"MODELS FOR IMPROVING OF SCHEDULING OF TASK AND CHECKING ITS EFFICIENCY IN CLOUD COMPUTING"

Thesis

Submitted for the award of
Degree of Doctor of Philosophy
In Computer Science and Engineering
by

Arvind Kumar Singh

Enrollment No: MUIT0117038073

Under the Supervision of
Dr. Vaishali Singh
Assistant Professor, Department of Computer Science & Engineering,
Maharishi University of Information Technology, Lucknow

Under the Maharishi School of Engineering and Technology

Session 2017-2018

Maharishi University of Information Technology

Sitapur Road, P.O. Maharishi Vidya Mandir Lucknow, 226013

Declaration by the Scholar

I hereby declare that the work presented in this thesis entitled **" Models For Improving Of Scheduling Of Task And Checking Its Efficiency In Cloud Computing "** in fulfillment of the requirements for the award of Degree of Doctor of Philosophy, submitted in the Maharishi School of Engineering and Technology, Maharishi University of Information Technology, Lucknow is an authentic record of my own research work carried out under the supervision of **Dr. Vaishali Singh**. I also declare that the work embodied in the present thesis-

i) is my original work and has not been copied from any journal/ thesis/ book; and

ii) has not been submitted by me for any other Degree or Diploma of any University/ Institution.

Arvind Kumar Singh

Maharishi University of Information Technology
Lucknow

Supervisor's Certificate

This is to certify that Mr. Arvind Kumar Singh has completed the necessary academic turn and the swirl presented by him/her is a faithful record is a bonafide original work under my guidance and supervision. He has worked on the topic **" Models For Improving Of Scheduling Of Task And Checking Its Efficiency In Cloud Computing"** under the School of Engineering and Technology, Maharishi University of Information Technology, Lucknow.

Supervisor
Dr. Vaishali Singh
Asst. Professor
Faculty of Engineering & Technology
MUIT, Lucknow

Date:

ACKNOWLEDGEMENTS

I extend my deepest gratitude and appreciation to my advisor, Dr. Vaishali Singh, whose guidance and expertise have been invaluable throughout the entire duration of my Ph.D. program. His unwavering support, insightful feedback, and encouragement have been instrumental in shaping the course of my research.

I would like to express my sincere thanks to Vice Chancellor Sir and Dean Research Sir for their guidance and support in completing my research work. Their intellectual guidance and scholarly insights have played a crucial role in broadening my understanding of the subject matter.

I would like to express my sincere thanks to the faculty members department of Computer Science and Engineering, Maharishi University of Information Technology for their valuable contributions to my academic development. Their intellectual guidance and scholarly insights have played a crucial role in broadening my understanding of the subject matter.

I am indebted to my colleagues and fellow researchers for their collaborative spirit, stimulating discussions, and shared experiences. Their camaraderie has created a rich academic environment that has greatly enriched my research endeavors.

My heartfelt thanks go to my friends and family for their unwavering support, encouragement, and understanding during the challenging phases of my Ph.D. journey. Their love and belief in my abilities have been a constant source of motivation.

Lastly, I extend my appreciation to all those who have directly or indirectly contributed to the successful completion of this thesis. Your support has been crucial in shaping my academic and personal growth.

Thank you all for being an integral part of my academic journey.

Arvind Kumar Singh
Ph.D. Scholar
Department of Computer Science and Engineering
Maharishi University of Information Technology

ABSTRACT

Cloud computing is a platform that can enable elastic applications in order to manage a limited number of virtual machines and computing servers to provide application services at a certain point in time. It is necessary to verify and schedule available resources in the cloud using an effective task scheduler so that they may be assigned to customers depending on their demands. Scheduling is the most important task in any working framework, and it is controlled by the CPU. This is due to the fact that resources are not stored in a particularly designated manner, but rather in a strictly handy manner in order to ensure maximum skills. Task scheduling and resource deployment have been unified controlled by cloud computing service providers through the use of virtualized technologies in the cloud computing environment. Cloud computing is a distributed and parallel computing paradigm that is becoming increasingly popular. This type of cloud computing comprises a group of heterogeneous linked datacenters that are reliant on virtualization techniques and that are provided as a dynamically to the customer through a negotiation between cloud service providers and cloud users. In this research The Deadline-Aware Priority Scheduling (DAPS) model will be the first of the scheduling models. When using the Budget-Aware Scheduling (BAS) model, tasks will be scheduled and assigned based on available resources, with the goal of lowering the total time required to complete the tasks while staying within the budget constraints. This tool should be flexible enough to support this environment while also receiving an increasing number of user requests. The experiments were carried out on the BAS model and compared to state-of-the-art scheduling algorithms, demonstrating that the BAS minimizes the makespan, response time, and number of violations for task execution on VMs, as well as increasing resource utilization and provider profit, and achieving an acceptable total gain cost for any user. Simulation results indicate that the Deadline Budget Scheduling (DBS) model outperformed state-of-the-art algorithms in lowering the makespan and cost in a variety of configurations, including low resources or high resources model, varied number of jobs and virtual machines (VMs). The violation ratio is lowered in the DBS model to meet user requirements while enhancing the provider's profit and resource utilization.

CONTENTS

Content Details	Page No.
Title Page	i
Certificate by the Supervisor(s)	ii
Declaration	iii
Acknowledgements	iv
Abstract	v
Contents	vi-viii
List of Figures	ix-x
List of Tables	xi-xii

CHAPTER 1	**INTRODUCTION**	**1-40**
1. 1.1 OVERVIEW		1
2. 1.2 CLOUD COMPUTING		4
1.2.1 Benefits of Cloud Computing		5
1.2.2 Difficulties of Cloud Computing		6
1.2.4 History of Cloud Technology		10
1.2.5 Characteristics of Cloud		11
1.2.6 Service Models of Cloud Computing		12
1.2.7 Disadvantages of Cloud Computing		14
1.2.8 Components of Cloud Computing		14
3. 1.3 CLOUD ARCHITECTURE		15
1.3.1 Cloud Consumer		15
1.3.2 Cloud Auditor		16
1.3.3 Cloud Provider		17
4. 1.4 TASK SCHEDULING		18
1.4.1 The Features of Task Scheduling In the Cloud Computing Environment		25
5. 1.5 TYPES OF SCHEDULING IN CLOUD		27
6. 1.6 PROCESS OF TASK SCHEDULING		29
7. 1.7 THE TARGET OF TASK SCHEDULING IN CLOUD ENVIRONMENT		29
1.7.1 Load balance		30
1.7.2 Quality of Service		30
1.7.3 Economic Principles		30
1.7.4 The best running time		31
1.7.5 The throughput of the system		31

8. 1.8 GUIDELINES OF SCHEDULING 31
9. 1.9 SCHEDULING CRITERIA 33
10. 1.10 TASK SCHEDULING ALGORITHMS IN CLOUD COMPUTING 34
11. 1.11 LOAD BALANCE AWARENESS IN TASK SCHEDULING 35
12. 1.12 OBJECTIVES OF THE STUDY 36
13. 1.13 SIGNIFICANCE OF OUR RESEARCH 37
14. 1.14 PROBLEM STATEMENT 37
15. 1.15 RESEARCH METHODOLOGY 38
1.15.1 Research Design 38
1.15.2 Proposed Models 38
1.15.3 Task scheduling steps 39
1.15.4 Simulation Tool 40
1.15.5 Datasets 40
CHAPTER 2 LITERATURE REVIEW 41-81
CHAPTER 3 ENHANCED TASK SCHEDULING IN CLOUD COMPUTING BASED ON DEADLINE-AWARE MODEL 83-140
16. 3.1 PREAMBLE 83
17. 3.2 DEADLINE RESTRICTION 84
18. 3.3 TASK SCHEDULING DEPENDENT ON DEADLINE-AWARE MODEL 85
3.3.1. Case Study 88
3.3.2 Steps of Deadline-Aware Priority Scheduling Model 93
19. 3.4. PERFORMANCE METRICS 95
3.4.1 Makespan 95
3.4.2 Response Time (RT) 95
3.4.3 Resource Utilization (RU) 96
3.4.4 Guarantee Ratio (GR) 96
3.4.5 Violation Ratio (VR) 97
3.4.6 Improvement of makespan ratio 97
20. 3.5 OUTCOMES OF EXPERIMENTS AS WELL AS ANALYSIS 97
3.5.1 Implementation Environment 97
3.5.2 Experiments Configuration 97
3.5.3 Dataset 98
21. 3.6 PERFORMANCE ASSESSMENT 102
22. 3.7 EFFICIENT MANAGEMENT OF CLOUD RESOURCES THROUGH BUDGET-AWARE TASK SCHEDULING TECHNIQUE 115
23. 3.8 BUDGET CONSTRAINT 117
24. 3.9 TASK SCHEDULING DEPENDENT ON BUDGET-AWARE MODEL 117

3.9.1 Application Model 120
3.9.2 Case Study 126
3.9.3 Steps of Budget-Aware Scheduling Model 135
25. 3.10 METRICS OF PERFORMANCE 137
26. 3.11 RESULTS OF EXPERIMENTS AND ANALYSIS 140
3.11.1 Implementation Environment 140
3.11.2 Experiments Configuration 140
3.11.3 Dataset 141
27. 3.12 PERFORMANCE EVALUATION 142
CHAPTER 4 DEADLINE BUDGET SCHEDULING FOR VIRTUAL CLOUD ENVIRONMENT 157-202
28. 4.1 INTRODUCTION 157
29. 4.2 BUDGET AND DEADLINE CONSTRAINTS 158
30. 4.3 TASK SCHEDULING BASED ON DEADLINE BUDGET MODEL 159
4.3.1 Type of Task Constraint 160
4.3.2 Resources Clustering 160
4.3.3 Scheduling Strategy 161
31. 4.4 PERFORMANCE METRICS 175
32. 4.5 EXPERIMENTAL RESULTS AND ANALYSIS 177
4.5.1 Environment for Implementation 177
4.5.2 Experiments Configuration 178
4.5.3 Dataset 179
33. 4.6 PERFORMANCE EVALUATION 180
34. 4.7 AUTO-SCALING TO MINIMIZE COST AND MEET APPLICATION DEADLINES IN CLOUD WORKFLOWS 189
4.7.1 Preprocessing 190
4.7.2 Dynamic Scaling-consolidation-scheduling 193
35. 4.8 EVALUATION 196
4.8.1 Application, Workload, and Virtual Machine 197
4.8.2 Cost and Resource Utilization 199
4.8.3 Heavy Workload vs Light Workload 201
4.8.4 Sensitivity to Inaccurate Parameters 203
4.8.5 Mechanism Overhead 205
4.8.6 Proposed addition in Thesis 208-225
CHAPTER 5 CONCLUSION 226-228
36. 5.1 CONCLUSION 226
37. 5.2 CONTRIBUTIONS 229

38. 5.3 WORK DIRECTIONS FOR THE FUTURE 230
REFERENCES **231-242**

LIST OF FIGURES

CHAPTER 1

Figure 1.1 Usage of Cloud 5
Figure 1.2 Cloud to denote Internet 11
Figure 1.3 Environment of Cloud Computing 14
Figure 1.4 the Conceptual Reference Model 16
Figure 1.5 Task Scheduling in Cloud Computing 21
Figure 1.6 Task Scheduling Constraints 23
Figure 1.7 Tasks Scheduling in Cloud 24
Figure 1.8 Task Scheduling Process 29

CHAPTER 3

Figure 3.1: Quality of service constraints 85
Figure 3.2: The primary symbols of attributes for our study 87
Figure 3.3: Suggested Deadline-Aware Priority scheduling model 87
Figure 3.4: The VMs obtain the task deadline 91
Figure 3.5: Flowchart of DAPS model 93
Figure 3.6: Dataset fields of DAPS model 99
Figure 3.7: Contrasting of average makespan 104
Figure 3.8: Contrast of mean of total average response time 105
Figure 3.9: Comparison of resource utilization 106
Figure 3.10: Comparison of guarantee ratio 107
Figure 3.11: Quantity of violations 109
Figure 3.12: Violation ratio 112
Figure 3.13: Budget constraint 117
Figure 3.14: Suggested Budget-Aware Scheduling model 119
Figure 3.15: Flowchart of BAS model 121
Figure 3.16: Comparison of average makespan 144
Figure 3.17: Comparison of mean of total average response time 145
Figure 3.18: Comparison of resource utilization 147
Figure 3.19: Number of violations 148
Figure 3.20: Provider profit 150
Figure 3.21: Total gain cost 152
Figure 3.22: Virtual machines usage time (hour) 153

CHAPTER 4

Figure 4.1: Metrics under deadline and budget constraints 159
Figure 4.2: Proposed Deadline Budget Scheduling model 161
Figure 4.3: Organogram of the DBS model 162
Figure 4.4: Comparison of average makespan 182
Figure 4.5: Total gain cost 183
Figure 4.6: Number of violations 184
Figure 4.7: Provider profit 185
Figure 4.8: Comparison of resource utilization 186
Figure 4.9: Task bundling 190
Figure 4.10: Deadline assignment 191
Figure 4.11: Parallelism reduction 192
Figure 4.12: Load vector 193
Figure 4.13: Instance consolidation 194
Figure 4.14: Application models 197
Figure 4.15: Workload patterns 198
Table 4.16: VM types and prices 198
Figure 4.16: The performance for pipeline applications 201
Figure 4.17: The performance for parallel applications 202
Figure 4.18: The performance for hybrid applications 202
Figure 4.19: Heavy workload and light workload 203
Figure 4.20: Inaccurate task execution time 204
Figure 4.21: Inaccurate instance acquisition lag 205
Figure 4.22: SCS overhead 206

LIST OF TABLES

CHAPTER 1

Table 1.1 Comparison of scheduling characteristics in different types of scheduling techniques 28

CHAPTER 3

Table 3.1: Configuration requirement for the experiments 98
Table 3.2: Experiments done for tasks with various VMs of the DAPS model 100
Table 3.3: Choosing real deadline utilizing DAPS related on deadline constraint and minimum completion time 101
Table 3.4: Average of makespan of DAPS, GA, Min-Min, SJF and Round Robin algorithms 103
Table 3.5: Mean of total average response time of DAPS, GA, Min-Min, SJF and Round Robin algorithms 104
Table 3.6: Resource utilization of DAPS, GA, Min-Min, SJF and Round Robin algorithms 106
Table 3.7: Guarantee ratio of DAPS, GA, Min-Min, SJF and Round Robin algorithms 107
Table 3.8: Number of violations of DAPS, GA, Min-Min, SJF and Round Robin algorithms 108
Table 3.9: Actual deadline for random ten tasks in DAPS, GA, Min-Min, SJF and Round Robin algorithms 109
Table 3.10: Violation ratio of DAPS, GA, Min-Min, SJF and Round Robin algorithms 111
Table 3.11: Improvement in makespan ratio for DAPS compared to GA, Min-Min, SJF and Round Robin algorithms 112
Table 3.12: T-test of DAPS model compared to GA, Min-Min, SJF, and Round Robin algorithms 113
Table 3.13: Tasks attributes 126
Table 3.14: VMs configurations 126
Table 3.15: Expected gain cost of each task into each VM 127
Table 3.16: Task priority and comparison of length and file size for each task 133
Table 3.17: Performance of each VM 134
Table 3.18: Configuration requirement for the experiments 140
Table 3.19: Experiments conducted for BAS model for tasks with 10 VMs 142
Table 3.20: Average of makespan of BAS, Max-Min, Round Robin and SJF algorithms 143

Table 3.21: Mean of total average response time of BAS, Max-Min, Round Robin and SJF algorithms 145
Table 3.22: Resource utilization of BAS, Max-Min, Round Robin and SJF algorithms 146
Table 3.23: Number of task violations of BAS, Max-Min, Round Robin and SJF algorithms 147
Table 3.24: Provider profit of BAS, Max-Min, Round Robin and SJF algorithms 149
Table 3.25: Total gain cost of BAS, Max-Min, Round Robin and SJF algorithms 150
Table 3.26: VMs usage time (hour) of BAS, Max-Min, Round Robin and SJF algorithms 152
Table 3.27: Improvement of cost ratio for BAS compared with Max-Min, Round Robin, and SJF algorithms 154
Table 3.28: T-test of BAS model compared to Max-Min, Round Robin and SJF algorithms 155

CHAPTER 4

Table 4.1: Task attributes 164
Table 4.2: VM configurations 164
Table 4.3 Expected deadline and gain cost of each task into each VM 165
Table 4.4: Task constraint type and comparison of length and file size for each task 171
Table 4.5: Performance of each VM 172
Table 4.6: Configuration requirement for the experiments 178
Table 4.7: Experiments conducted for tasks with different VMs of the DBS model 179
Table 4.8: Average of makespan of DBS, GA, Max-Min, Round Robin and SJF algorithms 181
Table 4.9: Total gain cost of DBS, GA, Max-Min, Round Robin and SJF algorithms 182
Table 4.10: Number of violations of DBS, GA, Max-Min, Round Robin and SJF algorithms 183
Table 4.11: Provider profit of DBS, GA, Max-Min, Round Robin and SJF algorithms 185
Table 4.12: Resource utilization of DBS, GA, Max-Min, Round Robin and SJF algorithms 186
Table 4.13: Improvement in makespan ratio for DBS compared with GA, Max-Min, Round Robin, and SJF algorithms 187
Table 4.14: Improvement of cost ratio for DBS compared with GA, Max-Min, Round Robin, and SJF algorithms 188
Table 4.15: T-test of DBS model compared to GA, Max-Min, Round Robin, and SJF algorithms 188

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The introduction of cloud computing systems is a watershed moment in modern information technology (IT), necessitating the development of an efficient and powerful architecture that can be used to a variety of systems that demand complicated processing and large-scale data storage. Cloud computing is a platform that can enable elastic applications in order to manage a limited number of virtual machines and computing servers to provide application services at a certain point in time. When it comes to multi-tenant computing environments, the cloud is a good choice since it allows users to share resources. It is necessary to verify and schedule available resources in the cloud using an effective task scheduler so that they may be assigned to customers depending on their demands. With the fast expansion of contemporary computer systems, the requirement for an effective task scheduler has become an essential necessity in order to attain and maintain maximum performance has become an urgent necessity. It is the responsibility of task scheduling algorithms to map workloads submitted to a cloud environment onto available materials in such a way that the overall response time and latency are reduced while the throughput and utilization of resources are maximized. Conventional task scheduling algorithms such as Shortest-Job-First (SJF), Round Robin (RR), and First-Come-First-Serve (FCFS), Multilevel queue scheduling (MQ), Max-Min, and Min-Min have achieved breathtaking results in different computer system types over the years, but they have always been plagued by major problems such as increased waiting time in RR and FCFS and starvation in SJF and Max-Min.

Computers and other equipment can benefit from the sharing of hardware resources and information available through cloud computing, which is primarily driven by the increase in Internet-related services, as well as the use and delivery model based on, which is typically delivered via the internet and offers dynamically scalable and often virtualized resources. Cloud computing has become increasingly popular in recent years. Virtualization can provide excellent technological support for cloud computing, and cloud computing may be thought of as a type of application virtualization

technology in its own right. A large number of cloud computing research and development groups, such as Google, IBM, Microsoft, Amazon, Alisoft (formerly known as Amazon Web Services), Huawei (formerly known as Huawei Web Services), Baidu (formerly known as Alibaba), and nearly all domestic and international well-known IT companies have launched a cloud computing solution in the last few years. At the same time, academic circles both at home and abroad are conducting extensive research on cloud computing and its essential technology-related concept.

Cloud Computing alludes to both the applications conveyed as administrations over the Web and the hardware and systems software in the datacenters that give those administrations. The administrations themselves have for quite some time been alluded to as Software as an Administration (SaaS) the datacenter hardware and software is the thing that we will call a Cloud at the point when a Cloud is influenced accessible in a compensation as-you-to go way to the overall population, we call it an Open Cloud; the administration being sold is Utility Computing. We utilize the term Private Cloud to allude to inner datacenters of a business or other association, not made accessible to the overall population. Along these lines, Cloud Computing is the total of SaaS and Utility Computing, yet does exclude Private Clouds. Individuals can be clients or providers of SaaS, or clients or providers of Utility Computing. Cloud computing is another innovation used to offer various types of assistance.

Cloud is partitioned into two significant models in particular, service model and arrangement model. The fundamental three service models are Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). The new four cloud sending models are private cloud, local area cloud, public cloud and half breed cloud. Certain significant difficulties in cloud computing are Security, Cost and Service level understanding. Data Transaction Management (DTM) is one of the challengeable occasions in cloud computing. IaaS is essential to deal with the cloud transaction management. The data are kept up in the database – as-a-service (DaaS) model. It has databases in cloud environment and gives highlights like data definition, data stockpiling and data recovery. The main cloud computing suppliers like Amazon, Google, IBM, Oracle and Microsoft furnish database – as-a-service with their cloud

DaaS arrangements. In Cloud assets are ordinarily flexible, with limitless measure of figure force and capacity accessible on request and pay-just for-what-you-use.

Cloud Computing has arisen as quite possibly the main new computing procedures in the undertaking. A blend of advances and cycles has prompted an insurgency in the manner that computing is created and conveyed to end client. Cloud computing is characterized by National Institute of Standards and Technology (NIST) as: "a model for empowering helpful, on demand network admittance to a common pool of configurable computing resources(e.g., networks, workers, stockpiling applications and services) that can be quickly provisioned and delivered with insignificant management exertion or service supplier connection". The cloud computing worldview upgrades dexterity, scalability, and accessibility for end clients and ventures. Cloud Computing gives advanced and proficient computing platform, and decreases equipment and software venture cost, just as carbon impression For instance, Netflix, as it out growed its data place capacities, settled on a choice to move its site and web-based feature from a customary data community execution to a cloud environment. This progression permitted the organization to develop and extend client base without building and supporting data place impression to meet its development prerequisites.

Managing the transactions progressively circulated processing system isn't simple, as it has heterogeneously arranged PCs to take care of a solitary issue. In the event that a transaction keeps running over some extraordinary sites, it might submit at a few sites and may disappointment at another site, prompting a conflicting transaction. The multifaceted nature is increment progressively applications by putting due dates on the reaction time of the database system and transactions processing. Such a system needs to process transactions before these due dates terminated. A progression of simulation examine have been performed to investigate the execution under various transaction management under conditions, for example, extraordinary workloads, conveyance strategies, execution mode-dispersion and parallel and so forth. The planning of information gets to are done with a specific end goal to meet their due dates and to limit the quantity of transactions that missed due dates. Another idea is acquainted with deal with the transactions in database measure for beginning site and

remote site instead of database estimate processing parameters. With this approach, the system gives a noteworthy change in execution.

1.2 CLOUD COMPUTING

Cloud computing is characterized as "a model for empowering omnipresent, advantageous, on request network admittance to a common pool of configurable computing assets (e.g., networks, workers, storage, applications, and services) that can be quickly provisioned and delivered with negligible management exertion or service supplier cooperation". Cloud computing follows compositional plans and service provisioning models that stray from the customary viewpoints, plan and services of conventional data innovation. Cloud computing offers data innovation assets provisioned (primarily through the Internet) in a Service Oriented Architecture (SOA) where clients just compensation for the assets they devour.

Contrasted with old style Internet and dispersed computing systems, cloud computing has different distinctive highlights, including pay-more only as costs arise plan of action, virtualization, huge scope storage offices, support for large data, high process force, and backing for misusing the strength of both incredible just as ware PCs. This model of pay-more only as costs arise services offered by cloud computing merchants gives customers the figment of admittance to endless computing assets. Assets are totally flexible and can be accommodated as and when required. Associations and people thusly don't have to make capital interests in computing assets. These highlights have engaged cloud computing innovations to turn into an alluring service facilitating and conveyance platform for different associations, instructive foundations, public area and enterprises like Google, Microsoft, Amazon and Facebook among others.

It is difficult to characterize what Cloud Computing is on the grounds that various creators have various definitions on Cloud Computing. In any case, as indicated by NIST (National Institute of principles and technology)"Cloud Computing is a model for empowering pervasive, helpful, on-request network admittance to shared pool of

configurable computing assets (e.g.., networks, workers, storage and applications) that can be quickly provisioned and delivered with insignificant management exertion or service supplier connection".

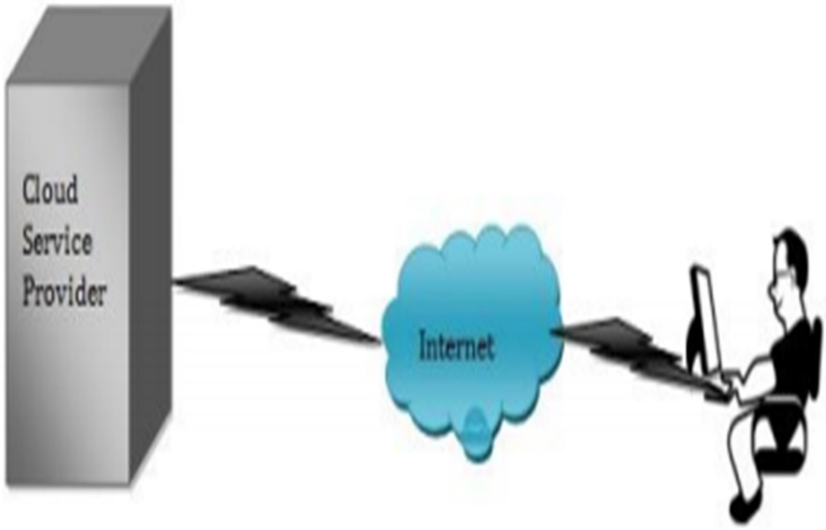


Figure 1.1 Usage of Cloud

Cloud Computing has five fundamental qualities (On-request self-service, Broad network access, Resource pooling, Rapid versatility and Measured service), three service models (Software as a service, Platform as a service and Infrastructure as a service) and four arrangement models (Private Cloud, community Cloud, public Cloud and Hybrid Cloud).

1.2.1 Benefits of Cloud Computing

Cloud Computing enjoys numerous benefits, a few models are:

- Masked intricacy - updates and support of the item or service can be stowed away from clients, without having them to partake;

- Cost flexibility - with the cloud computing there is no compelling reason to pay devoted software permit expenses, or to finance the structure of equipment and introducing software. The point of each business association is to have more clients and the present business are on the whole online as client expansion in getting to association application, the measure of data that an

application handles is expanding step by step as is the CPU power that can outfit. Association needs more transfer speed a cloud-based service can immediately fulfill the need due to the huge limit of the services of distant workers.

- Scalability - cloud empowers ventures to add computing assets at the time they are required;

- Adaptability - cloud computing assists ventures with adjusting to different client bunches with a different arrangement of gadgets;

- Ecosystem connectivity – cloud works with outside joint effort with shoppers and accomplices which prompts enhancements in efficiency and expanded development;

1.2.2 Difficulties of Cloud Computing

For those organizations which are anticipating giving cloud resources to their clients, they should project a picture that shows that they are truly solid to the level of the electric utility model. This could be an issue for organizations that depend on the cloud to keep basic business works fully operational. Having the option to keep significant data secure has consistently been a need in IT, yet with an innovation that takes data outside of the virtual secure dividers most enterprises have up will raise warnings. The use of dainty customers might actually be high-jacked if individuals are imprudent with data. Due to security concerns, cloud merchants generally are reluctant to introduce contextual investigations about organizations that are as of now utilizing their services. "Building new services in the cloud or in any event, receiving cloud computing into existing business setting, overall is a mind boggling choice including numerous components. Endeavors and associations need to settle on their decisions identified with services and deployment models just as to change their operational strategies into a cloud situated plan joined with an exhaustive danger evaluation work on coming about because of their requirements". By and large, cloud itself can't encounter disappointment; the service goes down. For instance, when Gmail had interference for 30 hours on the sixteenth of October, 2008, it was anything but a cloud disappointment; it was a service disappointment. "Blackout is the most basic issue that is making news in the cloud computing region. It alludes to the non-

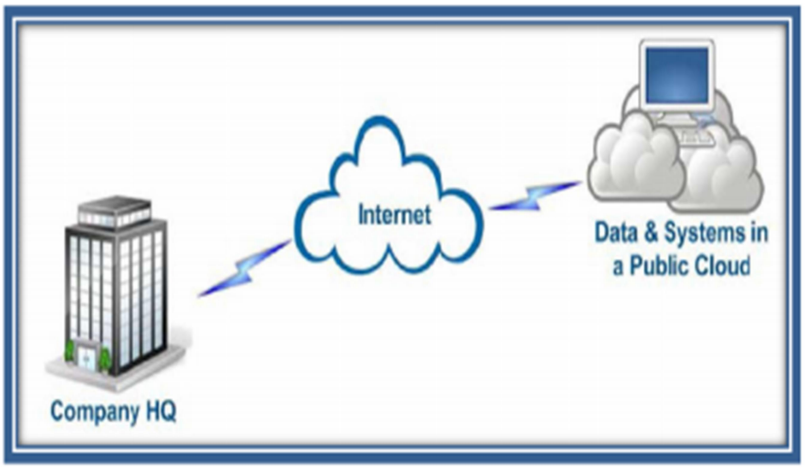
accessibility of service in the cloud throughout a specific time". For instance, in 2009, Microsoft Sidekick blackout brought about a deficiency of the clients data (a huge number of clients lost their data that were put away in the cloud).

1.2.3 Deployment Models Of Cloud Computing

Four sorts of deployment services accessible in the Cloud they are Private Cloud, Public Cloud, Hybrid Cloud, and Community Cloud.

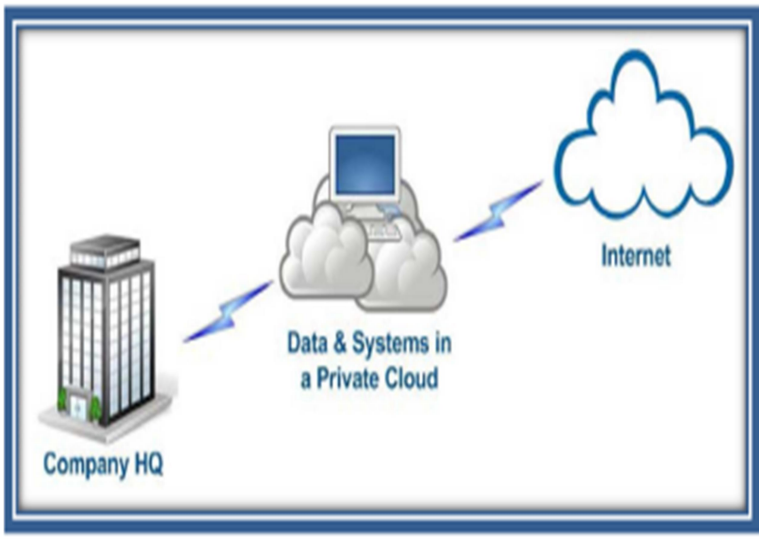
There are four fundamental Cloud computing models: public, private, hybrid and community cloud.

- **Public cloud:** "The cloud infrastructure is made accessible to the overall population or a huge industry bunch and is possessed by an association selling cloud services". Consumers need to pay just for the time term they utilize the service, i.e., pay-per-use which helps in diminishing expenses. They are less secure contrasting with other cloud models since every one of the applications and data are more opened to malignant assaults. Proposed answer for this worry is security approval keep an eye on the two sides. Community Cloud is a kind of infrastructure to share an asset to numerous associations from a particular community with basic concerns (for example security necessities, mission, strategy, consistence contemplations). A Community Cloud is a communitarian exertion in which infrastructure is divided among a few associations from a particular community with regular concerns (security, consistence, purview, and so forth), regardless of whether oversaw inside or by an outsider and facilitated inside or remotely.



