SLM energy efficient scheduling metrics was presented identifying the server's replication method and Virtual machine assortment and provisioning such as VM

allocation policy, backfilling Algorithm, dual-fold layer VM virtualization, overloaded and underutilized VMs conceptual detection and VM space-share policy and optimal Make span.

4.2 The Novelty of the SLM Scheduler

Applying a new idea of scheduling algorithms will help to control and optimize the mapping process time between the datacentre servers with the incoming tasks. The SLM scheduler will perform an optimal deployment of the datacentre resources to achieve good computing efficiency, network load minimization and reducing the energy consumption in the datacentre. What makes the SLM scheduler different from the previous models is combining the key energy controlling factors to have the ultimate energy optimization scheduling model and these main factors are presented in the following points:

- The SLM model uses the server replica scheme detailed in section 4.4 from chapter 4, which can be simply explained as the existence of the duplicated sever of each main one in the datacentres. This way will direct job at the first phase to the main hosts until the main ones are fully utilized then start mapping rest for the free or underutilized duplicated host which will eventually reduce the total energy consumption within the datacentre, because the average idle server consumes approximately 70% of the power consumed by the server running at full CPU speed [9].
- Virtualization of the existing physical machines in the datacentre which will manage resource deficiency and maximize the datacentre utilization as will be explained below in section 4.5.

- Synchronized homogeneous jobs processing under the SLM explicit constrains using the proposed backfilling guided algorithm in section 4.6.
- Direct job live migration as detailed in section 4.5, by transferring jobs to underutilized machines and moving processes from over loaded to less loaded servers which minimize the number of running physical machines, saves transmission time and reduce the network bandwidth.

4.3 Model assumptions and constrains

The SLM allocate jobs to physical resources, which are called hosts, and each process will be assigned to a VM in a way to minimize the number of migration and hosts consolidation to achieve the optimal energy within the cloud datacentre. The SLM scheduler employs a synchronized job characteristic (SJC) table for scheduling the incoming requests from the write and upload queue (WU) or the read and download queue (RD) based on the matching requested vacancy VMs in order to minimize the number of migrations and eliminate the cause of SLA violation. To develop the system model a number of N jobs requested by users assigned specifically to a VM using SJC table for host and VM allocation scheme, where each needed resource file by the job is listed with proper host as the designated destination, in order to minimize the migration costs among hosts as in Figure 16.

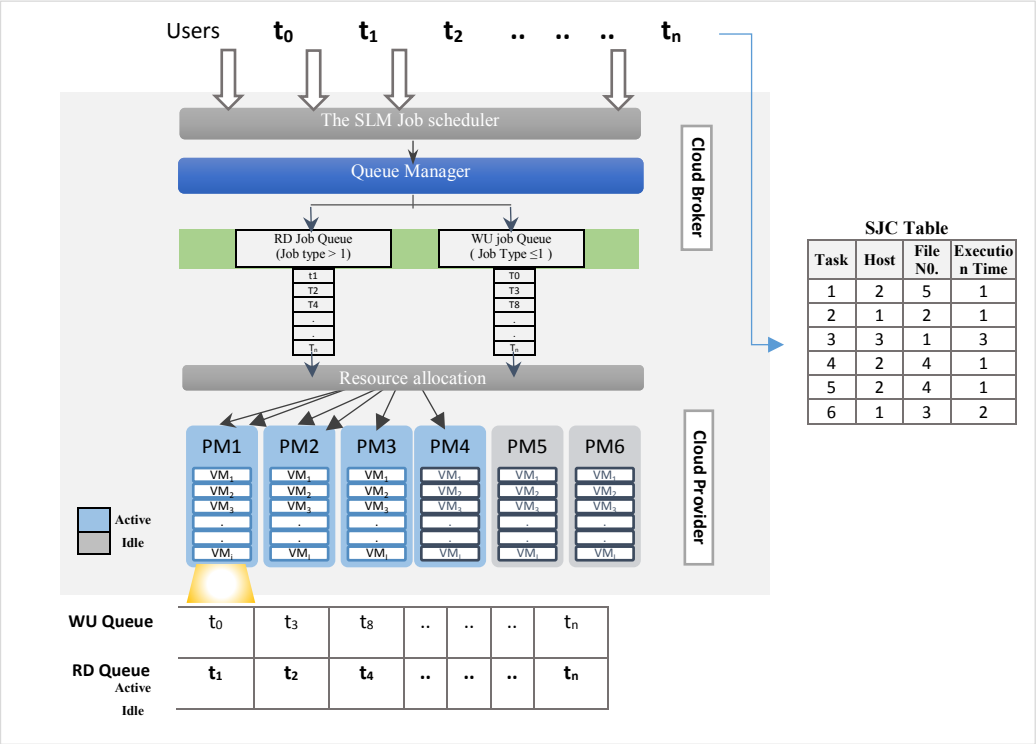


Figure 16. SLM system model

Assuming all jobs are independent In the SLM model, the following mathematical model defines the scheduling model problem:

$$\sum_{i=1}^{l}\sum_{x=1}^{r} Exec_{ix} \cdot J_{ix} + \sum_{i=1}^{l}\sum_{x=1}^{r} E_{ix} + \sum_{i=1}^{l}\sum_{x=1}^{r} cost_{ix} \qquad (2)$$

This is based on that the objective of the SLM independent scheduling problem is to minimize the execution time and the energy consumption. Since the SLM assumed communication cost⇔ 0:

$$\sum_{i=1}^{l}\sum_{x=1}^{r} Exec_{ix} \cdot J_{ix} + \sum_{i=1}^{l}\sum_{x=1}^{r} E_{ix} \qquad (3)$$

Formulating some constraint on the model, (4) and (5) ensures that each task is assigned only to one VM at specific time until it is fully processed. Equation (6) ensures that job processing required in million instruction assigned to specific

virtual machine x should not exceed the available processing power (MIPS) p_X for the VM_x. Equation (7) Assures that the total memory required for all jobs scheduled for x VM should not exceed the maximum available memory for the x VM.

$$J_{ix} = \begin{cases} 1, & if\ job\ is\ alocated\ to\ VM \\ 0, & otherwise \end{cases} \quad (4)$$

$$\sum_{i=1}^{l} J_{ix} = 1 \qquad \forall \mathrm{Job}, \quad (5)$$

$$\sum_{i=1}^{l} MI, J_{ix} \leq \mathrm{p}_X \qquad \forall \mathrm{VM} \quad (6)$$

$$\sum_{i=1}^{l} mem_I, J_{ix} \leq M_x \qquad \forall \mathrm{VM} \quad (7)$$

4.4 The proposed model

This thesis proposes a sharing with live migration Job Scheduling algorithm to find the optimal approach for identifying the status of the existing virtual machines and concurrently process incoming cloudlets based on similarity classification model in a cloud datacentre. The SLM scheduler perform an optimal deployment of the data centre resources to achieve good computing efficiency, network load minimization and reducing the energy consumption in the datacentre. Each host have three different source files that are needed by the cloud users. The datacentre has six hosts where the replica method is applied on half of the hosts so that three hosts have 9 different sources while the other half of hosts have the exact copy of the 9 files as in Figure 17.

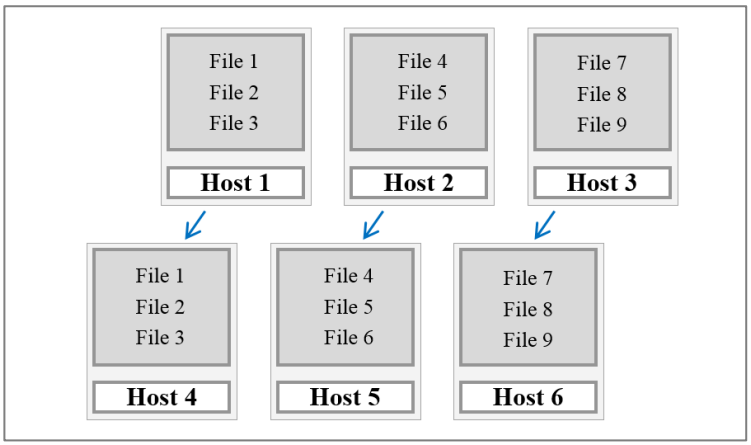


Figure 17. Identical hosts structure

The job workflow contains four types of cloudlets, which are reading, writing, uploading, and downloading with varying execution time. Based on the SLM concurrency control technique, two or more jobs can be executed concurrently if the job nature is identical in matters of the processing type needed and requested resource. The algorithm works on remapping jobs to be processed with similar arriving jobs and hibernating idle servers by executing the automatic live migration. Figure 18 represent an example where in a) a first fit decreasing placement is assigning job to a new VM based on bin packing algorithm after performing the two previously explained phases of the FFD scheduling. While the SLM scheduler is applied in b) if the file 3 is accessed by job $R_{2,3}$ for reading or download, in that case, another job $R_{3,3}$ requesting to access the same file for reading or download, then it will be allowed to access and completes its execution concurrently at the same time slot. On the other hand, if job $WR_{1,1}$ is accessing the file 1 for writing, then, no other job would be permitted to access the file 1 for reading or writing at that moment to avoid file overwriting. By permitting multiple job for reading or downloading, the same file concurrently will enhance parallel execution of job.

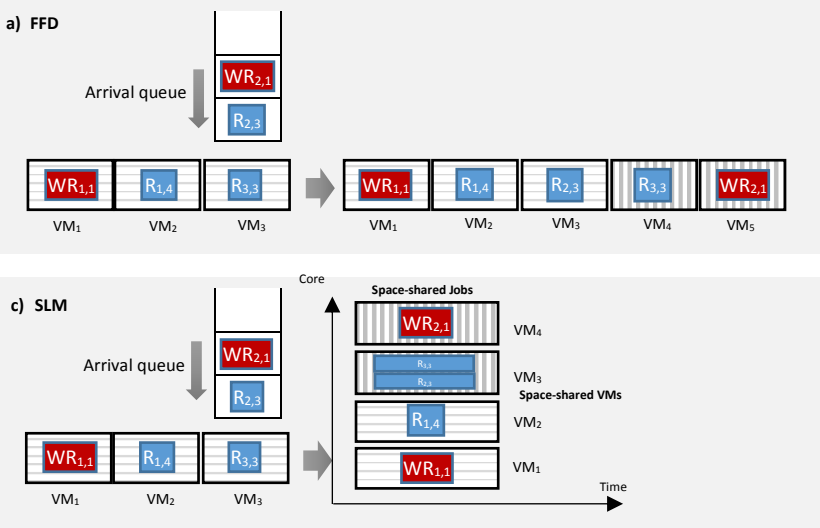


Figure 18. a) First fit decreasing scheduling, Vs b) Sharing with live Migration scheduling

4.5 SLM scheduler algorithm

Based on the previously described SLM model, the proposed sharing with live migration scheduling Algorithm reduce the total energy using four main phases: Virtual machine selection and placement, multiple job concurrent processing, active live migration and idle host shutdown. A random workflow is used where user requests are executed on non-pre-emption manner and all jobs are independent. The SLM algorithm is demonstrated in the following steps as shown in Figure 19:

Step 1: user request processing various types of jobs. Each job may fall under four different task types: reading file contents, updating data, uploading files and downloading software. The processing time fluctuates based on the type of job selected and the number of instructions for each task.

Step 2: The queue manager receives a set of user requests as an input in collaboration with the SLM scheduler that is connected to SCJ table.

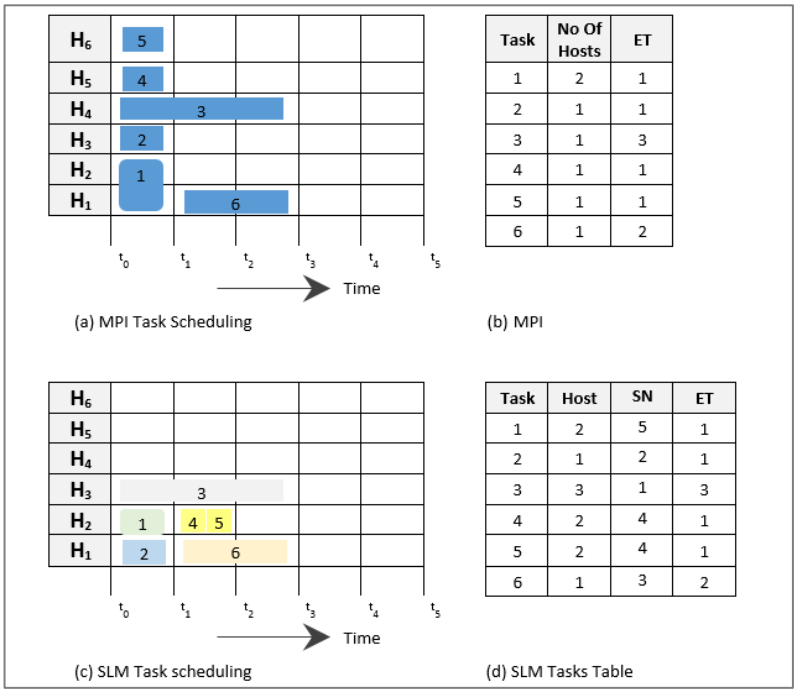
Step 3: The job classification in different queues in the earlier step, minimized the waiting time for the jobs, where the resource allocation here take place according

the SCJ ID of each host and its corresponding resource file which will reduce the transmission load and downgrade the network communication.

Step 4: The WU queued jobs will be executed in serial manner. On the other hand, a selective mechanism will be performed on the RD queue in order to assign jobs using the same resources with homogeneous processing request to the same VM, then space shared scheduling will be performed for multiple jobs concurrency processing. The amount of free PEs allocated previously to the VM will define the number of concurrent processed jobs.

Step 5: Each host energy consumption is shown as output of the algorithm and the processing time will be used to calculate the host's energy consumption after executing a single job.

Step 6: The sum of energy consumed by hosts processing all the jobs at the data centre for a given period will define the datacentre's energy consumption.



Task	No Of Hosts	ET
1	2	1
2	1	1
3	1	3
4	1	1
5	1	1
6	1	2

Task	Host	SN	ET
1	2	5	1
2	1	2	1
3	3	1	3
4	2	4	1
5	2	4	1
6	1	3	2

Figure 19. (a) Message Passing Interface (MPI) scheduling that supports shared memory job processing using MDP library to define the job parallelism, (b) SLM sharing VMs scheduler.

The following Figure 20 and 21 demonstrate the SLM scheduling algorithm and flowchart.

```
Create different types of cloudlet
Submit cloudlet to broker for execution
For each task from task queue
   For each VM from VM list
        Submit task to the VM
      If VM.host.utilization > uppethr
            Migrate VMs until host utilization is < uppethr
      Else
            If VM.host.utilization < lowthr
                Migrate to another host and hibernate current host
      If the read/download counter=0 & write/upload lock is in off mode for the
          current task required resource file
            Submit task for execution
                  If the task type is read/download
                        Increment the read/download counter
                  Else
                        Set write/upload lock on
     Else
                If read/download counter ≠0
                  If task type is read/download
                           Submit task for execution, increment counter
                  Else
                        Submit task to waiting queue
      Else
            Submit task to the waiting queue
After execution of cloudlet
If task has used file for read or download
     Decrement read/download count
Else
     Set write/upload off
IF cloudlet queue is Empty
     Terminate all VM, HOST
```

Figure 20. SLM scheduling algorithm

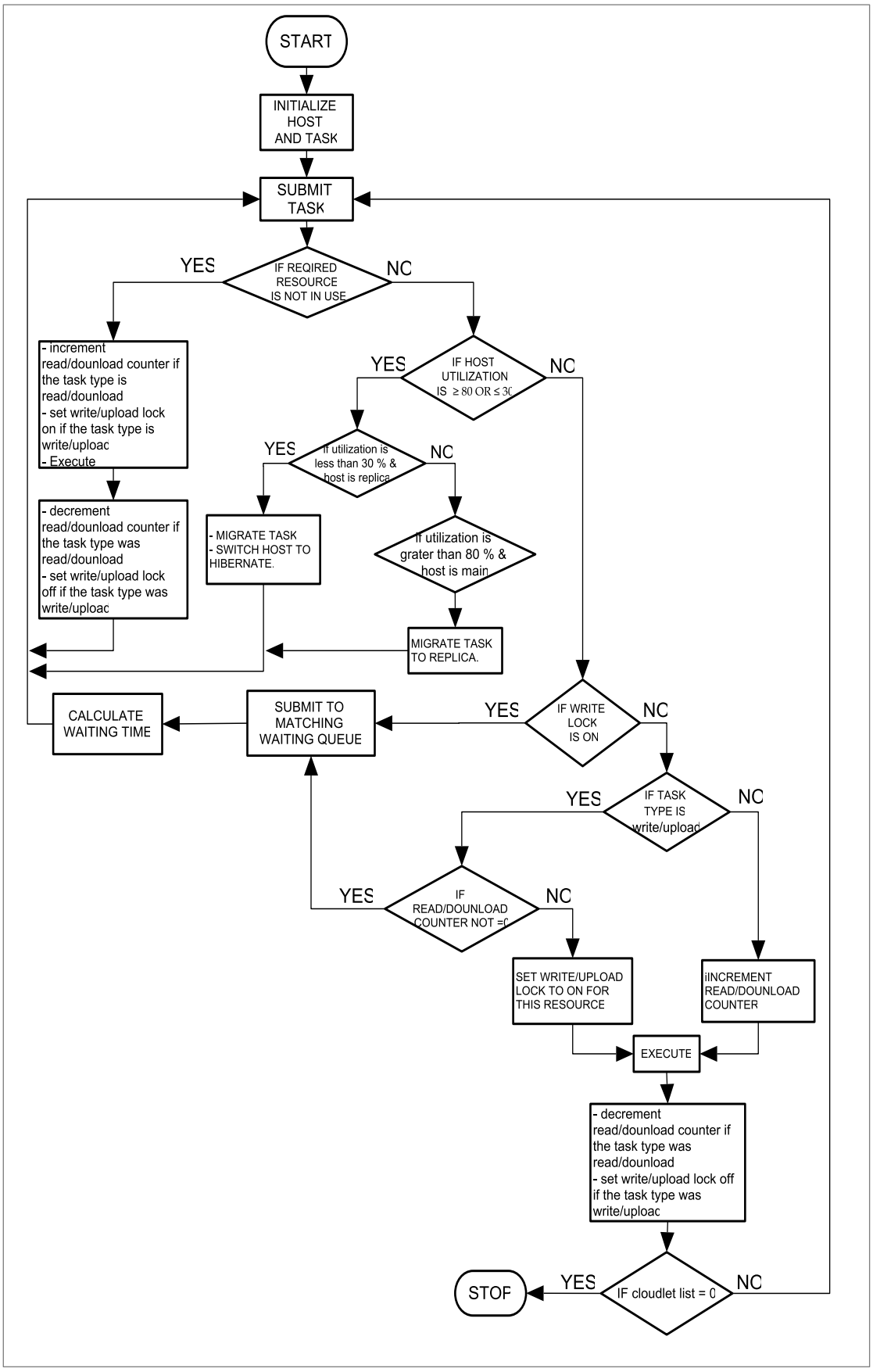


Figure 21. The SLM flow chart

4.6 The Model Parameter

H is the set of all the hosts (physical machines) in the SLM cloud datacentre is denoted by $Host = \{h_1, h_2 \dots h_v\}$ where v is the total number of hosts in the datacentre. The host is identified with its energy consumption denoted by Capacity and host utilization. Each host is virtualized to a set of VMs as VM_i where i s the number of VMs on one physical machine. Inputs: as the available m resource input to the system which should process n independent tasks represented by the set $T = \{T_1, T_2, \dots, T_n\}$, $i = 1, 2, \dots, n$.