- **Private cloud:** "The cloud infrastructure is worked exclusively for an association. It very well might be overseen by the association or an outsider and may exist on premise or off premise ". It is a data community possessed by a cloud computing supplier. "The primary benefit is that it is simpler to oversee security, upkeep and redesigns and furthermore gives more power over the deployment and u0se. Contrasted with public cloud where every one of the resources and applications are overseen by the service supplier, in private cloud these services are pooled together and made accessible for the clients at the authoritative level. The resources and applications are overseen by association itself". Private Cloud is additionally called as inner Cloud or corporate Cloud. Private Cloud is giving asset, storage of data to a predetermined number of facilitated services. This Cloud might be overseen and worked by the association behind a firewall. Private Cloud can get to who are situated inside the limits of an association. Private clouds are those that are assembled only for a solitary business. For some, organizations considering cloud computing, private clouds are a decent beginning stage. They permit the association to have applications, improvement conditions, and infrastructure in a cloud, while tending to concerns with respect to data security and control that can emerge in the public cloud climate. There are two sorts of private clouds: One sort of private cloud is an On Premises Private Cloud: This model, otherwise called an "Inward Cloud," is facilitated inside an association's own data place. A second sort of private cloud is a remotely

facilitated Virtual Private Cloud: This private cloud model is facilitated by a third-gathering Cloud Service Provider.



- **Community cloud:** "The cloud infrastructure is shared by a few associations and supports a particular community that has shared concerns (e.g., mission, security prerequisites, strategy, and consistence contemplations). It very well might be overseen by the associations or an outsider and may exist on premise or off premise ". Community Cloud is a kind of infrastructure to share an asset to numerous associations from a particular community with basic concerns (for example security necessities, mission, strategy, consistence contemplations). A Community Cloud is a communitarian exertion in which infrastructure is divided among a few associations from a particular community with regular concerns (security, consistence, purview, and so forth), regardless of whether oversaw inside or by an outsider and facilitated inside or remotely.

- **Hybrid cloud:** "The cloud infrastructure is an arrangement of at least two clouds (private, community, or public) that stay novel substances however are bound together by normalized or restrictive innovation that empowers data and application portability(e.g., cloud blasting for load-adjusting between cloud)". Hybrid cloud is safer approach to control data and applications and permits the gathering to get to data over the web. This cloud infrastructure is a mix of at least two unmistakable clouds. In this model an association gives

and deals for certain assets in-house and has others given remotely. It offers the advantages of numerous deployment models to the clients. We can decide to keep up certain systems and data in-house while utilizing outer services where they will be more successful for our business. A particularly consolidated arrangement is known as a Hybrid Cloud. A hybrid cloud is for the most part best-of-breed. It consolidates the solace level of a private cloud with the adaptability and flexibility of the public.

1.2.4 History of Cloud Technology

Cloud Computing traces all the way back to 1950 for example the enormous scope centralized servers were made accessible to huge undertakings. There the centralized computers' equipment infrastructure was gathered and introduced in worker room; the clients had the option to get to the data at worker by their imbecilic terminals. Later in 1970, IBM delivered an Operating System (OS) called Virtual Machine (VM) that permitted administrators to have numerous virtual system on a solitary actual hub. Each VM runs on visitor OS hand their own memory, CPUs and equipment gadgets alongside consoles, systems administration and CD-ROMs in spite of the way that those assets would be shared. Consequently virtualization idea turned into an innovation driver and it went about as an impetus for some greatest advancements in computing. In 1990s, the media transmission businesses came to its meaningful conclusion to-point data association as virtualized private organization associations with a similar service quality at a diminished rate. So the market pattern changed this way - "these workers are modest, allowed us to concentrate out how to join them". This move just cleared the venturing stone for Cloud Computing. A system would introduce the entirety of the assets as though they were in a solitary actual hub, by installing a piece of software called hypervisor across numerous actual hubs.

The significant advantage of the idea driving cloud computing is that the normal client doesn't need a PC that is very incredible to deal with complex database ordering assignments. With this resource, cloud computing clients from everywhere the world can appreciate the advantages of colossal processing power without significant capital or specialized expertise. The weighty utilization of transmission capacity that includes the web of today is the thing that makes the innovation work, as already networks were substantially less powerful because of moderate transfer and

download speeds that were accessible at that point. Another main consideration that changed the scene was the possibility that large numbers of modest PC equipment could be outfit to make a limitlessly organized data place similarly comparable to a more modest measure of more costly, better worker equipment.

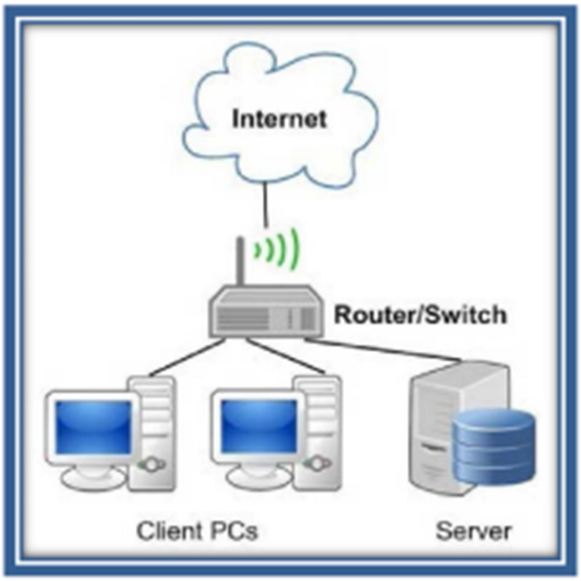


Figure 1.2 Cloud to denote Internet

It is fascinating to know the justification the name 'Cloud Computing'. The root of the term cloud computing is dark, yet it seems to get from the act of utilizing drawings of adapted clouds to indicate networks in outlines of computing and correspondences systems. The name 'Cloud Computing' was propelled by the 'Cloud' image (Figure 1.2) that is frequently used to address the Internet in stream outlines and chart since Cloud computing implies conveying facilitated service over the Internet.

1.2.5 Characteristics of Cloud

For a conveyed database system to be viewed as a cloud database, it should display five attributes that all cloud computing plans share: it should be an appropriated system, should be effectively expandable, should have a unique task of assets, should have high adaptation to non-critical failure, and, in conclusion, it should have a pay-more only as costs arise estimating system. Every one of these is inspected thusly to all the more obviously decide how the cloud contrasts from a conventional conveyed system. Initial, a cloud should be a disseminated system, however it will be shown

that it is a quite certain kind of conveyed system. The way that the cloud is a circulated system suggests that it utilizes more than one PC associated over a network to tackle a computational assignment. As demonstrated beneath, this first cloud trademark frames the reason for the vast majority of the other four attributes. The following two qualities go inseparably: the cloud should be effectively expandable and should powerfully be allocated assets. As the requirement for computational force expands, more CPU's and memory should be apportioned to the client's cloud network powerfully. The converse should likewise be valid. As assets are not, at this point required, they can be delivered so they are assignable to other cloud clients. Essentially, this permits clients to have a PC that can't run out of assets or computational force as long as the client can manage the cost of the additional assets. To make this unique assignment of assets, cloud PCs are regularly made as virtual PC pictures dwelling on a gathering of actual workers. As assets are mentioned, the facilitating workers will allot them to the mentioning virtual machines. If a worker runs out of assets to disperse, the virtual cloud occasion can be moved to an alternate actual worker that has more assets, which likewise opens up the entirety of the virtual PC's old assets for different clouds to utilize.

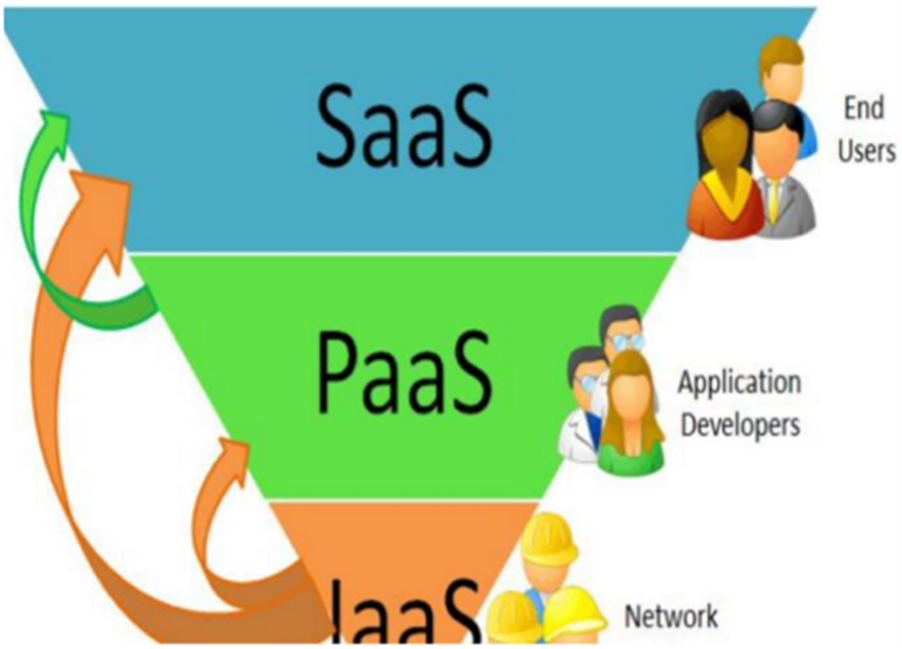
Another cloud highlight that normally ascends from the way that the cloud is a dispersed system is that it has a high adaptation to internal failure. This implies that an endeavor's data are upheld up across a few PCs inside the gathering of physical and virtual PCs that make up the cloud. On the off chance that one of those PCs were to go disconnected or get inaccessible, clients would in any case have the option to get to their data on one of different PCs in the cloud. Ultimately, clients just compensation for the measure of power, clock cycles, and memory they use. This pay-more only as costs arise valuing plan permits clients to perform CPU-concentrated calculations without paying for the development of an incredible PC. They need just compensation for the measure of time they spend running their projects.

1.2.6 Service Models of Cloud Computing

There are regularly three cloud service models: Software as a Service, Platform as a Service, and Infrastructure as a Service. These plans characterize how enormous of a cloud climate is given to the client and, thus, which parts of the system the client

controls. The Software as a Service (SaaS) model permits clients to have programs and their related data in the cloud, and to just deal with the subtleties of the cloud PC that relate to their software. SaaS turns out to be especially helpful when a gathering of clients wish to run a program. Maybe than having every client introduce the program on their nearby PC, the program could be facilitated in the cloud for the entirety of the clients to associate with by means of SaaS.

Rather than SaaS, Infrastructure as a Service (IaaS) gives the client power over each part of the facilitated cloud PC aside from the upkeep of the actual PC itself. This permits clients to completely control how their projects are run. They can choose everything from which working system is utilized to which processes are permitted to run at a given time. The last service model, Platform as a Service (PaaS), falls among SaaS and IaaS on the size of client control. PaaS clients are accountable for the "climate" that projects are executed in, yet are not liable for the remainder of the system. These three service models permit clients to choose precisely how much control they need over their cloud climate, however they are not a sign of their current circumstance's cloud type.



1.2.7 Disadvantages of Cloud Computing

Each coin has different sides and henceforth cloud likewise has certain impediments

- Dependability: For those organizations which are anticipating giving cloud assets to their clients, they should project a picture that shows that they are truly dependable to the level of the electric utility model. This could be an issue for organizations that depend on the cloud to keep basic business works fully operational.

- Security: Being ready to keep significant data secure has consistently been a need in IT, yet with an innovation that takes data outside of the virtual secure dividers most enterprises have up will raise warnings. The utilization of flimsy clients might actually be high-jacked if individuals are imprudent with data.

- Little or no Reference: Because of security concerns, cloud merchants generally are reluctant to introduce contextual analyses about organizations that are as of now utilizing their services.

1.2.8 Components of Cloud Computing

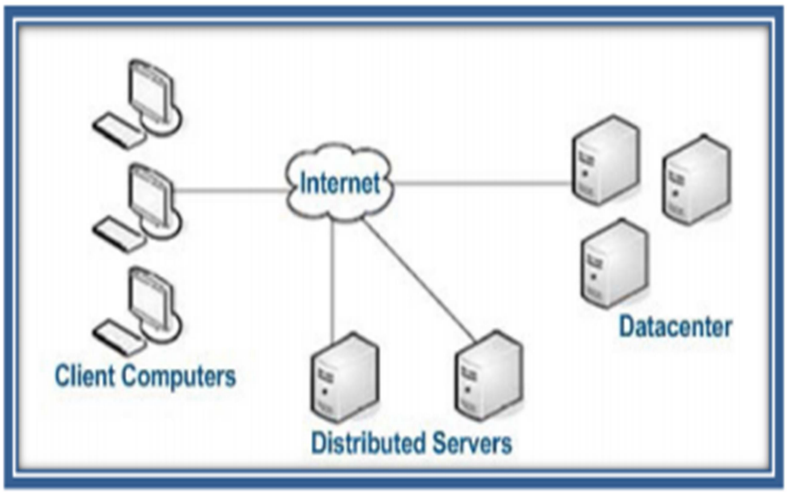


Figure 1.3 Environment of Cloud Computing

A cloud computing arrangement is comprised of a few components like clients, the data community and circulated workers (Figure 1.3). Every component has a particular job and reason in conveying a utilitarian cloud-based application

- **Clients:** Clients are the gadgets that the end clients collaborate with to deal with their data on the cloud. They can be PCs, workstations, tablet PCs, cell phones or PDAs all enormous drivers for cloud computing as a result of their versatility. For the most part clients can be versatile, slender or thick in nature.

- **Datacenter:** The datacenter is the assortment of workers where the application to which we buy in is housed. Virtualization is pertinent to cloud computing since it is one of the manners by which we will get to services on the cloud. There are two sorts of virtualization.

Distributed Servers: Often, workers are in geologically dissimilar areas Be that as it may, for the cloud supporter these workers go about as though they are murmuring ceaselessly directly close to one another. This gives the service supplier greater adaptability in alternatives and security.

1.3 CLOUD ARCHITECTURE

Basically, the cloud architecture is built with the combined perspective of cloud provider, user, and the broker. The conceptual picture of the cloud architecture is given in Fig. 1.8. The cloud conceptual reference model includes of cloud carrier, consumer auditor, provider and broker.

1.3.1 Cloud Consumer

The cloud consumer is regarded as a major stakeholder, and it represents a company or individual that obtains Cloud Computing services from a cloud provider for the purpose of doing their business. Cloud consumers can receive service from cloud providers in accordance with the terms of their service level agreements (SLAs). It outlines technical performance criteria, service contracts, and costs, quality of service, security and remedies in the event of a performance failure. A cloud customer can use any type of cloud service, including SaaS, PaaS, and IaaS, to accomplish their goals.

1.3.2 Cloud Auditor

A cloud auditor is a third party who has the ability to conduct an independent assessment of service checks with the purpose of expressing an opinion on the results. They may assess the privacy implications of cloud services, as well as their performance and security measures, among other things. A contractual condition allowing third parties to review the security measures of cloud service providers, according to Vivek Kundra, should be included in agencies' contracts".

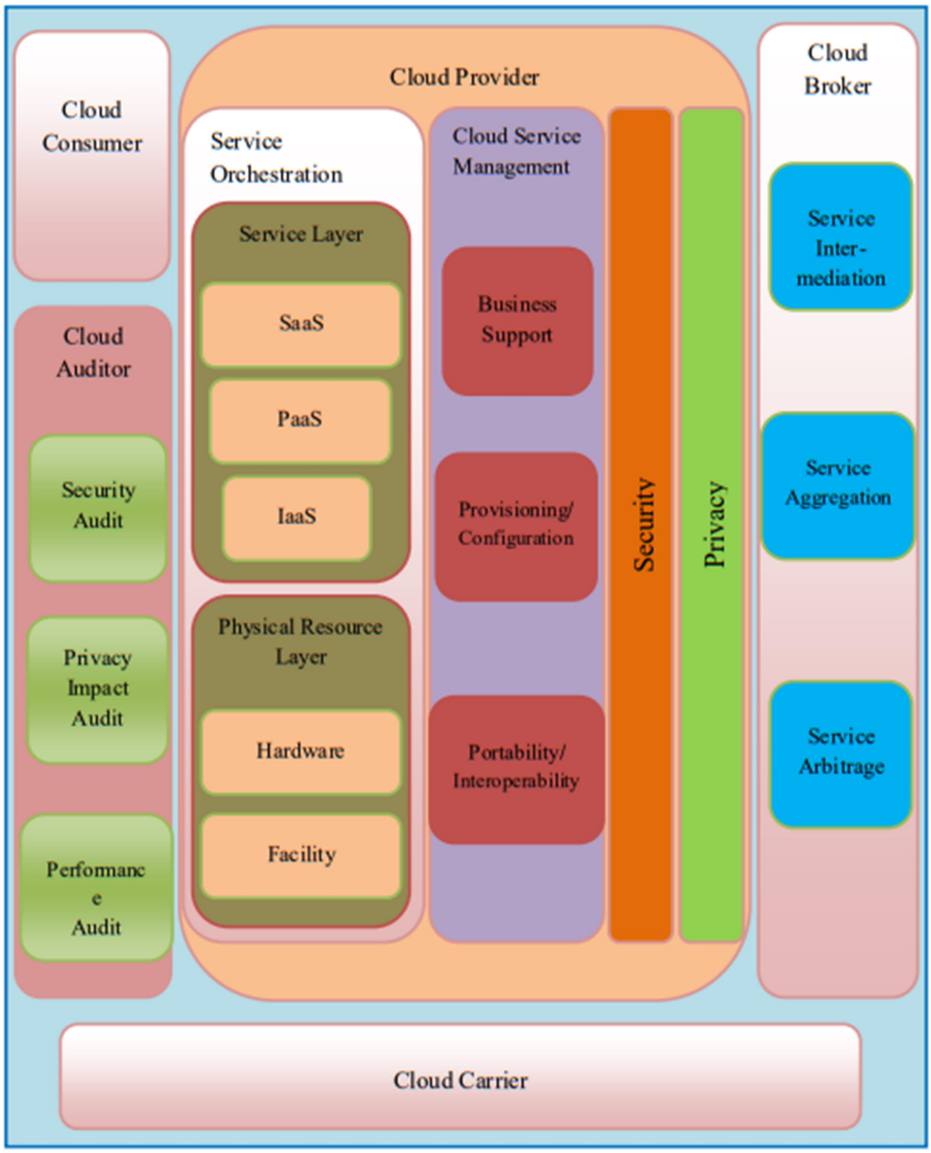


Figure 1.4 the Conceptual Reference Model

1.3.3 Cloud Provider

The job of the cloud provider is extremely crucial, as well as the most challenging. An elastic pool of resources such as servers, networks, storage and engineered systems is provided by a cloud provider, who uses cloud enabling technologies such as distributed computing, autonomic computing, internet technologies, and virtualization to meet the demands of customers on an as-needed basis. Service orchestration, cloud service management, service deployment, privacy and security are just a few of the categories in which cloud providers may be classified according to their capabilities.

- **Service Orchestration**

The paradigm for service orchestration is composed of three layers. The first is the service layer, which defines the interfaces that cloud users may use to access computing resources that are provided as services. SaaS (Software as a Service), PaaS (Platform as a Service), and IaaS (Infrastructure as a Service) are the three degrees of service offered. Individuals and groups can use the services, and they can access and use them at the same time. It is followed by a layer known as the resource abstraction and control layer, which allows physical resources to be conceptually partitioned with the use of virtualization technologies. This layer is in charge of access control, resource allocation, and use monitoring, among other things. Third, the physical resource layer, which includes all physical computing resources such as servers, networks, storage components, power, cooling systems, and other aspects of the physical components, as well as all other layers of computer resources.

- **Cloud Service Management**

All of the service-oriented tasks are addressed from the perspectives of provisioning and configuration, business support, portability or interoperability, and interoperability or portability. This section focuses on supplying resources quickly, modifying resources quickly, monitoring and metering, reporting and service level agreement (SLA) management, among other things. Business support activity is comprised of a collection of business-related services that are handled in conjunction with cloud users, companies, and supporting procedures. For cloud users, it provides features such as customer management, contract management, inventory management, accounting, and invoicing, as well as reporting, auditing, pricing, and

rating. The Mobility and Interoperability activity focuses on the portability of cloud customers' data or applications, as well as intercommunication across various cloud environments.

- **Security and Privacy**

Cloud security is a critical problem since it must be protected from the physical layer all the way up to the application layer. It handles cloud security needs such as authorization, availability, authentication, confidentiality, integrity, management, audit, identification, monitoring, threat detection, and security policy management, as well as cloud computing security controls such as encryption. In order to maintain privacy, the cloud provider should take appropriate measures to safeguard Personally Identifiable Information (PII). Information about a person's identification, such as their place of birth, date of birth, name, social security number, biometric information, and so on, is used to track or recognize that person's identity.

1.3.4 Cloud Broker

Because of the rapid advancement of Cloud Computing, the consolidation of Cloud Computing services has become too complex for cloud customers to handle. Instead of directly contacting a cloud provider, a cloud user may choose to contact a cloud broker to gain access to services. A cloud broker is a company that facilitates the negotiation of cloud services between cloud consumers and cloud providers on their behalf. A cloud broker can provide services such as service intermediation, service aggregation, and service arbitrage, among other things.

1.3.5 Cloud Carrier

A cloud carrier serves as an intermediary between cloud providers and cloud consumers, facilitating the delivery and connection of Cloud Computing services. User access is provided by a variety of means, including telecommunication networks, local area networks, transport agents, and other access devices. Moreover, in accordance with the service level agreement, a cloud carrier must provide dedicated and secure connections between cloud users and cloud providers.

1.4 TASK SCHEDULING

Scheduling is the most important task in any working framework, and it is controlled by the CPU. This is due to the fact that resources are not stored in a particularly designated manner, but rather in a strictly handy manner in order to ensure maximum skills. Because cloud services need a significant level of control and management of resources, effective scheduling is essential for the efficient management of projects and activities. Scheduling is critical to the operation of the management system. This problem is known to be NP-complete in generally due to the nature of the scheduling problem involved. Clients submit tasks to a broker or scheduler who then uses the information to execute the scheduling algorithm, allowing the system to begin taking task. In cloud-based infrastructure, real machines are virtualized into unified resources known as virtual machines, which are shared by multiple users (VMs). With the scheduler, you can control how and when resources (VMs) are used, and you can also control which tasks are done on each of these resources. It assigns tasks to the most appropriate virtual machine so that the computing process may be completed in order to meet the Quality of Service (QoS) restrictions imposed by users, such as deadlines and costs. This QoS-based optimization tries to minimize execution costs or to make execution time as fast as feasible while staying within a defined budget range. Scheduling refers to the process of allocating system resources to different activities. When it comes to cloud computing, scheduling is utilized to achieve excellent performance and the highest system throughput.

The type of scheduling that is used in a cloud-computing environment has a significant impact on the speed, efficiency, and efficient usage of resources. The maximum CPU usage and the maximum throughput are two of the scheduling requirements. The business model of the cloud computing environment is well-known in the industry. Users' demands for a wide range of jobs, a unified solution, and task scheduling continue to be major concerns. Between the allocation of resources in a cloud computing system and the distribution of social income riches, there are certain parallels: the resources given by foundation facility makers are comparable to the total amount of social wealth available. It is possible to abstractly represent social individuals as the needs of various users expressed through distinct task types. The amount of resources paid by the user might be viewed as a form of reward for the social individual's efforts. They divide money in diverse ways based on the disparities in work.

Cloud computing is a relatively new technology that allows users to access computer infrastructure and other services on a demand basis. Also of note, cloud computing has risen to become one of the most advanced technologies due to its rapid development and excellence in providing services to businesses, institutions, and users in an efficient, timely, and cost-effective manner in order to meet the needs of the users and ensure their satisfaction. User tasks are transmitted over the internet to cloud computing service providers, who are responsible for scheduling the tasks of customers into the resources available in a manner consistent with the service level agreement (SLA) contract between the user and service providers, as well as meeting the quality of service (QoS) requirements agreed upon in the SLA. Task scheduling in a cloud computing environment is a significant problem since it is concerned with increasing the use of resources such as processing power, memory, bandwidth, storage, and so on. To accomplish successful task scheduling, it is necessary to employ the shortest possible makespan and reaction time by completing the activities within a certain time frame. When it comes to information services, cloud computing is defined as a paradigm that is capable of disseminating submitted user requests over a resource pool that has been formed from a large number of physical computers that include virtual machines.

It helps consumers to have access to the information service that they desire. In response to an increment in the amount of available resources added to data centers, an increase in the number of alternate cloud service resources is seen. Users would only be charged for the services they really use, and cloud users will benefit from more efficient service at a lower cost. As a result, cloud computing service providers will see an increase in profits. The continued development of cloud computing apps will increase the number of users who can manage resources and also have access to the cloud computing platform. Resource management and job scheduling were complicated by the enormous reservoir of resources and services available in the cloud, as well as the demands of users and the constant updating of massive resources. As a result, one of the most difficult challenges to solve in cloud computing, which is also a hot topic in the academic community, is the scheduling and management of available resources.

In cloud settings, the cloud service provider must be more dynamic in that physical resources must be brought into and removed from clouds on a more frequent basis than in traditional systems. When it comes to cloud computing, task scheduling was among the most essential features that must be included in any cloud computing framework in order for the resources to be managed efficiently and the cloud customers to be served successfully. It has a significant impact on the allocation of resources for incoming activities that have strict performance optimization requirements. The major objectives of every cloud computing platform are high performance, high profit, usage (utilization), scalability, provision efficiency, and cost-effectiveness. These basic objectives can only be achieved by efficient task scheduling.

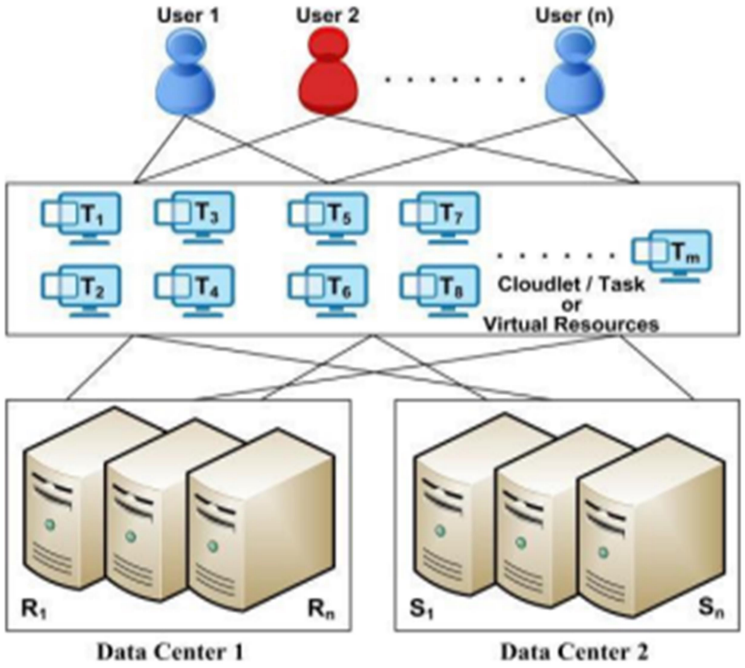


Figure 1.5 Task Scheduling in Cloud Computing

Typically, task scheduling does not always follow a uniform distribution, necessitating effective management and the ability to locate the best available VM resources to complete the tasks. The tasks presented to the cloud framework are diverse, with varying levels of quality of service (QoS). The virtual machine (VM) is a resource that offers a virtualized environment for the cloud user to utilize in order to complete the tasks assigned to it. According to the resource provisioning rules, a

single server machine can be multiplexed into numerous virtual machines (VMs) each with its own operating system and configuration. Every time a user submits a task to the cloud framework, the task scheduler module begins the process of locating the most appropriate virtual machine (VM) using the cloud framework's built-in scheduling algorithm. Real problems arise during the scheduling task, when tiny activities are assigned to a virtual machine with a lot of capacity, or huge tasks are assigned to a virtual machine with limited capabilities. Because to the extended waiting times and the rise in makespan, the overall performance of the system may be compromised. The usage of virtual machines (VMs), the profit margin, and the total throughput are all reduced from the cloud provider's standpoint. As a cloud user, it has a negative impact on user satisfaction due to longer waiting times, higher investments, and a failure to fulfill the Quality of Service (QoS) standards.

As a result, the task scheduling algorithm must be improved in order to meet the demands of both cloud providers and cloud consumers. The optimal scheduling algorithm must provide a method for assigning specific tasks to the most suitable virtual machine resources with the least amount of time spent and the least amount of money spent. According to the business perspective, the task scheduling algorithm must meet quality of service (QoS) standards such as makespan, resource usage, and cost. Typically, swarm-based algorithms are effective for solving NP complete task scheduling issues, such as those involving many tasks. These algorithms are a subset of evolutionary algorithms (EAs), which are a type of meta-heuristic stochastic search methods that are used to solve optimization problems in a deterministic manner. Technically speaking, they are population-based trial-and-error algorithms with meta-heuristic characteristics built into them. The method begins with a starting set of potential solutions that is generated and then updates them repeatedly. Every iteration, the less viable solution is replaced by adding a small number of random modifications, which makes the solution more feasible. Therefore, the algorithm continues to evolve until it finds the best feasible solution based on the objective function.

When it comes to cloud computing, task scheduling and resource distribution are critical components of the process. Users can choose from three different sorts of services provided by cloud computing. Users are expected to send their demands to

the cloud services through a dependable interface that is connected to the internet in order to take use of the three sorts of services available. The prime objective of the service provider is to satisfy the user's request through the use of task scheduling algorithms, which are described in more detail below. Because of the use of scheduling algorithms, the cloud service provider is able to efficiently manage both incoming requests and available IT resources. The scheduling and resource allocation are the two most important aspects that influence the overall performance of the cloud infrastructure. It is necessary to perform task scheduling in an effective manner in order to achieve a trade-off between income and resource usage. Task Scheduling organizes the user request from cloud users in a certain method, allowing the money is being spent in the most efficient manner.

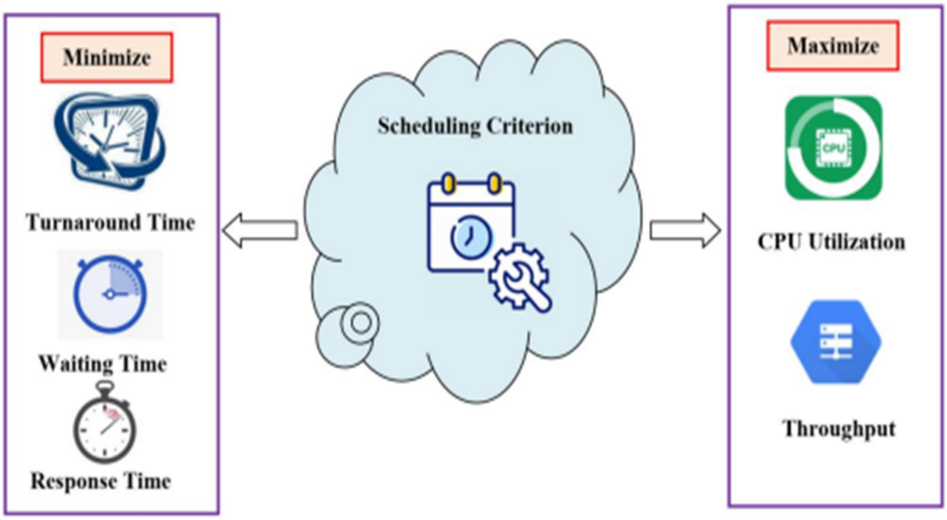


Figure 1.6 Task Scheduling Constraints

Users of cloud computing services make requests for IT resources through the Internet. For each service, there could be a large number of concurrent requests that arrive at the cloud point of service at the same time from a large number of users. If task scheduling is not correctly done, certain jobs will have to wait for a longer period of time, while the short-term activities will be automatically terminated. As a result, task scheduling is regarded as one of the most important quality improvement requirements in the cloud, and it is handled in this study experiment by employing innovative techniques. When conducting task scheduling, the scheduler must take into account the type of the task, improved Quality of Service (QoS), the size of the task,

the time required for execution, the availability of resources, task queues, and load-balancing methods, among other factors. The words load balancing and task scheduling are intertwined and are used in conjunction with one another. It is possible to achieve optimal allocation of resources by performing task scheduling in an appropriate manner, which the primary aim of cloud is computing. NP-hard problems, such as task scheduling, are believed to exist. As illustrated in Figure 1.6, task scheduling algorithms in the cloud are anticipated to maximize CPU utilization and throughput while also minimizing factors such as turnaround time, waiting time, and response time. Task scheduling in the cloud is intended to achieve two objectives: high processing speed and energy conservation, respectively.

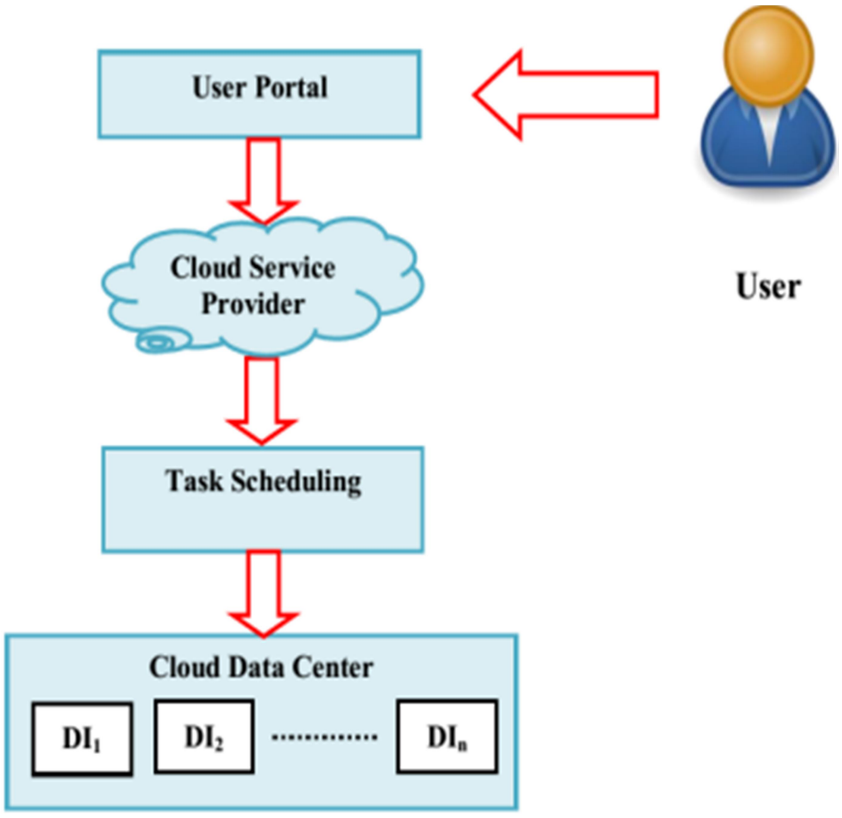


Figure 1.7 Tasks Scheduling in Cloud

As seen in Figure 1.7, the task scheduling cloud architecture is comprised of four components: the User Portal, the Cloud Service Provider, the Task Scheduler, and the Cloud Data Center, among others. The user portal acts as an interface via which cloud consumers may log in and request access to funds available on the cloud infrastructure. The jobs have been broken down into numerous subtasks, each of

which may be done on one of the actual nodes located inside the datacenters. In order to assist the task scheduler in appropriately allocating a task to the right node present in the datacenter, the Cloud Service Provider keeps track of every information regarding resource allocation, usage, and log information. The cluster can be any type of device, ranging from a tiny PC to a workstation in size and functionality. The cloud provider keeps track of the progress of each execution and the availability of available resources in order to control the scheduling of incoming tasks.

1.4.1 The Features of Task Scheduling In the Cloud Computing Environment

Task scheduling and resource deployment have been unified controlled by cloud computing service providers through the use of virtualized technologies in the cloud computing environment. They had been employed to conceal and accomplish users' tasks in an open and transparent manner. As a result of the visible and dynamic flexibility of the cloud platform, as well as the varying requirements for resources from different applications, task scheduling becomes more complicated. Using task scheduling techniques that are primarily concerned with equity or efficiency would raise the cost of time, space, and throughput while simultaneously improving the overall quality of service in the cloud computing environment. When it comes to cloud computing environments, the following features of task scheduling are present:

- **Task scheduling caters to a unified resources platform.**

With the help of virtualized technology, cloud computing allows us to abstract the underlying physical resources (all sorts of hosts, workstations or even PCs, for example) and group them together as an uniform resource pool, protecting heterogeneous resources from being used. It primarily distributes information through a significant number of dispersed computers and provides access to resources through the usage of a data center.

- **Task scheduling is global centralized.**

As cloud computing is a computing model that provides centralized resources to multiple distributed applications through the use of a mirror service, this mirroring deployment can make the execution of heterogeneous procedures and the interoperability of these procedures easier, which was previously difficult to deal with

in the past. As a result of virtualization technology and mirrored services, cloud computing task scheduling is able to produce a worldwide centralized scheduling model, which is advantageous.

- **Each node in the cloud is independent.**

As a result, in cloud computing, each cloud node has its own inner scheduling policy, and the schedulers in the cloud would not meddle with the scheduling policy of these nodes.

- **The scalability of task scheduling.**

In the initial phases of cloud computing, the size of resources available from cloud providers may be restricted. Adding more computational resources of different types might result in a significant increase in the size of the abstraction virtual resources, which in turn increases the demand for application resources. The scaling aspects of task scheduling in the cloud must be met in order for the throughput of task scheduling in the cloud to not be too poor.

- **Task scheduling can be dynamically self-adaptive.**

Depending on the demand, it may be essential to scale up or down apps on the cloud. It is also possible for virtual computing resources in a cloud system to grow or diminish simultaneously time. The resources are forever evolving, some resources may fail, and new resources can enter or resume in the clouds at any point in time.

- **The set of task scheduling.**

One task of task scheduling is used as a unified resource pool scheduling, and it is mainly responsible for such scheduling of applications and cloud APIs; the other part is used for unified port resource scheduling in the cloud, and it is liable for the scheduling of Map Reduce tasks, among other things. Each scheduling, on the other hand, is composed of two two-way processes: the scheduler leases resources from the cloud, and the scheduler callbacks the needing additional once they have been used. The former procedure is referred to as scheduling strategy, while the latter is referred to as callback strategy. The task scheduling set is made up of the scheduling strategy and the callback resource strategy combined together.

1.5 TYPES OF SCHEDULING IN CLOUD

Various types of task schedulers have been developed to make use of the vast amount of cloud-based resources users can access while also improving the management of data centers. These include Job scheduling, Virtual Machine scheduling, Workflow scheduling, Storage scheduling, and Task scheduling, to name a few. Detailed descriptions of the many types of scheduling strategies are provided in the subsequent sections:

- Job Scheduling: Job scheduling is comparable to how the operating task distributes tasks among its available resources. When evaluating a task, it considers the priority of other tasks currently in the queue, as well as the arrival time, waiting time, execution time, and deadline of the task in task. This aids in the accurate mapping of the task to the resources at the appropriate moment. Job scheduling features are included into the cloud computing environment. A job scheduler may be thought of as a user portal that has a greater sense of control over all of the jobs.

- Virtual Machine Scheduling: At first, the cloud service provider establishes a resource pool of virtual machines for its customers. One of the most difficult difficulties encountered by IaaS is the creation of virtual machines. There are several requirements that must be met before a virtual machine can be created, including meeting the Service Level Agreement (SLA), which defines how each host is linked with its Virtual Machine. In addition, throughout the scheduling procedure, a virtual machine transfer is carried out.

- Workflow Scheduling: Defined by Working flow diagrams are a collection of task representations that have a relationship to one another. Typically, a Directed Acyclic Graph (DAG) is used to describe workflows, in which each vertices represent a task and each link reflects the precedence and flow between two tasks in the workflow. According to the amount of resources and the type of resources assigned, as well as how it impacts the time required for the workflow to be completed, an assessment of the economic burden is made.

- Storage Scheduling: It is referred to as storage scheduling when a big data block with a large size and diverse data kinds (Video, Audio, and Text) is kept in clusters at different geographical locations. When the data amount rises, much like with big data, it creates significant complications. According to the information acquired from the resources and storage locations, storage scheduling divides a huge block of data into smaller pieces. Table 1.1 lists the various task scheduling kinds that are available in the Cloud, along with their respective features.

Table 1.1 Comparison of scheduling characteristics in different types of scheduling techniques

Scheduling Type	Main Objective	Important Attribute	Types of Requests
Job	To allocate a pool of task to a set of resources	Priority value(weight) and Size	Request for jobs and software applications
Virtual Machine	Allocating a set of Virtual Machines toa set of hosts	Consolidation and migration of Virtual Machines	Virtual Machine Request
Workflow	Sorting the tasks and allocating a setof ready tasks to resources	Bandwidth, data transmission, and pre-deadline of task	Workflow Request
Storage	Data is distributed in various geographic locationsand data blocks are assigned to its corresponding Resource Clusters.	Chunks of data	Big Data Analytics, Data Mining Application(DM), Storage Requests

1.6 PROCESS OF TASK SCHEDULING

It is possible that the broker in Figure 1.8 is a server or a datacenter where the scheduling rules is in effect. In Figure 1.8, you can see that the resources are available in the datacenters, and that their availability is connected with them. To better understand the task scheduling process, consider the following examples:

1. A first request is sent to the broker by the user in this stage.
2. The broker receives the request and investigates the nature of the request.
3. The task scheduling procedure is used to assign each available resource to the user task that has requested the resource.

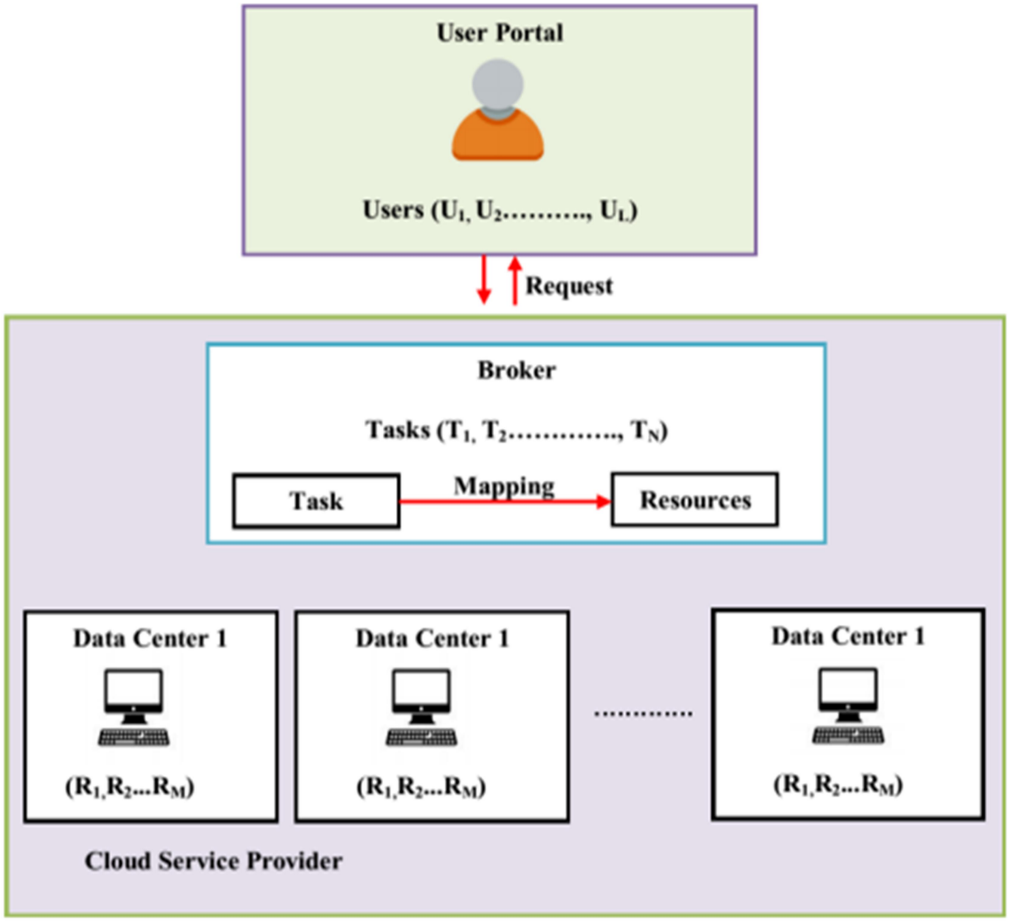


Figure 1.8 Task Scheduling Process

1.7 THE TARGET OF TASK SCHEDULING IN CLOUD

ENVIRONMENT

In cloud computing, the objective of task scheduling is to offer optimal task scheduling for clients while also ensuring that the actual cloud system's throughput and quality of service are maintained at the very same moment. Load balancing, quality of service (QoS), economic principle, and perhaps the most efficient operating time and provides benefits are all objectives to strive for.

1.7.1 Load balance

In the cloud environment, load balancing and task scheduling are in close communication with one another, with the task scheduling mechanism being responsible for the most efficient pairing of jobs and resources. In the cloud, load balancing has become an indispensable step due to the effectiveness of the task scheduling algorithm. Level two loads in task scheduling within cloud computing environments have existed since the load balancing state: the first stage is represented by the virtual machine load, and the second phase represents the resource layer load.

1.7.2 Quality of Service

The cloud is primarily used to offer consumers with computing and cloud storage solutions, and the relationship between workloads from users and resource supply from providers is demonstrated in the form of service quality. Ensure that the quality of service (QoS) of resources is maintained when task scheduling management is involved in work distribution.

1.7.3 Economic Principles

A large number of cloud computing resources are available all around the world. These resources might be owned by a variety of different entities. They are in charge of their own management rules and procedures. Cloud computing, as a business model, delivers relevant services that are tailored to meet the needs of various customers. As a result, the certain of are fair and reasonable. Task scheduling and resource management are driven by the market economy; we must ensure that they are beneficial to both consumers and service providers in order for cloud computing to progress even further.

1.7.4 The best running time

Tasks may be split into different categories based on the demands of the users, and then the optimal running duration and basis for various objectives for every task can be determined. This is especially true for apps. It will significantly enhance the quality of service (QoS) of task scheduling in a cloud context.

1.7.5 The throughput of the system

Throughput is a measure of system task scheduling that optimizes efficiency, and it is an objective that must be considered when developing a business model for cloud computing systems. Both consumers and cloud service providers could reap the benefits of increased throughput.

1.8 GUIDELINES OF SCHEDULING

Scheduled jobs are assigned to specified resources at specific times in accordance with a set of rules known as job scheduling. When it comes to cloud computing, the topic of work scheduling is a significant and fascinating subject. As a result, the work scheduler has to be active. Most of the time, job scheduling in cloud computing is focused on improving the well-organized usage of reserve resources such as bandwidth, memory, and a reduction in completion time. An efficient work scheduling approach must aim to harvest as little response time as possible so that the executing of given jobs takes place in the shortest amount of time possible and that there will be an incidence of in time when revenues fluctuate in the short term. Therefore, fewer job rejections will take place, and more jobs will be sent to the cloud by the customers, all of which will have cumulative effects in expediting the commercial initiation of cloud services. There are several types of scheduling that are based on different criteria, such as static vs. dynamic, centralized vs. distributed, offline vs. online, and so on, as described in the following sections:

1. **Static Scheduling:** Pre-Schedule occupations, all information perceived with regards to realistic assets and tasks and a task relegated once to a save, so it is more settled to adjust dependent on scheduler's standpoint.

2. **Dynamic Scheduling:** Jobs progressively exist for scheduling after some time through the scheduler. It is more flexible than static scheduling, to be capable

of conclusive run time in charge. It is more significant to incorporate burden balance as a principal factor to obtain steady, precise and successful scheduler calculation.

3. **Centralized Scheduling:** As expressed in powerful scheduling, it is a responsibility of incorporated/appropriated scheduler to settle on overall decision. The fundamental government aides of concentrated scheduling are solace of business; viability and more control and nursing on assets. Then again, such scheduler needs versatility, responsibility resistance and viable execution. Since of this burden, it isn't indorse for enormous scope matrices.

4. **Distributed/Decentralized Scheduling:** More of practical for genuine cloud in any case of its frail ability compared to bring together scheduling. There is no fundamental control substance, so neighborhood schedulers‟ solicitations to accomplish and maintain condition of jobs‟ line.

5. **Preemptive Scheduling:** This sort permits all task to add during execution and a task can went to another asset farewell its initially owed asset, accessible for different positions. On the off chance that limitations, for example, need are cautious, this sort is more useful.

6. **Non-Preemptive Scheduling:** Scheduling measure, in which capitals are not being permitted to be redistributed until the sequentially and planned occupation finished its execution.

7. **Co-employable scheduling:** Here, framework have currently a few schedulers, both one is liable for execution certain activity in scheduling interaction to normal framework wide reach dependent on the collaboration of occasions, given rubrics and present plan clients.

8. **Immediate/Online Mode:** Here, scheduler plans any newly showing up work when it comes to with no hanging tight for next time recess on accessible assets right then and there.

9. **Batch/Offline Mode:** The scheduler supplies internal positions as assortment of issues to tackle throughout successive time stretches, with the goal that it is well to plan a task for fit assets relying upon its components.

1.9 SCHEDULING CRITERIA

As previously noted, the various CPU scheduling methods have a variety of characteristics. The selection of a given algorithm may favor one class of processes over the other depending on the circumstances. When choosing an algorithm for a specific situation, we must take into account the characteristics of the algorithms under consideration. There have been several criteria proposed for evaluating CPU scheduling methods. Whose characteristics would be used in the comparison, and which can make a significant difference in the algorithm used to determine which one is the best, is being discussed. The following are some of the criteria to consider.

- **Context Switch:**

This is interaction of putting away and reestablishing setting (condition) of a seized cycle, with the goal that execution can continue from same point later. It is normally computationally escalated, lead to wastage of time and memory, which thusly expands the overhead of scheduler, so the plan of working framework is to upgrade just these switches, the objective is to limit it.

- **Throughput:**

This is a proportion of what amount occupied the CPU is. Typically, the objective is to boost the CPU usage.

- **Turnaround Time:**

This alludes to the all-out time, which is spend to finish the interaction and is the way long it takes the CPU to execute that cycle. The time span from the hour of accommodation of a cycle to the hour of fulfillment is the turnaround time.

- **Holding up Time:**

The absolute time a cycle has been holding up in prepared line. The CPU scheduling calculation doesn't influence the measure of time during which an interaction executes

or inputs yield; it influences just the measure of time that a cycle spends holding up in prepared line.

- **Reaction Time:**

It is the time from the accommodation of a solicitation until the main reaction delivered. Accordingly, the reaction time ought to be low for best scheduling. Thusly, we can reason that a decent scheduling calculation for constant and time-sharing framework should have following qualities:

1. Minimum context switches.
2. Maximum CPU utilization.
3. Maximum throughput.
4. Minimum turnaround time.
5. Minimum waiting time.

1.10 TASK SCHEDULING ALGORITHMS IN CLOUD COMPUTING

Different types of work scheduling algorithms have been developed and tested in the cloud environment, each with a somewhat different configuration. The primary goal of any task scheduling algorithm is to ensure fairness among the jobs throughout their execution and to decrease waiting time, as well as to enhance performance, quality of service, such as throughput end-to-end latency, and overall system performance. When providing services to a large number of users simultaneously in cloud computing, you must consider the reaction time of each individual user, and you cannot make consumers wait for an excessive amount of time.

Scheduling algorithms that are derived from different optimization criteria are prone to a variety of problems. Among the requirements for batch systems are turnaround time and throughput. In interactive systems, the requirements include reaction time and fairness. In real-time systems, fulfilling deadlines is a crucial component of operations. As a result, a scheduling algorithm should be chosen in such a way that it meets all of the essential requirements while also providing efficient service and

effective resource allocation. In a distributed computer system, there are many different types of scheduling algorithms. The majority of them may be implemented in a cloud environment with appropriate verification. The primary advantage of using a job scheduling algorithm is that it allows for high-performance calculating as well as the most optimal scheme amount. When used in a cloud environment, traditional task scheduling algorithms will not be able to provide scheduling. According to a straightforward classification, work scheduling algorithms in cloud computing may be divided into two primary categories: batch mode heuristic scheduling algorithms (BMHA) and online mode heuristic scheduling algorithms (OMHA). Jobs are queued and calmed into a set as soon as they come in the system at the BMHA, according to the organization. The scheduling process will begin when a secure date has been established. The First Come, First Served scheduling algorithm (FCFS), the Round Robin scheduling algorithm (RR), the Min–Min algorithm, and the Max–Min algorithm are all examples of BMHA-based algorithms that are often used. Jobs are scheduled as soon as they arrive in the scheme using the On-line mode experiential scheduling method. Due to the fact that the cloud environment is a dynamic system with varying processing speeds, the on-line mode experience scheduling methods are more suited for a cloud context. In the context of On-line mode experience scheduling algorithms, the much-fit task scheduling algorithm (MFTF) is a good example.

1.11 LOAD BALANCE AWARENESS IN TASK SCHEDULING

Another significant restriction in cloud settings is load balancing, which has a significant impact on the performance of the task scheduling process. Even during task scheduling process, it is necessary to ensure that the resources are neither overloaded nor under loaded in any way. In addition, because the majority of the resources are expected to be distributed at the same time, the scheduling system must guarantee that all of the resources are balanced in relation to its capacity throughout the allocation process. Due to the fact that the issue is not only overcrowded VMs, but also under loaded idle VMs, this helps to improve resource usage. When virtual machines (VMs) remain in an idle state, it has an impact on the service provider's earnings. As a result, in order to make better use of available resources, an effective load balance aware task scheduling algorithm must be included as a component of the task scheduling methodology. Load balancing may be accomplished in a variety of

methods, but in most situations, dynamic load balancing is preferable to static load balancing due to the fact that the cloud computing architecture is entirely real-time oriented. When it comes to dynamic load balancing, there are two options: one is that it may be performed during the task scheduling process, and another is that it can be performed after the task scheduling process.

As an example, in the first scenario, it is accomplished by concurrent monitoring and analysis of the load on VMs as well as resource matching capabilities, with judgments being made based on the load fluctuation before assigning the jobs. In the second instance, load balancing is performed after task scheduling, and if the load is not balanced across the VMs, a few jobs may need to be moved to other resources in order to keep the workload balanced. Migration of virtual machines (VMs) can sometimes result in performance loss in terms of makespan, cost, and extended waiting time. Load balancing during the task scheduling process is extremely fascinating and offers a number of important advantages, such as the ability to minimize the need for needless virtual machine migrations. Aside from that, the load on each virtual machine is examined in light of the present condition of the system, and decision variables are altered in order to direct the task scheduling process. The load balance-based task scheduling approaches utilized in some of the earlier research projects were founded on EA methodologies in some cases.

1.12 OBJECTIVES OF THE STUDY

1. To study the concept of cloud computing.

2. To build a scheduling model capable of utilizing the resources efficiently during load management by reducing the total completion time for the longest task (makespan) based on deadline constraint.

3. To formulate a model to reduce the total cost for task scheduling problem in a cloud computing environment, which consists of the cost of storage, cost of processing and cost of communication based on the budget constraint?

4. To design a model capable of executing tasks based on both deadline constraint and budget constraint. The performance of the proposed models is evaluated by comparing with other applicable algorithms.

1.13 SIGNIFICANCE OF OUR RESEARCH

Cloud computing is a distributed and parallel computing paradigm that is becoming increasingly popular. This type of cloud computing comprises a group of heterogeneous linked datacenters that are reliant on virtualization techniques and that are provided as a dynamically to the customer through a negotiation between cloud service providers and cloud users. With a rising number of cloud users and limited resources (e.g., processing capacity and bandwidth) in the cloud, ensuring high quality of service (QoS) and optimizing resource usage are still important issues in cloud computing, despite recent advances. A lot of the time, when scheduling activities over cloud resources, customers set limitations such as a deadline and a budget. This is because each service provider has a variety of resources with varying costs. Fast resources, on the other hand, are more expensive when compared to inexpensive resources.

In the face of these complicated conflicts between restrictions and the dynamic nature of the cloud, a trade-off solution is required. The most important question of the relevance of this research is how to accomplish effective job scheduling in a cloud computing environment while maintaining a balance between time and cost constraints. With the use of multiple constraints, this research manages the models that execute tasks in the suggested scheduling methods in order to achieve customer happiness while also ensuring that the service provider is satisfied. A scheduling method for reducing the overall completion time (makespan) and reaction time in the cloud computing environment will be given in order to achieve high - performance during task execution in heterogeneous resources will be demonstrated.

1.14 PROBLEM STATEMENT

Because of the flexibility provided by cloud service providers, the number of people who utilize the cloud is growing at an alarming rate. The service providers concentrated on increasing the capacity of their resources, such as infrastructure, to accommodate a high number of customers. However, there is no service provider with a limitless number of resources to meet the demands of peak or fluctuating users. In addition, service providers must ensure that there are an appropriate number of resources available to meet the criteria of QoS, such as execution time, deadline

restrictions, and budget limitations, which are guaranteed to users by the service provider. The implementation of too many tasks or demands on a single resource, on the other hand, will result in tasks interfering with one another. As a result, the performance will deteriorate and become unpredictable, resulting in users being discouraged. It is the assignment and execution of tasks on appropriate resources in the cloud environment that brings this difficult problem to a close. It is only after the user's request is successfully executed that the cloud service providers receive revenue. A provider, on the other hand, is charged a penalty fee when a SLA is violated. Each service provider has a unique service level agreement (SLA), which may differ from the SLAs of other service providers. However, a smaller amount of resources should be employed to complete the activities in order to maintain an acceptable level of quality of service (QoS) or to minimize the time it takes to complete the tasks providing the best task scheduling mechanism that determines the most effective resources while also taking into consideration the significance of QoS metrics such as the execution cost with budget and the execution time with a deadline, or in other words, achieving the satisfaction of both the user and the service provider, are the solutions to these problems.

1.15 RESEARCH METHODOLOGY

1.15.1 Research Design

In this research, tasks will be given and completed on the basis of the resources that will be made accessible within the limits of the study. The time limitations and financial constraints that will be employed in this study are as follows: Although a number of algorithms will be developed to deal with this sort of problem, the vast majority of them will not include the metrics that will be required to evaluate the performance of task scheduling. As a result, we will recommend that our models will be used to obtain effective performance in the task scheduling mechanism.

1.15.2 Proposed Models

The Deadline-Aware Priority Scheduling (DAPS) model will be the first of the scheduling models. As a result of this model, tasks will be planned and assigned to appropriate resources in order to decrease the amount of time required to complete them under the deadline constraint and to reduce the minimum completion time.

When we will be developing an efficient task scheduling model, one of the variables that must be taken into consideration is the level of happiness that users have with the system. The technique will pertain to a concentration on scheduling activities based on a deadline restriction, and as a result, the deadline will be considered to be the time required to complete the work.

When using the Budget-Aware Scheduling (BAS) model, tasks will be scheduled and assigned based on available resources, with the goal of lowering the total time required to complete the tasks while staying within the budget constraints. The cost of the resources that will be utilized is also computed. The service provider will provide services and resources to the user, and the user will determine which resources he requires and then pays for the resources that will be utilized within the constraints of his budget.

Last but not least, there will be the Deadline Budget Scheduling (DBS) paradigm. Tasks will be scheduled and assigned to diverse available resources in this model with two competing Quality of Service requirements: time and cost, all while ensuring that users will be satisfied. In the suggested DBS model, the most important aspects should to decrease the makespan under a user-defined deadline while maintaining or decreasing monetary expenses while not exceeding the user-defined budget.

1.15.3 Task scheduling steps

Cloud computing will work by sending a request or task to a cloud computing service provider, which may contain varied information about the user's requirements, such as a constraint, a priority, or other information. On the other hand, at the service provider that owns a special task scheduling system, the scheduler receives will request from users to schedule their tasks according to the terms of the service level contract between the users and the cloud service provider, in order to ensure the quality of service while earning a profit from the services that will be used by the customers. The scheduler will select the most appropriate resources from among the heterogeneous cloud resources to carry out these tasks in accordance with certain constraints, and the scheduler in this case will be represented by the proposed model and will serve as an intermediary between both the users and the service providers.

1.15.4 Simulation Tool

An efficient tool for simulating task scheduling in homogeneous and heterogeneous cloud computing environments will be required. This tool should be flexible enough to support this environment while also receiving an increasing number of user requests. CloudSim toolkit will be one of the finest open-source toolkits for the development of data centers and Physical Machines (PMs), as well as virtual machines. It will be available for free and open source. The CloudSim toolkit will be a simulator that operates on an event-driven model and will be hosted in the Java programming language. The CloudSim toolkit will enable you to design and enhance the policies that will be applied across all of the CloudSim components. As a result, it will be regarded as a useful study tool since it can replicate the complexity that emerge from different settings. In order to conduct the simulation tests, the following setups will be used on a laptop: 2.5 GHz Intel Corei5 processor, 4 GB of RAM, and a 512 GB hard drive.

1.15.5 Datasets

The datasets that will be used in the cloud computing environment vary depending on the applications that are being utilized. It will be utilized in this research to create two types of datasets the first, which will be obtained from the NASA site for research, and the second, which will be taken by utilizing the CloudSim toolkit, which will generate the dataset.

CHAPTER 2

LITERATURE REVIEW

Gagandeep Kaur (2021) Cloud computing has gotten famous in these days as it gives on-request accessibility of PC resources like processor, stockpiling, and data transfer capacity all through the world. Virtualization helps in the development of data and calculation focuses and makes responsibility adjusting a lot more straightforward and simpler. Virtualization in current days makes the things simpler for cloud clients as they at this point don't have to make enormous forthright speculations and get quick admittance to minimal expense and adaptable resources. Virtualization intends to decrease end-client worries concerning worker upkeep, limitations, and adaptability. Cloud computing utilizes the idea of virtualization and utilizations IT resources as utilities in todays' reality. Cloud computing utilizes cloud commercial center to get to and incorporate the administrations and contributions. Cloud clients and cloud suppliers have various points while dealing with the resources, strategies, and request designs in true situation. In this paper, three-level engineering has been introduced to deal with the resources in financial cloud market considering cutoff time and execution season of the errands for clients.