In the SLM model, cloudlet parameters such as the memory, processing time, bandwidth etc. are defined according to the job type where writing process is set to be the heaviest size while reading is lightest job size. According to size and requested resource similarity, jobs are given priority for processing concurrently with other jobs.

4.7 User Request Model

In the proposed scheme, a dynamic workload was conducted then all acknowledged jobs to datacentre go through a jobs inspection process in order to be convened before scheduling. Jobs are queued conferring to their size such that the first job assemblage will be processed in individual manner such as the writing and updating, on the other hand the multiple processing jobs guarantees a concurrency processing for similar jobs that have the same data and job type such as reading and downloading. Thus, the significant factor for queuing jobs is their data file required and job size that defines the job type. After submitting jobs to the scheduler, VMs are allocated to each host according to the available resources and two job groups will be categorizes.

The processer executes a set of instruction to each job. User provide information with the submitted job which is a file execution details $f_n = (E_n, J_n, D_n)$ where E_n is the execution time of job, J_n is the job type and D_n is the deadline. Jobs are assumed to be (1) independent to allow simultaneous job processing, (2) arrive based on batch mode where all the jobs characteristics are known in advance, (3) identical job CPU processing usage are less than 50% of the VM's CPU Allowing concurrent VM processing. Two-dimensional array is used to keep the job execution details where i is the job id and j is cloudlet file execution details.

$$TAlloc(i \times j) = \begin{bmatrix} TAlloc_{11} & TAlloc_{12} & TAlloc_{13} & TAlloc_{1j} \\ . & . & . & . \\ . & . & . & . \\ TAlloc_{i1} & TAlloc_{i2} & TAlloc_{i3} & TAlloc_{ij} \end{bmatrix} \quad (8)$$

4.8 SLM Energy Model

The total Power consumption by the datacentre servers is governed by the quality and capacity of resources. The host CPU utilization have a linear relation with the energy consumption where the CPU consume most of the datacentre power according to the power model [88]. The SLM use the power models of HP ProLiant ML110 G4 [89], where the power varies between $0\%\ to\ 100\%$ and the host power usage given by:

$$\mathbb{P}_m = \mathbb{P}_m^{Idle} + (\mathbb{P}_m^{max} - \mathbb{P}_m^{Idle}).\mathbb{u}_m \quad (9)$$

Where P_m is the host power consumption, $\mathbb{u}_m$ is the CPU usage, $\mathbb{P}_m^{max}$ and $\mathbb{P}_m^{Idle}$ are the highest and the lowest power consumption of idle host respectively. The resource utilization can be defined as (10):

$$Resource_u = \frac{\sum_{i=0}^{m} R_{VM_i}}{R_h} \quad (10)$$

Where the R_{VM_i} is the resource utilization of virtual machine i, and $R_h$is the host utilization total power. R_h is the summation of consumed power by all resources (such as the HDD, CPU, Bandwidth, RAM) based on the basic assumption of the SLM model which assume that all used resources have identical specifications and all created hosts have the same characteristics. Let α be the ideal CPU consumption percentage of resource power consumption for the known host giving range between 0 to 1where zero means no use and 1 is 100% usage, then the power is calculated after identifying the resource utilization in (10) as follow:

$$VM_U = \frac{L_i}{C_{MIPS}} \tag{11}$$

$$H_U = \frac{\sum_{i=1}^{n}(VM_{iU} \times VM_C)}{N_C \times C_C} \tag{12}$$

Using (11) and (12) in (9)

$$\mathbb{P}_C = \alpha \times \mathbb{P}_{max}^C + \left(\frac{\sum_{i=0}^{k} R_{vm_i}}{R_h} \times \frac{(1-\alpha)}{100}\right) \times \mathbb{P}_{max}^C \tag{13}$$

$$\mathbb{P}_R = \alpha \times \mathbb{P}_{max}^R + \left(\frac{\sum_{i=0}^{k} M_{vm_i}}{M_h} \times \frac{(1-\alpha)}{100}\right) \times \mathbb{P}_{max}^R \tag{14}$$

$$\mathbb{P}_B = \alpha \times \mathbb{P}_{max}^B + \left(\frac{\sum_{i=0}^{k} B_{vm_i}}{B_h} \times \frac{(1-\alpha)}{100}\right) \times \mathbb{P}_{max}^B \tag{15}$$

$$\mathbb{P}_{HD} = \alpha \times \mathbb{P}_{max}^{HD} + \left(\frac{\sum_{i=0}^{k} HD_{vm_i}}{HD_h} \times \frac{(1-\alpha)}{100}\right) \times \mathbb{P}_{max}^H \tag{16}$$

$$\mathbb{P}_H = \mathbb{P}_C + \mathbb{P}_R + \mathbb{P}_B + \mathbb{P}_{HD} \tag{17}$$

$$\mathbb{P}_D = \sum_{i=1}^{n} \mathbb{P}_H + v \tag{18}$$

$$\mathbb{E}_D = \mathbb{P}_D . \mathcal{T} \tag{19}$$

Equation (18) define the Overall Power consumed by a datacentre, where v represent the value of power produced from the power supply equipment and

cooling system in the cloud datacentre. The following Table 10 present equation 11-19 parameters.

Table 10. Equations Parameters

$\mathbb{P}_{\mathbf{C}}$	CPU Power Consumption
$\mathbf{R}_{\mathbf{vm_i}}$	The i_{th} VM memory (RAM)
$\mathbf{R_h}$	host memory (RAM)
$\mathbb{P}^{\mathbf{C}}_{\mathbf{max}}$	fully utilized CPU power consumption of active host
$\mathbb{P}_{\mathbf{R}}$	RAM Power Consumption
$\mathbf{M}_{\mathbf{vm_i}}$	processing power of VM_i in MIPS
$\mathbf{M_h}$	processing power of host in MIPS
$\mathbb{P}^{\mathbf{R}}_{\mathbf{max}}$	power consumed by RAM when at full utilisation of active host
$\mathbb{P}_{\mathbf{B}}$	Bandwidth Power Consumption
$\mathbf{B}_{\mathbf{vm_i}}$	The i_{th} VM bandwidth
$\mathbb{P}^{\mathbf{B}}_{\mathbf{max}}$	Fully utilized bandwidth power consumption of active host
$\mathbb{P}_{\mathbf{HD}}$	HDD Power Consumption
$\mathbf{HD_h}$	The host HDD
$\mathbb{P}^{\mathbf{H}}_{\mathbf{max}}$	Power consumed by HDD when at full utilisation of active host
$\mathbb{P}_{\mathbf{H}}$	Host Power Consumption
$\mathbb{P}_{\mathbf{D}}$	Datacentre Power Consumption
$\boldsymbol{\alpha}$	Percentage of idle CPU resource consumption
$\mathbf{v}$	Power supply equipment and cooling system power consumption
$\boldsymbol{VM_U}$	Virtual machine utilization
$\boldsymbol{N_C}$	Number of available cores
$\boldsymbol{C_C}$	Core capacity
$\boldsymbol{H_U}$	Host utilization

4.9 Conclusions

The energy efficiency concept has become a major consideration in the cloud computing designing filed and recently CSP seek for innovative ways to maximize their Return on Investment (ROI) while providing the best cloud services for their customers. Cloud datacentres energy consumption levels is one of the main controlling factors of profits degradation and it is one of the main problems facing the CSP beside hosts consolidation and workload prioritization which raised the need for innovative strategies deployment. Since resource management,

dynamic VM consolidation, scheduling techniques and hosts power-saving modes are powerful mechanisms and coalescing these energy efficient strategies will deliver a green cloud environment. This chapter propose an eco-friendly scheduling algorithm for large-scale datacentres where the proposed work deployed the previous factors as a bulk to maximize the system utilization and fulfil the energy optimization goal. The novelty of the proposed technique for managing dynamic workload scheduling base on the ground of providing a synchronized homogeneous jobs processing with in a replicated servers cloud using heuristics hosts virtualization to overcome the resources insufficiency problems with in datacentres and deploying a proposed live migration for system utilization. All the resources used in this model are identical in specifications and all created hosts have the same characteristics. This chapter analyses this novel approach description, problem statement, creation, and execution. Based on a detailed description of the SLM model in section 3.4, a sharing with live migration scheduling Algorithm was proposed to reduce the total energy using four main phases: Virtual machine selection and placement, multiple job concurrent processing, active live migration and idle host shutdown. The proposed work specifies independent user request model for each user which specifies the job type, execution time and deadline preparing the job for simultaneous job processing. The workload arrive based on batch mode where all the jobs characteristics are known in advance. The power model used within the datacentres is presented and the overall power consumption with a detailed energy usage aggregation and constrains for each host. What makes the SLM scheduler different from the previous conducted models is the deployment of innovative server consolidation assortment and provisioning method, live migration based on file locality, and underutilize VMs conceptual detection scheme to have the ultimate energy optimization scheduling model.

CHAPTER 5

SCHEDULER AMPLIFICATIONS AND MODEL ASPECTS CLASSIFICATION

Chapter 5. Scheduler Amplifications and Model Aspects Classification

5.1 Introduction

The cloud receives a dynamic workflow to be served by the available resources of the system based on the specified scheduling and allocating scheme provided by the CSP. This chapter propose the user request model based on significant characteristics to serve administration and queuing jobs by using their data file required and job size that defines the type. After which, the novel energy and performance efficient job scheduling algorithm is presented for the acknowledged dynamic workload by first adopting the best fit decreasing algorithm for VM placement strategy. Then presenting the proposed VM allocation scheme to reach the minimum VM to host allocation cost of the system. The employed synchronize processing of the workload within this scheduler is presented using proposed Guided Backfilling Algorithm (GBA) algorithm to define and partition the CPU architecture and usage among concurrent jobs. The SLM consolidate all the active VMs with in a single host using a local resource manager to govern the VMs live migration based on the SLM defined high and low threshold. The presented scheduler ensures a high level of energy efficiency under the SLA regulations based on the presented performance metrics that helps to deploy, evaluate and assist the existing algorithms such as the allocation scheme, GBA strategy. The SLM increase the resource usage rate and minimize the job make span at the same workload rate is an expected outcome due to the deployment of dual-fold VM layers scheme and regulated using the proposed high and low threshold margin setup.

5.2 SLM model aspects arrangements

The model algorithm deploys a power aware model and the scheduling energy matrix used to maximize the CPU utilization and the VM allocation strategy. The VMs are allocated according to the best underutilized vacant host provided by the scheduler algorithm and this scheme gives the algorithm more control over the host heterogeneity. Different assortments and Provisioning of the allocated VMs is controlled by different presented polices such as VMs layering and load balancing in order to schedule incoming cloudlet to the efficiently. Guided Backfill scheduling scheme is used to fulfil the generated dual-fold vacant slots of the underutilized VMs. The SLM scheduler aims to minimize the overall migration processes and this is accomplished by setting a threshold boundaries strategy to assent in controlling and managing the limits of migration processes.

5.3 SLM job classifier

Another two-dimensional array is used to store the source files allocation details (files needed by the job to finish their execution). The file type F= (f_0, f_1, f_2, ..., f_k) represent the datacentre stored files needed during the execution time for each job, and k is the number of available data file in each host. S is file lock notation where:

$|FL| = \begin{cases} 0, & if\ job\ type\ is\ R\ or\ D \\ 1, & if\ file\ type\ is\ W or\ U \end{cases}$, R and D are read and download job type while W and U are the write and upload processing type.

$$FAlloc(F \times S) = \begin{bmatrix} FAlloc_{11} & FAlloc_{1s} \\ . & . \\ FAlloc_{f1} & FAlloc_{fs} \end{bmatrix} \quad (20)$$

The job classifier works on categorizing incoming workload into two classes of created queues where the first one is the WU queue, which have jobs requesting

for write or upload type of process. On the other hand, the second RD queue stores the applicable jobs to concurrent processing such as reading and downloading jobs with that requests.

One read/download counter and one write/upload lock are defined and used as in Figure 21 and 21, identified as read/download lock and write/upload lock. The write/upload lock is turned on while performing a write or upload process. Once the process is completed, the lock will be switched back to the off mode. This lock is used to prevent any other process from starting if it is on. For the read/download counter, one counter is used to count the read and download operations because a write or upload operation on a resource can start only if no other process is using it; since many read or download tasks can be processed in parallel but not necessarily starting and ending at the same time. The counter will be incremented each time a read or download process starts and decremented each time a read or upload process finished. The write or upload process can't start unless if this counter is zero and that's why we use this counter.

5.4 Host utilization

The overhead for each host is different from the other and a CPU utilization number is used to declare if the VM located in the host have enough resources to serve the user task or not. Since under loaded hosts are consuming more that 50% of the total energy at peak workload [90], the SLM scheduler use a replicated hosts to save the wasted idle hosts energy by maximizing the basic hosts utilization as long as the system can deal with workflow while setting replicated host on idle mode. Which means that at least one of each replicated host should stay on run mode at the system initialization. This consolidation will result with a better resource utilization and optimize the energy consumption in the cloud datacentre. Each host utilization H_U is calculated as:

$$VM_U = \frac{l_i}{c_{MIPS}} \quad (21)$$

$$H_U = \frac{\sum_{i=1}^{n}(VM_{iU} \times VM_C)}{N_C \times C_C} \quad (22)$$

Where $VM_U$is the virtual machine utilization defined as the current VM_i load over the total capacity of the VM in MIPS. $N_C \times C_C$ total host capacity defined by the number of available cores and core capacity respectively while VM_{iU} is the VM_i load and VM_C is the total MIPS assigned for VM_i.

5.5 Dual-fold layered VM Virtualization

Limited system utilization has extensively been a serious concern in commercial datacentres and only up to 50% of the host utilization is reached while operating an ordinary datacentre [92]. To manage the SLM scheduler job allocation and execution process, a proposed dual-fold layered VM is used to reduce identical jobs response time, where the available resources to each allocated VM is distributed before job dispatching phase on two levels in the VM, the (TVML) as the top VM layer and the (LVML) as lower layer of the virtual machine is obtained by virtualizing the existing portion on CPU available to this VM as in Figure 22. The job processing starting time of identical requests is the same denoted by t_i. The processing elements $\mathcal{P}_j$ assigned for each of the similar jobs are disjoint and the job processing time of $J_{identical1}$ and $J_{identical2}$ is reduced by 50% for the RD queue jobs because two jobs are processed at the same time slot on single VM at two different layers. The response time RT is $RT_{J_{identical1}+J_{identical2}} = CT_{J_{identical1}+J_{identical2}} - ST_{J_{identical1}+J_{identical2}}$, where CT and ST are the job completion and submission time in a row. To improve the host utilization, identical job processing is assumed to not fully utilize the processing speed assigned and portioned to the located VM while this assumption will forbid the

non-identical jobs to be processed at the same time on TVM and LVM using predefined VM lock because it will exceed the maximum allowed utilization.