Shaw, Rachael (2021) throughout the last number of years there has been high speed increase and far and wide reception of cloud-based administrations by a wide range of businesses to work on the adaptability and dependability of their administrations while additionally lessening costs. The prominence of the cloud has summoned significant worry for the high energy utilization and fossil fuel byproducts clear in the activity of data focuses today. One of the vital reasons for high energy utilization is wasteful resource management. While much exertion as of late has been committed to accomplishing further developed energy productive resource management methodologies, execution vulnerability has likewise become a significant prevention for cloud resource management frameworks. Execution vulnerabilities present critical difficulties for arranging and provisioning Virtual Machine (VM) resource portion and planning for the cloud while having adverse consequences by and large in accomplishing more prominent energy efficiencies and worked on Quality of Service (QoS). Not at all like conventional resource management methodologies that depend

on heuristics this work had presents novel anticipating based methodologies for booking and dispensing the computational resources of public cloud foundation. We will probably expand and make headways after existing examination by applying and looking at the exhibition of cutting-edge learning algorithms utilizing genuine data. We investigate how these high-level methodologies can be consolidated to bring about savvy resource management frameworks with the ability to settle on further developed choices under vulnerability. Specifically, we propose and examine a few fascinating varieties and augmentations to issues, for example, work process booking, dynamic VM combination and furthermore VM movement. The outcomes acquired from this examination demonstrate the gigantic capability of embracing learning-based techniques to upgrade resource use, further develop execution and by and large endeavor to arrive at new outskirts in energy effectiveness.

Harvinder Singh et al (2020) Cloud resource management is pivotal for proficient resource allotment and booking that needs for satisfying clients' assumptions. However, it is hard to foresee a fitting coordinating in a heterogeneous and dynamic cloud climate that prompts execution corruption and SLA infringement. Hence, resource management is a difficult assignment that might be compromised in light of the unseemly portion of the necessary resource. This paper presents a precise audit and insightful correlations of existing studies, research work exists on SLA, resource designation and resource planning for cloud computing. Further, conversation on open exploration issues, ebb and flow status and future examination headings in the field of cloud resource management.

Mohd Ameen Imran et al (2020) Cloud computing is a platform that is becoming used these days. Because of the use of designs such as SASS, PASS, and IASS in cloud computing, we have gained numerous advantages over our current platforms, including economies of scale and availability, security as well as other significant enhancements to our computing platforms. Many different types of research are being conducted to improve the reliability of cloud computing platforms for users (individual or corporate) and customers. This research paper examines log management in cloud computing and demonstrates how logs may be utilized as a useful information source on cloud platforms such as Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP), and others. It is proposed that cloud

platforms preserve log files in non-volatile storage in a consistent format, which may be used for virtual machine restoration and account monitoring for faults, as well as for the forensics process. This framework is called Lass scheme. Cloud platforms employ different types of services, and Lass provides a framework for collecting logs from multiple sources based on the type of service used. Lass provides a mechanism through which the user's log may be secured and the privacy of the user's log can be maintained.

Jong Beom Lim et al (2019) as of late, computerized reasoning strategies have been broadly utilized in the software engineering field, like the Internet of Things, enormous data, cloud computing, and portable computing. Specifically, resource management is of most extreme significance for keeping up with the quality of services, service-level arrangements, and the accessibility of the framework. In this paper, we audit and break down different approaches to meet the prerequisites of cloud resource management dependent on man-made consciousness. We partition cloud resource management procedures dependent on man-made brainpower into three classes: haze computing frameworks, edge-cloud frameworks, and insightful cloud computing frameworks. The point of the paper is to propose a shrewd resource management plot that oversees versatile resources by checking gadgets' situations with anticipating their future security dependent on one of the man-made brainpower methods. We investigate how our proposed resource management plan can be stretched out to different cloud-based frameworks.

J. Antony John Prabu (2019) Cloud is a computing innovation; it offers a few types of assistance in the application for a dependable service. The principle highlight of cloud is a capacity to deal with enormous measure of data without thinking about hardware structure and support. Albeit, the upkeep of consistency in data transaction is considered as a significant issue of cloud database. It's one kind of ACID properties. So data transaction in cloud requires better way to deal with maintain the consistency state. In existing, a large portion of the analysts have fostered a few methodologies for this issue yet it is in the earliest stages level. To take care of this issue, this paper focuses on fostering another methodology that is D1TFBC to guarantee more significant level consistency. Further, the performance of the

proposed approach is investigated and results are confirmed with existing methodologies.

Dr. Diwakar Ramanuj Tripathi (2019) Cloud computing ensures different central focuses for the organization of data-raised applications. One essential assurance is decreased expense with a remuneration as-you-go business show. Another assurance is (all things considered) endless throughput by including workers if the responsibility increases. This paper records elective constructions to affect cloud computing for database applications and reports on the outcomes of a thorough evaluation of existing business cloud benefits that have gotten these plans. The point of convergence of this work is on trade dealing with (i.e., peruse and invigorate jobs), rather than assessment or OLAP jobs, which have as of late got a ton of thought. The results are surprising in a couple of ways. Most importantly, it gives the idea that each and every huge dealer have accepted a substitute designing for their cloud organizations. Accordingly, the expense and execution of the organizations contrast basically depending upon the responsibility.

Gawali (2018) Modern technology necessitates the use of cloud computing. The scheduling of tasks and the distribution of resources are critical components of cloud computing. This paper proposes a heuristic approach for performing task scheduling and resource allocation that incorporates the modified analytic hierarchy process (MAHP), bandwidth aware divisible scheduling (BATS) + BAR optimization, longest expected processing time preemption (LEPT), and divide-and-conquer methods. Each job is processed prior to being allocated to cloud resources in this technique, which is accomplished through the use of an MAHP process. The resources are assigned utilizing a combined BATS + BAR optimization technique, which takes into account the bandwidth and load of the cloud resources as restrictions while allocating the resources. The suggested system also preempts resource-intensive activities using LEPT preemption, which is an extension of the previous system. When turnaround time and response time are used as performance metrics, the divide-and-conquer approach improves the proposed system, as demonstrated experimentally through comparison with the existing BATS and improved differential evolution algorithm (IDEA) frameworks when the divide-and-conquer approach is used.

Mitrevski et al (2017) Cloud computing is a new appealing term in the IT world. The expression "Cloud Computing" emerges from the thought for incorporating the capacity and calculation in appropriated data. Its drawn out objectives are to give an adaptable, on – request bundle to the cloud client, giving him considerably more opportunity, adaptability and dependability simultaneously, accomplishing the entirety of the above by utilizing a straightforward "utility computing model". It vows to welcome on-request estimating, less IT overhead and a capacity to scale IT here and there rapidly. The focal point of this work tumbles down on transaction processing applications which work in multi – processing and cloud conditions. All significant sellers have embraced an alternate engineering for their cloud administrations. Therefore, in this paper we will survey some of them and their basic methodologies on improving Cloud Transactions.

Qusay Kanaan Kadhim et al (2017) Cloud computing is the most encouraging current execution of utility computing in the business world, since it gives some critical highlights over exemplary utility computing, for example, flexibility to permit clients progressively increase and scale-down the resources in execution time. In any case, cloud computing is as yet in its untimely stage and encounters absence of normalization. The security issues are the primary difficulties to cloud computing appropriation. Hence, basic ventures like government associations (services) are hesitant to believe cloud computing because of the dread of losing their touchy data, as it lives on the cloud with no information on data area and absence of straightforwardness of Cloud Service Providers (CSPs) systems used to get their data and applications which have made a hindrance against receiving this spry computing worldview. This examination means to audit and group the issues that encompass the execution of cloud computing which a hot region that should be tended to by future exploration.

Sultan Aldossary (2016) Cloud computing switched the world up us. Presently individuals are moving their data to the cloud since data is getting greater and should be available from numerous gadgets. In this manner, putting away the data on the cloud turns into a standard. Nonetheless, there are numerous issues that counter data put away in the cloud beginning from virtual machine which is intend to share assets in cloud and finishing on cloud stockpiling itself issues. In this paper, we present

those issues that are keeping individuals from embracing the cloud and give a review on arrangements that have been done to limit dangers of these issues. For instance, the data put away in the cloud should be private, safeguarding uprightness and accessible. Additionally, sharing the data put away in the cloud among numerous clients is as yet an issue since the cloud specialist organization is dishonest to oversee verification and approval. In this paper, we list issues identified with data put away in cloud stockpiling and answers for those issues which vary from different papers which center around cloud as broad.

Artan Mazreka et al (2016) Cloud Computing is one of the innovations with quick improvement lately where there is expanding interest in industry and the scholarly world. This innovation empowers numerous services and assets for end clients. With the ascent of cloud services number of organizations that offer different services in cloud framework is expanded, accordingly making a rivalry on costs in the worldwide market. Cloud Computing suppliers offer more services to their customers going from infrastructure as a service (IaaS), platform as a service (PaaS), software as a service (SaaS), storage as a service (STaaS), security as a service (SECaaS), test environment as a service (TEaaS). The motivation behind suppliers is to amplify income by their value plans, while the primary objective of clients is to have nature of services (QoS) at a sensible cost. The reason for this paper is to think about and examine a few models and evaluating plans from various Cloud Computing suppliers.

Joundy (2016) Cloud computing is becoming increasingly popular as a substitute for conventional physical hardware computing, particularly in the areas of parallel and distributed computing. Virtualized resources that may be provided on demand, based on the demands of the users, make up cloud computing infrastructures (also known as clouds). Cloud computing has to deal with a large number of user groups, as well as a large number of tasks and a large amount of data, thus the amount of processing required is enormous. The effective scheduling of work has emerged as a critical challenge in the realm of cloud computing that must be addressed. From over years, task scheduling has been a key study topic in a variety of architectures and contexts, beginning with single processors and progressing through multiprocessors and cloud computing. When it comes to computing resources, cloud computing is a paradigm for providing ubiquitous network access to a shared pool of customizable computing

resources where available resources must be verified and planned using an efficient task scheduler before being allocated to clients. The majority of currently available task schedulers failed to meet the necessary criteria and requirements. As part of this thesis, we present a unique hybrid task scheduling algorithm termed (SRDQ) that incorporates the best features of both the shortest-job-first and round robin schedulers, as well as a dynamic variable task quantum that considers splitting the ready queue into two sub-queues, Q1 and Q2. The assignment of jobs to resources from either Q1 or Q2 is done in a mutually beneficial manner, with two assignments from Q1 and one task from Q2. The proposed algorithm was implemented in two different environments, C# and CloudSim, where the results of the experiments and tests revealed that the proposed algorithm significantly improved the average waiting and response times while also partially reducing the starvation when compared to the state-of-the-art algorithms, as demonstrated by the results of the experiments and tests.

J. Antony John Prabu (2015) Cloud Computing is quite possibly the main zones in the current IT businesses. It gives diverse kind of administrations (SaaS, PaaS, and IaaS) contingent upon the clients' necessities. Rather than buying, introducing and keeping up programming in IT businesses can utilize Cloud computing. In this paper, necessities of ACID properties for cloud databases, benefits and impediments of cloud databases are talked about. It additionally pictures the significant issues and difficulties of Database Architectures and Data Transaction Management in Cloud Environment. There are various techniques used to deal with the data, yet every single one of them has its own impediments in cloud climate. At long last this paper gives a comprehensive examination on Transaction Processing System, Cloud Database as a Service (DBaaS), Cloud RDBMS and Cloud data stockpiles to plan a novel design for cloud databases with the help of conventional ACID properties.

D. S. B. R. Kumar (2015) Cloud Computing is quite possibly the main territories in the current IT enterprises. It gives distinctive kind of administrations (SaaS, PaaS, and IaaS) contingent upon the clients' necessities. Rather than purchasing, introducing and keeping up programming in IT businesses can utilize Cloud computing. In this paper, requirements of ACID properties for cloud databases, benefits and impediments of cloud databases are talked about. It likewise pictures the significant issues and

difficulties of Database Architectures and Data Transaction Management in Cloud Environment. There are various strategies used to deal with the data, however every last one of them has its own limits in cloud climate. At last, this paper gives a comprehensive investigation on Transaction Processing System, Cloud Database as a Service (DBaaS), Cloud RDBMS and Cloud data stockpiles to plan a novel design for cloud databases with the help of customary ACID properties.

R. Velumadhava Rao (2015) Cloud Computing pattern is quickly expanding that has an innovation association with Grid Computing, Utility Computing, Distributed Computing. Cloud service suppliers, for example, Amazon IBM, Google's Application, and Microsoft Azure and so on, give the clients in creating applications in cloud climate and to get to them from anyplace. Cloud data are put away and gotten to in a far-off server with the assistance of services given by cloud service suppliers. Giving security is a significant worry as the data is sent to the far off server over a channel (web). Prior to executing Cloud computing in an association, security provokes should be tended to first. In this paper, we feature data related security challenges in cloud-based climate and answers for survive.

Usman Namadi Inuwa (2015) Cloud computing is a computing innovation expecting to share storage, calculation, and services straightforwardly among a gigantic client. Current cloud computing systems present genuine constraint to ensuring the secrecy of client data. Since the data share and put away is introduced in decoded structures to distant machines claimed and worked by outsider service suppliers regardless of it affectability (model contact address, sends), the dangers of uncovering client classified data by service suppliers might be very high and the danger of assaulting cloud storage by outsider is likewise expanding. The motivation behind this examination is to survey investigates done on this innovation, recognize the security hazard and investigate a few strategies for shielding users‟ data from aggressors in the cloud.

Azim et al (2014) Cloud computing has arisen as an effective worldview for web application deployment. Economies-of-scale, flexibility, and pay-per use valuing are the greatest guarantees of cloud. Database management systems serving these web applications structure a basic part of the cloud environment. To serve thousands and an assortment of applications and their immense measures of data, these database

management systems should not just scale-out to groups of product servers, yet in addition act naturally overseeing, issue open minded, and profoundly accessible. In this paper we review, investigate the presently applied transaction management methods and we propose a worldview as indicated by which, transaction management could be portrayed and dealt with.

Nesrine Ali Abd-El Azim (2014) Cloud computing has arisen as an effective worldview for web application sending. Economies-of-scale, versatility, and pay-per use estimating are the biggest guarantees of cloud. Database management systems serving these web applications structure a basic part of the cloud climate. To serve thousands and an assortment of utilizations and their enormous measures of data, these database management systems should not just scale-out to groups of item servers, yet additionally act naturally overseeing, flaw lenient, and profoundly accessible. In this paper we study, investigate at present applied transaction management procedures and we propose a worldview as per which, transaction management could be portrayed and dealt with.

Petter Svard (2014) A vital part of cloud computing is the guarantee of boundless, versatile resources, and that cloud services should increase and down on request. This proposition examines techniques for dynamic resource designation and management of services in cloud datacenters, presenting new methodologies just as upgrades to set up advancements. Virtualization is a vital innovation for cloud computing as it permits a few working framework examples to run on a similar Physical Machine, PM, and cloud services ordinarily comprises of various Virtual Machines, VMs, that are facilitated on PMs. In this proposal, a novel virtualization approach is introduced. Rather than running every PM segregated, resources from various PMs in the datacenter are disaggregated and presented to the VMs as pools of CPU, I/O and memory resources. VMs are provisioned by utilizing the perfect measure of resources from each pool, accordingly empowering both bigger VMs than any single PM can have just as VMs with customized determinations for their application. Another significant part of virtualization is live relocation of VMs, which is the idea moving VMs between PMs without break in service. Live movement considers better PM use and is likewise helpful for regulatory purposes. In the postulation, two upgrades to the standard live movement algorithm are introduced, delta pressure and page move

reordering. The upgrades can diminish relocation personal time, i.e., the time that the VM is inaccessible, just as the all-out movement time. Post copy relocation, where the VM is continued on the objective before the memory content is moved is likewise contemplated. Both client space and in-bit post copy algorithms are assessed in a top to bottom investigation of live movement standards and execution. Effective planning of VMs onto PMs is a critical issue for cloud suppliers as PM usage straightforwardly impacts income. At the point when services are acknowledged into a datacenter, a choice is made on which PM should have the service VMs. This proposition presents an overall methodology for service booking that considers a similar planning software to be utilized across various cloud structures. Various booking algorithms to improve destinations like income or use are additionally considered. At long last, a methodology for constant datacenter solidification is introduced. As VM workloads vary and worker accessibility shifts any underlying planning will undoubtedly become imperfect after some time. The consistent datacenter solidification approach changes this VM-to-PM planning during activity dependent on mixes of management activities, such as suspending/continuing PMs, live moving VMs, and suspending/continuing VMs. Proof-of idea software and a bunch of algorithms that permits cloud suppliers to constantly enhance their worker resources are introduced in the theory.

Linlin Wu (2014) The Cloud computing Software-as-a-Service (SaaS) model has changed the business model for software suppliers. The SaaS model changes the conventional permit-based model to a membership model, which permits clients to get to applications over the Internet without software and equipment forthright expenses and gives decreased upkeep costs. Nonetheless, the key for deals is still consumer loyalty which is at the core of the selling cycle. To ensure Quality of Service (QoS) for consumer loyalty subsequently, the Service Level Agreement (SLA) is carried out among clients and SaaS suppliers, where the principle goals are profit augmentation and expanded portion of the overall industry. To accomplish these goals, there are a few difficulties because of the unique idea of the Cloud climate. First and foremost, the SaaS supplier uses shared framework and different kinds of solicitation loads which can prompt unconventionality in execution and accessibility of resources. Furthermore, there is plausible that current clients may make changes in prerequisites, which can prompt resource redistribution. In that capacity, resource designation may

cause SLA infringement which could diminish the SaaS providers‟ profit edge and notoriety, which means a potential loss of existing clients and likely new clients. Thirdly, SaaS suppliers need to draw in clients with extraordinary requirements and consider market contest from different suppliers to build profit and portion of the overall industry. To conquer the above challenges, most proposed arrangements are centered on the resource management determined to limit cost without adequately thought of customer‟ needs. Hence, to address these difficulties, this proposition proposes algorithms and methods for ideal provisioning of Cloud resources fully intent on amplifying profit and client base by taking care of the dynamism related with SLAs and heterogeneous resources.

Shri V D Garde (2013) The boundless fame of Cloud computing as a favored stage for the organization of web applications has brought about a gigantic number of utilizations moving to the cloud, and the immense achievement of cloud service suppliers. Because of the expanding number of web applications being facilitated in the cloud, and the developing size of data which these applications store, process, and serve – adaptable data management systems structure a basic piece of cloud foundations. There are issues identified with the database security while database is on cloud. The major testing issues are multi-tenure, versatility and the protection. This paper centers around the issues looked in the data security of Relational Cloud. The issues looked by different kinds of occupants and the sort of access into the database causes to modify on the security of data, by dissecting appropriate locking systems on the records got to from the database. Data security in cloud computing tends to the kind of access mode by the clients (for scientific or transaction reason) and the recurrence of data access from the actual area (in shared or no shared plate mode). As needs be, the different data locking techniques are considered and suitable locking system will be carried out for constant applications as in internet business.

Pranita P. Khairnar (2013) Cloud computing has created a great deal of interest and rivalry in the business and it is perceived as one of the main 10 advancements of 2010. It is a web-based service conveyance model which gives web-based services, computing and storage for clients in all market including monetary, medical care and government. Cloud security is turning into a key differentiator and serious edge between cloud suppliers. From the supplier’s perspective a Cloud is an enormous

dispersed system which presents numerous challenges. Cloud computing is plainly one of the present most alluring innovation territories to its expense productivity and flexibility. There is a developing pattern of utilizing cloud services for truly developing storage and data processing needs.

Kashif Munir (2013) Cloud computing has altered the whole procedure that distributed computing was previously characterized by, for example, Grid computing and server client computing, among other things. Many existing IT systems have undergone recent advancements, and cloud computing is a term that refers to the separation of application and information from the underlying infrastructure. A critical component of the quality of service provided by cloud service providers is the security of their cloud computing infrastructure. The moment one begins to execute programs outside of the authorized firewall and moves closer to the public domain, there are security considerations to be considered. Any component of the cloud that is compromised in terms of security can be disastrous for both the business (the client) and the cloud service provider. In this research, we offer a cloud security model and security framework that identify security issues in cloud computing and provide recommendations for addressing them.

Sindhu S (2011) Cloud computing refers to the usage of computers, platform, and software as a service, and it is becoming increasingly popular. It is a type of utility computing in which the client does not need to own the essential facilities and just pays for the services that they actually utilize. Virtual machines serve as delivery vehicles for computing resources. Since a result, task scheduling algorithms play a crucial role in this situation, as their goal is to schedule activities efficiently in order to decrease turnaround time and maximize resource usage. This work offers two scheduling methods for scheduling tasks that take into account the computational complexity of the jobs as well as the computing capacity of the processing units in the system. Experimentation is carried out using the CloudSim toolbox. The experimental findings demonstrate that the suggested algorithms work well even when subjected to high loads.

Muhammad Usman Sana and Zhanli Li (2021) somewhat recently, cloud computing turns into the most requesting stage to determine issues and oversee demands across the Internet. Cloud computing brings tremendous freedoms to run

savvy logical work processes without the prerequisite of having any set-up for clients. It makes accessible practically limitless assets that can be accomplished, coordinated, and utilized as required. Asset scheduling assumes a basic part in the efficient portion of assets to each task in the cloud climate. Anyway alongside these increases many difficulties are needed to be considered to propose an effective scheduling calculation. A productive Scheduling calculation should upgrade the execution of objectives like scheduling cost, load adjusting, makespan time, security mindfulness, energy utilization, dependability, administration level understanding upkeep, and so forth To accomplish the previously mentioned objectives many best in class scheduling strategies have been proposed dependent on half breed, heuristic, and meta-heuristic methodologies. This work surveyed existing calculations according to the viewpoint of the scheduling objective and techniques. We direct a near examination of existing methodologies alongside the results they give. We feature the disadvantages for understanding into additional exploration and open difficulties. The discoveries help scientists by giving a guide to propose proficient scheduling calculations.

Fahd Alhaidari (2021) as of late, there has been critical development in the prominence of cloud computing frameworks. One of the primary issues in building cloud computing frameworks is task scheduling. It assumes a basic part in accomplishing significant level execution and remarkable throughput by having the best advantage from the assets. Subsequently, improving task scheduling calculations will upgrade the QoS, in this way prompting greater supportability of cloud computing frameworks. This paper presents a clever method called the dynamic cooperative heuristic calculation (DRRHA) by using the cooperative calculation and tuning its time quantum in a powerful way dependent on the mean of the time quantum. In addition, we applied the leftover burst season of the task as a factor to choose the progression of executing the task during the flow round. The trial results got utilizing the CloudSim Plus device showed that the DRRHA essentially beat the opposition as far as the normal holding up time, turnaround time, and reaction time contrasted and a few considered calculations, including IRRVQ, dynamic time cut cooperative effort, further developed RR, and SRDQ calculations.

J. Kok Konjaang (2021) Workflow scheduling includes planning enormous tasks onto cloud assets to further develop scheduling efficiency. This has drawn in light of a

legitimate concern for some scientists, who dedicated their time and assets to work on the exhibition of scheduling in cloud computing. Be that as it may, logical work processes are large information applications, thus the executions are costly and tedious. To resolve this issue, we have expanded our past work "Cost Optimized Heuristic Algorithm (COHA)" and introduced a clever work process scheduling calculation named Multi-Objective Workflow Optimization Strategy (MOWOS) to mutually diminish execution cost and execution makespan. MOWOS utilizes tasks parting system to divide huge tasks into sub-tasks to lessen their scheduling length. Besides, two new calculations called MaxVM determination and MinVM choice are introduced in MOWOS for task allotments. The plan reason for MOWOS is to empower all tasks to effectively fulfill their time constraints at a diminished time and financial plan. We have painstakingly tried the presentation of MOWOS with a rundown of work process inputs. The recreation results have shown that MOWOS can successfully perform VM distribution and organization, and well handle approaching streaming tasks with an irregular showing up rate. The presentation of the proposed calculation increments essentially in enormous and extra-huge work process tasks than in little and medium work process tasks when contrasted with the condition of-craftsmanship. It can incredibly lessen cost by 8%, limit makespan by 10% and further develop asset usage by 53%, while additionally permitting all tasks to fulfill their time constraints.

Farooq Hoseiny et al (2021) volunteer computing is an Internet-based disseminated computing framework in which volunteers share their extra accessible assets to oversee enormous scope tasks. Be that as it may, computing gadgets in a Volunteer Computing System (VCS) are exceptionally unique and heterogeneous as far as their preparing power, financial expense, and information moving dormancy. To guarantee both the excellent of Service (QoS) and minimal expense for various solicitations, the entirety of the accessible computing assets should be utilized productively. Task scheduling is a NP-difficult issue that is viewed as one of the fundamental basic difficulties in a heterogeneous VCS. Because of this, in this paper, we plan two task scheduling calculations for VCSs, named Min-CCV and Min-V. The fundamental objective of the proposed calculations is mutually limiting the calculation, correspondence and defer infringement cost for the Internet of Things (IoT) demands. Our broad reenactment results show that proposed calculations can allot tasks to chip

in mist/cloud assets more proficiently than the best in class. In particular, our calculations further develop the cutoff time fulfillment task rates by around 99.5% and decline the complete expense between 15 to 53% in examination with the hereditary based calculation.

Honglin Zhang (2021) in cloud computing, task scheduling and asset designation are the two center issues of the IaaS layer. Proficient task scheduling calculation can work on the coordinating with efficiency among tasks and assets. In this paper, an upgraded heterogeneous soonest finish time dependent on rule (EHEFT-R) task scheduling calculation is proposed to advance task execution efficiency, nature of administration (QoS) and energy utilization. In EHEFT-R, requesting rules dependent on need imperatives are utilized to upgrade the nature of the underlying arrangement, and the improved heterogeneous most punctual completion time (HEFT) calculation is utilized to guarantee the worldwide presentation of the arrangement space. Reenactment tests confirm the viability and predominance of EHEFT-R.

Zeinab Shahbazi (2021) the cutting-edge industry, creation, and manufacturing center is creating dependent on smart manufacturing systems and digitalization. Smart manufacturing's reasonable and significant plan follows data, data, and operational innovation through the block chain, edge computing, and AI to create and work with the smart manufacturing system. This current process' proposed smart manufacturing system thinks about the incorporation of block chain, edge computing, and AI draws near. Edge computing makes the computational responsibility adjusted and comparably gives an ideal reaction to the gadgets. Block chain innovation uses the data transmission and the manufacturing system's transactions, and the AI approach gives progressed data investigation to an enormous manufacturing dataset. Concerning manufacturing systems' computational surroundings, the model takes care of the issues utilizing a multitude knowledge-based methodology. The test results present the edge computing component and comparatively improve the processing season of an enormous number of undertakings in the manufacturing system.

Dinh C. Nguyen (2020) the block chain innovation is surprising the world. Block chain with its decentralized, straightforward and secure nature has arisen as a problematic innovation for the up-and-coming age of various modern applications. One of them is Cloud of Things empowered by the mix of cloud computing and

Internet of Things. In this specific situation, block chain gives creative answers for address difficulties in Cloud of Things as far as decentralization, data protection and organization security, while Cloud of Things offer versatility and adaptability functionalities to improve the effectiveness of block chain tasks. Hence, a novel worldview of block chain and Cloud of Things mix, called BCoT, has been broadly viewed as a promising empowering agent for a wide scope of utilization situations. In this paper, we present a best-in-class survey on the BCoT coordination to furnish general perusers with an outline of the BCoT in different viewpoints, including foundation information, inspiration, and incorporated design. Especially, we additionally give an in-depth review of BCoT applications in various use-case areas like smart medical care, smart city, smart transportation and smart industry. At that point, we survey the new BCoT improvements with the arising block chain and cloud stages, administrations, and examination projects. At last, some significant exploration difficulties and future headings are featured to prod further examination in this promising region.

Rasha Makhlouf (2020) Looking only from the neoclassical viewpoint, cloud computing is cost viable. In any case, as per institutional and transaction cost financial aspects, cloud clients should appraise different expenses past the cost. Such expenses may not be known to cloud clients, prompting neglected assumptions and execution challenges. The point of this paper is to contemplate transaction expenses of cloud computing from the client viewpoint to make the cloud venture less cloudy, for example more educated and very much arranged. This paper applies transaction cost hypothesis to cloud computing through a 360-degree industry investigation. Master interviews with seller, client and consultancy sides were directed to comprehend costs related with cloud computing. Discoveries were approved through a contextual investigation. Discoveries of this examination show that cloud has high 'resource explicitness' because of progress management costs, Meta administrations expenses and business process reengineering costs. Cloud additionally has a significant degree of 'vulnerability' requesting overseeing contracts, putting resources into cloud-explicit checking arrangements and deliberately investigating of the lawful consistence. At last, cloud has high 'transaction recurrence', which makes up for the required ventures set off by 'vulnerability' and 'resource explicitness'.

Tahani Aladwani (2020) Cloud computing is quite possibly the main technology utilized lately, it permits clients (people and associations) to get to computing assets (programming, equipment, and stage) as administrations distantly through the Internet. Cloud computing is recognized from customary computing ideal models by its versatility, movable expenses, openness, dependability, and on-request pay-more only as costs arise administrations. As cloud computing is serving a large number of clients all the while, it should can meet all clients' demands with superior and assurance of nature of administration (QoS). In this way, we need to carry out a fitting task scheduling calculation to reasonably and proficiently meet these solicitations. Task scheduling issue is the one of the most basic issues in cloud computing climate since cloud execution relies primarily upon it. There are different sorts of scheduling calculations; some of them are static scheduling calculations that are considered reasonable for little or medium scale cloud computing; and dynamic scheduling calculations that are considered appropriate for enormous scope cloud computing conditions. In this examination, we endeavor to show the most famous three static task scheduling calculations execution there are: first started things out help (FCFS), short occupation first scheduling (SJF), MAX-MIN. The CloudSim test system affects calculation intricacy, asset accessibility, absolute execution time (TET), complete holding up time (TWT), and all out-finish time (TFT).

Zhihao Peng (2017) Energy utilization has been one of the principles worries to help the fast development of cloud server farms, as it not just expands the expense of power to specialist co-ops yet additionally assumes a significant part in expanding ozone harming substance outflows and hence natural contamination, and contrarily affects framework unwavering quality and accessibility. Subsequently, energy utilization and efficiency measurements have turned into a fundamental issue for equal scheduling applications dependent on tasks performed at cloud server farms. In this paper, we present a period and energy-mindful two-stage scheduling calculation called best heuristic scheduling (BHS) for coordinated non-cyclic diagram scheduling on cloud server farm processors. In the main stage, the calculation allots assets to tasks by arranging, in view of four heuristic techniques and a grasshopper calculation. It then, at that point chooses the most proper technique to play out each task, in view of the significance factor controlled by the end-client or specialist organization to accomplish an answer planned at the perfect opportunity. In the subsequent stage,

BHS limits the makespan and energy utilization as indicated by the significance factor dictated by the end-client or specialist organization and taking into account the beginning time, arrangement time, end time, and energy profile of virtual machines. At long last, a test dataset is created to assess the proposed BHS calculation contrasted with the multi-heuristic asset designation calculation (MHRA). The outcomes show that the proposed calculation works with 19.71% more energy stockpiling than the MHRA calculation. Besides, the makespan is decreased by 56.12% in heterogeneous conditions.

Aida Amini Motlagh (2019) Today, cloud computing has created as one of the significant eminent advances in correspondence and Internet. It offers on request, pay per use admittance to foundation, stages, and applications. Because of the expansion in its fame, the gigantic number of solicitations should be dealt with in a proficient way. Task scheduling as one of the difficulties in the cloud computing upholds the solicitations for appointing a specific asset to perform viably. In the asset the executives, task scheduling is performed where there is the reliance between tasks. Many methodologies and contextual analyses have been produced for the scheduling of these tasks. Up to now, a precise writing survey has not been introduced to find and assess the task scheduling approaches in the cloud computing climate. To survive, this paper presents a SLR-put together examination with respect to the task scheduling approaches that arrange into (a) solitary cloud conditions that assess cost-mindful, energy-mindful, multi-objective, and QoS-mindful methodologies in task scheduling; (b) multicloud climate that assesses cost-mindful, multi-objective, and QoS-mindful task scheduling; and (c) portable cloud climate that is energy-mindful and QoS-mindful task scheduling. The logical conversations are given to show the benefits and limits of the current methodologies.

Preethi Sheba Hepsiba (2019) the need of productive arrangement assets in cloud computing is basic in gathering the presentation necessities. The plan of any asset designation calculation is subject to the sort of responsibility. BoT (Bag-of-Tasks) which is comprised of bunches of autonomous tasks are prevalent in huge scope disseminated frameworks like the cloud and productively scheduling BoTs in heterogeneous assets is a known NPComplete issue. In this work, the smart specialist utilizes support figuring out how to gain proficiency with the best scheduling heuristic

to use in a state. The essential goal of BISA (BoT Intelligent Scheduling Agent) is to limit makespan. BISA is sent as a specialist in a cloud test bed and manufactured responsibility and various arrangements of a private cloud are utilized to test the adequacy of BISA. The standardized makespan is thought about against 15 group mode and prompt mode scheduling heuristics. At its best, BISA produces a 72% below standardized makespan than the customary heuristics and as a rule tantamount to the best conventional scheduling heuristic.

Danlami Gabi et al (2019) Inefficient scheduling of tasks on cloud datacenter assets can bring about underutilization prompting helpless income age. To show effective tasks scheduling on cloud datacenter, the makespan time should be limited. In this paper, we presented a customary Cat Swarm Optimization (CSO) task scheduling procedure as an optimal arrangement. Albeit the CSO is promising as far as assembly speed, certain upgrades are needed to make it productive for cloud task scheduling since it endures capture at the nearby hunt. To beat this, we fused a Linear Descending Inertia Weight (LDIW) condition at the neighborhood search of the CSO method. This prompted better assembly speed and conceivably guaranteed effective tasks planning on virtual assets that limits the makespan time. The proposed CSO-LDIW strategy is carried out on CloudSim test system apparatus with five (5) heterogeneous Virtual Machines (VMs) viable to show its exhibition. The aftereffects of the recreation demonstrate that a correlation with that of the Particle Swarm Optimization-Linear Descending Inertia Weight (PSO-LDIW) and the CSO shows that our proposed CSO-LDIW can plan task viably on cloud asset with a promising makespan time.

N.P. Saravanan (2019) Cloud computing brings computing assets like programming and equipment, it serve administration to the clients through an organization. Significant idea of cloud computing is to share the radiant stockpiling area. In cloud computing, the client occupations are ready and executed with suitable assets to effectively convey the administrations. There are huge measure of task distribution methods that are utilized to achieve task arranging. To further develop the task scheduling strategy, so we proposed technique for proficient task scheduling calculation. Optimization strategies are tackling NP-difficult issues is exceptionally well known. In this proposed procedure, client tasks are put away in the request for

line techniques. The need is planned and dispensed appropriate assets for the task. New tasks are researched and kept in the on-request need of line. The yield of the on-request line is given to the MWOA. It has been demonstrated that this calculation is fit to wipe out optimization issues and beat the current calculations. The technique is proposed to the necessary more number of emphases is decreased. The proposed calculation is contrasted and different scheduling calculations, for example, hereditary calculation, subterranean insect state, standard dim wolf optimization and particle swarm optimization. The results of tests show the better efficiency of the MWOA in articulations of makespan and energy utilization.

Simanta Shekhar Sarmah (2019) Block chain innovation is later and prominent monetary innovation that totally change the deals. It's a decentralized organization that help and utilize assortment of cryptography models. This strong and adaptable got transactions is being incorporated with another prominent computing worldview, cloud computing. In this paper, we make an endeavor to audit about the utilization of block chain in cloud computing system. Initially, the idea of block chain is momentarily talked about with their benefits and inconveniences. Second, the idea of cloud computing is momentarily shown with block chain innovation. At long last, earlier papers are looked into and introduced in even structure. It directs that the examination holes, actually, relates in field of block chain dependent on cloud computing systems. This paper helps the forthcoming scientists in this field for planning novel got models.

Abhishek A. Singh et al (2019) Wide-region Edge Database (WedgeDB) range all around the world and store data nearer to the clients. We term this the Global Edge Data management issue. In such a climate, angles like data stockpiling, recovery, transaction processing, and assurance from vindictive entertainers should be addressed by any data management system that plans to see itself as a reasonable arrangement. Despite the fact that block chain innovation (both permissioned and permission less) has given an approach to address these worries, transaction processing in these conditions is as yet testing. WedgeDB is an endeavor to address these security issues in edge-cloud data systems. The principle objectives of WedgeDB are to help distributed transaction processing combined with secure transaction execution. Data put away in WedgeDB is divided into groups with each

bunch taking care of a novel arrangement of keys. Wedge DB is an assortment of such groups. Each bunch Cx stores a novel arrangement of keys. This parceling plan permits us to construct a distributed transaction model which runs transactions on a subset of groups in the organization. Transactions in WedgeDB are serializable. In WedgeDB, customers perform read activities as a feature of a transaction by means of read demands which can be shipped off any of the WedgeDB hubs. The read activities are added to the transaction's set of experiences. Compose tasks are reserved by the customer until submit is called which sends the transaction object containing the read history and compose activities to a WedgeDB hub to be submitted. Transactions are processed in groups called Epochs. Each bunch keeps a pioneer which gets transactions and gatherings them into ages. A bunch in WedgeDB contains 3f + 1 hubs (where f is the quantity of decent defective hubs) and PBFT is utilized to achieve agreement among the hubs when executing transactions. Keys adjusted during an age are added to a Merkle tree which is utilized to confirm changes to keys dealt with by the group. During transaction execution, evidence of transaction execution is produced as marked data blocks by the hubs in the group. At any rate f + 1 marked messages should be accumulated before an age can be submitted. These data blocks alongside the base of the Merkle tree are put away in a SMR log where every passage in the SMR log compares to an age. Transactions that contain keys from various bunches are executed through two-stage submit. During the get ready stage a distant group executes PBFT inside its neighborhood bunch and checks for reliance infringement prior to pushing forward with the submit stage. Submitted age might not have submitted transactions and consequently an extra boundary is utilized to demonstrate the last dedicated age. This boundary joined with the reliance vector help in discovering serializability infringement and cut short transactions. With WedgeDB, transactions that influence a couple of groups don't need worldwide agreement to submit. Transactions that read keys from various groups don't need agreement at the server hubs and permit data to be perused from a predictable preview of the WedgeDB organization. We in this way, present WedgeDB as a reasonable answer for the issue of worldwide edge data management.

J. Antony John Prabu (2019) Cloud is a computing innovation; it offers a few types of assistance in the application for a dependable service. The principle highlight of cloud is a capacity to deal with enormous measure of data without thinking about

hardware structure and support. Albeit, the upkeep of consistency in data transaction is considered as a significant issue of cloud database. It's one kind of ACID properties. So data transaction in cloud requires better way to deal with maintain the consistency state. In existing, a large portion of the analysts have fostered a few methodologies for this issue yet it is in the earliest stages level. To take care of this issue, this paper focuses on fostering another methodology that is D1TFBC to guarantee more significant level consistency. Further, the performance of the proposed approach is investigated and results are confirmed with existing methodologies.

Gui Huang Et Al (2019) Alibaba runs the biggest online business stage on the planet serving in excess of 600 million clients, with a GMV (net product esteem) surpassing USD 768 billion in FY2018. Online web-based business transactions have three outstanding qualities: (1) extraordinary increment of transactions each second with the opening shot of significant deals and advancement occasions, (2) countless hot records that can undoubtedly overpower system supports, and (3) speedy move of the "temperature" (hot vs. warm vs. cold) of various records because of the accessibility of advancements on various classifications throughout various brief timeframe periods. For instance, Alibaba's OLTP database bunches encountered 122 times increment of transactions on the beginning of the Singles' Day Global Shopping Festival in 2018, processing up to 491,000 deals transactions each subsequent which mean in excess of 70 million database transactions each second. To address these difficulties, we present X-Engine, a compose improved capacity motor of POLARDB worked at Alibaba, which uses a layered stockpiling engineering with the LSM-tree (log-organized union tree) to use equipment speed increase, for example, FPGA-sped up compactions, and a set-up of advancements incorporating non-concurrent writes in transactions, multi-arranged pipelines and gradual reserve substitution during compactions. Assessment results show that X-Engine has beat other capacity motors under such transactional responsibilities.

Dr. Diwakar Ramanuj Tripathi (2019) Cloud computing ensures different central focuses for the organization of data-raised applications. One essential assurance is decreased expense with a remuneration as-you-go business show. Another assurance is (all things considered) endless throughput by including workers if the responsibility

increases. This paper records elective constructions to affect cloud computing for database applications and reports on the outcomes of a thorough evaluation of existing business cloud benefits that have gotten these plans. The point of convergence of this work is on trade dealing with (i.e., peruse and invigorate jobs), rather than assessment or OLAP jobs, which have as of late got a ton of thought. The results are surprising in a couple of ways. Most importantly, it gives the idea that each and every huge dealer have accepted a substitute designing for their cloud organizations. Accordingly, the expense and execution of the organizations contrast basically depending upon the responsibility.

Dileep Mardham (2018) in dispersed transactional systems sent over some hugely decentralized cloud workers, access strategies are ordinarily recreated. Interdependencies promotion irregularities among arrangements should be tended to as they can influence execution, throughput and precision. A few severe degrees of strategy consistency imperatives and implementation ways to deal with ensure the dependability of transactions on cloud workers are proposed. We characterize a look-into table to store strategy renditions and the idea of "Tree-Based Consistency" way to deal with keep a tree design of the workers. By coordinating look-into table and the consistency tree-based methodology, we propose an upgraded form of Two-stage approval submit (2PVC) convention incorporated with the Paxos submit convention with decreased or practically a similar execution overhead without influencing exactness and accuracy. Another storing plan has been proposed which contemplates Military/Defense uses of Delay-lenient Networks (DTNs) where data that should be reserved follows an entire diverse need level. In these applications, data notoriety can be characterized based on demand recurrence, yet additionally based on the significance like who made and positioned point of interests in the data, when and where it was made; higher position data having a place with some particular area might be more significant however recurrence of those may not be higher than more mainstream lower need data. In this way, our reserving plan is planned by thinking about various necessities for DTN networks for safeguard applications. The presentation assessment shows that our reserving plan lessens the general access inertness, store miss and utilization of store memory when contrasted with utilizing storing plans.

Aasha Begum (2018) the cloud foundation gives Database Management System as a help with adaptability and versatility. Cloud DBMS is a more affordable stage for dealing with our data assets. Cloud data are reproduced for high data accessibility. Simultaneous transactions are done in the cloud database. Database clients expect that data ought to be steady when two client's admittance to a similar data simultaneously and can see a similar worth. Because of replication of data in cloud climate, data irregularity may happen between various duplicated hubs. Henceforth Effective locking instruments are expected to deal with such recreated cloud database. This paper proposes a novel calculation for transaction processing to settle irregularity and to control simultaneousness utilizing lock directors and line chief.

Alejandro Zlatko Tomsic (2018) the capacity systems fundamental the present huge scope cloud administrations handle a high volume of solicitations from clients all throughout the planet. These administrations should give quick reaction and a "consistently on" experience. Neglecting to do so brings about diminished client commitment, which straightforwardly impacts incomes. To satisfy these necessities, stockpiling systems duplicate data at various areas around the world. Clients limit latency by associating with their nearest site and, on account of site disappointments, clients can interface with other solid ones. In addition, every area dissipates data across countless servers. Thusly, each site can deal with volumes of solicitations bigger than what a solitary machine can deal with. Transactional ensures work on the improvement of uses that depend on capacity systems. Specifically, transactional separation conceals peculiar conduct sourced in simultaneousness. Nonetheless, in a distributed climate, their application can convert into clients seeing high latencies and administration vacations. This has prompted creation stores — for example, the ones basic administrations like Facebook and Amazon—to shun segregation. This proposal concentrates how to implement segregation in a cloud climate without influencing accessibility and responsiveness. Our first commitment is Cure, a transactional protocol that guarantees significant degree of semantics viable with accessibility: Transactional Causal Consistency (TCC), an intuitive transactional interface, and backing for Convergent data types. TCC guarantees there are no requesting peculiarities, nuclear multi-key updates and steady preview peruses. Fix's intuitive interface permits perusing and refreshing articles in a solitary transaction. CRDTs uncover an engineer amicable API and resolve simultaneous updates securely,

ensuring combination and that no updates are lost. When contrasted with systems that shun consistency and detachment, these certifications limit the oddities brought about by parallelism and appropriation, in this way working with the advancement of uses. Fix includes a component to make refreshes noticeable regarding causal request that brings about insignificant overhead over systems that don't ensure causal consistency. It depends on a novel metadata encoding to improve execution and progress as for condition of-workmanship arrangements. Tentatively, Cure is pretty much as adaptable as a pitifully predictable protocol, despite the fact that it gives more grounded semantics. Transactional protocols like Cure improve on application advancement without trading off accessibility. In any case, their transactional systems show latency overheads that have obstructed their selection at scale. Our subsequent commitment is to investigate how to execute distributed seclusion with no additional deferrals concerning a non-transactional system. In this mission, we find, measure and show a three-path compromise between read disengagement, delay (latency), and data newness. For our examination, we distinguish a read-disconnection property called Order-Preserving Visibility. Request Preserving peruses are more fragile than Atomic peruses, ensured by TCC and more grounded models (e.g., Snapshot Isolation and Serializability). They don't deny a simultaneousness peculiarity called Read Skew, which permits noticing the updates of different transactions mostly. On the positive side, as Atomic Visibility, Order-Preserving Visibility forbids perusing uncertain data and noticing (for example causal) requesting irregularities. The three-path compromise between read confinement, delay (latency), and data newness can be summed up as follows: (I) To ensure perusing data that is the most new immediately is conceivable just under a pitifully detached mode, like that given by the standard Read Committed. (ii) Conversely, peruses that uphold more grounded disconnection at negligible postponement force perusing data from an earlier time (not new). (iii) Minimal-defer Atomic peruses power transactions to peruse refreshes that were submitted and recognized before. (iv) In actuality, insignificant defer Order-Preserving peruses can peruse the updates of simultaneous transactions. (v) Order-Preserving and Atomic peruses at maximal newness require totally unrelated peruse and compose tasks, which may block peruses or composes uncertainly. These outcomes hold freely of different highlights, for example, update semantics (e.g., monotonically requested or not) or data model (e.g., organized or unstructured). Spurred by these outcomes, we propose two separation properties: TCC− and PSI−.

They come about because of corrupting the (Atomic) read assurances of TCC and PSI to Order Preserving Visibility. Utilizing the consequences of the compromise, we use Cure, which is perused calculation in some cases blocks, to make three protocols which are latency ideal. AV keeps up Cure's TCC ensures by debasing newness. The excess two protocols improve newness by debilitating the confinement ensures. Operation gives TCC−, and CV gives Read Committed Isolation, where peruses authorize Committed Visibility. The trial assessment of these protocols upholds the hypothetical outcomes. Every one of the three protocols display comparative latency. The exemption is Cure, which here and there displays higher latency because of blocking. With respect to, CV consistently peruses the most exceptional data. Operation corrupts newness unimportantly under completely tried responsibilities, while under Cure and AV, the newness debasement is extreme.

AR. Arun Arani (2018) Cloud computing deals with an assortment of virtualized assets, which makes scheduling a basic part. In the cloud, a customer might use a few thousand virtualized resources for each task. Thusly, manual scheduling is certifiably not a doable arrangement. The fundamental thought behind task scheduling is to record tasks to limit time misfortune and amplify execution. A few examination endeavors have analyzed task scheduling before. This paper presents a far reaching study of task scheduling systems and the related measurements appropriate for cloud computing conditions. It examines the different issues identified with scheduling procedures and the impediments to survive. Unmistakable scheduling methods are concentrated to find which attributes are to be remembered for a given framework and which ones to ignore. The writing overview is coordinated dependent on three alternate points of view: techniques, applications, and boundary based measures used. Furthermore, future exploration issues identified with cloud computing-based scheduling are recognized.

Fatema Akbar Lokhandwala (2018) In Cloud Data Centers (CDCs), energy cost is the significant cost and numerous scientists intend to lessen it at various levels. Decreasing the energy cost and further developing its efficiency should be possible at the DC level and at the worker level where task scheduling happens. Task scheduling centers around further developing the energy efficiency as well as to distribute the assets productively in an opportune way by streamlining energy utilization. With the

ascent of new innovations in this period, one of the advancements is Block chain. Block chain innovation has been utilized with the cloud as far as security and capacity. This paper proposes a heuristic way to deal with task scheduling in the cloud utilizing block chain. The work is completed to further develop the boundaries, for example, stand by time, execution time, and Service Level Agreement (SLA) that will ultimately assist with decreasing the energy utilization and its expense. The point of this examination is to distinguish the advantages of block chain in the cloud to plan the tasks and investigate its attainability.

Ankit Patel et al (2017) Cloud computing has been an arising innovation throughout the previous few years in which computing administrations and assets are set accessible to clients on expectations through the Internet on a rental premise. Expanded utilization of Cloud has come about into huge enhancement in various Cloud server farms and thus enormous measure of energy utilization. Subsequently a few scientists have attracted their considerations resolving the issue of energy utilization of Cloud server farms. In this work, we expect to resolve the issue of decreasing the force needed to execute earnest or high-need tasks. In light of different factors like degree of criticalness, task cutoff time, task runtime, VM reusability and suspension of a non-earnest task, we characterize the need of task and likewise allocate the task to appropriate virtual machines (VMs) on a viable host. We propose an adjusted force mindful scheduling for dire tasks that consolidates Dynamic Voltage Frequency Scaling (DVFS) and VM Reusability. The proposed technique plans to decrease power-utilization while keeping up with accessibility for need tasks without compromising responsibilities to the client. In future, we intend to reproduce the proposition on CloudSim and contrast it and existing methods for checking its plausibility and measure the upgrades.

Mokhtar A. Alworafi et al (2017) Cloud computing is another age of computing climate which conveys the applications as an assistance to clients over the web. The clients can choose any help from a rundown given by specialist co-ops relying upon their requests or needs. The idea of this new computing climate prompts tasks scheduling and load adjusting issues which become a flourishing examination region. In this paper, we have proposed Scheduling Cost Approach (SCA) that ascertains the expense of CPU, RAM, transfer speed, stockpiling accessible. In this methodology,

the tasks will be disseminated among the VMs dependent on the need given by client. The need relies upon the client spending fulfillment. The proposed SCA will attempt to further develop the load balance by choosing reasonable VM for each task. The consequences of SCA are contrasted and the aftereffects of FCFS and SJF calculations which demonstrates that, the proposed SCA approach fundamentally diminishes the expense of CPU, RAM, transfer speed, stockpiling contrasted with FCFS and SJF calculations.

Qiang Guo (2017) in request to advance the task scheduling system in cloud climate, we propose a cloud computing task scheduling calculation dependent on subterranean insect province calculation. The primary objective of this calculation is to limit the makespan and the all-out cost of the tasks, while making the framework load more balanced. In this paper, we set up the target capacity of the makespan and expenses of the tasks, characterize the load balance work. In the interim, we likewise work on the introduction of the pheromone, the heuristic capacity and the pheromone update strategy in the subterranean insect province calculation. Then, at that point, a few examinations were done on the Cloudsim stage, and the outcomes were contrasted and calculations of ACO and Min-Min. The outcomes shows that the calculation is more effective than the other two calculations in makespan, expenses and framework load adjusting.

Mitrevski et al (2017) Cloud computing is a new appealing term in the IT world. The expression "Cloud Computing" emerges from the thought for incorporating the capacity and calculation in appropriated data. Its drawn-out objectives are to give an adaptable, on – request bundle to the cloud client, giving him considerably more opportunity, adaptability and dependability simultaneously, accomplishing the entirety of the above by utilizing a straightforward "utility computing model". It vows to welcome on-request estimating, less IT overhead and a capacity to scale IT here and there rapidly. The focal point of this work tumbles down on transaction processing applications which work in multi – processing and cloud conditions. All significant sellers have embraced an alternate engineering for their cloud administrations. Therefore, in this paper we will survey some of them and their basic methodologies on improving Cloud Transactions

Qusay Kanaan Kadhim et al (2017) Cloud computing is the most encouraging current execution of utility computing in the business world, since it gives some critical highlights over exemplary utility computing, for example, flexibility to permit clients progressively increase and scale-down the resources in execution time. In any case, cloud computing is as yet in its untimely stage and encounters absence of normalization. The security issues are the primary difficulties to cloud computing appropriation. Hence, basic ventures like government associations (services) are hesitant to believe cloud computing because of the dread of losing their touchy data, as it lives on the cloud with no information on data area and absence of straightforwardness of Cloud Service Providers (CSPs) systems used to get their data and applications which have made a hindrance against receiving this spry computing worldview. This examination means to audit and group the issues that encompass the execution of cloud computing which a hot region that should be tended to by future exploration.

Jin Ho Park et al (2017) Block chain has drawn consideration as the cutting edge monetary innovation because of its security that suits the informatization time. Specifically, it gives security through the confirmation of companions that share virtual money, encryption, and the age of hash esteem. As indicated by the worldwide monetary industry, the market for security-based block chain innovation is relied upon to develop to about USD 20 billion by 2020. Moreover, block chain can be applied past the Internet of Things (IoT) climate; its applications are relied upon to grow. Cloud computing has been significantly embraced in all IT conditions for its proficiency and accessibility. In this paper, we talk about the idea of block chain innovation and its hot exploration patterns. Also, we will concentrate how to adjust block chain security to cloud computing and its safe arrangements exhaustively.

Sultan Aldossary (2016) Cloud computing switched the world up us. Presently individuals are moving their data to the cloud since data is getting greater and should be available from numerous gadgets. In this manner, putting away the data on the cloud turns into a standard. Nonetheless, there are numerous issues that counter data put away in the cloud beginning from virtual machine which is intend to share assets in cloud and finishing on cloud stockpiling itself issues. In this paper, we present those issues that are keeping individuals from embracing the cloud and give a review

on arrangements that have been done to limit dangers of these issues. For instance, the data put away in the cloud should be private, safeguarding uprightness and accessible. Additionally, sharing the data put away in the cloud among numerous clients is as yet an issue since the cloud specialist organization is dishonest to oversee verification and approval. In this paper, we list issues identified with data put away in cloud stockpiling and answers for those issues which vary from different papers which center around cloud as broad.

Ali Gholami (2016) Cloud computing offers the possibility of on-request, flexible computing, given as a utility help, and it is reforming numerous spaces of computing. Contrasted and before techniques for processing data, cloud computing conditions give critical advantages, like the accessibility of mechanized instruments to amass, associate, design and reconfigure virtualized assets on request. These make it a lot simpler to meet hierarchical objectives as associations can without much of a stretch send cloud administrations. Nonetheless, the move in worldview that goes with the reception of cloud computing is progressively bringing about security and protection contemplations identifying with features of cloud computing, for example, multi-occupancy, trust, loss of control and responsibility. Thusly, cloud stages that handle delicate data are needed to send specialized measures and hierarchical shields to stay away from data security breakdowns that may bring about tremendous and expensive harms. Touchy data with regards to cloud computing incorporates data from a wide scope of various territories and spaces. Data concerning wellbeing is an ordinary illustration of the kind of delicate data took care of in cloud computing conditions, and clearly most people will need data identified with their wellbeing to be secure. Henceforth, with the development of cloud computing lately, security and data insurance prerequisites have been advancing to ensure people against observation and data exposure. A few instances of such defensive enactment are the EU Data Protection Directive (DPD) and the US Health Insurance Portability and Accountability Act (HIPAA), the two of which request security conservation for dealing with by and by recognizable data. There have been extraordinary endeavors to utilize a wide scope of instruments to improve the protection of data and to make cloud stages safer. Methods that have been utilized include: encryption, confided in stage module, secure multi-party computing, homomorphic encryption, anonymization, holder and sandboxing innovations. Notwithstanding, it is as yet an

open issue about how to effectively construct usable protection safeguarding cloud systems to deal with touchy data safely because of two examination challenges. In the first place, existing protection and data assurance enactment request solid security, straightforwardness and discernibility of data use. Second, absence of experience with an expansive scope of arising or existing security answers for assemble productive cloud systems. This thesis centers on the plan and advancement of a few systems and strategies for dealing with touchy data suitably in cloud computing conditions. The vital thought behind the proposed arrangements is implementing the security necessities ordered by existing enactment that means to ensure the protection of people in cloud-computing stages. We start with an outline of the primary ideas from cloud computing, trailed by recognizing the issues that should be tackled for secure data management in cloud conditions. It at that point proceeds with a portrayal of foundation material as well as inspecting existing security and protection arrangements that are being utilized in the space of cloud computing. Our first fundamental commitment is another technique for displaying dangers to security in cloud conditions which can be utilized to recognize security necessities as per data insurance enactment. This strategy is then used to propose a system that meets the security prerequisites for dealing with data in the space of genomics. That is, wellbeing data concerning the genome (DNA) of people. Our subsequent commitment is a system for safeguarding protection when distributing test accessibility data. This system is significant on the grounds that it is equipped for get connecting over numerous datasets. The proposition proceeds by proposing a system called ScaBIA for security saving mind picture examination in the cloud. The last part of the thesis depicts another methodology for evaluating and limiting the danger of working system bit misuse, notwithstanding the improvement of a system call mediation reference screen for Lind - a double sandbox.

Akon Samir Dey (2016) lately, cloud or utility computing has reformed the way programming, equipment and organization framework is provisioned and conveyed into creation. A critical part of these huge, various and heterogeneous systems is the constancy layer given by an assortment of data store and database administrations, comprehensively ordered into what is alluded to as NoSQL (Not just SQL) databases or data stores. These come in numerous flavors from basic key-esteem stores and section stores to database administrations with help for SQL-like interfaces. These

systems are basically intended to work at web scale in light of high adaptability and adaptation to internal failure. Subsequently, they regularly penance consistency ensures and frequently support just single-thing predictable activities or no transactions by any stretch of the imagination. While these consistency constraints are fine for a wide class of uses, there are a couple or in some cases just pieces of bigger applications that need ACID transactional ensures to work effectively. To address this, we characterize a data store customer API, we call REST+T (REST with Transactions), an expansion of HTTP that upholds transactions on one store. At that point, we utilize this to characterize a customer facilitated transaction responsibility convention and library, called Cherry Garcia, to empower simple applications improvement across assorted, heterogeneous data stores that each help single-thing transactions. We expand the notable YCSB benchmark, to introduce YCSB+T, to empower us to bunch numerous data store tasks into ACID transactions and assess properties like throughput. YCSB+T likewise gives the capacity to identify and measure data store inconsistencies that outcome from the execution of the responsibility. At long last, we portray our model executions of REST+T in a system called Tora, and our customer facilitated transaction library, additionally called Cherry Garcia, that upholds transactions across Windows Azure Storage (WAS), Google Cloud Storage (GCS) and Tora. We assess these utilizing both YCSB+T and miniature benchmarks.

Artan Mazreka et al (2016) Cloud Computing is one of the innovations with quick improvement lately where there is expanding interest in industry and the scholarly world. This innovation empowers numerous services and assets for end clients. With the ascent of cloud services number of organizations that offer different services in cloud framework is expanded, accordingly making a rivalry on costs in the worldwide market. Cloud Computing suppliers offer more services to their customers going from infrastructure as a service (IaaS), platform as a service (PaaS), software as a service (SaaS), storage as a service (STaaS), security as a service (SECaaS), test environment as a service (TEaaS). The motivation behind suppliers is to amplify income by their value plans, while the primary objective of clients is to have nature of services (QoS) at a sensible cost. The reason for this paper is to think about and examine a few models and evaluating plans from various Cloud Computing suppliers.

Chaowei Yang et al (2016) Big Data has arisen in the previous few years as another worldview giving bountiful data and freedoms to improve as well as empower exploration and choice help applications with phenomenal incentive for advanced earth applications including business, sciences and designing. Simultaneously, Big Data presents difficulties for advanced earth to store, transport, and process, mine and serve the data. Cloud computing offers central help to address the difficulties with shared computing assets including computing, stockpiling, organizing and scientific software; the utilization of these assets has cultivated noteworthy Big Data progressions. This paper studies the two wildernesses – Big Data and cloud computing – and surveys the benefits and outcomes of using cloud computing to handling Big Data in the advanced earth and important science spaces. From the parts of an overall presentation, sources, challenges, innovation status and exploration openings, the accompanying perceptions are offered: (i) cloud computing and Big Data empower science revelations and application improvements; (ii) cloud computing gives significant answers for Big Data; (iii) Big Data, spatiotemporal reasoning and different application areas drive the progression of cloud computing and pertinent advancements with new prerequisites; (iv) inborn spatiotemporal standards of Big Data and geospatial sciences give the source to discovering specialized and hypothetical answers for advance cloud computing and processing Big Data; (v) open accessibility of Big Data and processing ability present social difficulties of geospatial importance and (vi) a mesh of developments is changing Big Data into geospatial examination, designing and business esteems. This audit presents future advancements and an examination plan for cloud computing supporting the change of the volume, speed, assortment and veracity into upsides of Big Data for nearby to worldwide computerized geology and applications.

Yousri Mhedheb (2016) a data community is regularly likewise a Cloud place, which conveys its computational and capacity limit as administrations. To empower on-request asset arrangement with versatility and high unwavering quality, the host machines in data habitats are generally virtualized, which brings a difficult examination subject, i.e., how to plan the virtual machines (VM) on the hosts for energy efficiency. The objective of this Work is to enhance, through scheduling, the energy efficiency of data focus. To help this work a clever VM scheduling instrument plan and execution will be proposed. This component addresses on both load-

offsetting and temperature-mindfulness with a last objective of decreasing the energy utilization of a data place. Our scheduling plan chooses an actual machine to have a virtual machine dependent on the client prerequisites, the load on the hosts and the temperature of the hosts, while keeping up with the nature of the help. The proposed scheduling component on CloudSim will be at last approved, a notable test system that models data focuses provisioning Infrastructure as a Service. For a relative report, we additionally carried out other scheduling calculations i.e., non-power control, DVFS and force mindful ThrMu. The exploratory outcomes show that the proposed scheduling plan, consolidating the force mindful with the warm mindful scheduling methodologies, essentially lessens the energy utilization of a given Data Center as a result of its warm mindful methodology and the help of VM relocation systems.

Jichao Hu (2015) we proposed a task scheduling in cloud computing dependent on knowledge firefly calculation focused on the weaknesses of cloud computing task scheduling. First and foremost, based on cloud model, utilized knowledge firefly calculation with solid capacity of worldwide looking to track down the better arrangement of cloud computing task scheduling then, at that point transformed the better arrangement into the underlying pheromone of further developed firefly calculation, and discovered the cloud computing task scheduling and the calculation's worldwide ideal arrangement through further developed firefly data correspondences and inputs. At last, made correlation trial of the three benchmark work based on MATLAB, the outcomes showed, contrasted and customary insight firefly calculations, the further developed calculation can ideally dispense the assets in cloud computing model, the impact of expectation model time is all the more near genuine time, can proficiently restrict the chance of falling into neighborhood combination, the ideal arrangement's season of target work esteem is abbreviate which address the client's issues more.