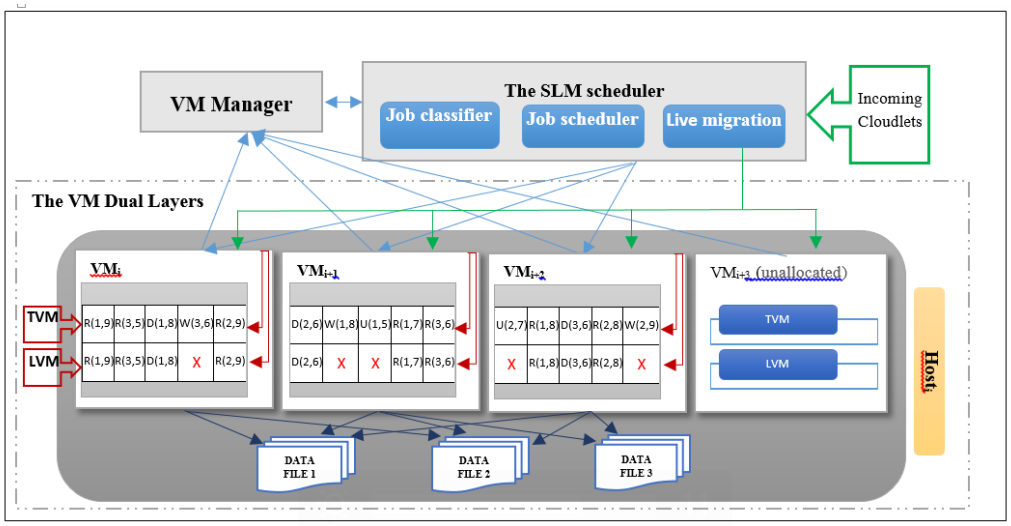


Figure 22. Virtual Machine Dual-fold layer architecture

5.6 Job scheduling scheme

At a specific given time, n number of jobs must be executed either as individuals or paired to be processed based on the job characteristics similarity as specified previously. As a first stage of scheduling, the initial user requests are stored in a batch queue Q_b before performing job classification in the first phase of job administration. Afterwards, jobs are dispatched either to the WU or to the RD queue based on type and data file needed by the job comparison.

5.6.1 Guided Backfilling Algorithm (GBA)

The basic backfill scheduling scheme works on synchronized processing of two jobs at the same time slot by selecting the top job from the designated queue for processing then take the next following job that have the lowest processing time from the same queue to start its execution instantaneously and concurrently with first one [91]. The proposed guided backfilling algorithm (GBA) conducts synchronized job execution by filling

the smallest unused left gaps within the destination VM dual-fold layers by reserving one layer (TVL) of the VM to the first job in RD queue then again reserve the second VM layer (LVM) to the next identical job request from the same queue to be executed concurrently with the (TVM) job by conducting a queue search among arrival jobs with the lowermost processing time to start processing regardless the queue order, which guarantee the ultimate utilization of available resources. To avoid system starvation resulting from long weighting time for large jobs in the traditional backfilling algorithm, the GBA ranks incoming jobs with a predefined job weight for each processing type needed by the job with specific number of processing elements. For immediate idle VM utilization,

SLM jobs are divided based on matching requested data file and type compatibility into two main queues (WU and RD) where the WR queue use random job to VM allocation for heavy Weight users requests such as write and upload data files processed based on arrival time. On the other hand, the Guided backfilling algorithm is used to schedule the jobs with identical characteristics from the RD queue to a single VM by dividing the available resources on two levels of the VM. RD queue take read and download jobs and ranked each identical request to paired in dual-fold VM layer using the GBA algorithm in descending manner where each identical jobs are dispatched together to fulfil the dual-fold VM layered vacant slot to make the best VM utilization by synchronized processing at the same time slot. The total time $\mathcal{T}_{Qwu}$needed by the WU Queue $\mathcal{Q}_{Wu}$ to fully process all the jobs is:

$$\mathcal{T}_{Qwr} = \sum_{K=1}^{J} \left(VM_{req}^{K} \cdot Exc_{t}^{K} \right) \tag{23}$$

Where J is the total number of jobs in the RD queue and VM_{req}^{K} is the number of VMs required for processing the J_{th} job, while Exc_t^K is the execution time for the J_{th} job. On the other hand, the RD queue Q_{RD} jobs will take the following processing time $\mathcal{T}_{Qrd}$ to execute its jobs:

$$\mathcal{T}_{Qrd} = \sum_{r=1}^{m} \left(VM_{req}^{r} \cdot Exc_t^{r}\right) * \sum_{s=1}^{n} \frac{\left(VM_{req}^{s} \cdot Exc_t^{s}\right)}{2} \qquad (24)$$

Where $r + s = total\ number\ of\ jobs\ in\ the\ RD\ queu$ and r is the number of non-identical jobs while s is the number of paired processed jobs using the dual-fold VMs.

Algorithm 1: Guided Backfilling Algorithm (GBA)

Input:
RD_j: jobs waiting in the DR queu,
WU_j: jobs waiting in the WR queu, based on arriaval
T_m: System make span
Output:
Schedule RD_J and WU_J for the dual-fold VM execution
Begin
Step1: dispatch $WU_j \cup RD_j$ to TVM and LVM based resource availability
Sort RD_J in descending order
For each cloudlet in $WU_j \cup RD_j$ & $arr_{RD} < arr_{WU}$ do
    $S_{j1} \leftarrow top\ job\ in\ RD_j$
    For J=0 to J = RD_n do
        IF $J_n = S_j$ then
            $S_{j2} \leftarrow J_n$
        Else
        Calculate S_j waiting time
        If S_{j1} $W_t < 2 * execution\ time$ then
        Repeat
        Else
        $S_{j2} \leftarrow null$
    Dispatch (S_j,S_{j2})
    Sort RD queue in descending order
Else
    $S_{j1} \leftarrow top\ job\ in\ WU_j$
    $S_{j2} = null$
End
Step2 : Schedule $WU_j \cup RD_j$ to TVML based resource availability

---

IF required S_{j1}&S_{j2}PEs available in required VM then
    $TVM_n = S_{j1}$
    $LVM_n = S_{j2}$
    $VM_n \leftarrow$ Busy
    IF $S_{j1} \cup S_{j2} \in RD\ Then$
        $T_m = t_{current} + S_{j1}$execution time+$S_{j2}$ execution time
    Else
        $T_m = t_{current} + S_{j1}$execution time
Elese
    $VM_n \leftarrow$ Idel
If $WU_j \cup RD_j$ are empty then
Break;

5.7 Virtual machine assortment and provisioning

To maximize the VMs utilization within the provisioned cloud datacentre, this thesis proposes a synchronize cloudlets processing using the propose VMs allocation and management scheme. The VMs are allocated to host based on a specific strategy to reach the ultimate number of active hosts to overcome the high energy consumption.

5.7.1 VM allocation policy

In order to reduce the host overload problems, minimize VM migration and reach the highest datacentre productivity, multiple VMs must be allocated properly to run at the same time on a single host. The recently created VM starts the placement request to run on existing hosts after creation based on resources convenience. At run time, best fit decreasing (BFD) algorithm, that works on characterizing and arranging every host that have multiple running VMs by its free memory and make it ready for next execution. The algorithm was adapted because of its good performance results which was the aim for electing and applying this algorithm [69]. If all host have equal power consumption the following steps are deployed for VM allocation:

(1) Arrange existing hosts based on their CPU utilization from the highest to the lowest then allocate to lowest utilized vacant host machine based on necessity. Defining β_i as the possibility of a specific VM to be allocated to $host_i$ satisfying the following constrains to reduce the number of running host as much as possible to accommodate all the allocations needed for VMs:

$$\min H = \sum_{i=1}^{v} \beta_i \qquad (25)$$

Where β_i is a variable that states if a certain host has been designated to start receiving VMs, and $\beta_i = 1$ once the VM is allocated to host h_i, else it is zero and the allocation of single job is assumed to be for only one VM until it is fully processed.

(2) To guarantee the degradation of power levels and that the SLM scheduling strategy will reduce the number of active hosts in the cloud datacentre, the total power consumption of all existing VMs with in a single host will be assumed to be limited by maximum power limit of the used host.

Algorithm 2: VM allocation algorithm

```
Input: Host-List, VM-List
Output: VMalloc
    Arrange Host-list based on Best-Fit-Decreasing
    For each VM in VM-List do
        For each Host in Host-List do
        If Host Host-Utliz < Thr && enough Resources for VM
         Then
           Host←Alloc-VM
           Host-power← estimate.newP
           β_i = 1
        If  β_i = 1  Then
        Add VM to VM-allocated
    Return
```

5.7.2 VMs threshold margin's setup

The SLM scheduler aims to minimize the overall migration processes because the number of executed migrations varies linearly with energy consumption of the host, which means that large number of migrations leads to more power consumption due to the resulting performance degradation of the datacentre. The estimated performance degradation resulting from migration is 10% [93], which is

deducted eventually from the VM total CPU utilization. In order to control and optimize migrations, an accurate threshold must be defined and set as a migration trigger.

5.7.2.1 Upper and lower thresholds

The fixed threshold number is the adopted and used method for defining the SLM scheduler consolidation and migration threshold where if we have a set of numbers $\mathrm{Y}_1, \mathrm{Y}_2, \ldots, \mathrm{Y}_i$ then the Median Absolute Deviation (MAD) is used to define the threshold utilization Up_t as in equation (27).

$$MAD = med_n(|\mathrm{Y}_n - med_m(\mathrm{Y}_m)|) \qquad (26)$$

$$Up_t = 1 - \vartheta . MAD \qquad (27)$$

The Up_t the upper limit of threshold while ϑ is the variant that defines the VM consolidation strength and by experiments it is set to $\vartheta = 0.4$ because using higher percentage will raise the VMs consolidation state which will increase the number of migrations which is undesirable in the energy optimization case. Using the Interquartile measure in (27) and dividing the upper threshold by the median the lower thresholds lo_t of live migration is generated. The over load migration starts if the load is higher than Up_t because the value of the upper utilization threshold Up_t minimize the CPU utilization and optimize the power in the datacentre. Less than low_t is the underutilization state, which evoke life migration for all remaining VMs as first phase then convert the host state to off mode if no more running VMS and $\beta_i = 0$.

5.7.3 VM Space-share policy and optimal Make span

Cloud datacentres utilize tow-scheduling policies in CloudSim for processing submitted jobs, Space Shared scheduling and Time-Shared scheduling policy. Each of these two polices can be performed either on host or VMs level using

both policies. The SLM utilize allocating jobs to host using the space-shared policy to allow multiple VMs to run at the same time sharing the processing units such as the RAM, storage and CPU. Also, the scheduling policy at the VMs level is implemented using space shared policy, where space-share is applied for achieving job concurrency processing at matching job size and resources.

5.8 SLM Live migration

The first step in migration is to go through all existing host at each instance of time t to perform underutilized host detection process, and then elect a normal utilized running host to be the destination to the migrated VMs. Then perform the migration phase where live migration (LM) implies performing jobs transaction between the two hosts with no interruption and the current capacity of the new host is used to estimate the additional expanse of time to file transferring.

Algorithm 3. SLM host consolidation and VM migration

Step-1 for each host H_i
Calculate Power Utilization of H_i
P_i = ResourceUtiization by H_i * constant
// constant depend on host architecture
HP[i] = P_i
Step-2 for i=0 to No_of_host
if HP[i] > Up_t // overloaded host
Overloaded hostlist = HP[i]
if HP[i] < lo_t // underloaded host
Under-loaded hostlist = HP[i]
Step-3 for each $host_i$ in overloaded host
select VM_j from $host_i$ and Migrate VM_j
Step-4 for each $host_i$ in under-loaded host
Migrate all VM from under-loaded host and shutdown that host.
Step-5 Repeat Step – 1 to Step – 4 in every 5 sec.

5.8.1 Live migration Time aspect aggregation

Live migration inversely linear with the bandwidth B and varies linear with the LM_k, which is the available memory needed for this migration residing in the other host. The number of similar jobs J_k copied as a dirty page during the live migration minimizes the total number of re-coping which affect the total time. The Live migration execution time is calculated as:

$$l_d = \frac{NR_{MPIS}}{H_{MPIS}} \tag{28}$$

$$M_k = \frac{LM_k}{B * J_k} \tag{29}$$

$$P_d = \int_{t_0}^{t_0+M_k} l_d(t)dt, \tag{30}$$

Where P_d is the performance degradation cause by VM_d and deducted from the VM total CPU utilization, and $l_d(t)$ is the VM_d CPU utilization as the VM load, NR_{MPIS} is the number of requested Million Instruction Per Second (MPIS) by the VM and H_{MPIS} is the host MIPS. The overload migration starts if the load is higher than upper threshold percentage (Up_t%), on the other hand, less than lower threshold percentage (lo_t%) is the underutilization stat, which evoke life migration as first phase then convert to idle stat to be ready for off mode.

The SLM scheduler aims to minimize the overall migration processes because the number of executed migrations varies linearly with energy consumption of the host.

5.9 Underutilized host detection and consolidation

VM consolidation via migration relies on decreasing the total low utilized active hosts within the datacentre under the threshold boundaries regulation then turn it state to off mode. On the other hand, the VM migration algorithm maintain the

minimum number of performed migrations due to the homogeneous power consumption cost triggered by this procedure. The targeted destination active hosts $\hat{h} < h$ for performing VMs live migration is the host with low power consumption than the highest utilized host with maximum power $p_{h,MAX}$ where the process of increasing the number of free hosts follow the ultimate migration functions:

$$Idel_{max} = \sum_{n=1}^{\hat{h}} p_{idel}.\alpha_n - \sum_{n=1}^{\hat{h}} \sum_{l=1}^{\hat{h}} \sum_{j=1}^{v_n} c_j \chi_{nlj} \quad (31)$$

$$\sum_{n=1}^{\hat{h}} \sum_{j=1}^{v} c_j \chi_{nlj} \leq (p_{l,MAX} - p_{l,present}) \quad (32)$$

$$\sum_{l=1}^{\hat{h}} \sum_{j=1}^{v_n} c_j \chi_{nlj} = v_n.\alpha_n, \forall\, n = 1,..,\hat{h}, l \neq n \quad (33)$$

Where p_{idel}in (31) represent the idle hosts power consumption and α is 1 if host n is not active otherwise it is set to 0, while v is number of VMs in host n and the c_j represent the fixed VM_j power consumption cost produced by migration. The variable χ_{nl} defines the migration source n and the destination l. Equation (32) assures that the power consumption of the destination host l doesn't go over the maximum power of the host after performing the migration before launching the migration phases. The source host n of the migrated VM will continue migrating all its VMs due its low utilization until it is idle preparing for sleep mode following the equation (33).

5.10 Over utilized host detection and load balancing

Once the host reach its maximum utilization threshold applied by the SLM scheduler, the host will be detected dynamically using this algorithm then there will be an activation process for the identical host within the datacentre. This

section works on reducing the number of active hosts in order to increase the efficiency of the load balancer. Since the relation between load balancing and the mean number of the total active hosts is inverse then:

$$Q = \frac{1}{m}\sum_{i=1}^{m} C_i \qquad (34)$$

Where the lower number Q represent a better quality of VM load balancing, C_i is the active hosts number within the datacentre. To see the impact of the over utilized host detection on the VMs load balancing quality, assuming that the SLM perform a migration at each over utilized host detection. Let p be the probability of the first scenario where there will be an activation of one more host which is the replicated host to the current (because the replica was in off mode), while (1-p) represent the probability of the second scenario when migrating VM to an active host that is not fully utilized.

If τ is the time interval between two migrations of VM load balancing resulting from the high utilized host detection algorithm, while estimated number of migrations is $E[\tau]$, then the estimated random number that represent the time during which an additional host will be activated is denoted by D, will be defined as:

$$E[D] = \sum_{i=1}^{m}(m - (i-1)\,E[\tau])p \qquad (35)$$

$$E[D] = \left\lfloor \frac{m}{E[\tau]} \right\rfloor \frac{p}{2}\left(m + m - \left(\left\lfloor \frac{m}{E[\tau]} \right\rfloor - 1\right) E[\tau]\right) \qquad (36)$$

$$\leq \frac{mp}{2}\left(1 + \frac{m}{E[\tau]}\right)$$

From (34):

$$Q = \frac{1}{m}\sum_{i=1}^{m} C_i \qquad (34)$$

$$= \frac{1}{m}\sum_{i=1}^{m} C_1 + \frac{1}{m}\sum_{i=1}^{m}(C_i - C_1)$$

$$= C_1 - \frac{1}{m}\sum_{i=1}^{m}(C_{i-} C_1) \quad (37)$$

C_1 is the number of that denotes the total active hosts number until the end, while Q* which is $\frac{1}{m}\sum_{i=1}^{m}(C_{i-} C_1)$ is the activated host resulting from the VMs load balance process.

Since Q* is promotional to the product of anticipated number additional hosts that will be activated as a result from the VMs load balancing and the activation time then:

$$E[Q^*] = \frac{mp^2}{2E[\tau]}\left(1 + \frac{m}{E[\tau]}\right) \quad (38)$$

The way to improve the VMs load balancing algorithm is minimize the $E[Q^*]$ value (which is the expected number of active host) and from (38) it is clear that the way to minimize $E[Q^*]$ is maximizing $E[\tau]$ which is the VMs migration interval time.

5.11 Conclusions

This research has been focusing on designing a new paradigm for minimizing the energy consumption in cloud computing datacentres. In order to fulfil the PhD research, this thesis focused on saving energy in the cloud computing data centres using the role of managing and scheduling the tasks processing. Existing scheduling techniques have been reviewed and analysed before structuring our novel scheduler. The SLM scheduler requires a specific technique of provisioning and processing of the incoming tasks to improve the resource utilization and power optimization, keeping in mind the balances

between Quality of serves QOS and fairness among the jobs so that the efficiency will be increased. This chapter proposed novel heuristics for dynamic workload scheduling technique with VM consolidation that rely on workload type investigation with maximum hosts CPU utilization. Section 5.3 present the SLM cloud servers' structure that apply hosts replication which provide a mirror servers copies of the main existing servers in the datacentre to manage servers provisioning by hibernating unused server copies to preserve hosts energy. Then the GBA algorithm was proposed as a job administration strategy for sorting and preserving VMs for queued jobs based on defined constrains. Section 5.4 and 5.5 clarifies that the scheduler take advantage of two main cloud characteristics, parallel processing and VMs migration to reduce the bandwidth utilization and transmission time required for source file transactions between different resources to optimize the consumed energy in a datacentre. Section 5.6 present the scheduling policy that is deployed in the SLM model: first the space-shared policy used for allocating jobs to host to maximize the system capability and tolerate running multiple VMs simultaneously using the same RAM, storage and CPU. Additionally, a space shared policy is deployed at the VMs level for synchronized similar job processing. Then a multiple dynamic VM placement and consolidation algorithm was defined using a dual-fold layered VMs triggered using two threshold margins for provisioning the dynamic workflow migration process.

CHAPTER 6

SLM SIMULATION EXECUTION ENVIRONMENT

Chapter 6. SLM Simulation Execution Environment and Results

6.1 Introduction

Scheduling is considered to be a complex procedure regarding to the difficulties to scale and manage the resources sharing among the cloud system applications which raised the need to use simulation tool for analysing the scheduling performance and evaluate its efficiency that would've been hard to accomplished using real cloud system test bed. Experimentation in a real environment is quite a problem due to the high financial cost and the time required to accomplish it and the tests are not repeatable, because many variables can't be controlled with in the test which may affect the results. To evaluate the SLM scheduling algorithm presented in the previous chapter, CloudSim tool kit was used as a large scaled simulation infrastructure for virtualized datacentre that generates an infinite computing resources and components to cloud customers. Various classes are embedded in CloudSim tool to verify and define the datacentre components structuring and functionalities. The host and the VMs classes provide storage computations, power calculation schemes and offer the deployment of new scheduling policies possibility. A simulation experiment was conducted in this chapter using a CloudSim toolkit to elevate the datacentre utilization and manage an efficient job mapping between active VMs. A simulation setup section explains the system hardware and software requirements along with a detailed run scenario and results.

6.2 The scheduler simulation requirements

The proposed scheduling technique following requirements must be fulfilled and satisfied by the needed simulator:

- Support easy and fast scheduler model development and the model description shall be supported by a development tool.
- The possibility to deploy an energy optimization model that adopts automatically with varying cloud structure designs and workload managements schemes.
- A simple and comfortable possibility to design and compose the scheduler algorithm logic.
- Specific technique of provisioning and processing dynamic workload to improve the resource utilization and power optimization, keeping in mind balancing between the Quality of Serves (QOS) and fairness among the jobs to improving the system efficiency.

  Based on the previous SLM scheduling requirements, CloudSim tool kit is used for testing the SLM model because it offers a free testing services for the proposed algorithm and can accredit its accuracy and performance. CloudSim has been developed by the clouds Laboratory of the Computer Science and Software Engineering Department of the University of Melbourne, Australia by Prof. Rajkumar Buyya. As a simulation platform, it enables modelling and simulating the cloud computing systems offering a wide range of application environments that supports both system and behaviour modelling of cloud systems. Datacentres, workload (cloudlets), resource management techniques and servers with virtualized machines are relevant components available to be programmed with java using several classes with conceptual support of many other basic functionalities in the SLM scheduling technique

such as cloudlet queuing, processing and resources allocation [94]. On the other hand, CloudSim offers the deployment of any desired power model for efficient energy environment modelling allowing the evaluation impact of the desired model on energy consumption under different configurations. Many entities are offered by CloudSim platform to set the main datacentre architectural components such as Datacenter.java, cloudlet.java, VM_List.java and FileAttribute.java. Each component uses predefined events as for example the VM entity relate to some events such as VM_Allocat, VM_Reserve and VM_Distroy.

6.3 CloudSim deployment

CloudSim simulation tool provide fixable environment for the SLM development with pack of multiple customized patterns of choices to configure different cloud scenarios to the algorithm resolutions all under specific constrains of performance, provisioning and security measures. The aim of the simulation is to evaluate the efficiency of the algorithm in a typical datacentre scenario, including the SLM scheduler model Assumptions.

6.3.1 CloudSim architectural component

The simulator structured using different customized interconnected classes assembled with in packages and the deployed scheduling policy and cloudlets allocation on any given host is modelled using these abstract classes as in Figure 23.

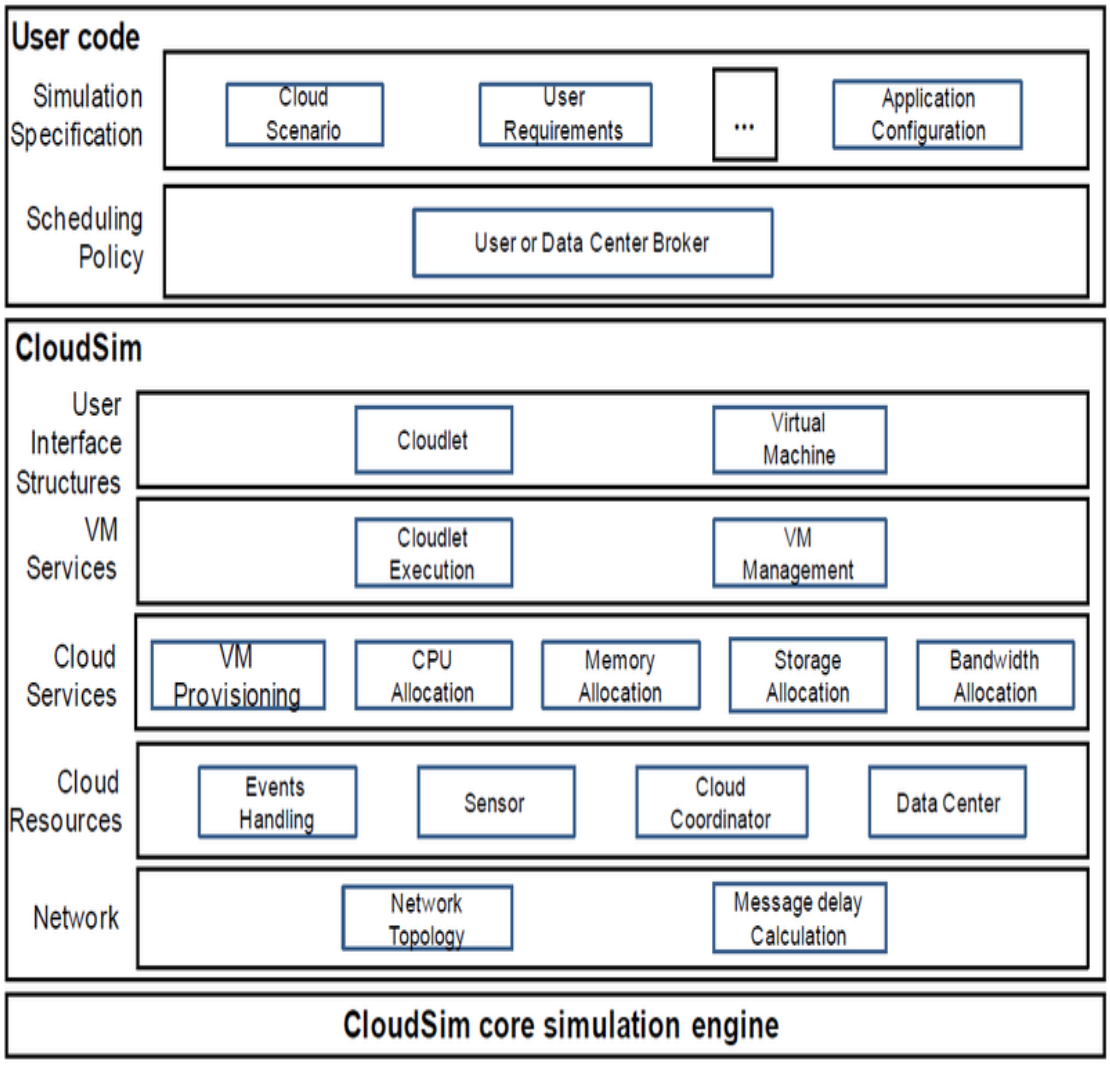


Figure 23. CloudSim Architectural components

6.4 Simulator creation steps

Figure 24 and 25 shows the creation steps of the simulation starting from the first query sent from users that followed by checking he datacentre availability. Then using the cloud information system (CIS) to match jobs with the available VM based on its characteristics for the suitable cloud provider until reaching the VMs destruction final phase at the end of any simulation circle [97].

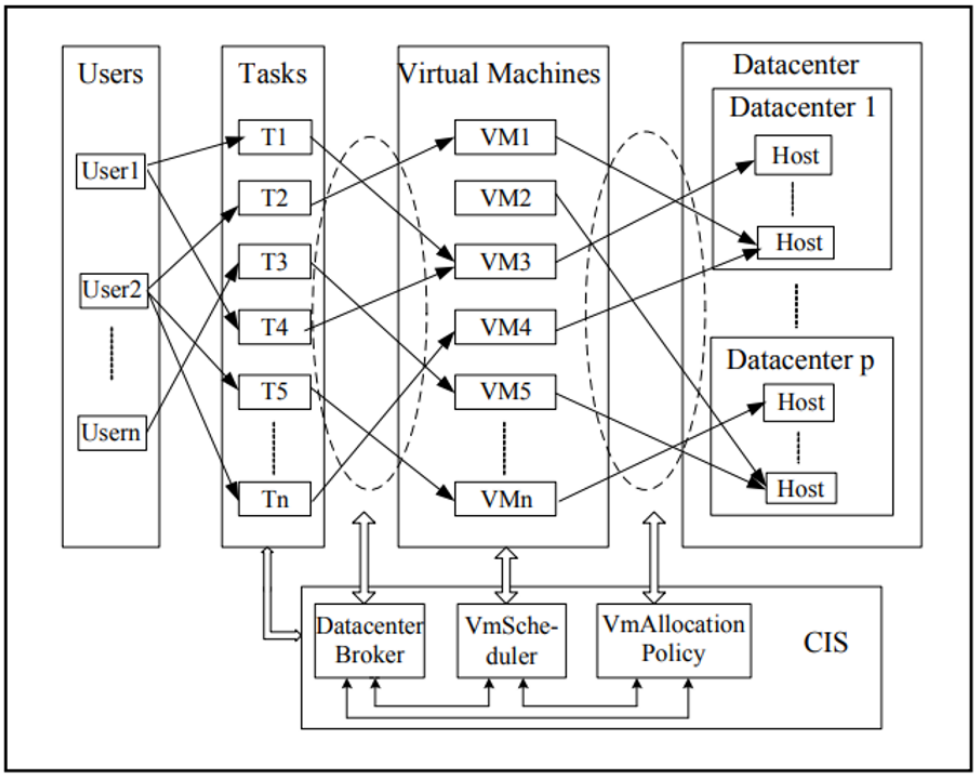


Figure 24. CloudSim simulation cycle

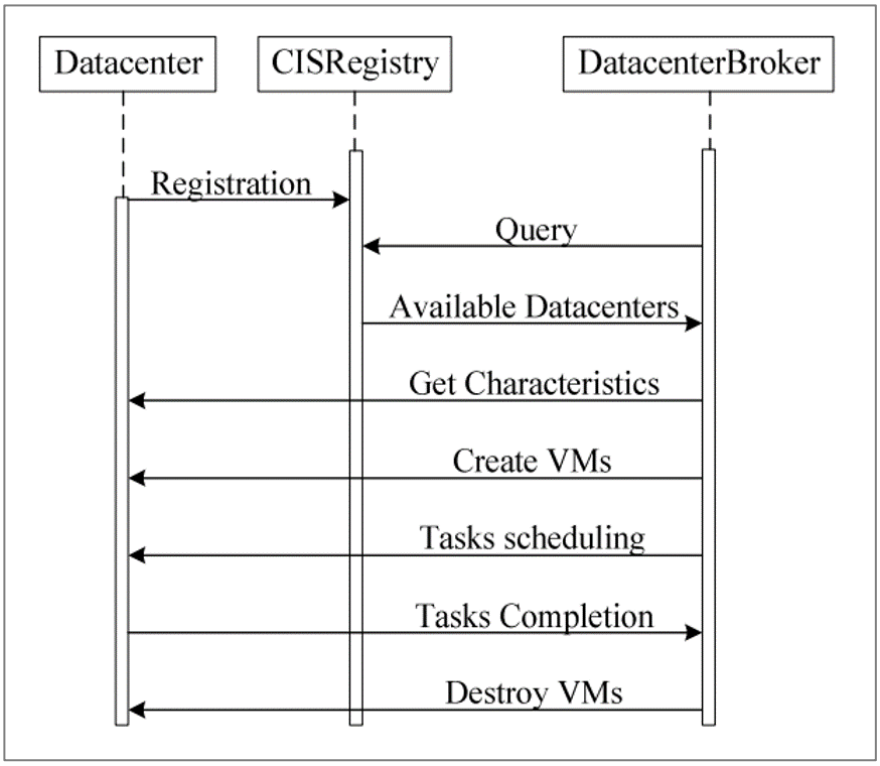


Figure 25. Simulation phases

6.5 CloudSim scheduling model

The SLM Workload scheduling in CloudSim is offered and managed via the structure of the following three basic scheduling policy classes based on the required resources availability.

6.5.1 Vm.Allocation.Policy Class

Allocating created VMs to the most suitable host creates challenge with in the datacentre and Vm.Allocation.Policy class is used to apply the allocation method and take care of the matching process based on a predefined method under the availability of the required resources condition. VM scheduling use two levels of implementations to specify the required scheduling policy where the first level is accomplished on the host while the other level is based on the VM component.

6.5.2 Cloudlet.Scheduler.SpaceShared Class

The (Cloudlet.Scheduler.SpaceShared) class is the controlling class of the VMs scheduling scheme ,where the scheduling strategy reposition each cloudlet to a specified virtual machine without violating any concurrency control rule. The rest of upcoming delivered cloudlets are held in a waiting list in case of no available PEs. A (Cloudlet. Submit) method is responsible for validating the user conditions regarding the cloudlet execution under the availability of the host resources.

6.6 The SLM Model Simulation and Assumptions

The CloudSim provide the basic IaaS cloud environment characteristics such as virtualization and support both SaaS and PaaS job scheduling algorithms integration, deployment and testing using a Java framework. The virtualization concept of the cloud datacentre model is presented using stack of CloudSim simulation layers that provide functional virtualize cloud activities starting from user's task generation and ending with the datacentre utilities destruction.

Devoted hosts management of memory, storage and allocation process and execution progression with continues provisioning of the datacentre stat is offered by the CloudSim simulation layers. In this simulation we implied the following assumptions in the predefined java classes:

1. All PEs under the same Machine have the same MIPS rating.
2. All host have similar characteristics such as CPU, storage.
3. Dynamic workload policy is deployed, and independent jobs are assumed.
4. The network delay is not considered, and communication cost is ignored.
5. The space shared management policy is used as a datacentre characteristic, which means that jobs use space slots instead of share time.
6. A random data set is generated to offer different job parameter on random base each new simulation run.
7. The initial number of cloudlets simulation attempts are 500 cloudlets of the system which would represent the datacentre workload.
8. Cloudlets are classified into 4 different types which define the expected core and RAM requirements as well as the processing times of cloudlet and each type have different length.

6.7 The execution model

All the elements of the simulation framework, which are relevant for the evaluation, were defined precisely, and to have a meaningful framework all the following parameters were identified:

- Datacentre
- Servers
- Hosts
- Cloudlets
- Processing type

- Resources
- Processing time
- Power Consumption
- Backup servers

The proposed SLM scheduler algorithm execution phases are discussed in the following:

1. Users send number of jobs for the datacentre.
2. The scheduler starts processing jobs in case if the needed server is zero utilized.
3. Queuing is applied to incoming tasks if their needed file is in use.
4. For each queue, job check-up procedure is performed by the scheduler for each of the queued job, and in case any of the jobs need to perform a similar type of process on the available resource, then it will be given the permission to start processing instantly with the previous job without waiting in the queue if the designated host is not overloaded.
5. Migrate job to the replica underutilized servers if the needed main server is overloaded.
6. Migrate job from the under loaded replica server to the main server when it becomes not overloaded then set the replica back to off mode.
7. Apply queuing on the tasks with minimum waiting time in case of resources being unavailable.