J. Antony John Prabu (2015) Cloud Computing is quite possibly the main zones in the current IT businesses. It gives diverse kind of administrations (SaaS, PaaS, and IaaS) contingent upon the clients' necessities. Rather than buying, introducing and keeping up programming in IT businesses can utilize Cloud computing. In this paper, necessities of ACID properties for cloud databases, benefits and impediments of cloud databases are talked about. It additionally picture the significant issues and difficulties

of Database Architectures and Data Transaction Management in Cloud Environment. There are various techniques used to deal with the data, yet every single one of them has its own impediments in cloud climate. At long last this paper gives a comprehensive examination on Transaction Processing System, Cloud DataBase as a Service (DBaaS), Cloud RDBMS and Cloud data stockpiles to plan a novel design for cloud databases with the help of conventional ACID properties.

D. S. B. R. Kumar (2015) Cloud Computing is quite possibly the main territories in the current IT enterprises. It gives distinctive kind of administrations (SaaS, PaaS, and IaaS) contingent upon the clients' necessities. Rather than purchasing, introducing and keeping up programming in IT businesses can utilize Cloud computing. In this paper, requirements of ACID properties for cloud databases, benefits and impediments of cloud databases are talked about. It likewise picture the significant issues and difficulties of Database Architectures and Data Transaction Management in Cloud Environment. There are various strategies used to deal with the data, however every last one of them has its own limits in cloud climate. At last this paper gives a comprehensive investigation on Transaction Processing System, Cloud Database as a Service (DBaaS), Cloud RDBMS and Cloud data stockpiles to plan a novel design for cloud databases with the help of customary ACID properties.

III Albert Horvath (2015) The Internet has created to such a point that numerous insightful articles are considering it the fifth utility, behind water, force, sewage, and phone. The value of the fifth utility is verifiable and will surely just develop. Before long, regular web clients will actually want to accomplish something beyond use it for amusement and shopping. Arising advances can insightfully associate individuals to the data they need to improve their lives. For example, pulse screens can be associated distantly to the web and patients can live at home realizing that if there are any issues, assist will with being called regardless of whether they are separated from everyone else and don't have the ability to settle on the actual decision. The biggest impediment keeping the normal individual away from utilizing the web in more significant manners is trust. The patient with the heart screen should comprehend that his own data, telephone number, distinguishing proof numbers, address, and other individual data is protected from the individuals who may exploit a wiped-out individual. It is hard for the normal web client to believe that their actually recognizable data (PII) is

protected on the web. Practically week after week the American news media reports new, decimating breaks of individual data in big business. Infrequently do they distribute how well a few organizations secure their clients? As per a 2010 study led by the Fujitsu organization, 88% of clients, around the world, are stressed over who approaches their data and practically that amount is stressed over where their data is genuinely put away (Sato, 2010). We offer a study and investigation to show that there is a buyer issue with trust and that there are ways for cloud service suppliers to acquire that trust. A definitive objective of the investigation is to teach clients and CSPs of the difficult that exists and propose approaches to defeat it.

R. Velumadhava Rao (2015) Cloud Computing pattern is quickly expanding that has an innovation association with Grid Computing, Utility Computing, Distributed Computing. Cloud service suppliers, for example, Amazon IBM, Google's Application, and Microsoft Azure and so on, give the clients in creating applications in cloud climate and to get to them from anyplace. Cloud data are put away and gotten to in a far off server with the assistance of services given by cloud service suppliers. Giving security is a significant worry as the data is sent to the far off server over a channel (web). Prior to executing Cloud computing in an association, security provokes should be tended to first. In this paper, we feature data related security challenges in cloud based climate and answers for survive.

Usman Namadi Inuwa (2015) Cloud computing is a computing innovation expecting to share storage, calculation, and services straightforwardly among a gigantic client. Current cloud computing systems present genuine constraint to ensuring the secrecy of client data. Since the data share and put away is introduced in decoded structures to distant machines claimed and worked by outsider service suppliers regardless of it affectability (model contact address, sends), the dangers of uncovering client classified data by service suppliers might be very high and the danger of assaulting cloud storage by outsider is likewise expanding. The motivation behind this examination is to survey investigates done on this innovation, recognize the security hazard and investigate a few strategies for shielding users‟ data from aggressors in the cloud.

Ettazi et al (2015) Advances in remote correspondences and versatility have expanded the utilization of savvy portable applications. Because of the noteworthy

increment of cell phones and the unavoidable remote organizations, an enormous number of portable clients are requiring personalization services redid to their specific situation. The versatile cloud-computing worldview from a setting mindful viewpoint means to discover viable approaches to make cloud services mindful of the setting of their clients and applications. Another significant test for setting mindful cloud services is to abuse the advantages of cloud computing to oversee transaction processing for the duration of the existence pattern of a service. In this paper, we zeroed in on the need of approximately coupled setting supporting parts that work with a transaction-mindful service infrastructure to adjust services to the setting of the client and his cell phone. We propose a cloud-based middleware for transactional service transformation (CM4TSA) by adding the "Variation as a Service" layer into fundamental cloud architecture, to play out the right execution of transactional service as indicated by the client setting.

Azim et al (2014) Cloud computing has arisen as an effective worldview for web application deployment. Economies-of-scale, flexibility, and pay-per use valuing are the greatest guarantees of cloud. Database management systems serving these web applications structure a basic part of the cloud environment. To serve thousands and an assortment of applications and their immense measures of data, these database management systems should not just scale-out to groups of product servers, yet in addition act naturally overseeing, issue open minded, and profoundly accessible. In this paper we review, investigate the presently applied transaction management methods and we propose a worldview as indicated by which, transaction management could be portrayed and dealt with.

Vinit A Padhye (2014) the rise of cloud computing and huge scope Internet services has led to new classes of data management systems, ordinarily alluded to as NoSQL systems. The NoSQL systems give high scalability and accessibility, anyway they give just restricted type of transaction support and frail consistency models. There are numerous applications that require more helpful transaction and data consistency models than those as of now given by the NoSQL systems. In this proposal, we address the issue of giving adaptable transaction support and proper consistency models for bunch based just as geo-repeated NoSQL systems. The models we create in this postulation are established upon the depiction segregation (SI) model which

has been perceived as alluring for scalability. In supporting transactions on group-based NoSQL systems, we present a thought of decoupled transaction management in which transaction management capacities are decoupled from capacity system and incorporated with the application layer. We present two system structures dependent on this idea. In the principal system design all transaction management capacities are executed in a completely decentralized way by the application processes. The subsequent engineering depends on a cross breed approach in which the contention location capacities are performed by a diehard commitment. Since the SI model can prompt non-serializable transaction executions, we explore two methodologies for guaranteeing serializability. We play out a similar assessment of the two models and approaches for ensuring serializability and exhibit their scalability. For transaction management in geo-reproduced systems, we propose a SI based transaction model, alluded to as causal preview seclusion (CSI), which gives causal consistency utilizing non-concurrent replication. The causal consistency model gives more valuable consistency ensures than the possible consistency model. We expand upon the CSI model to give a productive transaction model to halfway recreated databases, addressing the special difficulties raised because of incomplete replication in supporting depiction seclusion and causal consistency. Through trial assessments, we show the scalability and execution of our systems.

Padhye, Vinit A. (2014) the development of cloud computing and huge scope Internet services has brought about new classes of data management systems, ordinarily alluded to as NoSQL systems. The NoSQL systems give high versatility and accessibility, anyway they give just restricted type of transaction support and feeble consistency models. There are numerous applications that require more helpful transaction and data consistency models than those right now given by the NoSQL systems. In this postulation, we address the issue of giving adaptable transaction support and proper consistency models for bunch based just as geo-reproduced NoSQL systems. The models we create in this proposition are established upon the depiction detachment (SI) model which has been perceived as alluring for versatility. In supporting transactions on bunch-based NoSQL systems, we present an idea of decoupled transaction management in which transaction management capacities are decoupled from storage system and coordinated with the application layer. We present two system models based on this idea. In the main system engineering all transaction

management capacities are executed in a completely decentralized way by the application processes. The subsequent engineering is based on a half and half methodology in which the contention location capacities are performed by a diehard loyalty. Since the SI model can prompt non-serializable transaction executions, we explore two methodologies for guaranteeing serializability. We play out a relative assessment of the two structures and approaches for ensuring serializability and show their adaptability. For transaction management in geo-recreated systems, we propose a SI based transaction model, alluded to as causal depiction disconnection (CSI), which gives causal consistency utilizing non-concurrent replication. The causal consistency model gives more valuable consistency ensures than the possible consistency model. We expand upon the CSI model to give an effective transaction model to incompletely repeated databases, tending to the interesting difficulties raised because of halfway replication in supporting preview detachment and causal consistency. Through trial assessments, we exhibit the versatility and execution of our systems.

Shri V D Garde (2013) The boundless fame of Cloud computing as a favored stage for the organization of web applications has brought about a gigantic number of utilizations moving to the cloud, and the immense achievement of cloud service suppliers. Because of the expanding number of web applications being facilitated in the cloud, and the developing size of data which these applications store, process, and serve – adaptable data management systems structure a basic piece of cloud foundations. There are issues identified with the database security while database is on cloud. The major testing issues are multi-tenure, versatility and the protection. This paper centers around the issues looked in the data security of Relational Cloud. The issues looked by different kinds of occupants and the sort of access into the database causes to modify on the security of data, by dissecting appropriate locking systems on the records got to from the database. Data security in cloud computing tends to the kind of access mode by the clients (for scientific or transaction reason) and the recurrence of data access from the actual area (in shared or no shared plate mode). As needs be, the different data locking techniques are considered and suitable locking system will be carried out for constant applications as in internet business.

Jagirdar et al (2013) Cloud Computing is an adaptable innovation that can uphold a wide range of uses. The minimal effort of cloud computing and its dynamic scaling

renders it an advancement driver for little organizations, especially in the creating scene. Cloud sent undertaking asset arranging (ERP), store network management applications (SCM), client relationship management (CRM) applications, clinical applications and versatile applications can possibly arrive at a great many clients. In this paper, we investigate the various ideas associated with cloud computing. Utilizing our encounters on different clouds, we inspect clouds from specialized, and service viewpoints. We feature a portion of the chances in cloud computing, underlining the significance of clouds and showing why that innovation should succeed. At long last, we talk about a portion of the issues that this region should manage.

Waleed Al Shehri (2013) Cloud computing has been the most adoptable innovation in the new occasions, and the database has additionally moved to cloud computing now, so we will investigate the subtleties of database as a service and its working. This paper incorporates all the essential data about the database as a service. The working of database as a service and the challenges it is confronting are examined with a suitable. The construction of database in cloud computing and its working in a joint effort with hubs is seen under database as a service. This paper likewise will feature the significant things to note down prior to embracing a database as a service gives that is best among the other. The benefits and impediments of database as a service will let you to conclude either to utilize database as a service or not. Database as a service has effectively been received by numerous online business organizations and those organizations are getting profits by this service.

Pranita P. Khairnar (2013) Cloud computing has created a great deal of interest and rivalry in the business and it is perceived as one of the main 10 advancements of 2010. It is a web-based service conveyance model which gives web-based services, computing and storage for clients in all market including monetary, medical care and government. Cloud security is turning into a key differentiator and serious edge between cloud suppliers. From the supplier's perspective a Cloud is an enormous dispersed system which presents numerous challenges. Cloud computing is plainly one of the present most alluring innovation territories to its expense productivity and flexibility. There is a developing pattern of utilizing cloud services for truly developing storage and data processing needs.

Katarina Grolinger et al (2013) development in Web innovation and the expansion of cell phones and sensors associated with the Internet have brought about enormous processing and capacity necessities. Cloud computing has arisen as a worldview that vows to meet these necessities. This work centers around the capacity part of cloud computing, explicitly on data management in cloud conditions. Customary social databases were planned in an alternate equipment and software period and are confronting difficulties in gathering the presentation and scale necessities of Big Data. NoSQL and NewSQL data stores introduce themselves as options that can deal with tremendous volume of data. Due to the enormous number and variety of existing NoSQL and NewSQL arrangements, it is hard to fathom the space and surprisingly more testing to pick a fitting answer for a particular undertaking. Accordingly, this paper audits NoSQL and NewSQL arrangements with the goal of: (1) giving a viewpoint in the field, (2) giving direction to specialists and scientists to pick the proper data store, and (3) distinguishing difficulties and openings in the field. In particular, the most unmistakable arrangements are looked at zeroing in on data models, querying, scaling, and security related capacities. Highlights driving the capacity to scale read asks for and compose demands, or scaling data stockpiling are examined, specifically apportioning, replication, consistency, and simultaneousness control. Moreover, use cases and situations in which NoSQL and NewSQL data stores have been utilized are talked about and the appropriateness of different answers for various arrangements of uses is inspected. Therefore, this investigation has recognized difficulties in the field, including the enormous variety and inconsistency of phrasings, restricted documentation, inadequate examination and benchmarking measures, and nonexistence of normalized query languages.

Lizheng Guo (2012) Cloud computing is an arising innovation and it permits clients to pay as you need and has the superior. Cloud computing is a heterogeneous framework too and it holds enormous measure of utilization data. During the time spent scheduling some escalated data or computing a concentrated application, it is recognized that upgrading the moving and handling time is urgent to an application program. In this paper to limit the expense of the handling we figure a model for task scheduling and propose a particle swarm optimization (PSO) calculation which depends on little position esteem rule. By uprightness of contrasting PSO calculation and the PSO calculation installed in hybrid and transformation and in the nearby

exploration, the examination results show the PSO calculation combines quicker as well as runs quicker than the other two calculations in an enormous scope. The trial results demonstrate that the PSO calculation is more appropriate to cloud computing.

Xiaoli Wang et al (2012) High energy utilization of data focuses has turned into an incredible obstruction to the advancement of cloud computing. This paper essentially centers on how to further develop the energy efficiency of workers in a data place by proper task scheduling systems. In light of MapReduce, Google's gigantic data handling system, another energy-productive task scheduling model is proposed in this paper. To tackle this model, we set forward a viable hereditary calculation with useful encoding and interpreting techniques and uncommonly planned hereditary administrators. In the meantime, with the end goal of speeding up this present calculation's joined speed just as improving its looking through capacity, a neighborhood search administrator is presented. At last, the examinations show that the proposed calculation is viable and effective.

CHAPTER 3

ENHANCED TASK SCHEDULING IN CLOUD COMPUTING BASED ON DEADLINE-AWARE MODEL

In this chapter, an essential component of task scheduling in the cloud computing environment will be discussed, namely, scheduling jobs based on a deadline time restriction in order to decrease makespan, reaction time, and various other metrics, among other things. We will also operate in a heterogeneous environment, employing separate tasks and distributing them across existing funds, while taking into consideration the load balance among the virtual machines, to achieve our goals. Also included are the metrics employed in the study, such as the makespan, reaction time, and resource usage underneath a deadline restriction, as well as how these are altered whenever the suggested approach is applied.

3.1 PREAMBLE

It is a distinct benefit of cloud computing because the service provider lends resources to the customer while also allowing them to satisfy level of service criteria by purchasing the rights to utilize those resources.

Today's world is plagued by challenges such as task scheduling and load balancing, which are exacerbated by the fact that many algorithms in the cloud computing environment strive to provide high performance and efficient work scheduling. Thus, developing an algorithm that makes effective use of available resources is difficult. Scheduling may be defined as the allocation of a number of resources for the execution of activities in a straightforward manner. When faced with work restrictions such as a strict deadline and a limited budget, the user wants to determine the overall cost of completing his tasks within the constraints of his budget and schedule. As soon as the user specifies a budget and a deadline, the tasks must be completed within those parameters. However, when the activities have a smaller budget or require immediate processing/attention, certain jobs may be done and others may not, depending on the limits imposed on them. The purpose of a task scheduling optimization algorithm on deadlines is to ensure that each performed job is completed by the deadline. As a result, the task scheduling algorithm can decide which jobs must

be completed in a short period of time. As a result, one of the most significant objectives of the deadline concept is the work scheduling inside that time constraints. Aside from task assignment, the primary purpose of task scheduling in cloud computing is to minimize the makespan (total completion time), which is defined as the time it takes for a work to be completed. To achieve efficient scheduling mechanisms based on deadlines in cloud computing environments, a number of algorithms have been developed that not only concentrate on achieving the deadline constraint once implementing the tasks, but also on going to meet the user satisfaction requirements and load balancing between available resources while maximizing resource utilization. When a scheduling algorithm is capable of absorbing a certain amount of misunderstanding or ambiguity throughout the course of a task's execution, it may be considered to be extremely efficient. In order to increase application reliability, achieving the final deadline is a critical problem, which must be addressed.

3.2 DEADLINE RESTRICTION

It is the time limit that dictates when a job or collection of activities will be finished that is represented by the deadline constraint. Each job has a deadline constraint that is determined by the users, and this deadline should be equal to or higher than the entire completion time (makespan) of the work under consideration. It is also well recognized that the cloud computing system provides a variety of services, and that it is required to assess their quality, which is accomplished through the use of Quality of Service (QoS), where the QoS is included in the SLA agreement. Aside from that, the QoS incorporates a number of limitations such as deadlines and budgets, all of which are agreed upon between the cloud user as well as the cloud service provider, as seen in Figure 3.1.

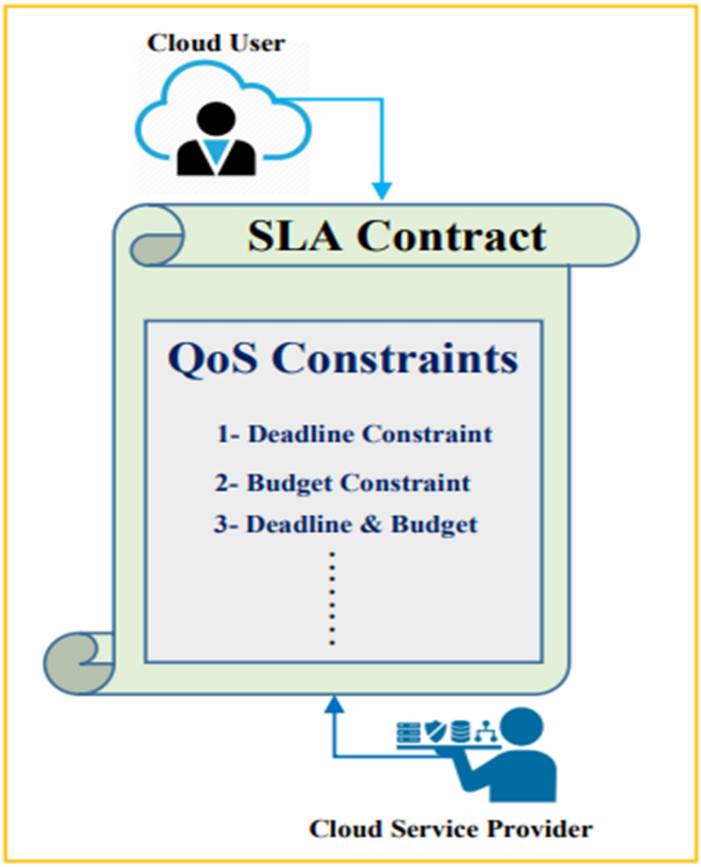


Figure 3.1: Quality of service constraints

3.3 TASK SCHEDULING DEPENDENT ON DEADLINE-AWARE MODEL

In cloud computing, job scheduling is a significant topic, and academics are working to develop the most efficient methods for task scheduling using the existing funds in the cloud computing environment. The cloud service provider and the cloud user are the two most essential participants in the cloud computing ecosystem. The cloud service provider is a company that provides cloud computing services. The criteria of each of them must be met when the tasks ought to be completed using cloud resources that are readily available to them. If a cloud service provider wants to maximize the usage of available resources, for example, that is what they want to do. When completing operations in the cloud, the cloud user is concerned with attaining the best possible productivity and effectiveness while keeping the overall completion time to a minimum (makespan). Work features such as task length, projected execution time,

and deadline time are all distinct for each user's task. Beginning with the proposal of the Deadline-Aware Priority Scheduling (DAPS) model, this research will proceed to examine the many aspects of the model. The DAPS model's method is to schedule tasks and allocate them to available resources while reducing the makespan based on deadline constraints and shortest remaining time requirements, among other things. When developing an efficient work scheduling model, one of the variables that must be taken into consideration is the level of pleasure experienced by the users. In this research topic, we are concerned with scheduling tasks based on a deadline restriction. As a result, we make the assumption that the deadline is the time required to complete the work, and we will describe how we came to this conclusion later. According to this concept, the cloud environment incorporates a data centre comprised of heterogeneous physical computers or servers that are made up entirely of virtual machines (cloud computing).

On Figure 3.2, you can see the primary symbols for the qualities that were employed in the situation for this study.

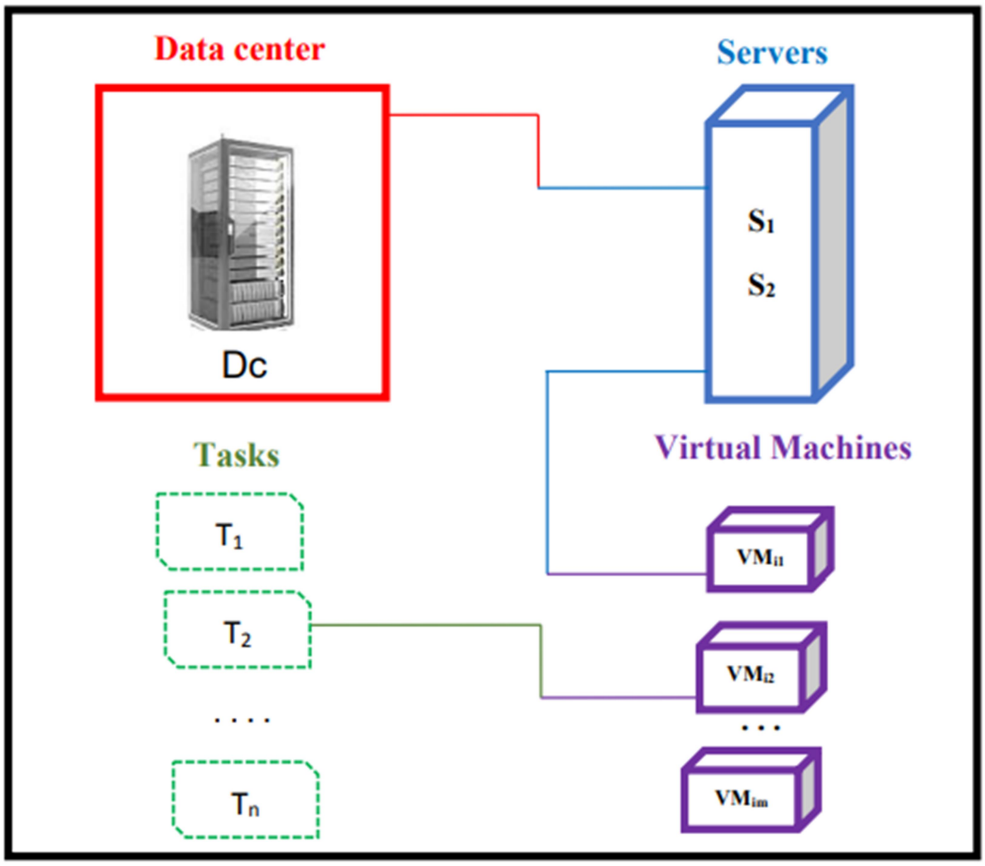


Figure 3.2: The primary symbols of attributes for our study

The deadline constraint is emphasized in the DAPS model since it is a key criterion in determining the success of the job scheduling mechanism. When a task's runtime does not fulfill the deadline restriction, the task is considered violated. We concentrated our efforts on developing our proposed DAPS model for this challenge. Our suggested approach, which is illustrated in Figure 3.3, states that tasks with a deadline are delivered to the service provider by the end-users themselves. On the other hand, we presume that the deadline corresponds to the time required to complete the assignment. In the following step, the service provider will look for resources available to schedule jobs in accordance with our model. The DAPS model finds the virtual machine (VM) that fulfils the deadline, then quantifies the projected completion time for distributing the tasks to the right VM that has less competition time while still meeting the deadline, as described above. Last but not least, the DAPS refreshes the waiting time for the virtual machine. If the job is not completed by the deadline, it signifies that no VM can complete the task before the deadline expires.

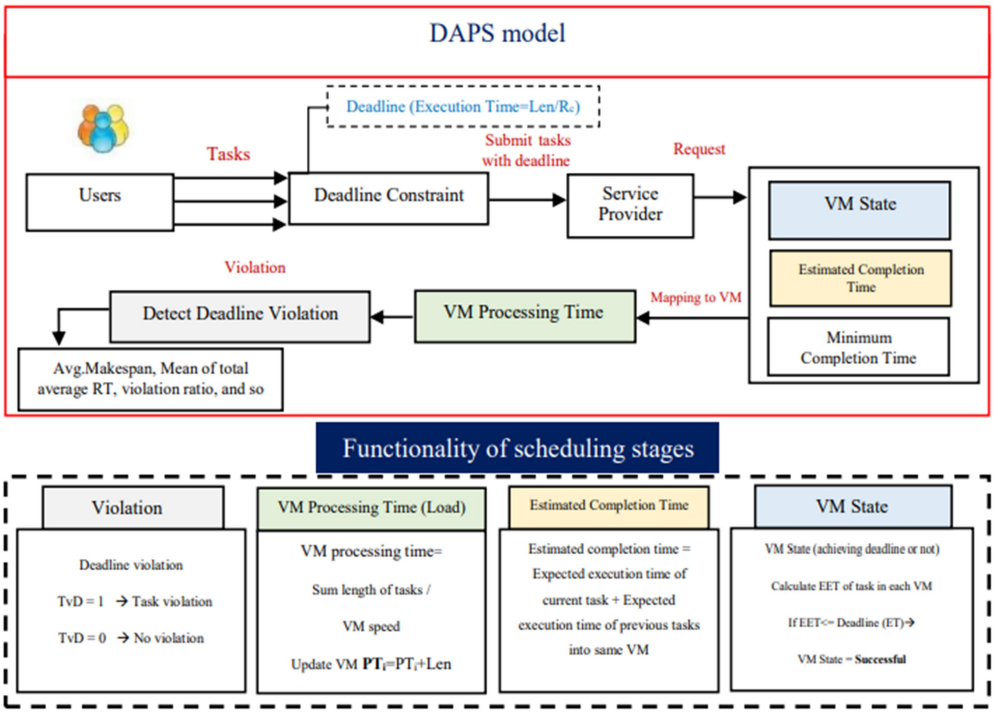


Figure 3.3: Suggested Deadline-Aware Priority scheduling model

In the DAPS model, tasks are prioritized in increasing order depending on their length, with the longest tasks being placed first. The principle of first come, first served, however, is applied when two jobs are almost the same duration, as opposed to when they are not. The deadline constraint indicates the Execution time (EX) of the job, which is computed using Condition 3.1 and specifies the time required to complete the work.

$$EX = \frac{Len}{R_C} \quad (3.1)$$

Where Len is the duration of the work and RC denotes the CPU of the virtual machine

3.3.1. Case Study

Consider the following scenario:

3 virtual machines= {VM1, VM2, VM3}

Tasks= {T1, T2, T3, T4, T5}

VMs' speed= {700, 500, 300}

Length of task= {1000, 40000, 300000, 400, 125000}

We anticipate that the following jobs will be allocated across the virtual machines:

Then there's the task's deadline:

$$\text{Task1.deadline: } \frac{1000}{700} = 1.4. \quad \rightarrow VM_1$$

$$\text{Task2.deadline: } \frac{40000}{500} = 80. \quad \rightarrow VM_2$$

$$\text{Task3.deadline: } \frac{300000}{300} = 1000. \quad \rightarrow VM_3$$

$$\text{Task4.deadline: } \frac{400}{700} = 0.5. \quad \rightarrow VM_1$$

$$\text{Task5.deadline: } \frac{125000}{500} = 250. \quad \rightarrow VM_2$$

The Expected Execution Time (EET) is the time it is expected that tasks will take to complete in all virtual machines (EET=EETT1, EETT2,...EETTn). The job in each virtual machine is then found to be comparable to the deadline constraint to determine the status of the virtual machine, i.e., to determine whether of the virtual machines meets the deadline, and is classified as effective or failed, as determined utilizing Equation 3.2

$$\textbf{VM's state} = \begin{cases} \text{Successful} & \text{if EET} \leq \text{ deadline} \\ \text{Unsuccessful} & \text{otherwise} \end{cases} \quad (3.2)$$

In the preceding sentence, the predicted execution time of job in every virtual machine was calculated.

T_1. Expected execution time = $\{\frac{1000}{700}, \frac{1000}{500}, \frac{1000}{300}\}$.

T_2. Expected execution time = $\{\frac{40000}{700}, \frac{40000}{500}, \frac{40000}{300}\}$.

T_3. Expected execution time = $\{\frac{300000}{700}, \frac{300000}{500}, \frac{300000}{300}\}$.

T_4. Expected execution time = $\{\frac{400}{700}, \frac{400}{500}, \frac{400}{300}\}$.

T_5. Expected execution time = $\{\frac{125000}{700}, \frac{125000}{500}, \frac{125000}{300}\}$.

As shown in Figure 3.4, when the deadline of each job is compared with its predicted processing time, the task deadline is met, and the stages are explained in more detail in Condition 3.2:

Task1.deadline obtained in {VM1}.

Task2.deadline obtained in {VM1, VM2}. →{Choose VM that returns minimum CT}

Task3.deadline obtained in {VM1, VM2, VM3}. →{Choose VM that returns minimum CT}

Task4.deadline achieved in {VM1}.

Task5.deadline achieved in {VM1, VM2}. →{Select VM that returns minimum CT}

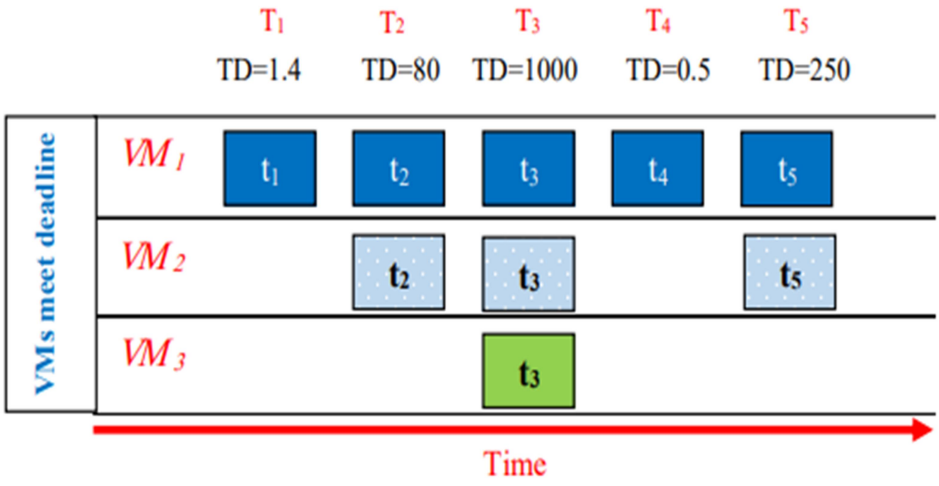


Figure 3.4: The VMs obtain the task deadline

Following a recent status check of the virtual machines, tasks are routed to those that have the shortest completion time. This is accomplished by computing the anticipated completion time of each job and then finding which virtual machine has the shortest completion time. If the VMs fail to complete the work before the deadline, the VMs are regarded failed as well as the task is considered breached. As a result, the task that was violated will be assigned to the VM that has the shortest finish time between all failed VMs.