6.8 Experiment setup

A random generated test data was used in this simulation runs with a fluctuating workload length as an input to the algorithm with different job types. All jobs have a predefined execution deadline and identical characteristics host and VMs while the data communication cost was

neglected. As shown in Table 10 and 11, the cloud datacentre proposes six physical servers, each with maximum number of 10 virtual machines, there are several characteristics for all the VMs and Hosts where each is given the exact same specific characteristics as processing capability, storage, and bandwidth. For the hardware setup, each host have 8 GB RAM, 2 TB storage, four CPUs that have a capacity power of 10,000 MIPS and the datacentre is 16 DVS-enabled processors [100].

Table 10. SLM host configuration

Number of Hosts	**6**
Host Types	1
Host Storage (TB)	2
RAM (GB)	8
Scheduling Strategy	Space shared
Processing Power (MIPS)	10,000

Table 11. Virtual machine configuration

CPU (processing element)	**1**
Number of VMs	60 (10 to each host)
VMs Types	1
RAM (MB)	800
Scheduling Strategy	Space shared
Processing Power (MIPS)	1000

Each host have three different source files that are needed by the cloud users, and since the datacentre have six hosts we applied the replica method on half of the hosts which means that three hosts have 9 different sources while the other half of hosts have the exact copy of the 9 files. The random workload was generated using the RAND utility in CloudSim tool kit and the number of cloudlets were wide-ranging from 100 to 500.

6.9 Simulation Scenario

Each user in the cloud is connected to a Broker entity. The task, which is called a cloudlet, of a user is first submitted to its broker and the broker schedules the tasks based on a scheduling policy. The broker dynamically before scheduling the task gets a table of all the available resources and hosts, they are located in. The simulated scenario includes the generation of cloudlets, with 500 jobs, which are passed to the scheduler to decide on admission control and prioritization. The simulation environment had been set up and executed following the next steps scenario:

1. Host and VMs creation phase
2. Generating cloudlets:
   - Setting up task types to Reading, writing, uploading, and downloading with task length 10000, 40000, 80000,200000.
   - Generating randomly several cloudlets and setting their configuration (type, size, number).
3. Primary admission control: Datacentre broker submit cloudlets to a class called cloudlet submit to manage transferring it to the proper VM without violating any concurrency control rule and while there are enough free processing elements (PEs). This phase was controlled using two different queues.
4. Application-specific analysis and ranking: In the simulation, each task is described by cloudlet file execution details (file VM number; file host number, file number, task type). Cloudlets are assigned using the SLM scheduling policy to the first under loaded VM layer that has the corresponding needed file under the regulation of the dual-fold VMs.

Cloudlet type represents the type of the task process which could be either reading, downloading, writing or uploading. The code for reading =0, download=1, writing=2 and upload=3. The SLM scheduler will allow multiple job processing at the same time only if the job type are read or download but it will not permit that in case if the current task type that is under processing is either write or upload due to the over writing problem. Hence, SLM scheduler will provide concurrency control.

5. Quality of services and serves level agreement control: Providing a reliable Quality of services is one of the most important and essential elements in the cloud computing environments. Quality of services is defined by the serves level agreement which guarantees a certain level of service to the user [101]. In our SLM simulation model an effective continuous observation class was added to the simulation to detect changes of the QoS and the SLA conditions and prevent any violation to those agreements.
6. Hosts, VMs and datacentre destruction.

6.10 Experimental results

This section starts with experiment results of running simulation to all the proposed algorithms in section 4.5 and 4.6 based on the scenario presented in the previous chapter using the proposed SLM scheduler and the VM consolidation model detailed in chapter 3 with the help of a conventional model simulation. The proposed approach was tested via simulations using the CloudSim toolkit with one datacentre and six host where each host was virtualized to ten VMs with total of sixty VMs. The job workflow has four types of cloudlet which are reading, uploading, downloading and writing. Processing cost of task

reading, uploading, downloading and writing are in increasing order. The conventional method datacentre model conducts dynamic cloudlet processing mechanism. However, the SLM simulation model allow multiple job management under specific constrains at same time. The following table 12 and 13 demonstrate a sample test of dynamic workload processing simulation run for 500 cloudlets varying between four different jobs Types, where the resulting execution time of each cloudlet is presented as an output using the SLM scheduler.

Table 12. 1-10 cloudlets SLM Simulation Run Results

Cloudlet_ID	Status	Cloudlet_Type	Source ID	VM ID	Execution_Time	Start_Time	Finish_Time
0	SUCCESS	Reading	2	0	10	0.1	10.1
1	SUCCESS	writing	3	3	40	2.1	42.1
2	SUCCESS	uploading	0	6	80	4.2	84.2
3	SUCCESS	downloading	1	8	200	6.4	206.4
4	SUCCESS	uploading	1	10	80	7.8	87.8
5	SUCCESS	writing	3	9	40	10.1	50.1
6	SUCCESS	writing	0	14	80	8.5	88.5
7	SUCCESS	downloading	3	17	150	9.2	159.2
8	SUCCESS	Reading	3	19	30	12.6	42.6
9	SUCCESS	writing	2	20	40	14.1	54.1

Table 13. Energy Consumption with number of Cloudlets

No. of Cloudlet	Energy Consumption in the conventional method	Energy Consumption in the SLM model
100	0.16	0.08
200	0.22	0.13
300	0.30	0.17
400	0.34	0.22
500	0.40	0.27

The resulting power consumptions as shown in Figure 29 were found to be between 0.08 and 0.27 kilo watts per hour for a randomly generated workload with maximum of 500 cloudlets within the cloud datacentre indicating no losses or rejected jobs. The performance of the used consolidation algorithm result with a few migrations based on the incoming workload in order to release and shut down some hosts. The assessment was performed on active VMs with in the

simulation duration resulting with 4 migration process were performed for 100 cloudlets, 7 for 200 cloudlets, 13 for 300 cloudlets, 18 for 400 cloudlets and 20 for 500 cloudlets.

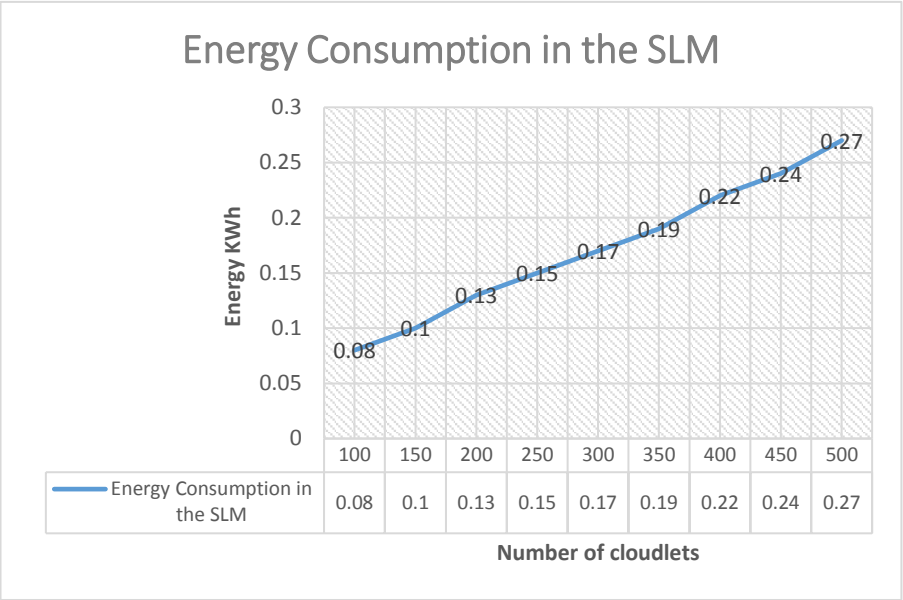


Figure 26. SLM Energy consumptions levels

Make span and migration number where the two factors that where presented and recorded with the energy consumption levels as an output of this simulation to demonstrate the effectiveness of the SLM scheduler performance as shown in Figure 27.

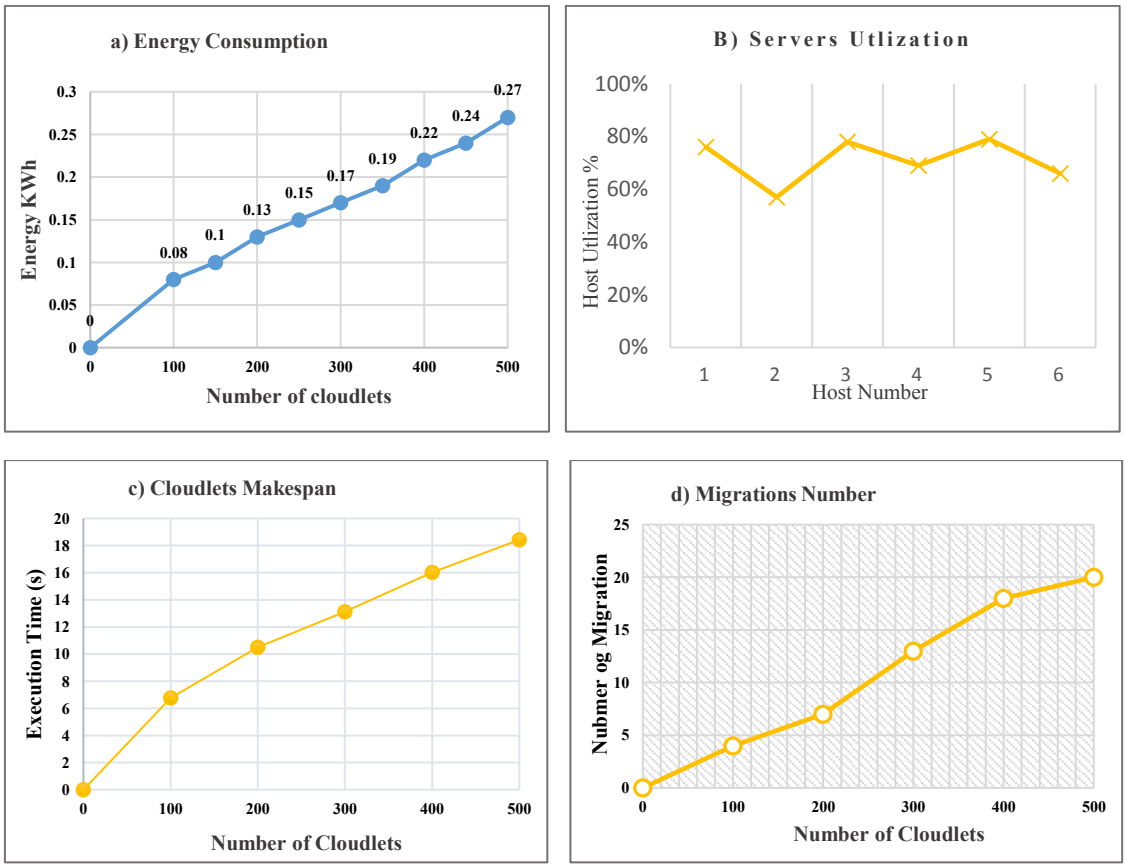


Figure27 . Energy, Servers utilization, Make span and Migration number of the SLM scheduler

6.11 Conclusions

Several simulators can be used as test beds to fulfil the challenging cloud system performance evaluation. Electing the best simulator is challenging but the decision could be decisive once the simulator is reviewed and selected based on the needed outcomes. This chapter introduce CloudSim as the SLM simulation environment that was used to test the novel job scheduling algorithms The deployment of CloudSim tool to simulate the SLM scheduling model was based on its ability to provide a conceptual virtualized layers of the cloud infrastructure which was not presented in other simulators, above that its capability of supporting the ultimate energy modelling for green cloud computing environment through predesigned packages for providing the supplementary objects for

structuring and managing scheduling strategies. The proposed approach simulation is conducted using a single structure datacentre, six host and sixty VMs. The dynamic workflow has four types of cloudlet varying in type between reading, uploading, downloading and writing with different processing cost.

Multiple random workload experiments are conducted to test and evaluate the performance of the proposed scheduler regarding energy consumption, make span and number of conducted migrations. The experimental results and estimated energy, make span and migration number was plotted according to the dynamic workload of 500 cloudlets. The following chapter provide the SLM evaluation in regards to energy consumption, make span and migration numbers compared to Ant Colony Optimization and Practical Swarm Optimization scheduling techniques.

CHAPTER 7

MODEL EVALUATION AND DISCUSSIONS

Chapter 7. Model Evaluation and Discussions

7.1 Introduction

The previously conducted simulation and presented in chapter 6 used a dynamic workload to model and simulate a cloud datacentre deploying the proposed SLM scheduler under different system regulations and assumptions. The SLM adapt a dynamic workflow management and established a queuing strategy for a regulating and monitoring the relationships between cloudlets types and VMs processing reservations to active parallel processing. The proposed model performance was experimented in the previous chapter a gains conventional method in the filed while this chapter provide experimental results evaluation, comparison and analyses against two existing meta-heuristic optimization techniques: Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO) scheduling schemes which were defined in section 2.8. Experimental results confirm that this approach guarantees performance and resource efficiency for cloud datacentres services.

7.2 Evaluation of the proposed SLM scheduler

This section evaluates the proposed SLM scheduler, the results and evaluation use the energy consumption levels and make span in order to prove the functionality of the proposed SLM to be compared with the Ant colony optimization (ACO) and particle swarm optimization (PSO) because both algorithms gave the best results in energy consumption optimization. As PSO manage to efficiently determine the best solution with less computational cost, ACO dos not apply a job scheduler to allocate resources under time constrains [102][103]. The numerical evaluation of the SLM is used to calculate the

percentage of energy consumption produced by the cloud datacentre after combining the VMs consolidation with the task scheduling algorithm.

7.2.1 Performance evaluation matrix

Different experiments are conducted several times on a dynamic workload of 500 cloudlets then the SLM scheduler model performance was analysed based on the experimental results. To further verify the performance of the proposed SLM scheduler and illustrate its effectiveness, the SLM performance test results was benchmarked ACO and PSO scheduling strategies in regard of four main performance matrix elements: energy consumption levels, migration size, server utilization and make span.

7.2.2 Servers utilization

To evaluate the SLM scheduler performance six hosts were used to execute 500 cloudlets within the datacentre, and the maximum utilization reached by each server is considered as one of the main optimization factors of the energy levels. The SLM increased the hosts utilization compared to ACO and PSO where Figure 28 and Table 14 results shows that the utilization of server one to six fluctuated between 57% to 78% On the other hand ACO and PSO maximum utilization fluctuated between 10%-89% and 25%-75% respectively, which indicate a high percentage of underutilized host within the system and inefficient resource's usage.

On the other hand conventional method preserve a dynamic workload distribution equally among host which resulted with narrow host's utilization that led to power loss due to resource inefficiency. The SLM demonstrates a high server utilization compared to ACO and PSO schedulers due to the used allocation strategy and the GBA algorithm which managed the CPU utilization of host from 1 to 6 within

the range of the upper and lower threshold margins to assure the server utilization while maintaining energy optimization.

Table 14. Host utilization in SLM, ACO and PSO

Host Number	Servers Utilization			
	Conventional Method	ACO	PSO	SLM
1	55%	53%	49%	76%
2	48%	32%	25%	57%
3	60%	36%	53%	78%
4	57%	10%	43%	69%
5	58%	67%	75%	79%
6	46%	89%	39%	66%

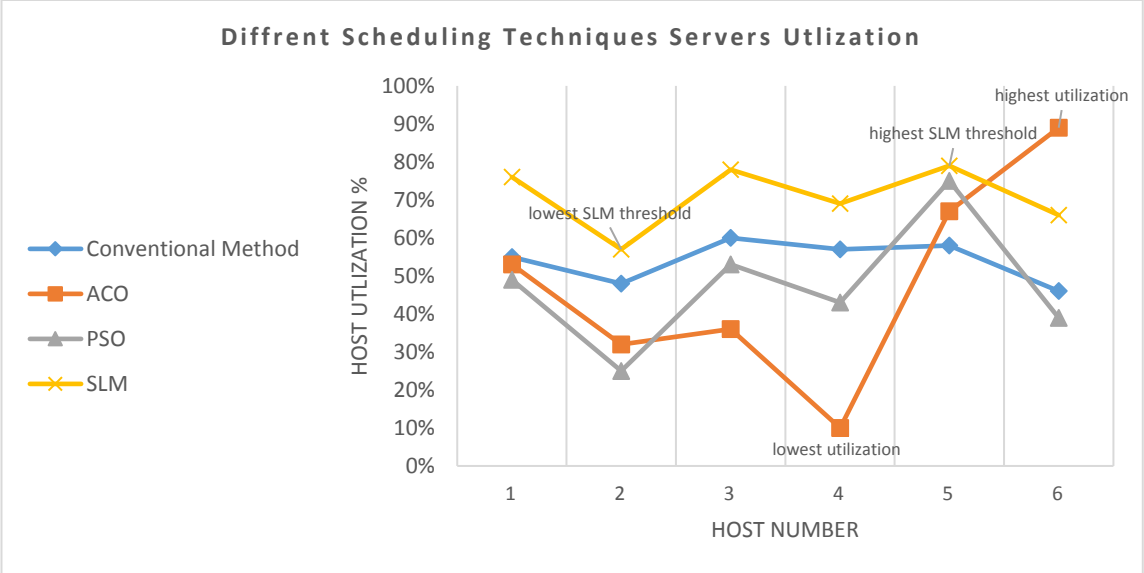


Figure 28. Server's utilization comparison in SLM, Conventional, ACO and PSO

The lowest and the highest levels of utilization was reached by the ACO at 10% and 89% but this doesn't mean that the ACO reached the best hosts utilization levels because at these numbers the host either will maximize its energy use at 89% or it will waste energy at 10% of utilization as underutilized host. On the other hand, the SLM maintained the host utilization levels between the upper threshold

79% and the lower threshold 57% which is the best host utilization for preserving the host energy according to the energy reading in figure 27.

7.2.3 Number of executed live migrations

One of the resolutions behind the proposed scheduling strategy was minimizing the number of accomplished live migration iterations based on the defined threshold The simulation of SLM further verify the performance of the proposed SLM scheduler. Hence, the SLM model provide job admission control and provide a parallel processing feature with migration. Table 15 shows the distinction number of VM migrations obtained by various techniques where the total number of VM migrations is decreased in regard to the growing number of jobs due to the high and low load utilization detection algorithm used in the SLM which manged to minimized the number of performed migrations. The migration intervals and time duration is maximized and the number of migration processes are minimized as the dynamic workload grow, which means that the high load detection algorithm manged to reduce the migrations iteration. Hence, at the beginning of the workload arrival from 100 jobs to 300 the number of performed migration is higher than its progression between 400 and 500.

Figure 29 results shows that proposed SLM scheduler has better performance in regard of reducing migrations when compared with the conventional, PSO and ACO. Precisely, and according to the migration Table 15 the number has reduced by 11.3% compared to PSO while minimizing migrations approximately 21% compared to the average of migration number in ACO algorithm.

Table 15. Different scheduler's migrations number

No. of Cloudlet	Number of VMs Migration			
	Conventional Method	ACO	PSO	SLM
100	7	6	5	4
200	11	10	9	7
300	15	14	14	13
400	21	20	19	18
500	25	25	22	20

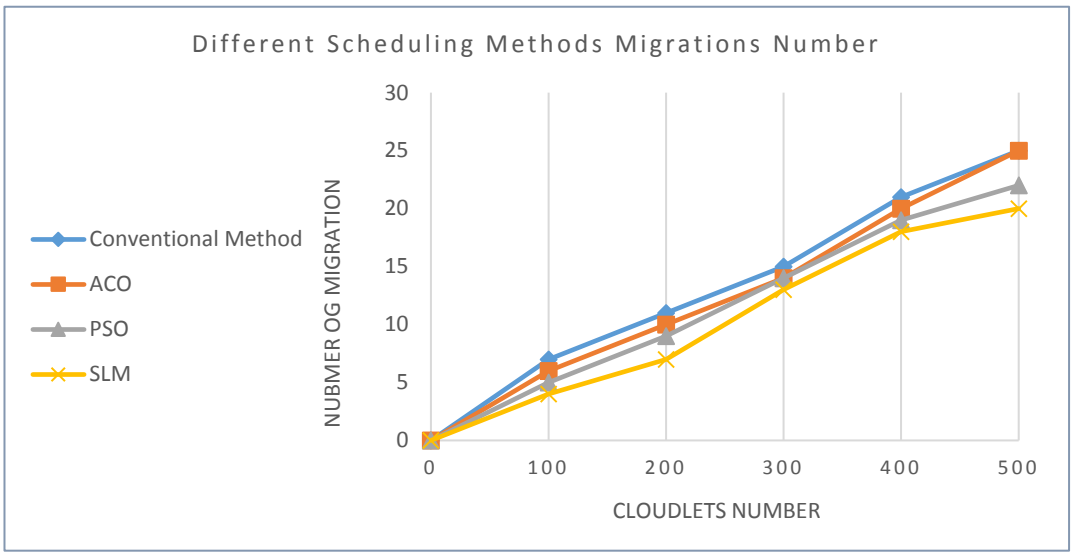


Figure 29. Migration levels comparison

7.2.4 Impact on Energy Usage

In this section, the energy consumptions rates of the SLM algorithm are analyzed based on the results of the previously conducted experiment. The SLM scheduler avoided the performance degradation as shown in Figure 30 by applying concurrent and parallel processing of identical incoming jobs within dual fold layered VMs architecture which led to reach the ultimate CPU, low numbers of migrations and energy consumption.

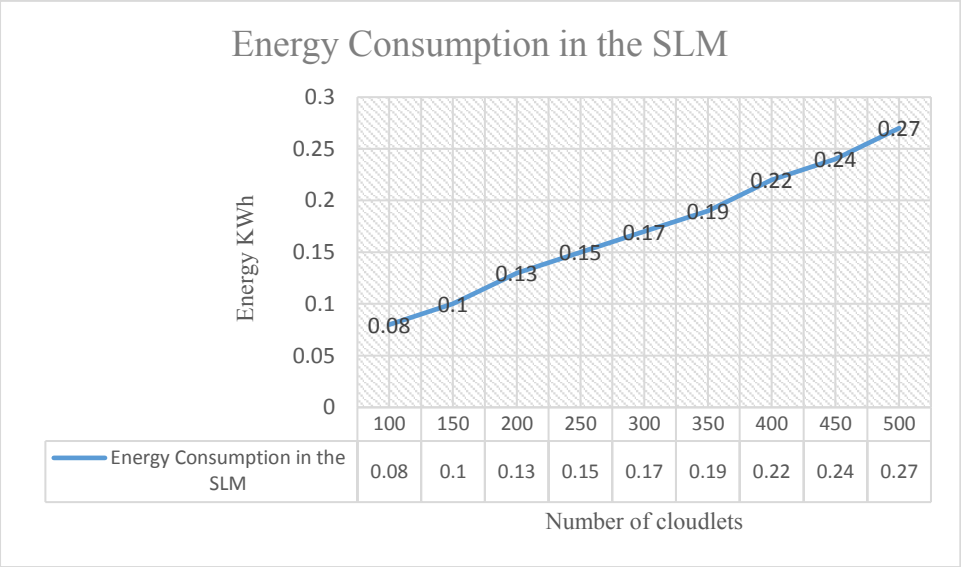


Figure 30. Energy levels using the SLM algorithm

The simulation results of the SLM model in Table 16 and Figure 31 show the improvement in reducing energy consumption. The results shows improvements in reducing Energy Consumption using the SLM model and by comparing it with the conventional method, PSO and ACO, For instance, a batch of 100 cloudlets in the cloud datacentre consumed 0.08 kWh for processing using the SLM algorithm while the same number of cloudlets required 0.13 and 0.12 in ACO and PSO respectively. Through many simulation experiments runs, this algorithm provided a good performance and energy optimization scheduling abilities and given Table 17, the SLM shows clear distinctions for Energy Consumption with varying number of cloudlets. Average improvement of energy consumption of SLM over Conventional method, ACO and PSO is 35.11%, 23% and 18.1% respectively. As per elevating host utilization along combined with the migration optimization in the previous two sections, energy consumption is directly proportional to migration and host utilization hence this system will also reduce the total energy during execution.

Table 16. Energy Consumption by SLM, ACO and PSO with varying No. of cloudlets

Number of Cloudlet	Energy Consumption (kwh)			
	Conventional Method	ACO	PSO	SLM
100	0.16	0.13	0.12	0.08
150	0.19	0.15	0.14	0.10
200	0.22	0.18	0.17	0.13
250	0.28	0.21	0.19	0.15
300	0.30	0.24	0.22	0.17
350	0.32	0.26	0.23	0.19
400	0.34	0.28	0.27	0.22
450	0.38	0.31	0.29	0.24
500	0.4	0.33	0.31	0.27

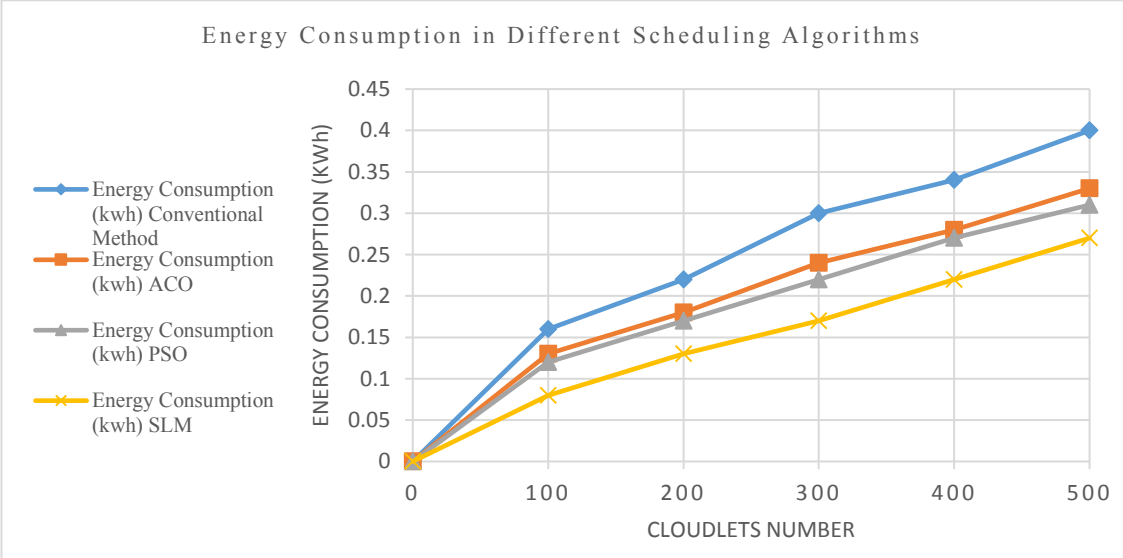


Figure 31. Energy consumptions levels in SLM scheduler compare to ACO and PSO

7.2.5 Make span time of SLM scheduler

Regarding make span, as shown in Table 17, the SLM method manged to maintain a steady make span control and optimize the execution time of jobs with the same performance as the job number grows from 100 to 500 compared to ACO and PSO schedulers. The SLM execution scheme of multiple homogenous jobs concurrently minimized the needed time for synchronized jobs processing while other scheduler doesn't hence the make span of SLM is reduced among other scheduling method.

In fact, ACO has the highest make span among all tested schedulers. The reason is that ACO method reduce all of its computational problems to achieve the best paths for resources but it still cannot result with lower job processing time compared with PSO, because PSO considered to have a better optimization method specifically at low-intensive workload which result with less computational cost. With the SLM scheduler, the VM consolidation and the dual fold layers VMs allocation are effective for improving and minimize the processing cost and the overall make span of the system specifically at high intensive workload.

Table 17. Make span with varying number of Cloudlets

Number of Cloudlet	Make span Time			
	Conventional Method	ACO	PSO	SLM
100	9.06	8.76	8.05	6.78
200	14.5	13.34	12.09	10.5
300	17.53	15.67	14.56	13.13
400	19.45	17.5	17.67	16.02
500	22.09	23.29	19.34	18.43

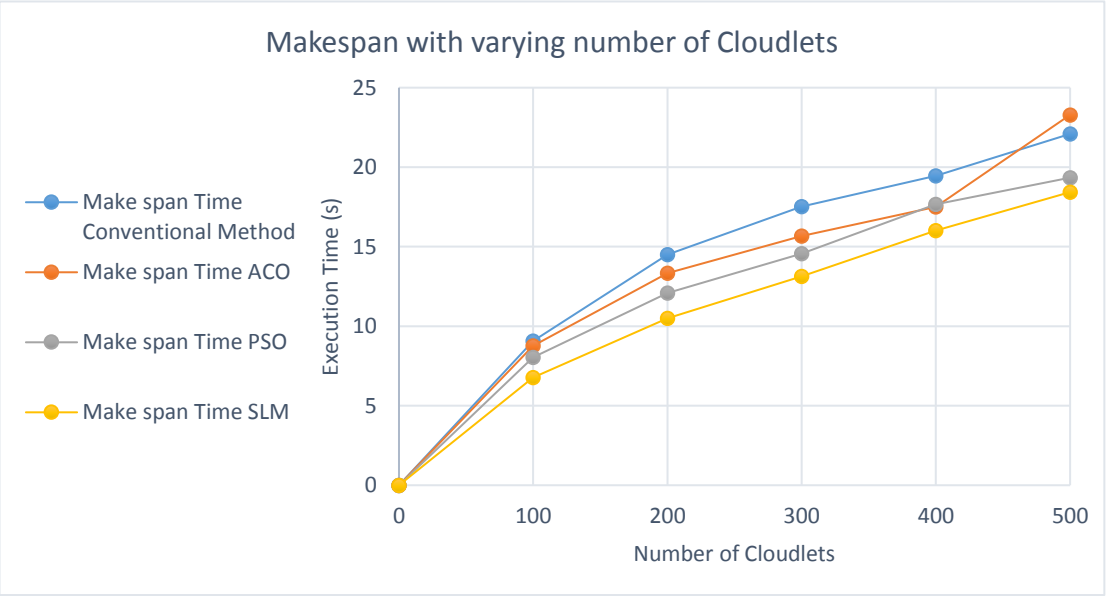


Figure 32. Improvement in Make span of proposed SLM against PSO and ACO

Figure 32 shows the make span of different scheduling algorithms and proposed method SLM clearly outperforms with the other method. Overall make span improvement in SLM over Conventional method, ACO and PSO is near about 19%, 15% and 10% respectively.

7.3 Effectiveness of the SLM Scheduling Technique and Its Impact on the Power Consumption

The SLM simulation results outperforms the ACO and the PSO methods s on all four benchmarks which effected the total energy improvements within the datacentre and led to reducing Energy Consumption because each factor have a significant impression on the power usage of every physical machine. By comparing the SLM with the ACO and PSO approaches, the simulation results and readings shows a clear distinction for energy Consumption. The SLM model require less power than both schedulers with varying number of cloudlets. Average improvement of energy consumption of the SLM model over the ACO and PSO is 23% and 18.1%. As per cloud computing cost calculation, energy consumption is directly proportional to total cost hence this system will also reduce the total cost during execution. Low Energy Consumption also reduces the emission of Carbon Dioxide (CO_2), making it eco-friendlier.

7.4 Conclusions

In the perception of effective cloud scheduling and forward cloud computing optimization algorithms, this work presents an energy efficient scheduling technique for dynamic workload systems. A sharing with live migration-based scheduler was introduced in this thesis and the proposed approach was evaluated on various standard benchmarks. Experimental results show that proposed approach yields significant energy savings and make span optimization

as result of applying concurrent job allocation and execution, VMs consolidation and migration according to a dynamic thresholds margins. Regarding the system make span, the results showed a minimization approximated to 15% and 10% in the datacentre make span compared to ACO and PSO scheduling algorithm respectively. On the other hand, the impact of SLM scheduler deployment over the power consumption was 23% and 18.1% levels reduction compared to ACO and PSO. On the other hand the six servers achieved the highest utilization rate among the other two scheduler's servers. This model successfully managed to drive a cloudlets provisioning service and scheduling technique which improves the resources efficiency and maximize the system ability to improve the energy consumption and achieved the performance target.

CHAPTER 8

CONCLUSION AND FUTURE WORK

Chapter8. Conclusion and Future Work

8.1 Introduction

The idea of providing computing resources over the Internet will extremely thrive as the technology advances. Hence, Cloud datacentres will grow and collect a greater portion of the world's computing resources, which will make the energy efficient administration of datacentre a vital problem to the operating costs and CO2 emissions to the environment. The recent researches struggle with the energy consumptions rates in cloud computing environment and that was the result of the escalading energy-hunger in the cloud computing datacentres and the escalading rates of the CO2 emissions. Reducing the energy consumption of the cloud computing datacentres became a research challenge in the recent years. The goal of energy efficient scheduling in cloud computing is to reduce cost and computing infrastructure. This chapter summarizes the research work on energy-efficient scheduling in Cloud datacentres presented in this thesis and highlights the main findings. It also discusses research limitations in the area and outlines several future research guidelines.

8.2 Achievements of the Research

This research was dedicated to the development of a novel approach for functional testing scheduling technique in cloud computing datacentres and establish a useful workflow management system by providing the flowing achievements and research accomplishments:

- The general cloud structure was analysed presenting the main cloud elements and essential cloud computing regulation and agreements.

- Different service structure concepts were discussed with an overview of the cloud deployment models and basic types (see chapter 2). Then cloud resources management strategies were presented focusing on the scheduling element and its heuristics classifications.
- The fields of technologies for scheduling execution and provisioning were analysed. Individual advantages and disadvantages of each technique were shown, and research projects related to the topic were discussed.
- A new energy aware based scheduling policy and VMs layering and consolidation algorithms are proposed (see chapter 3). The cloud job scheduling algorithm allocate resources based on energy optimization methods called Sharing with Live Migration (SLM). This method learns and predicts the homogeneity between the cloudlets then allocates each of it to a suitable VM layer which demonstrates improved virtual machine efficiency and resource utilization. The scheduler policy results in maximizing the resources utilization, reduction of energy consumption and make span time.
- The proposed scheduler amplifications and model aspects classification was presented (see chapter 4) indicating the service model specifications, workflow scheduling matrix, VMs consolidation strategy and provisioning scheme.
- A variety of GUI and programming based simulation tools were presented and evaluated regarding the criteria (see chapter 5), which are (1) underlying platform, (2) software, (3) performance, (4) availability, (5) simulation time, and (6) programming language and graphical support.
- The cloud datacentre resource usage rate was maximized while the system make span was reduced under the same workload rate by

deploying the proposed dual fold VM strategy governed by a threshold margin.