The Completion Time (CT):

'Completion Time (CT)' is a number that represents the total of expected execution times for all prior tasks plus the estimated execution times for the currently running job. The time required to complete the task may be determined using Condition 3.3.

$$\mathbf{CT} = \mathrm{EET_t} + \sum_{j=1}^{T-1} \mathrm{EET_j} \qquad (3.3)$$

As previously said, the cloud computing ecosystem is comprised of a diverse range of resources. Because varied resource characteristics result in varying task completion times in different VMs, the completion time matrix for each job [1, m] is generated using Condition 3.3 for each task [1, m]. The Completion Time Matrix (CTM) displays the expected time required to complete a job on each virtual machine. The

item (j,i) in the CTM specifies the projected completion time of job j on the virtual machine i. There are two parameters for each requested task: t for task heterogeneity and VM for virtual machine heterogeneity. For each requested task, there are two criteria: t for task heterogeneity & VM for virtual machine heterogeneity.

In this case, if we have tasks T= T1, T2, T3,.... Tn and VM = VM1, VM2,....,VMm, then we may deduce the CTM from the given tasks as follows:

$$CTM = \begin{array}{c} \\ T_1 \\ T_2 \\ . \\ . \\ . \\ T_n \end{array} \begin{array}{c} \begin{array}{cccc} VM_1 & VM_2 & \dots & VM_m \end{array} \\ \begin{pmatrix} CT_{11} & CT_{12} & \dots & CT_{1m} \\ CT_{21} & CT_{22} & \dots & CT_{2m} \\ . & . & \dots & . \\ . & . & & . \\ . & . & & . \\ CT_{n1} & CT_{n2} & \dots & CT_{nm} \end{pmatrix} \end{array}$$

The method of operation consists in converting the bare minimum of CTji into particular VMi for each job. Following the assignment of the job, the Processing Time (PTi) of the virtual machine will be changed, as well as the work that has been scheduled will be deleted from the task list set. The processing time is computed in the same way as in the previous condition.

$$PT_i = \frac{\sum_{j=1}^{n} Len_j}{R_C} \quad (3.4)$$

Following that, PTi is modified in accordance with Condition 3.5.

$PT_i = PT_i + Len$ (3.5)

At the end of the process, the number of tasks that have been violated is determined, where the number of violations is equivalent to the sum of tasks that have been late. The number of violations is specified by Condition 3.6, where the violated task equals 1 if the task deadline (TvD) is exceeded, and otherwise TvD equals 0.

$$\textbf{Number of Violations} = \sum_{j=1}^{n} TvD \quad (3.6)$$

Figure 3.5 depicts the steps we took in the DAPS model to schedule models depend on a deadline restriction, as shown in the example.

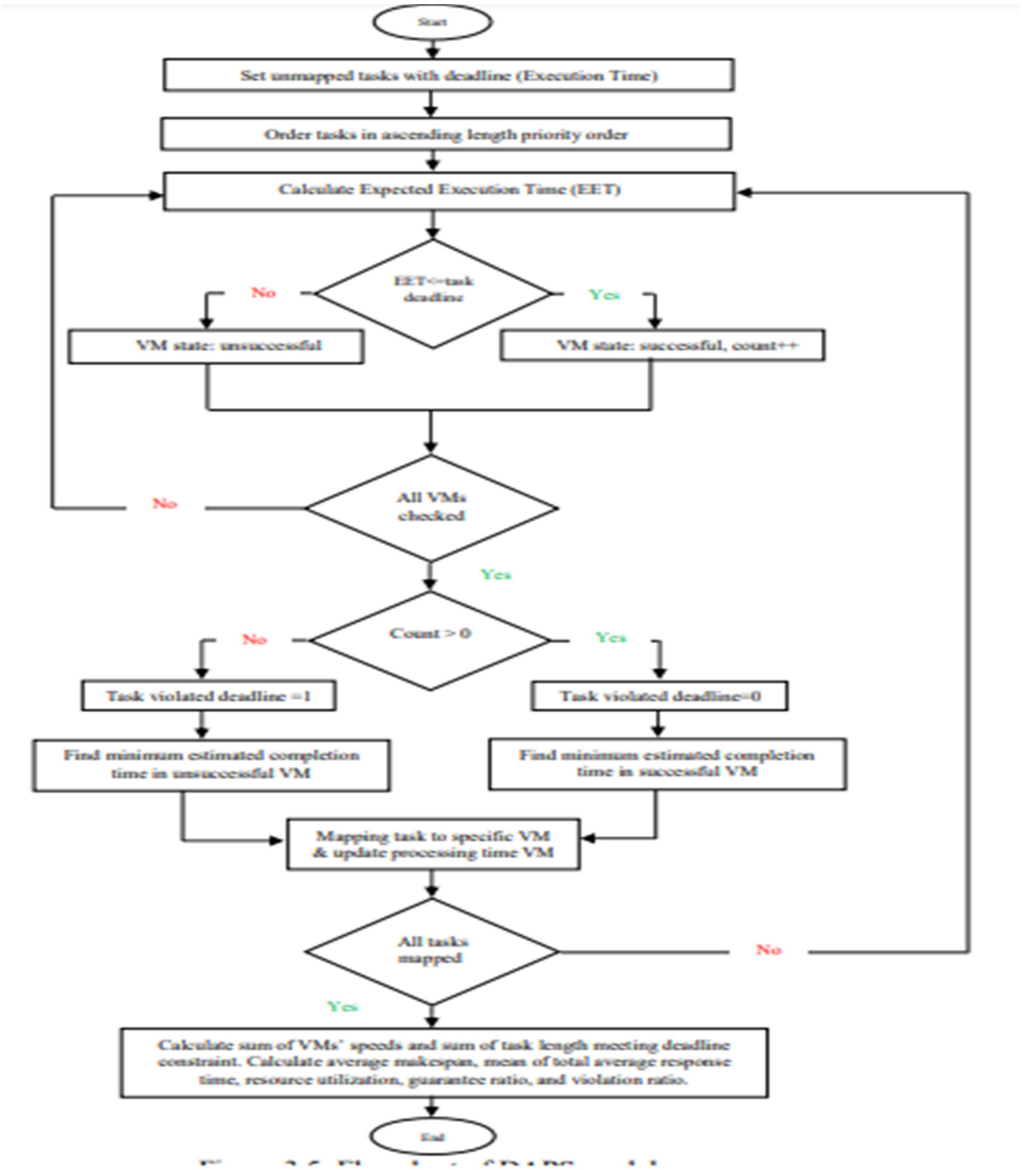


Figure 3.5: Flowchart of DAPS model

3.3.2 Steps of Deadline-Aware Priority Scheduling Model

The DAPS model that has been suggested is as regards:

Input	*Set the unmapped tasks with length priority time order*
Output	*Minimize the makespan based on deadline constraint*
1.	**For** j=1 to n // 'n' order tasks in ascending order based on length priority

2.	// Check VM state:
	For i=1 to m //In each VM resource
	i. Calculate the Expected Execution Time (EET) of task from ET that is calculated in the **Equation 3.1**
	ii. **If** EET_T <= task deadline
	iii. VM state: successful
	iv. Count++
	v. **Else**
	vi. VM state: unsuccessful
	End for

3.	**If** count > 0 i. TvD =0 ii. Find minimum estimated CT in specific successful VM from CTM **Else** i. TvD= 1 ii. Find minimum estimated CT in specific unsuccessful VM from CTM
4.	Mapping task to specific VM Update processing time VM (PT_i) based on the **Equation 3.5** **End for** // All task mapped
5.	**Computation performance** 1. Calculate sum of VMs speeds and sum of task meet deadline 2. Calculate: i. Average makespan based on the **Equation 3.8**

	ii. Mean of total average response time based on the **Equation 3.11** iii. Resource utilization based on the **Equation 3.12** **iv.** Guarantee ratio based on the **Equation 3.13** **v.** Violation number based on the **Equation 3.6** vi. Violation ratio based on the **Equation 3.14.** vii. Improvement of makespan ratio based on the **Equation 3.15.**

3.4. PERFORMANCE METRICS

We employed a variety of indicators to assess the efficacy of the DAPS model, including:

3.4.1 Makespan

The term "makespan" relates to the utmost amount of time required to complete a task. To put it another way, it is the amount of effort spent providing resources to end users. In order to achieve great performance, we must keep the time required for wealth distribution to a bare minimum. As seen in Condition 3.7, this is written as

Makespan = max {CTi} (3.7)

When i ∈ VMs are concerned this is followed by the calculation of the averaged of makespan, which would be described in Condition 3.8.

$$\textbf{Avg. Makespan} = \frac{\sum_{i=1}^{m} \textbf{makespan}_i}{\textbf{m}} \quad (3.8)$$

Where m is number of VMs

3.4.2 Response Time (RT)

Response time is defined as the amount of time required to finish a procedure that includes the time required to accomplish the job in a cloud computing system. As seen in Condition 3.9, responsive period is measured as follows:

$RT_j = CT_j - SB_j$ (3.9)

Where SBj is the time of submission

In the following step, we compute the mean of reaction times as well as the mean of the overall average time of all virtual machines. They are given in the Conditions 3.10 and 3.11, respectively.

$$\textbf{Avg.RT}=\frac{\sum_{j=1}^{n} RT_j}{N} \quad (3.10)$$

$$\textbf{M. Total Avg. RT} = \frac{\sum_{j=1}^{m} \text{Avg.RT}_j}{m} \quad (3.11)$$

Where N denotes the amount of jobs running in the virtual machine and M.Overall Avg. RT is the average of the overall combined reaction time.

3.4.3 Resource Utilization (RU)

The amount of useful work completed by virtual machines is referred to as resource utilization. Good work is defined as a task that was completed on a virtual machine in accordance with the deadline constraints. The resource use of virtual machines (VMs) is specified by Condition 3.12.

$$\textbf{RU}=\frac{\sum_{j=1}^{n} \text{len of TmD}}{\sum_{k=1}^{m} R_C} \quad (3.12)$$

3.4.4 Guarantee Ratio (GR)

The guarantee factor is the proportion between the varieties of tasks that fulfil the time limitation as compared to the total number of tasks completed. According to Condition 3.13, the guarantee index is known as

$$\textbf{GR} = \frac{\sum_{j=1}^{n} \text{No.of TmD}}{n} \quad (3.13)$$

Where n is total number of task

3.4.5 Violation Ratio (VR)

The violation ratio is defined as the ratio of the amount of task violations to the total number of tasks multiplied by 100, based on number of tasks. The violation ratio is defined as Calculation 3.14.

$$\mathbf{VR} = \frac{\mathbf{NoV}}{\mathbf{n}} \times 100 \quad (3.14)$$

3.4.6 Improvement of makespan ratio

Makespan is a critical parameter for evaluating the success of our model, and it is kept in mind. As a result, Condition 3.15 may be used to compute the change in the makespan ratio.

$$\textbf{Makespan ratio} = (1 - \frac{\sum_{i=1}^{e} \text{Makespan DAPS}}{\sum_{i=1}^{e} \text{Makespan of other algorithm}}) * 100 \quad (3.15)$$

Where e is the quantity of experiments

3.5 OUTCOMES OF EXPERIMENTS AS WELL AS ANALYSIS

3.5.1 Implementation Environment

Using the CloudSim toolkit, we were prepared to achieve our experiments for this section of the study project. Following that, we tested the effectiveness of our suggested DAPS model. CloudSim is a toolbox for modelling and replicating data centres, hosts, virtual machines (VMs), and resource allocation algorithms, among other things. It can assess the viability of a project and fine-tune its performances without spending any expenses.

3.5.2 Experiments Configuration

Table 3.1 describes the configuration needs for the experiments in further detail.

Table 3.1: Configuration requirement for the experiments

Configurations	**Datacenter**	**Server**	**Virtual machine**
Numbers		2	6, 8, 10, 12, and 14
Core		Quad-Core and Dual-Core	1 processing element
Memory (RAM)		16 GB	0.5 GB
Storage		1 TB	10 GB image size
Bandwidth		100 GB/s	1 GB/s
VM scheduling algorithm	**1**	Time-Shared	Time-Shared
Architecture		X86 Architecture	Xen
Operating system		Linux	
Virtual machine monitor		Xen	
Speed		10,000 MIPS	300, 400 and 500 MIPS

3.5.3 Dataset

They were developed using the NASA Ames Intel Personal Super Computer (iPSC/860), which has 128 processor components communicating in a hypercube and served as a baseline for the task creation process. The iPSC/860 record consists of three months' worth of acknowledged registers for the 128-node iPSC/860 that was

installed at NASA Ames Research Center's Numerical Aerodynamic Simulation (NAS) Systems Division. Records of tasks and special records are two types of records that make up the trace of the execution of this program. Records of tasks include the following information: a pseudo user ID, a pseudo command name relating to the running file name, the size of the network that were utilized, the start time, the runtime (in seconds), and the date. In addition to the code that distinguishes the event (planned or unscheduled downtime, dedicated time, hardware or software problems, or other), the interval, the date, and the start time are included in the special records. The dataset fields that were employed in the DAPS model are depicted in Figure 3.6.

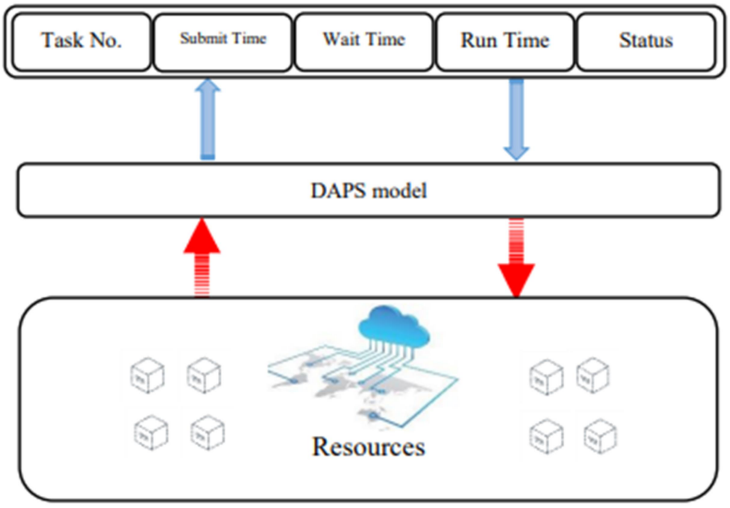


Figure 3.6: Dataset fields of DAPS model

Tasks were produced and separated into groups of 250, 500, 1000, 1500, and 2000 tasks, with each block consisting of 500 tasks.

For the purpose of evaluating the proposed DAPS model, we employed the following metrics: the average of makespan, the average of total average response time, resource utilization, guarantee ratio, and the amount of breaches for the DAPS model, as explained in Table 3.2.

Table 3.2: Experiments done for tasks with various VMs of the DAPS model

Experiments	No. of VMs	No. of tasks	Average makespan	Mean of total average RT	Resource utilization	Guarantee ratio	No. of violations	Violation ratio
1	6	250	262818	34490	5621	0.992	2	0.8%
2	8	500	380478	45741	8403	0.99	5	1%
3	10	1000	568431	73229	14355	0.989	11	1.1%
4	12	1500	696511	88045	18389	0.992	12	0.8%
5	14	2000	716719	85337	19747	0.992	16	0.8%

Based on the DAPS scheduling method, the deadlines for ten selected tasks are shown in Table 3.3, which are planned on 3 virtual machines (VMs). We observed that the suggested DAPS model distributed the sorted tasks among the VMs in accordance with two conditions: the VMs had to fulfil the deadline and the tasks had to be completed in a minimum amount of time. First, the DAPS model examines all of the virtual machines (VMs) that are accessible to decide which VM fulfils the deadline limitations. Following that, a job will be allocated to the VM that gives the shortest predicted completion time from among these VMs.

Table 3.3: Choosing real deadline utilizing DAPS related on deadline constraint and minimum completion time

Sl. No.	TaskID	Deadline constraint	VM1		VM2		VM3		Actual deadline (DAPS)		Selected VM
			EET	CT (All Tasks)	EET	CT (All Tasks)	EET	CT (All Tasks)	EET (deadline met)	CT (All Tasks)	
1.	669	**1.0**	0.3	90.3	0.18	90.18	**0.225**	90.0	**0.225**	90	**VM3**
2.	170	**1.0**	**0.9**	270.3	0.54	270.54	0.675	270.67	**0.9**	270.3	**VM1**
3.	60	**2.0**	2.1	-	1.26	773.82	**1.575**	652.5	**1.575**	652.5	**VM3**
4.	92	**2.0**	2.1	-	1.26	773.82	**1.575**	676.57	**1.575**	676.57	**VM3**
5.	218	**5.0**	5.1	-	3.06	1.52	**3.825**	1.01	**3.825**	1.01	**VM3**
6.	188	**7.0**	7.2	-	4.32	9.74	**5.4**	6.46	**5.4**	6.46	**VM3**
7.	291	**1.0**	**1.8**	549.6	1.08	683.82	1.35	562.5	**-**	*549.6	**VM1**
8.	650	**30.0**	30.9	-	**18.54**	1.02	23.175	1.35	**18.54**	1.02	**VM2**

9.	89	**31.0**	31.2	-	18.72	4.085	**23.4**	2.71	**23.4**	2.71	**VM3**
10.	209	**47.0**	47.7	-	**28.62**	6.69	35.775	8.88	**28.62**	6.69	**VM2**

* Task violated (not meeting the deadline)

According to Table 3.3, even though task 1 and task 2 had deadlines that were reached in VMs 1, 2, and 3, the tasks 1 and 2 were assigned to VMs 3, 1, and 2, which resulted in a shorter completion time than the tasks 1 and 2. Task 7 was also violated in all of the VMs, however when we looked at the projected completion time of task 7, we found that it was completed in VM1 with a shorter anticipated completion time. As a result, our suggested DAPS model initially concentrated on the task's deadline and subsequently got a reduction in the total time required to complete all tasks. A number of other jobs that met their deadlines in VM2 and VM3 were also routed to another VM that yielded a lesser duration to completion.

3.6 PERFORMANCE ASSESSMENT

In comparison to existing scheduling algorithms, the suggested DAPS model provides a better task scheduling approach based on a deadline constraint. The DAPS model employed a variety of performance measures to reduce the average makespan and the number of tasks that were violated at the same time. Throughout this chapter, the following five experiments were investigated in order to assess the overall quality and reliability of the suggested DAPS model:

i. 250 tasks with six VMs.

ii. 500 tasks with eight VMs.

iii. 1000 tasks with ten VMs.

iv. 1500 tasks with twelve VMs.

v. 2000 tasks with fourteen VMs

Using measures such as average of makespan, average of total average response time, average of total average response time, resource utilisation, guarantee ratio, amount of violations and violation ratio, we evaluate our results DAPS model to the GA, Min-Min, SJF, and Round Robin algorithms.

One of the most important criteria and the goal of the proposed DAPS model is to reduce the makespan, which refers to the total time it takes for all tasks within a virtual machine to be completed. As shown in Table 3.4, the suggested DAPS model surpassed other algorithms by determining the overall makespan while increasing the number of tasks when compared to GA, Min-Min, SJF, and Round Robin algorithms. The results of all experiments revealed that the proposed model performs better other algorithms by lowering makespan while increasing the variety of tasks

Table 3.4: Average of makespan of DAPS, GA, Min-Min, SJF and Round Robin algorithms

Experiments		**Average makespan**				
No. of tasks	**No. of VMs**	**DAPS**	**GA**	**Min-Min**	**SJF**	**Round Robin**
250	6	**262818**	305004	280130	307421	323316
500	8	**380478**	430264	422428	440850	474695
1000	10	**568431**	626303	598599	645316	630876
1500	12	**696511**	763337	733366	785707	751648
2000	14	**716719**	795613	760132	814443	826665

Figure 3.7: The X-axis represents the number of tasks, and the Y-axis represents the rise of the average makespan over time. When compared to existing algorithms, the suggested DAPS model has achieved a shorter average makespan in all of the trials conducted.

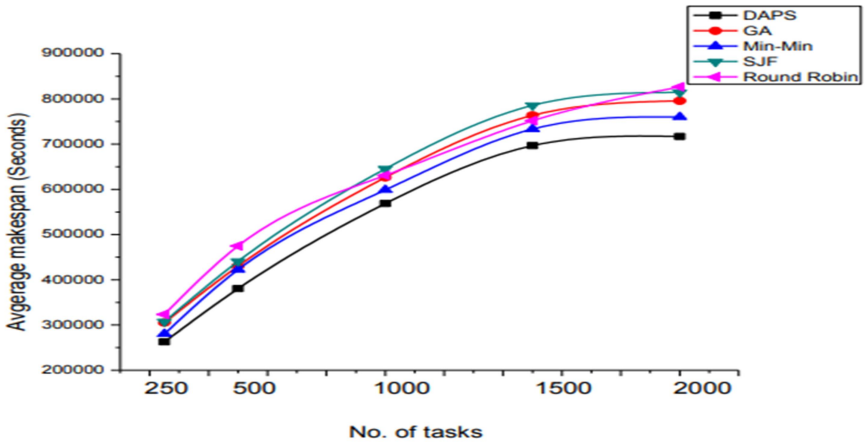


Figure 3.7: Contrasting of average makespan

Another measure that is being used as a performance statistic is the mean of total average response time, which is calculated as the sum of all responses. When compared to current algorithms, the findings of the measuring performance shown in Table 3.5 shows that the suggested DAPS model reduce the average of total average reaction time.

Table 3.5: Mean of total average response time of DAPS, GA, Min-Min, SJF and Round Robin algorithms

Experiments		**Mean of total average response time**				
No. of tasks	**No. of VMs**	**DAPS**	**GA**	**Min-Min**	**SJF**	**Round Robin**
250	6	**34490**	42618	37668	38590	42636
500	8	**45741**	56257	65225	51413	60862
1000	10	**73229**	85400	78772	83188	84606
1500	12	**88045**	100291	93720	97821	98724
2000	14	**85337**	98795	90922	96222	100107

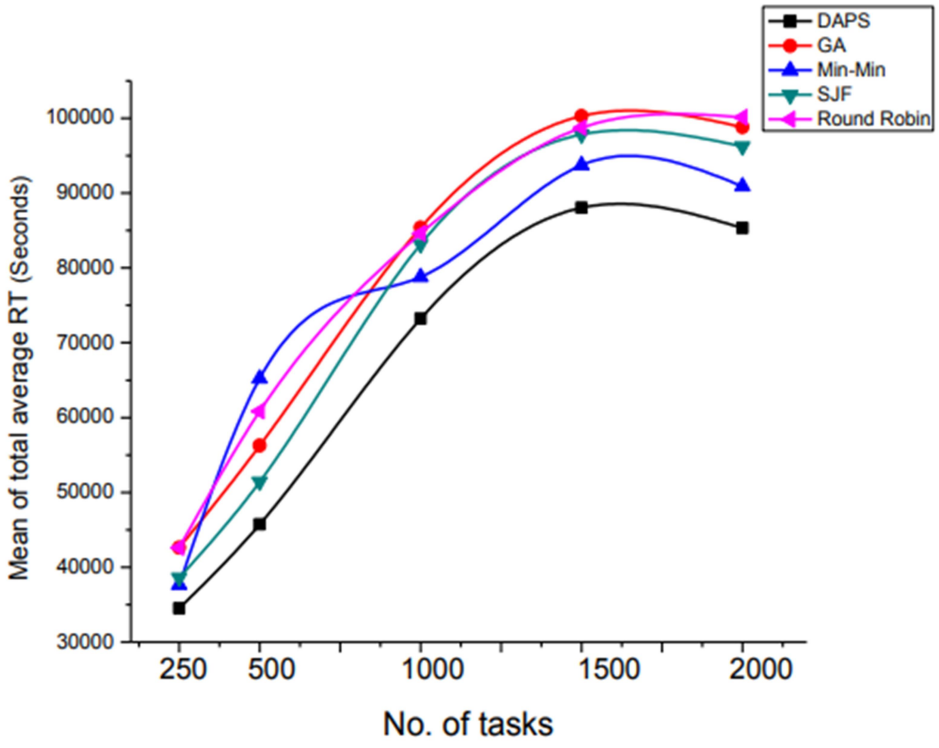


Figure 3.8: Contrast of mean of total average response time

Figure 3.8 depicts the outcome of the mean of the entire average response time calculation. Compared to GA, Min-Min, SJF, and Round Robin algorithms, the suggested DAPS model optimizes system response time based on deadline by minimizing the mean of total average response time as the number of jobs and virtual machines (VMs) increases. The task scheduling mechanism should be designed to maximize the usage of available resources in order to enhance the effectiveness of cloud computing. When compared to other approaches, the results of the proposed DAPS model shown in Table 3.6 demonstrate that our model is capable of optimizing the usage of resources under the constraints of a deadline restriction.

Table 3.6: Resource utilization of DAPS, GA, Min-Min, SJF and Round Robin algorithms

Experiments		Resource utilization				
No. of tasks	**No. of VMs**	**DAPS**	**GA**	**Min-Min**	**SJF**	**Round Robin**
250	6	**5621**	4307	4381	3541	3246
500	8	**8403**	5739	6540	5249	4411
1000	10	**14355**	9686	10336	9064	9276
1500	12	**18389**	13477	14364	13171	12699
2000	14	**19747**	13928	15836	13565	12825

Figure 3.9 depicts resource consumption when different tasks (250-2000) are employed in conjunction with a deadline restriction, demonstrating that our suggested model outperforms existing techniques.

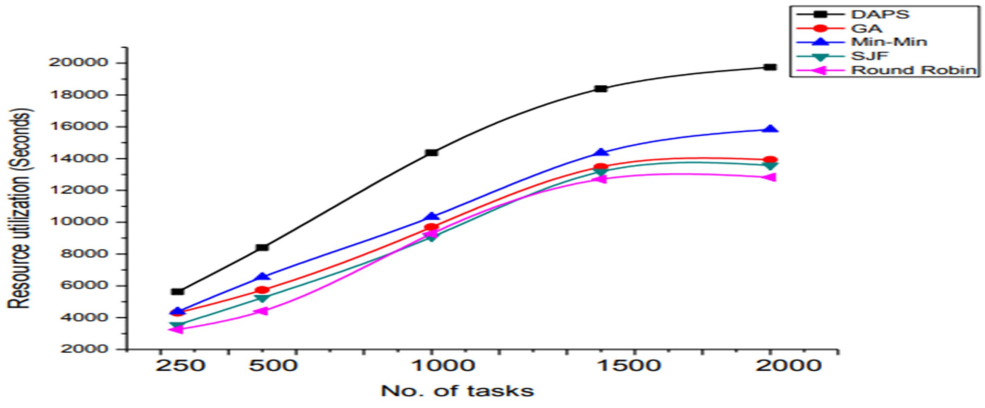


Figure 3.9: Comparison of resource utilization

Task scheduling is the process of allocating tasks in order to satisfy deadline limits and, as a result, obtaining the optimal ratio of deadline guarantee to workload. In all evaluations of guarantee ratio, the DAPS model performed far better than in the other algorithms, as indicated in Table 3.7, which can be seen inset of Figure 3.10.

Table 3.7: Guarantee ratio of DAPS, GA, Min-Min, SJF and Round Robin algorithms

Experiments		**Guarantee ratio**				
No. of tasks	**No. of VMs**	**DAPS**	**GA**	**Min-Min**	**SJF**	**Round Robin**
250	6	**0.992**	0.695	0.82	0.684	0.684
500	8	**0.99**	0.665	0.868	0.662	0.668
1000	10	**0.989**	0.641	0.724	0.632	0.634
1500	12	**0.992**	0.693	0.789	0.694	0.691
2000	14	**0.992**	0.673	0.768	0.667	0.666

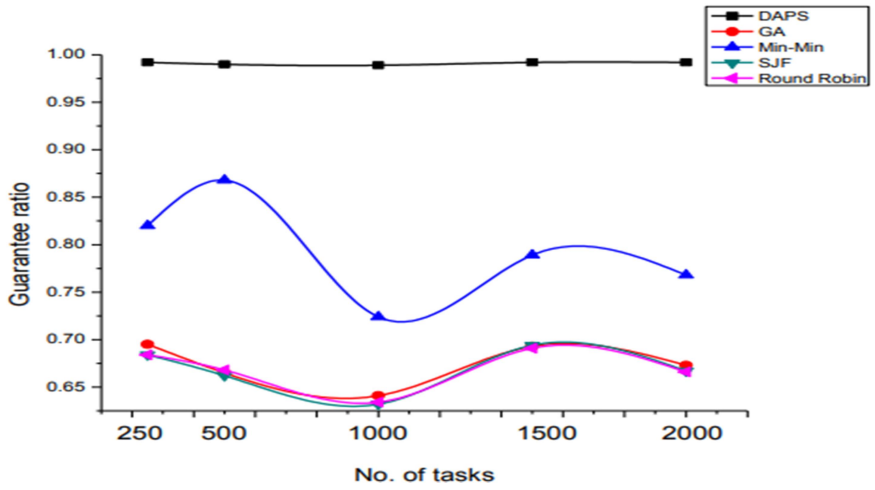


Figure 3.10: Comparison of guarantee ratio

Users' pleasure is the primary goal of the DAPS suggested model, which prioritizes satisfying deadline constraints and assigning tasks to achieve it. As an example, Table 3.8 involves the comparison of task violations in our suggested DAPS model to the amount of task violations in existing algorithms. It was discovered that our suggested DAPS model caused less infractions than previous algorithms, which was a pleasant surprise.

Table 3.8: Number of violations of DAPS, GA, Min-Min, SJF and Round Robin algorithms

Experiments		**No. of violations**				
No. of tasks	**No. of VMs**	**DAPS**	**GA**	**Min-Min**	**SJF**	**Round Robin**
250	6	**2**	76	45	79	79
500	8	**5**	167	66	169	166
1000	10	**11**	358	276	368	366
1500	12	**12**	459	317	459	463
2000	14	**16**	654	464	667	669

The amount of task violations for the proposed DAPS model is illustrated in Figure 3.11, which is significantly smaller than the amount of task violations for those other algorithms.

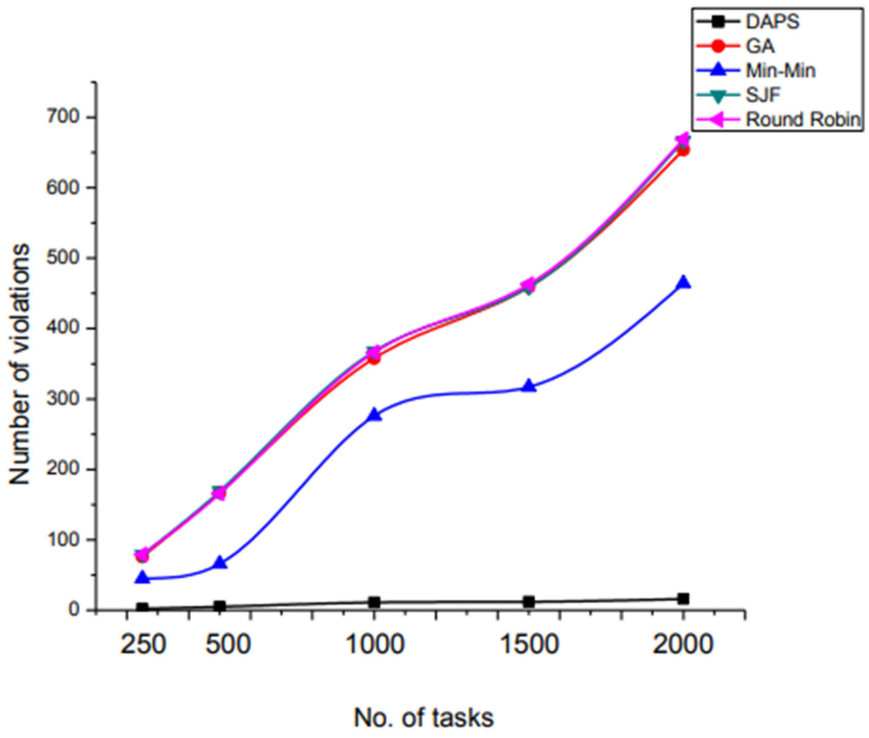


Figure 3.11: Quantity of violations

This table shows the difference in real deadline between our suggested DAPS model and the GA, Min-Min, SJF, and Round Robin algorithms, in which we picked ten random jobs out of 250 tasks using 6 virtual machines and executed them on six virtual machines.