- The SLM model experimental setup, run and result was introduced (see chapter 6) where the SLM scheduler perform an optimal deployment of the datacentre resources and achieved good computing efficiency, network load minimization and energy consumption minimization.
- A model evaluation and discussions are presented (see chapter 7), the results showed performance improvement compared to the ACO and PSO scheduling techniques regarding VMs migrations number, energy consumption and total make span.
- For each watt consumed by computing resources an additional 0.5-1 watt is required by the cooling systems, which say that the SLM scheduler save almost double the percentage declared in section 7 because the algorithm doesn't just minimize the consumed energy with in the datacentre but also save as much power consumed by the cooling systems used to minimize that energy.

8.3 Thesis Contribution

This research has achieved all the objectives introduces in chapter 1 by designing and developing a novel energy efficient job scheduling technique for a cloud computing environment that allocate dynamic workload to resources based on energy optimization methods called Sharing with Live Migration (SLM). The designed scheduling algorithms were developed and implemented using CloudSim toolkit using Eclipse as a run time environment. A series of experimental simulations have been conducted toward testing and evaluating the work.

The main contributions of our work are as follows:

- A novel energy efficient job scheduling technique described in (chapter 3) provides job allocation efficiency, datacentre resource utilization and optimize job waiting time. The SLM scheduler composed of three main algorithms (SLM job classifier, SLM scheduler, VMs live migration) applied in serial order with in the job execution cycle to minimize the scheduling expense, total make span and minimize the energy consumption based on an accumulative effect on the entire datacentre performance.
- A job classification scheme is proposed to work in collaboration with the SLM scheduler. This scheme dynamically receives and manage user's requests as an input which are connected to synchronized job characteristics table SJC to provide jobs classification process and dynamic queue sorting based on FFD algorithm. Two classes of queues are created where the first one is the WU queue (Write and upload), which manage jobs that have job requesting for write or upload type of process. On the other hand, the second RD queue (Read and Download) which stores the applicable jobs to concurrent processing such as reading and downloading jobs with that requests.
- To manage the SLM scheduler job allocation and execution process, a proposed dual-fold layered VM is used (chapter 4) to reduce identical jobs execution time, where a dual-fold allocation algorithm to manage the distribution of the available resources to each allocated VM before job dispatching phase on two levels in the VM, the (TVML) as the top VM layer and the (LVML) as lower layer of the virtual machine is obtained by virtualizing the existing portion on CPU available to this VM. The job processing starting time of identical requests within the dual fold VMs is

the same denoted, while processing elements assigned for each of the similar jobs are disjoint.

- Using a VM assortment and provisioning scheme that works under a dynamic workload detection algorithm with a regulation of high and low threshold levels to apply VMs consolidation and load balancing among VMs. This scheme provide a dynamic load strategy to control the datacentre utilization levels, manage active server's state and prevent the performance degradation resulting from multiple VMs migration.
- SLM host consolidation and VM migration algorithm is used (chapter 4) that implies VMs migration by performing jobs transaction between two under or over utilized detected VMs with no interruption, where the current capacity of the new host is used to estimate the additional expanse of time to file transferring. The source host of the migrated VM will continue migrating all its VMs in case of low utilization detected until it is idle then switched to off mode state.
- In order to manage the scheduled administrated job to the VMs structure, a guided backfilling algorithm (GBA) is used to works on identical jobs dispatch and conduct synchronized job execution by filling the smallest unused lifted gabs within the selected VM dual-fold layers and reserve the appropriate VM layer to the next identical job request. This guarantee the ultimate utilization of available resources and contributes in avoiding the system starvation resulting from long weighting time for large jobs.
- Experimental results have been gathered then evaluated in regard to energy consumptions, CPU utilization, make span and total migration process numbers compared to two different scheduling techniques adapted for similar problem.

The overall aim of efficiently optimize the energy consumption levels, make span and improve the datacentre performance is achieved by maximizing the PMs and VMs efficiency, enhanced resource allocation strategies and active servers states management. The energy loss problem was controlled using a replicated host datacentre structure, where wasted idle hosts energy can be preserved using this scheme by maximizing the basic hosts utilization as long as the system can deal with workflow while setting replicated host on idle mode (chapter 3 and chapter 4). The applied replicated host structure achieve its goals by the deployment of a dynamic underutilized server's detection scheme (chapter 4) to manage the datacentre host's states. Moreover, applying virtualization on the existing physical machines in the datacentre managed the resource deficiency and maximize the datacentre utilization (chapter 2). In order to minimize the waiting time for the jobs, an identical jobs classification strategies reduced the load and downgrade the network communication based on a synchronized job control table of each host and its corresponding resource file.

8.4 The research Limitations

Despite substantial contributions of the current thesis in energy-efficient datacentres, the assumptions of simulation environment used in this thesis led to some challenges and limitations that need to be addressed in order to further advance the area. The key limitations are summarized as follows.

1. There is no specific methodology defined within the proposed SLM to change the task types of the datacentre so that a test case execution can be indiscriminate. It is the task of the test developer to take this into consideration.

2. In the simulation model, the host machines are of similar configuration, and the VMs created on host machines are also of similar configuration. However, the majority of the CSPs have different host machines and VMs (different configuration).
3. This model simulation doesn't provide the GUI feature (Graphical user interface). The GUI is required for a real-world product but is not relevant for the prototype.
4. In this research, CloudSim was selected to simulate the model, which consists of numerous elements. Only those CloudSim activities are implemented that are required for the proof of the concept.
5. The considered cloudlets units are of same length (containing same number of Instructions) in the simulation model but in real-time scenario, task units are not consisting of similar number of instructions.
6. The simulation runs on a dynamic offline jab batch mode and doesn't consider soft aperiodic cloudlets which have no deadlines.

Despite these limitations, the research has made valid contributions to knowledge and provided enough proof of concept for the proposed approaches.

8.5 Suggestions and Scope for Future Work

Cloud computing is still an emergent field, with several opportunities to conduct research. Scheduling mechanism efficiency in cloud computing relates to the efficient management of processing the incoming tasks and increasing the server performance as well as resources. There are numerous problems in preceding scheduling mechanisms as we have discussed earlier, which needs to be reduced in order to elevate the efficiency. A substantial number of earlier studies have hence targeted the scheduling in the cloud. Energy-efficient management

of Cloud infrastructure resources, and particularly dynamic VM consolidation explored in this thesis, will enable resource providers to successfully offer scalable service provisioning with lower energy requirements, costs, and CO2 emissions. Research, such as presented in this thesis, will undoubtedly drive further innovation in Cloud computing and development of next generation computing systems.

In present cloud datacentres infrastructures, it is essential to evolve a growing breadth of techniques to overcome the power consumption and make this the primary motivation for network administrators to investigate and focus on the overall network energy control.

In future work, the research focus will be on developing inclusive model for scheduling and provisioning high workload over 10,000 cloudlets in cloud environments. The model will expand by relying on adopting auto-fit scheduling technique using multi queuing model for heterogynous resources to satisfy the cloud service requests and reach the ultimate cloud datacentre utilization cost and energy. The aimed improved version of scheduling model will tolerate a robust monitoring system establishment for scalable real-time data management to achieve a major cloud field advancement.

8.6 Conclusions

This thesis introduced a novel scheduling algorithm, named SLM, which aims to reduce the overall energy consumption of a datacentre by combining performing a parallel cloudlets processing on homogenous VMs while deploying live migration policies based on a proposed threshold high and low margins. The mapping process time between the datacentre servers and the incoming jobs was controlled and optimized using this scheduling technique. The proposed

algorithm has a different constrains to manage the arriving cloudlets so that the datacentre can have the ultimate deployment of the processing time and resource management.

The evaluation of the SLM scheduler was based on a CloudSim simulation using JAVA language. A set of simulation tests demonstrated that the method leads to a more balanced workload consolidation for each VM and lower system make span. The SLM scheduling algorithm perform an optimal deployment of the datacentre resources to achieve good computing efficiency, network load minimization which had a huge effect on reducing the energy consumption in the datacentres. Compared to ACO and PSO scheduling algorithms, the SLM algorithm has a better deployment of the processing time and better overall optimization performance and the results shows a well performance in energy utilization and make span. The proposed algorithms are expected to be applied in real-world cloud platforms, aiming at network load minimization and reducing the energy costs for cloud datacentres.

References

[1] R.Buyya, C.Vecchiola, S.T.Selvi, “Mastering Cloud Computing: Foundations and Applications Programming,” The Netherlands: Elsevier, 2013.

[2] A.Tiwari, P.Richhariya, S.Patra, “Ant Colony based Cloud VM Allocation and Placement Approach for Resource Management in Cloud,” International Journal of Computer Applications, Vol.158, pp.8-11, 2017.

[3] M.Attaran, “Cloud Computing Technology: Leveraging the Power of the Internet to Improve Business Performance,” Journal of International Technology and Information Management, Vol.26, Jan 2017.

[4] Rackspace Cloud, December 2018. http://www.rackspacecloud.com/.

[5] Amazon EC2, December 2018. http://aws.amazon.com/ec2.

[6] Cisco global cloud index, https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.html.

[7] D.Laganà, C.Mastroianni, M.Meo, D,Renga, “Reducing the Operational Cost of Cloud Data Centers through Renewable Energy,” MDPI Algorithms, Vol.11, 2018.

[8] D.F.Cerero, A.F.Montes, F.Velasco, “Productive Efficiency of Energy-Aware Data Centers,” Energies. Vol.11, pp.3-17, Aug 2018.

[9] D.Armstrong, K.Djemame, R.Kavanagh, “Towards energy aware cloud computing application construction” Journal of Cloud Computing Advances, Systems and Applications, Vol.6, pp.1-13, 2017.

[10] G.Han, W.Que, G.Jia, L.Shu, “An Efficient Virtual Machine Consolidation Scheme for Multimedia Cloud Computing,” sensors, Vol.16, 2016.

[11] H.Cheung, S.Wang, C.Zhuang, "Development of a simple power consumption model of information technology (IT) equipment for building simulation," The 9th International Conference on Applied Energy ICAE2017, Cardiff, UK, 2017.

[12] J.K.Verma, C.P.Katti, P.C.Saxena, "MADLVF: An Energy Efficient Resource Utilization Approach for Cloud Computing," International journal of Information Technology and Computer Science, Vol.7, pp.56-64, 2014.

[13] C.P.Chandgude, G.B.Gadekar, "Core of Cloud Computing," Journal of Engineering Research and Application, Vol.7, pp.76-81, 2017.

[14] M.Azzouzi, F.Neri, "An Introduction to the Special Issue on Advanced Control of Energy Systems WSEAS Trans," Power System, Vol.8, pp.103, 2015.

[15] S.Srichandan, T.A.Kumar, S.Bibhudatta, "Task scheduling for cloud computing using multi-objective hybrid bacteria foraging algorithm," Future Computing and Informatics Journal, Vol.3, pp.210-230, 2018.

[16] S.Vakilinia, "Energy efficient temporal load aware resource allocation in cloud computing datacenters," Journal of Cloud Computing Advances, Jan 2018.

[17] I.Indu, P.M.Anand, V.Bhaskar, "Identity and access management in cloud environment: Mechanisms and challenges," Engineering Science and Technology an International Journal, Vol.21, pp. 574-588, 2018.

[18] J.Lindström, A.Hermanson, F.Blomstedt and P.Kyösti, "A Multi-Usable Cloud Service Platform: A Case Study on Improved Development Pace and Efficiency," Journal of Applied sciences: MDPI, Vol.8, pp.2-14, 2018.

[19] Amazon, December 2018, https://aws.amazon.com/compute/sla.

[20] S. A. Kolte, P.E. Ajmire, "A Survey- Cloud Computing," International Journal of Advanced Research in Computer and Communication Engineering, Vol. 5, April 2016.

[21] S.Singh, Y.S.Jeong, J.H.Park, “A survey on cloud computing security: Issues, threats, and solutions,” Network Computer Applications, Vol.75, pp. 200–222, 2016.

[22] P.Lubomski, A.Kalinowski, and H.Krawczyk, "Multi-level Virtualization and Its Impact on System Performance in Cloud Computing," International Conference on Computer Networks, pp. 247- 259, 2016.

[23] L.Wu, S.K.Garg, R.Buyya, “Service Level Agreement (SLA) based SaaS Cloud Management System,” IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS), 2016.

[24] S.Latif, S.M.Gilani, R.L.Ali, M.Liaqat, K.M.Ko, “Distributed Meta-Brokering P2P Overlay for Scheduling in Cloud Federation,” MDPI, Vol.8, pp.1-25, 2019.

[25] S.K.Yoo, B.Y.Kim, “A Decision-Making Model for Adopting a Cloud Computing System,” MDPI Sustainability, Vol.10, pp.1-15, 2018.

[26] J.W.Rittinghouse, J.F.Ransome, “Cloud computing: implementation, management, and security,” CRC press, 2016.

[27] A.Baiboz, “Energy-Efficient Data Center Concepts under the EXPO-2017,” Journal of Multidisciplinary Engineering Science and Technology (JMEST), Vol.2, pp.1126–1128, 2015.

[28] M.Ullrich, J.Lassig, M.Gaedke, “Towards efficient resource management in cloud computing: a survey,” IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud), 2016.

[29] V.Mojtaba, Y.Zhang, “Cloud Mobile Networks. In: From RAN to EPC,” Springer International Publishing AG, pp.11-31, 2017.

[30] H.Rahman, G.Wang, J.Chen and H.Jiang, “Performance Evaluation of Hypervisors and the Effect of Virtual CPU on Performance,” 2018 IEEE Smart World, Ubiquitous Intelligence & Computing, (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), 2018.

[31] N.Ahmad, A.Kanwal, M.A.Shibli, “Survey on secure live virtual

machine (VM) migration in cloud," Conference Proceedings - 2013 2nd National Conference on Information Assurance NCIA, pp.101–106, November 2015.

[32] A.Gupta, P.Dimri, R.M.Bhatt,"Virtual Machine Live Migration Procedures in Cloud Computing Environment," International Journal of Engineering Research in Computer Science and Engineering (IJERCSE), Vol.5, Issue.2, 2018.

[33] B.H.Malik, M.Amir, B.Mazhar, S.Ali, R.Jalil, J.Khalid," Comparison of Task Scheduling Algorithms in Cloud Environment," International Journal of Advanced Computer Science and Applications, Vol.9, No.5, 2018.

[34] P.H.Srimathi, "Survey and Analysis of Task Scheduling in Cloud Environment," Indian Journal of Science and Technology, Vol.9, No.37, 2016.

[35] S.p.chahal,A.Sharma, "Design An Optimal Scheduling Algorithm That Minimize The Cost And Task Completion Time," International Journal of Science Engineering and Advance Technology,Vol.2, Issue.9,2014.

[36] L.Luo, "A resource scheduling algorithm of cloud computing based on energy efficient optimization methods," IEEE Green Computing Conference (IGCC), pp.1–6, 2012.

[37] G.Monika, Y.Kalpana,"Data Security is the Major Issue in Cloud Computing - A Review," Indian Journal of Science and Technology, Vol.9, pp.2-6, 2016.

[38] M.Haghighat, S.Zonouz, M.AbdelMottaleb, "CloudID: Trustworthy Cloud-based and Cross-Enterprise Biometric Identification", Expert Systems with Applications, Vol.42, 2015.

[39] M.Avgerinou, P.Bertoldi, L.Castellazzi, "Trends in Data Centre Energy Consumption under the European Code of Conduct for Data Centre Energy Efficiency," Energies MDPI, Vol.10, 2017.

[40] T.Ma, M.Tang, W.Shen, Y.Jin, "Improved FIFO Scheduling

Algorithm Based on Fuzzy Clustering in Cloud Computing," Information 2017, Vol.5, pp.25, 2015.

[41] S.Sharma, N.Kumar, "Task Scheduling Algorithm in Cloud Computing based on Power Factor," International Journal of Innovative Research in Science, Engineering and Technology, Vol. 6, 2017.

[42] A.K.Yadav, S.B.Rathod, "Study of Scheduling Techniques in Cloud Computing Environment," International Journal of Computing Trends Technology, Vol.29, pp.69–73, 2015.

[43] A.I.Awad, N.A.ElHefnawy, H.M.Abdelkader, "Enhanced Particle Swarm Optimization for Task Scheduling in Cloud Computing Environments," Proceeding of Computer Sciences, Vol.65, pp.920–929, 2015.

[44] B.,P.Mishra, H.Das, S.Dehuri and A.K.Jagadev, " Cloud Computing for Optimization: Foundations, Applications, and Challenges," Springer, Vol. 39,pp.51-63,2018.

[45] H.Goel, N.Chamoli, "Job Scheduling Algorithms in Cloud Computing: A Survey," International Journal of Computer Applications, Vol.95, No.23, June 2014.

[46] N.Kaur, A.Chhabra,"Comparative Analysis of Job Scheduling Algorithms in Parallel and Distributed Computing Environments," International Journal of Advanced Research in Computer Science, Vol. 8, No. 3, April 2017.

[47] Z.Cai, X.Li, R.Ruiz, Qianmu Li,"A delay-based dynamic scheduling algorithm for bag-of-task workflows with stochastic task execution times in clouds" Future Generation Computer Systems, Vol.71, pp.57-72, June 2017.

[48] N.Almezeini, A.Hafez, "Task Scheduling in Cloud Computing using Lion Optimization Algorithm," (IJACSA) International Journal of Advanced Computer Science and Applications, Vol.8, No.11, 2017.

[49] A. Razaque, N. R. Vennapusa, N. Soni, G. S. Janapati, k. R. Vangala "Task Scheduling in Cloud Computing," IEEE Long Island Systems, Applications and Technology Conference (LISAT), pp. 1–5,April 2016.

[50] A.Mahmood, S.A.Khan, "Hard Real-Time Task Scheduling in Cloud Computing Using an Adaptive Genetic Algorithm," Computers2017, 2017.

[51] N.Moganarangan, R.G.Babukarthik, S.Bhuvaneswari, M.S.Saleem Basha, P.Dhavachelvan, "A novel algorithm for reducing energy-consumption in cloud computing environment: Web service computing approach," Journal of King Saud University - Computer and Information Sciences, Vol.28, PP 55-67, Jan 2016.

[52] D.Saxena, R.K.Chauhan And Ramesh Kait, "Dynamic Fair Priority Optimization Task Scheduling Algorithm In Cloud Computing: Concepts And Implementations," I. J. Computer Network and Information Security (IJCNIS), 2016.

[53] D.I.Esa, A.Yousif, "Scheduling Jobs on Cloud Computing using Firefly Algorithm," International Journal of Grid and Distributed Computing, Vol.9, No.7, pp.149-158, 2016.

[54] D.Komarasamy, V.Muthuswamy, "Content-Based Federated Job Scheduling Algorithm in Cloud Computing," Australian Journal of Basic and Applied Sciences, Vol.10, pp. 52-59, 2016.

[55] M.Lawanya Shri, M.B.Anbumalar, K.Santhi And M.Deepa, "Task Scheduling Based On Efficient Optimal Algorithm In Cloud Computing Environment," International Conference on Recent Research Development in Science, Engineering and Management (ICRRDSEM), May 2016.

[56] Z.Tang, L.Qi, Z.Cheng, K.Li, S.Khan, "An Energy-Efficient Task

Scheduling Algorithm in DVFS-enabled Cloud Environment," Journal of Grid Computing, Vol.14, pp.55-74, 2015.

[57] C.Wu, R.Chang, H.Chan, "A green energy-efficient scheduling algorithm using the DVFS technique for cloud datacenters," Future Generation Computer Systems, Vol.37, pp.141-147, 2014.

[58] S.A.Ali, J.M.Islamia, "A Relative Study of Task Scheduling Algorithms in Cloud Computing Environment," 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I), 2016.

[59] Q.Zhao, C.Xiong, C.Yu, C.Zhang, X.Zhao, "A new energy-aware task scheduling method for data-intensive applications in the cloud," Journal of Network and Computer Applications, Vol.59, pp.14-27, 2016.

[60] C.Lin, Y.Syu, C.Chang, J.Wu, P.Liu, P.Cheng, W.Hsu, "Energy-efficient task scheduling for multi-core platforms with per-core DVFS," Journal of Parallel and Distributed Computing, Vol.86, pp.71-81, 2015.

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

[62] B.Pavithra, R.Ranjana, "Energy efficient resource provisioning with dynamic VM placement using energy aware load balancer in cloud," 2016 International Conference on Information Communication and Embedded Systems (ICICES), Chennai, India, Feb 2016.

[63] S.M.AlIsmail, A.Kurdi, "Green algorithm to reduce the energy consumption in cloud computing data centers," 2016 SAI Computing Conference (SAI),pp. 557-561, 2016.

[64] D.Ziqian, N.Liu, R.Cessa, "Greedy scheduling of tasks with time constraints for energy-efficient cloud-computing data centers," Journal of Cloud Computing: Advances Systems and Applications 2015, Vol.4, 2016.

[65] Intel's cloud computing 2015 vision. http://www.intel.com/content/www/us/en/cloud-computing/cloudcomputing-intel-cloud-2015-vision.html.

[66] L.Ismaila, A.Fardoun, "EATS: Energy-Aware Tasks Scheduling in Cloud Computing Systems," The 6th International Conference on Sustainable Energy Information Technology (SEIT 2016), pp.870 – 877, 2016.

[67] K.Rajagopal,"Simulation of Online Bin Packing in Practice," University of Nevada, pp.67, 2016.

[68] N. Sharma, S. Tyagi, "A Survey on Heuristic Approach for Task Scheduling in Cloud Computing," International Journal of Advanced Research in Computer Science, Vol.8, No.3, 2017.

[69] A.K.Mishra, Y.Mohapatra, A.K.Mishra, "A fundamental study of scheduling algorithm for Cloud Computing Job Selection," Journal of Environmental Science, Computer Science and Engineering & Technology, Vol.5, pp.282-293, 2016.

[70] C. S. Bindu, A. Y. Reddy, P. D. K. Reddy,"Intelligent SRTF: A New Approach to Reduce the Number of Context Switches in SRTF," Springer Science and Business Media Singapore, 2017.

[71] S.Patel, U.Bhoi, "A Preemptive Priority Based Job Scheduling Algorithm in Green Cloud Computing," 2016 6th International Conference - Cloud System and Big Data Engineering, Jan 2016.

[72] S.K.Nager, N.S.Gill,"An Improved Shortest Job First Scheduling

Algorithm to Decrease Starvation in Cloud Computing," International Journal of Computer Science and Mobile Computing (IJCSMC), Vol.5, PP.155 – 161, Aug 2016.

[73] N.Er-raji, F.Benabbou, "Priority Task Scheduling Strategy for Heterogeneous Multi-Datacenters in Cloud Computing," (IJACSA) International Journal of Advanced Computer Science and Applications, Vol.8, No.2, 2017.

[74] Z.Qian, G.Yufei, L.Hong, S.Jin, "A Load Balancing Task Scheduling Algorithm based on Feedback Mechanism for Cloud Computing," International Journal of Grid and Distributed Computing, Vol.9, No.4, pp.41-52, 2016.

[75] M.Mohammadi, A.M.Rahmani, "De-centralized dynamic task scheduling using hill climbing algorithm in cloud computing environments," International Journal of Cloud Computing (IJCC), Vol.6, No.1, 2017 .

[76] M.Gautam, R.Kumar,"A Survey on Workflow Scheduling in Cloud Computing Environment," International Journal of Innovative Research in Computer and Communication Engineering, Vol.4, 2016 .

[77] M.Patel, R.Kadian, "A Review on ACO based Scheduling Algorithm in Cloud Computing," International Journal of Computer Science and Mobile Computing (IJCSMC), Vol.5, pp.489 – 493, 2016.

[78] W.Sun, Z.Ji, J.Sun, N.Zhang, Y.Hu, "SAACO: A Self Adaptive Ant Colony Optimization in Cloud Computing," 2015 IEEE Fifth International Conference on Big Data and Cloud Computing, 2015 .

[79] N.Sadhasivam, P.Thangaraj, "Design of an improved PSO algorithm for workflow scheduling in cloud computing environment," Intelligent Automation & Soft Computing, Vol.23, pp.493-500, 2016.

[80] H.Chen, X.Zhu, D.Qiu, L.Liu, "Uncertainty-Aware Real-Time Workflow Scheduling in the Cloud," The Journal of Supercomputing, 2017 .

[81] J.Meena, M.Kumar, M.Vardhan, "Cost Effective Genetic Algorithm for Workflow Scheduling in Cloud under Deadline Constraint," IEEE Access, Vol.4, pp.5065–5082, 2016 .

[82] P.Banga, S.Rana, "Heuristic based Independent Task Scheduling Techniques in Cloud Computing: A Review," International Journal of Computer Applications (0975 – 8887), Vol.166, 2017 .

[83] M.Jaeyalakshmi, P.Kumar, "Task Scheduling Using Meta-Heuristic Optimization Techniques in Cloud Environment," International Journal of Engineering and Computer Science, Vol.5, pp.19050-19053, 2016.

[84] J.Jiao, W.Yu, L.Guo, "Research on Batch Scheduling in Cloud Computing" Teklanika, Vol.14, No.4, pp.1454-1461, 2016.

[85] G.Patel, R.Mehta, U.Bhoi,"Enhanced Load Balanced Min-min Algorithm for Static Meta Task Scheduling in Cloud Computing," Procedia Computer Science, Vol.57, pp.545-553, 2015.

[86] Z.Chen, Y.Zhu, Y.Di, S.Feng,"A Dynamic Resource Scheduling Method Based on Fuzzy Control Theory in Cloud Environment," Journal of Control Science and Engineering, Jun 2015.

[87] X.Liu, Y.Zhou, "A Self-Adaptive Layered Sleep-Based Method for Security Dynamic Scheduling in Cloud Storage," 4th International Conference on Information Science and Control Engineering (ICISCE) (2017), pp.99-103, 2017.

[88] W.Tian, M.He, W.Guo, W.Huang, X.Shi, M.Shang, A.N.Toosi, R.Buyya, "On minimizing total energy consumption in the scheduling

of virtual machine reservations," Journal of Network and Computer Applications, Vol.113, pp.64–74, 2018.

[89] R.Rahmani,"A Complete Model for Modular Simulation of Data Centre Power Load," Journal of IEEE Transactions on Automation Science and Engineering, Vol.14, No.8, 2017.

[90] N.patel, H.Patel, "Energy efficient strategy for placement of virtual machines selected from under loaded servers in compute Cloud," Journal of King Saud University - Computer and Information Sciences, 2017.

[91] D.Komarasamy, V.Muthuswamy, "Priority scheduling with consolidation based backfilling algorithm in cloud," World Wide Web, Vol.21, pp.1453–1471, 2018.

[92] L.Barroso, U.Holzle,"The case for energy-proportional computing," Computer, Vol.40, No.12, pp.33-37, 2017.

[93] S.A.Ali, J.M.Islamia, "A Relative Study of Task Scheduling Algorithms in Cloud Computing Environment," Proceedings of the 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I), pp.14–17, 2016.

[94] S.M. Ranbhise, K.K.Joshi, "Simulation and Analysis of Cloud Environment", International Journal of Advanced Research in Computer Science & Technology (IJARCST 2014), Vol.2, 2014.

[95] A.Nunez, J.L.Poletti, A.C.Caminero, G.G.Castae, "ICanCloud: A Flexible and Scalable Cloud Infrastructure Simulator," Journal of Grid Computing, 2013.

[96] S.Sotiriadis, N.Antonopoulos, N.Bessis, A.Anjum,"SimIC: Designing a New Inter-cloud Simulation Platform for Integrating Large-Scale Resource Management," IEEE 27th International Conference on Advanced

Information Networking and Applications (AINA), 2013.

[97] A.Mohan, s.Shine, "Survey on Live VM Migration Techniques," International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), Vol.2, 2013.

[98] A.v.Sajitha, A.C.Subhajini, "Analysis of Cloud Sim Toolkit for Implementing Energy Efficient Green Cloud Data Centers," International Journal for Research in Applied Science & Engineering Technology (IJRASET), Vol.6, 2018.

[99] A.Sajjad, A.A.Khan, M.Aleem, "Energy-Aware Cloud Computing Simulators: A State of the Art Survey," International Journal of Applied Mathematic Electronics and Computers, Vol.6, pp.15-20, 2018.

[100] W.Shu, W.Wang, "A novel energy-efficient resource allocation algorithm based on immune clonal optimization for green cloud computing," Journal on Wireless Communications and Networking, 2014.

[101] I.G.Hemanandhini, C.Ranjani," A Study on Various Techniques for Energy Conservation in Data Centers for Green Computing," International Journal of Engineering Trends and Technology (IJETT), Vol.46, 2017.

[102] D.Wang, D.Tan, L.Liu, "Particle swarm optimization algorithm: an overview," Springer, Vol.22, pp.387-408, 2018.

[103] K.Anjaria, A.Mishra," Thread scheduling using ant colony optimization: An intelligent scheduling approach towards minimal information leakage," Karbala International Journal of Modern Science, Vol.3, pp. 241-258, 2017.

Appendix A – Abbreviations

ACO	Ant Colony Optimization
Amazon EC2	Amazon Elastic Compute Cloud
ACM	Association for Computing Machinery, Inc.
API	Application Programming Interface
BFD	Best Fit Decreasing
CapEx	Capital expense
CPU	Central Processing Unit
CFJS	Content-based Federated Job Scheduling
CSP	Cloud service providers
CIS	Cloud Information Service
DCB	datacentre broker
DC	datacentres
EMF	Eclipse Modelling Framework
ETS	Executable Test Suite
FCFS	First Come First Serve
FIFO	First in First Out
GA	Genetic Algorithm
GBA	Guided Backfilling Algorithm
HEFT	Heterogeneous Earliest Finish Time
IaaS	Infrastructure as a Service
I/O	Input / Output
IT	Information Technology
IEEE	Institute of Electrical and Electronics Engineering
ISA	Instruction Set Architecture
ITU	International Telecommunication Union
IoT	Internet of Things
JAR	Java Archive
JSR	Java Specification Request
LOA	Lion Optimization Algorithm
LAN	Local Area Network
MAD	Median Absolute Deviation
MIPS	Million Instruction per Second
NIST	National Institute of Standards and Technology

NIC	Network interface controller
OS	Operating System
PaaS	Platform as a Service
PACO	Period Ant Colony Optimization
PEs	Processing elements
PSO	Particle Swarm Optimization
QoS	Quality of Services
RAM	Random Access Memory
RD	Read and Download
RR	Round Robin
ROI	Return on investment
SaaS	Software as a Service
SJC	Synchronized Job Characteristic
SLA	Service Level Agreement
SMBs	Small and Medium Businesses
SAACO	Self-Adaptive Ant Colony Optimization
SJF	Shortest Job First
SUT	System/Service under Test
SLM	Sharing with Live Migration
TEE	Test Execution Environment
TCO	Total Cost Ownership
VM	Virtual Machine
VMM	Virtual Machine Monitor
WU	Write and Upload

Appendix B – Glossary

Cloudlet

Cloudlet is an extension to the cloudlet.

Cloudlet Scheduler

Cloudlet Scheduler is an abstract class that represents the policy of scheduling performed by a virtual machine.

Cloudlet Scheduler Space-Shared

Cloudlet Scheduler Space-Shared implements a policy of scheduling performed by a virtual machine.

Cloudlet Scheduler Time-Shared

Cloudlet Scheduler Time-Shared is a policy of scheduling performed by a virtual machine.

Datacentre

A facility that centralizes an organization's IT operations and equipment, as well as where it stores, manages, and distributes its data.

Datacentre Broker

Datacentre Broker represents a broker acting on behalf of a user.

Datacentre Characteristics

Datacentre Characteristics represents static properties of a resource such as resource architecture, Operating System (OS), management policy (time- or space-shared), cost and time zone at which the resource is located along resource configuration.

Host

A cloud host is a server that provides hosting services.

Make span

The total time needed to finsh processing all the tasks.

Pe

Processing Element represents CPU unit, defined in terms of Millions Instructions Per Second (MIPS) rating.

QoS

In Cloud Computing the term Quality of Services (QoS) denotes the levels of availability, reliability and performance offered by the infrastructure and by the platform and or an application that hosts it.

SLA

A contract document and a formal negotiation agreement between the cloud service providers and the cloud users based on the purpose and objectives of the cloud services.

VM

A virtual machine (VM) is a software program or operating system that not only exhibits the behaviour of a separate computer but is also capable of performing tasks such as running applications and programs like a separate computer.

VM Allocation Policy

VM Allocation Policy represents the provisioning policy of hosts to virtual machines in a Data centre.

Appendix C – Own Publications and Presentations

The following list includes publications and presentations related to the area of this research, to which the author of this thesis has contributed during the course of research.

- Alshathri, S. (2016). Towards an Energy Optimization Framework for Cloud Computing Data Centres. Eleventh International Network Conference INC 2016 Proceedings, Frankfurt, Germany, 2016, pp.9-12.
  Cscan:
  https://www.researchgate.net/publication/330054162_Towards_Green_Cloud_Computing_an_Algorithmic_Approach_for_Energy_Minimization_in_Cloud_Data_Centers

- Alshathri, S. (2018). Contemporary Perception Of Task Scheduling Techniques In Cloud: A Review, 2nd European Conference on Electrical Engineering & Computer Science EECS 2018, Bern, Switzerland, pp. 201-205.
  DOI: https://doi.org/10.1109/EECS.2018.00045
  IEEE Xplore: https://ieeexplore.ieee.org/document/8910077/
- Alshathri, S.; Ghita, B.; Clarke, N. (2018). Sharing with Live Migration Energy Optimization Scheduler for Cloud Computing Data Centres. Future Internet 2018, Vol.10, pp.86.
  DIO: https://doi.org/10.3390/fi10090086
  MDPI: https://www.mdpi.com/1999-5903/10/9/86/pdf
- Alshathri, S. (2019). CloudSim and Comparative Study on Cloud Computing Simulation Platforms. Under review with the IEEE Internet Computing 2019.