Table 3.9: Actual deadline for random ten tasks in DAPS, GA, Min-Min, SJF and Round Robin algorithms

Task No.	T.ID	Deadline Constraint	Actual Deadline (DAPS)	Actual Deadline GA	Actual Deadline Min-Min	Actual Deadline SJF	Actual Deadline Round Robin
1.	60	**2.0**	1.575	1.7115	1.26	1.575	1.26
2.	291	**1.0**	**1.8**	**1.26**	**1.08**	**1.8**	**1.8**

3.	221	**1.0**	0.9	**1.3275**	0.9	**1.125**	0.9
4.	417	**1.0**	0.72	80661	0.72	**1.2**	0.9
5.	170	**1.0**	0.675	0.7695	0.54	0.675	0.54
6.	191	**1.0**	0.72	0.984	0.72	0.9	**1.2**
7.	320	**3.0**	2.475	2.739	2.475	1.98	**3.3**
8.	304	**1.0**	**1.8**	**1.251**	**1.08**	**1.08**	**1.08**
9.	266	**3.0**	2.34	**3.003**	2.34	2.925	2.925
10.	93	**54.0**	40.725	3703093	40.725	**54.3**	**54.3**
Number of tasks violated			**2**	**4**	**2**	**5**	**4**

Note: The cells in blue color indicate tasks violated

On the basis of the data in Table 3.9, we can conclude that, when 10 selected tasks are picked to be performed on six virtual machines, the violation number in SJF is the highest, trailed by Round Robin and GA, while Min-Min and DAPS have just two task violations. However, when 250 tasks are taken into consideration, our suggested DAPS model has the lowest violation rate when contrasted with other methods, as previously noted in Table 3.8. The violation ratio is another indicator that is associated with task noncompliance. As shown in Table 3.10, the violation ratios for the DAPS model and other available methods are comparable.

Table 3.10: Violation ratio of DAPS, GA, Min-Min, SJF and Round Robin algorithms

Experiments		Violation ratio				
No. of tasks	**No. of VMs**	**DAPS**	**GA**	**Min-Min**	**SJF**	**Round Robin**
250	6	**0.8%**	30.4%	18.%	31.6%	31.6%
500	8	**1%**	33.4%	13.2%	33.8%	33.2%
1000	10	**1.1%**	35.8%	27.6%	36.8%	36.6%
1500	12	**0.8%**	30.4%	21.1%	30.6%	30.8%
2000	14	**0.8%**	32.7%	23.2%	33.35 %	33.45%

When contrasted to the GA, Min-Min, SJF, and Round Robin algorithms, the DAPS model, which is founded on a deadline constraint, has a lower violation ratio of jobs because of the deadline restriction. As a result, as indicated in Figure 3.12, our suggested DAPS model has a significantly higher operating efficiency than existing models.

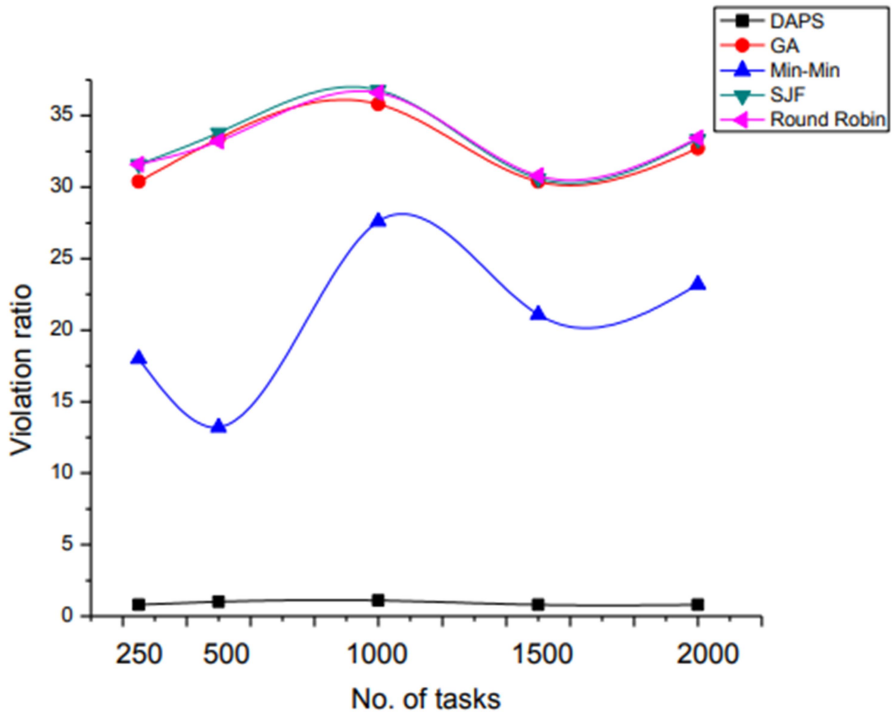


Figure 3.12: Violation ratio

This table shows that the change in makespan ratio for our suggested DAPS model with the GA, Min-Min, SJF, and Round Robin strategies is 10% better than the increase in makespan ratio for the DAPS model using the Min-Min, SJF, and Round Robin algorithms.

Table 3.11: Improvement in makespan ratio for DAPS compared to GA, Min-Min, SJF and Round Robin algorithms

Improvement in makespan ratio					
Experiments	**DAPS**	**GA**	**Min-Min**	**SJF**	**Round Robin**
	262818	305004	280130	307421	323316
	380478	430264	422428	440850	474695
	568431	626303	598599	645316	630876

	696511	763337	733366	785707	751648
	716719	795613	760132	814443	826665
Sum of Expts.	**2624957**	**2920521**	**2794655**	**2993737**	**3007200**
Improvement DABS vs other algorithm		**10%**	**6%**	**12%**	**12.7%**
Sum of all makespan ratio		40.7			
Average		**10.25 %**			

In order to conduct a statistical study of our suggested DAPS model, we employed the T-test. The T-test is an effective factor which determines the mean of the DAPS model, demonstrating the use of normality in the process. DAPS model results in a T-test when compared to other algorithms such as GA, Min-Min, SJF, and Round Robin are presented in Table 3.12, which is a table of findings.

Table 3.12: T-test of DAPS model compared to GA, Min-Min, SJF, and Round Robin algorithms

Metrics	**DAPS model vs other algorithms**			
	GA	**Min-Min**	**SJF**	**Round Robin**
Average makespan	0.3	0.3	0.2	0.2
Mean of total average response	0.2	0.3	0.3	0.2

time				
Resource utilization	0.1	0.2	0.1	0.09
Guarantee ratio	5.0	1.9	7.6	4.2
Number of violations	0.005	0.01	0.005	0.006
Violation ratio	6.1	1.9	7.5	4.6

We suggested a model based on a deadline constraint that is capable of allocating work while also responding to the satisfaction of both users and service providers in the following manner:

i. Tasks are prioritized in ascending order depending on length of the tasks and the importance of the tasks.

ii. A good state of the VM is defined as one in which it has met the deadline restriction.

iii. Choosing an appropriate virtual machine (VM) that would complete the work in the shortest amount of time

In summary, we can say that this chapter provides an answer to the first question: how to develop a scheduling model that will efficiently utilize resources during load process and improve total completion time while still meeting the deadline constraint of the project. We introduced the Deadline-Aware Priority Scheduling model to reduce the amount of time required to complete tasks depending on deadline restrictions. The improvement of both makespan and resource usage underneath a deadline restriction may be ensured by using our model in comparison to GA, MinMin, SJF, and Round Robin algorithms when a deadline constraint is imposed. The performance evaluation of our proposed DAPS model also indicates that our model achieves the best performance by minimizing metrics such as average

makespan, mean of total average response time, number of violations, and violation ratio while maximizing resource utilization and guarantee ratio, amongst other factors. The improvement in makespan ratio for our suggested DAPS model over the GA, Min-Min, SJF, and Round Robin algorithms was 10%, 6%, 12.7 percent, and 12.7 percent, respectively, compared to the GA, Min-Min, SJF, and Round Robin algorithms. Also computed was the T-test, which was used to compare our suggested DAPS model against other algorithms, such as GA, Min-Min, SJF, and Round Robin, in order to conduct statistical analysis.

3.7 EFFICIENT MANAGEMENT OF CLOUD RESOURCES THROUGH BUDGET-AWARE TASK SCHEDULING TECHNIQUE

The cost of performing activities in a cloud computing environment is one of the most important areas of interest for the user in the cloud computing environment. As a result, there must be a budget limit in order to identify the valid budget to be given to the user once they have negotiated and agreed with the service providers. Based on the foregoing, we will examine what the budget constraint is and the suggested model for scheduling jobs based on this limitation in this chapter, as well as evaluating and discussing the outcomes achieved after using the proposed model. Cloud computing has evolved into a large-scale paradigm of cloud applications that can be tailored to meet the needs of individual users. Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service are the three most prevalent cloud services. Unlike on-premises computing, cloud computing allows customers to rent infrastructure and select from a variety of resources and services to meet their specific needs.

Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service are the three most prevalent cloud services. Unlike on-premises computing, cloud computing allows customers to rent infrastructure and select from a variety of resources and services to meet their specific needs. Before utilizing and paying for services or resources, the user should determine which services or resources he requires. Because the resources of the cloud are being charged for their usage, the cost of execution must be taken into consideration. Thus, there is a negative relationship

between performance and cost: in order to achieve greater performance, the cost must be increased. In order to get the highest performance while also saving costs, it is difficult to strike a balance between the two objectives.

Cloud computing makes use of virtualization technology, which provides a range of virtual machines (VMs). Each of them has a unique set of computing skills, and they are the ones who are responsible for drastically increasing resource usage while simultaneously lowering the cost of maintenance. The number of cloud service providers has expanded dramatically in recent years, and each provider provides a unique set of services and charges differently for different resources (CPU, bandwidth, memory). In light of the services and prices now accessible, customers may find it challenging to identify the most cost-effective option when attempting to match their instance requirements with the cloud services currently available. Users can pick the most appropriate instance for performing their jobs with high performance of task scheduling from a large number of cloud service providers, such as Amazon (Elastic Cloud Computing (EC2)), which give pricing lists and resource and instance offerings. In addition, Google Cloud Platform (GCP) is a well-known cloud service provider that provides a cost list for its resources and services, such as Google Compute Engine and Google Cloud Storage (GCE).

When working in a cloud computing environment, the duties of the users are executed on either real or virtual computers, depending on the job needs. Because of the virtualization idea, each virtual machine may perform a variety of functions while sharing its same settings as the corresponding real computer. However, assigning the work to appropriate resources depending on quality of service (QoS) is depicted as a difficult problem. The mapped tasks will also be executed on the virtual machine that fulfills the QoS restrictions, with the price of task implementation being kept to a bare minimum under the limits of the budget. As a result, the cost is an important barrier in the work scheduling process in cloud computing. Most significantly, the service level agreement (SLA) between the customer and cloud provider, which includes QoS measurements, must be taken into account by the provider (e.g. response time, bandwidth, storage, resources utilization, and cost). These metrics must be quantifiable and must be evaluated while delivering service in order to detect and resolve violations of the SLA that have been decided upon.

3.8 BUDGET CONSTRAINT

The budget is a limit on the amount of money that may be spent on the tasks that must be completed. For want of a better expression, the budget is the financial limitation or uses accrual accounting for the services that are given by a cloud service provider to a customer. When processing and performing the activities, the service provider must consider how to reduce the amount of time spent on each activity while still staying within the budget constraints. The total cost of the project should not be larger than the budget limitation that was specified by the end customer. The overall cost of resources, like the expense of a CPU, memory, and other components of the system, should be determined. Figure 3.13 illustrates how a service level agreement (SLA) is a contract that comprises quality of service measures such as the budget restriction, that's used as a barrier to cope with the cost of funds that are made available by the service provider to the consumers.

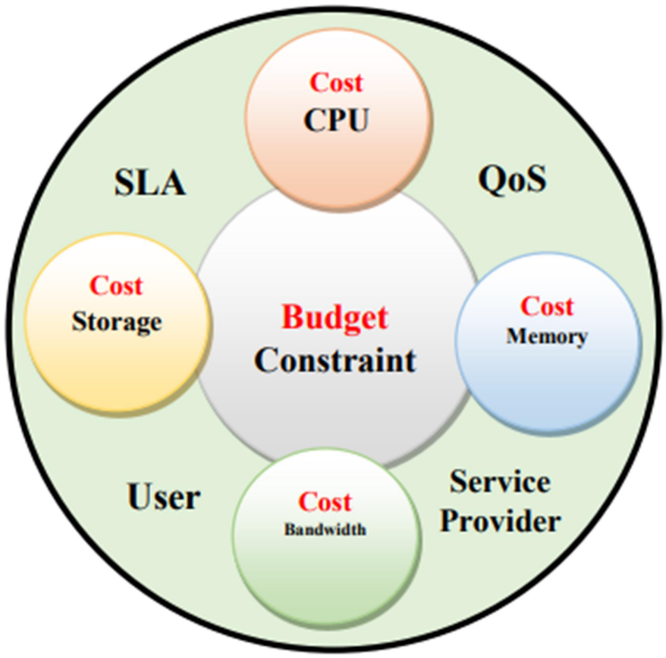


Figure 3.13: Budget constraint

3.9 TASK SCHEDULING DEPENDENT ON BUDGET-AWARE MODEL

The Budget-Aware Scheduling (BAS) paradigm that has been presented is dependent on a budget limitation. In this scenario, a data center is made up of a collection of diverse hosts that are then separated into virtual machines. The CPU, memory, bandwidth, and storage capacity of the host and virtual machines (VMs) may be different.

- **Definition 1 (assigned tasks):** (Leni, Fiti, and TB). These properties are the length, file size, and task budget limit for each job that is submitted by users, and they are described in detail below.

- **Definition 2 (Resources):** Resources are defined as follows: In order to define a VM resource, it must have the following key attributes: RVM = (RC, RM, RB, RS), where these symbols denote the CPU, memory, bandwidth, and storage of the virtual machine. The cost of each resource is represented by the equation CR= (CRC, CRM, CRB, CRS), where CRC represents the cost of the CPU, CRM represents the expense of memory, CRB represents the cost of bandwidth, and CRS represents the cost of storage.

We are attempting to complete the jobs in the shortest amount of time possible based on the budget constraints for each activity in order to achieve the lowest possible cost execution while maximizing resource usage. As seen in Figure 3.14, the BAS model displays the resource needs of the tasks and describes the link between resource costs and user budget constraints.

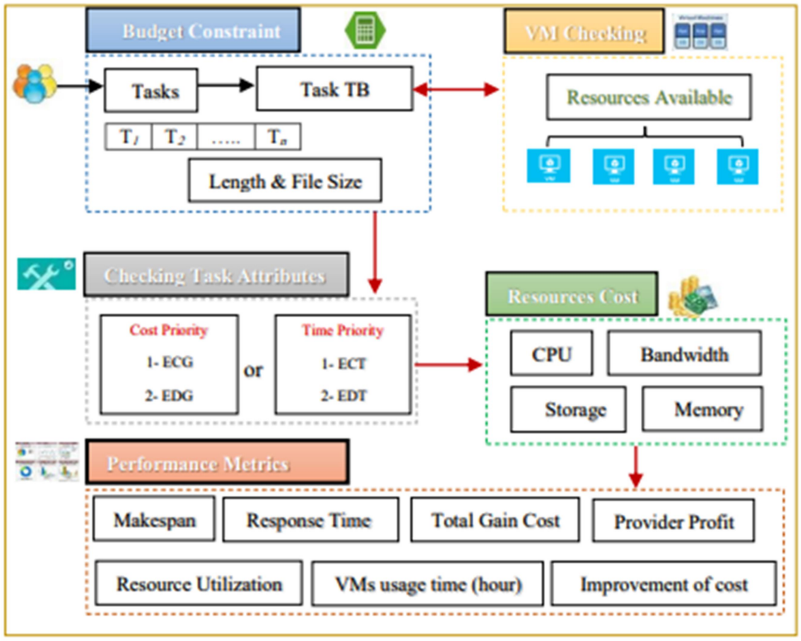


Figure 3.14: Suggested Budget-Aware Scheduling model

Figure 3.15 depicts the scheduling process in the BAS model, which accepts submission requests from users and then conducts the work, which is divided into four steps:

- **Step 1:** Calculate the average task budget constraint (Avg. TB) and contrast it to the total task budget constraint (TB).
- The second step is to compute the expected gain cost (EGC).
- Third, look for the virtual machine that fulfills the Task Budget constraint (TB). This indicates that if the EGC in this VM is equal to or less than TB, it signifies that the VM complies with the user limitation.
- Fourth, map the job to the appropriate virtual machine and compute the cost of resources consumed.

The users have a large number of tasks that need to be scheduled at cloud resources, and each task has an EGC that varies from job to task. The EGC finds the most appropriate virtual machine (VM) that can be utilized to carry out the given job. Alternatively, both the duration of the work (Len) and the size of the file (Fit) are

critical criteria that must not be overlooked. The duration of the work and the size of the file would decide whether the activity required time priority or cost priority. When it comes to task performance metrics, the resource cost may be impacted by some of the metrics that are important for our suggested BAS model (makespan, reaction time, gain cost, resource usage, and profit), which are detailed below. A scheduler is a component that is in charge of allocating jobs to available resources in the system. As previously stated, the optimal scheduling approach suggested in this chapter maps tasks to resources based on this information. The tasks are then scheduled based on this information. In the beginning, the information about the tasks and resources is gathered. Second, it determines if the Resource (Rj) meets the task (Ti) need and makes a decision based on that determination. Among the requirements are that the VMs fulfil the TB and have sufficient resources. Determine if the work is time- or cost-sensitive, and then map the task to the VM that best fulfills the task's needs according to the priority determined in the previous step. At the end of the process, the scheduler assigns the resource to the job.

3.9.1 Application Model

Cloud computing is characterized by a wide range of activities and resources. Because cloud resources come in a variety of configurations, the cost of each resource differs from the price of the other. In addition, the costs of various jobs vary. As a result, we take into consideration the complexity of the cloud environment and propose the BAS model, which more accurately reflects the costs of certain jobs. According to the concept of resources, it separates the cost of resources into four categories: CPU, bandwidth, storage, and memory (or a combination of these). The budget limitation is located at the very top of the pyramid in our suggested BAS paradigm. Budget-Aware Scheduling establishes the precedence of the job in all virtual machines that fulfil the TB requirements. The BAS model examines the properties of each job (e.g., length, file size) in order to assign each work to the most appropriate resource. For the sake of scheduling work inside the BAS model, we have implemented the processes outlined in Figure 3.15 depending on the budget restriction.

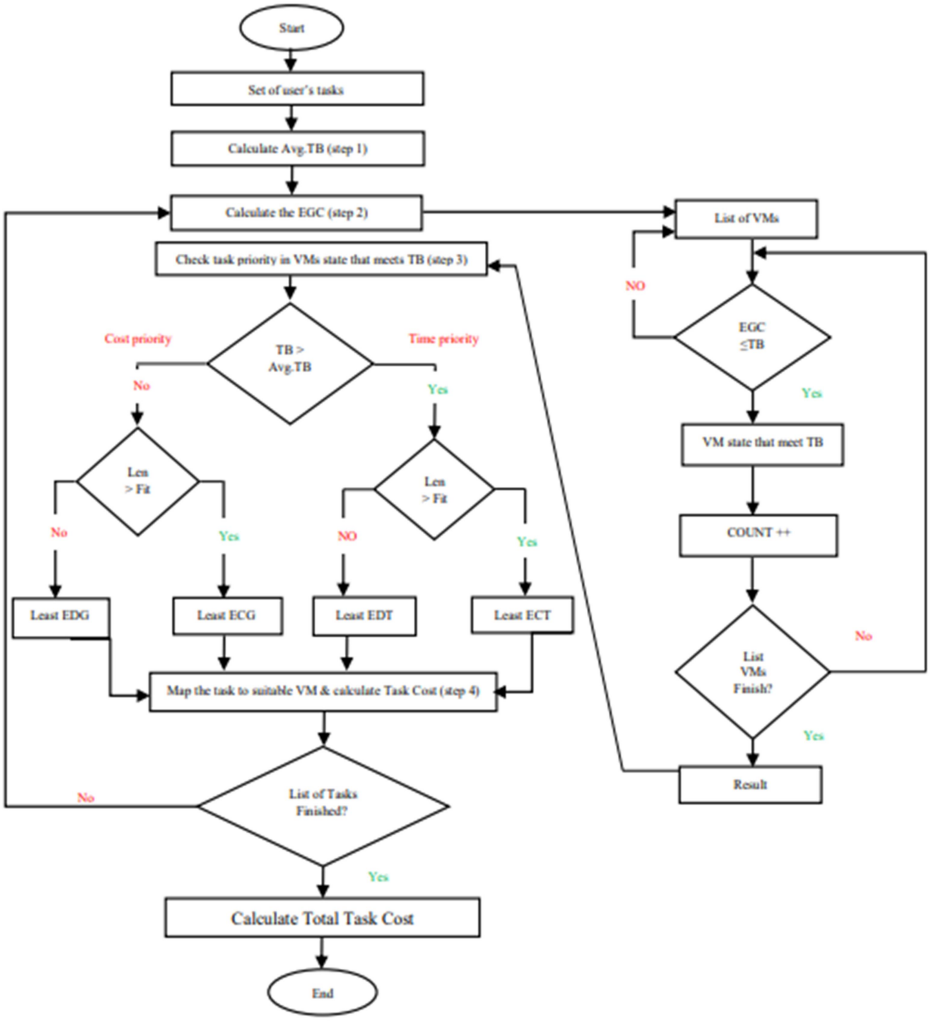


Figure 3.15: Flowchart of BAS model

In order to begin, the average of all users' budget (Avg. TB) is determined in accordance with the following condition 3.16:

$$\mathbf{Avg.TB} = \frac{\sum_{i=1}^{n} TB_i}{n} \quad (3.16)$$

Where n is the complete quantity of tasks

The list of virtual machines (VMs) is examined to determine which VM provides the highest level of customer satisfaction by getting the EGC as described in Condition 3.17:

$$\textbf{Expected Gain Cost (EGC)}=\left(\frac{Len_i}{R_{Cj}}*C_{RC_j}+\frac{Fit_i+Fot_i}{R_{Bj}}*C_{RB_j}+\frac{Fit_i+Fot_i}{R_{Mj}}*C_{RM_j}+\right.$$

$$\left.\frac{Fit_i+Fot_i}{R_{Sj}}*C_{RS_j}\right) \quad (3.17)$$

Where Fot is the file output size of task

The EGC is then generated within every VM and matched to the TB using Equation 3.18:

$$\textbf{VMs state}=\begin{cases}1 & GC\le TB\\ 0 & Otherwise\end{cases} \quad (3.18)$$

Finally, the task's importance is assessed by comparing the user's total time with the average total time. According to Equation 3.19, if the TB value is more than the Avg.TB constraint value, the job is said to have time priority, and if the TB value is less than the Avg.TB restriction valuation, the work was shown to have cost priority.

$$\textbf{Task priority}=\begin{cases}Time & if\ \ TB>Avg.TB\\ Cost & Otherwise\end{cases} \quad (3.19)$$

When a job has a high priority in terms of time, the time is computed; otherwise, the project cost is estimated. Following this is a determination of task needs based on the task characteristics, where our suggested BAS model would contrast the length of the job and the file size, as specified in Condition 3.20.

$$\textbf{Time}=\begin{cases}Completion\ Time & Len>Fit\\ Data\,transfer\,Time & Len\le Fit\end{cases} \quad (3.20)$$

As a result, if the task's length is more than the file size, the completion time is computed; otherwise, the data transfer is calculated as follows:

In the third definition (the expected completion time (ECT) matrix)

This matrix represents the anticipated completion time for every job across all virtual machines. The following is the matrix of the predicted completion time displayed by ECT:

$$ECT = \begin{array}{c} \\ T_1 \\ T_2 \\ \vdots \\ T_n \end{array} \begin{array}{c} \begin{array}{cccc} VM_1 & VM_2 & \dots & VM_m \end{array} \\ \begin{Bmatrix} ECT_{11} & ECT_{12} & \dots & ECT_{1m} \\ ECT_{21} & ECT_{22} & \dots & ECT_{2m} \\ \vdots & \vdots & \dots & \vdots \\ ECT_{n1} & ECT_{n2} & \dots & ECT_{nm} \end{Bmatrix} \end{array}$$

The work will be assigned to the virtual machine with the shortest completion time in this matrix. Equation 3.21 is used to estimate the projected completion time for this project.

$$\mathbf{ECT} = \sum_{i=1}^{p} EX_i + EX\ of\ current\ task \quad (3.21)$$

Where p is the number of previous tasks in a specific VMj,

EX is the time required to complete the work, as determined by Equation 3.22.

$$\mathbf{EX} = \frac{Len_i}{R_{Cj}} \quad (3.22)$$

- **Definition 4 (Expected time of Data Transfer (EDT) matrix)**

The following is the matrix that describes the estimated data transfer latency of a process running on a virtual machine:

$$EDT = \begin{array}{c} \\ T_1 \\ T_2 \\ . \\ . \\ . \\ T_n \end{array} \begin{array}{c} \begin{array}{cccc} VM_1 & VM_2 & \dots & VM_m \end{array} \\ \begin{pmatrix} EDT_{11} & EDT_{12} & \dots & EDT_{1m} \\ EDT_{21} & EDT_{22} & \dots & EDT_{2m} \\ . & . & \dots & . \\ . & . & & . \\ . & . & & . \\ EDT_{n1} & EDT_{n2} & \dots & EDT_{nm} \end{pmatrix} \end{array}$$

The work will be allocated to the virtual machine that requires the least amount of data transmission time. It is necessary to determine Data Transfer Time (DT) in accordance with the Equation 3.23.

$$\mathbf{DT} = \frac{Fit_i + Fot_i}{R_{Bj}} \quad (3.23)$$

Using Equation 3.24, it is therefore possible to compute the estimated data transmission time.

$$\mathbf{EDT} = \sum_{i=1}^{p} DT_i + DT\,of\,current\,task \quad (3.24)$$

When the job has a low priority, our suggested BAS model gives it the next highest priority. For tasks with low priority, checking the cost of execution should be done in accordance with the assignment brief, in which case the task duration is matched to the file size as specified in Equation 3.25:

$$\mathbf{Cost} = \begin{cases} CPU\ Gain & Len > Fit \\ Data\,transfer\ Gain & Len \leq Fit \end{cases} \quad (3.25)$$

Described by Equation 3.25, we can derive two more definitions, which are as:

- **Definition 5 (Expected CPU Gain (ECG) matrix)**

The estimated cost for every job in each VM is represented by the ECG matrix, which looks like this:

$$ECG = \begin{array}{c} \\ T_1 \\ T_2 \\ . \\ . \\ . \\ T_n \end{array} \begin{array}{c} \begin{array}{cccc} VM_1 & VM_2 & \dots & VM_m \end{array} \\ \begin{Bmatrix} ECG_{11} & ECG_{12} & \dots & ECG_{1m} \\ ECG_{21} & ECG_{22} & \dots & ECG_{2m} \\ . & . & \dots & . \\ . & . & & . \\ . & . & & . \\ ECG_{n1} & ECG_{n2} & \dots & ECG_{nm} \end{Bmatrix} \end{array}$$

When a task has a low priority and its length exceeds the file size, a quicker machine is chosen to complete it, and the machine's ECG is determined as shown in Equation 3.26.

$$\mathbf{ECG} = \frac{Len_i}{R_{Cj}} * C_{RC_j} \quad (3.26)$$

When the length is small or equal, however, the data transfer cost is kept high.

- **Definition 6 (Matrix of Expected Data Transfer Gain (EDG)):**

The Estimated Data Transfer Gain (EDG) matrix displays the expected cost of data transfer for each task in each VM. It can be expressed numerically as follows:

$$EDG = \begin{array}{c} \\ T_1 \\ T_2 \\ . \\ . \\ . \\ T_n \end{array} \begin{array}{c} \begin{array}{cccc} VM_1 & VM_2 & \dots & VM_m \end{array} \\ \begin{Bmatrix} EDG_{11} & EDG_{12} & \dots & EDG_{1m} \\ EDG_{21} & EDG_{22} & \dots & EDG_{2m} \\ . & . & \dots & . \\ . & . & & . \\ . & . & & . \\ EDG_{n1} & EDG_{n2} & \dots & EDG_{nm} \end{Bmatrix} \end{array}$$

As shown in Equation 3.27, the task would be routed to a quicker VM.

$$\mathbf{EDG} = \frac{Fit_i + Fot_i}{R_{Bj}} * C_{RB_j} \quad (3.27)$$

3.9.2 Case Study

Tables 3.13 and 3.14 define the tasks attributes and VMs configurations, respectively, to demonstrate how our suggested BAS model works.

Table 3.13: Tasks attributes

Task _ID	**Length**	**Input file size (Fit)**	**Output file size (Fot)**	TB
1	1000	2200	300	10
2	2000	2750	100	4
3	5000	1500	200	3
4	4000	2000	500	7

Table 3.14: VMs configurations

VM_Id	**VM_MIPS**	**VM-BW**	**VM-RM**	**VM-ST**	**Cost-MIPS**	**Cost-BW**	**Cost-RM**	**Cost-ST**
1	500	60	113	250	0.01	0.001	0.2	0.02
2	2000	80	210	250	0.04	0.001	0.3	0.02
3	1000	40	170	350	0.02	0.001	0.1	0.04

Equation 3.16 is used to get the budget average (Av.TB). To decide task priority, if the task's budget exceeds Avg.TB, the task is given a time priority; otherwise, it is given a cost priority. In this case, Avg.TB = (10 + 4 + 3 + 7= 24/4 = 6), then EGC is determined for each task using Equation 3.17, and each VM state is labeled using Equation 3.18. The VM state is equal to 1 if the EGC is less than or equal to the TB,

otherwise it is equal to 0. As a result, we assume EGC for each task in each VM, as shown in Table 3.15.

Table 3.15: Expected gain cost of each task into each VM

Task _ID	Expected gain cost			VM_ State		
	VM1	VM2	VM3	VM1	VM2	VM3
1	4.681	3.821	1.837	1	1	1
2	5.355	4.373	2.106	0	0	1
3	3.264	2.677	1.336	0	1	1
4	4.741	3.881	1.897	1	1	1

The procedure for obtaining the values in Table 3.15 is as follows:

- **EGC of task 1 in VM1:**

Gain CPU Cost of task 1= length of task / MIPS of VM1 * cost of CPU= 1000 / 500 * 0.01= 0.02

Gain Bandwidth Cost of task 1= Task Fit + Task Fot / Bandwidth * cost of bandwidth= 2200 + 300 / 60 * 0.001= 0.041

Gain Memory Cost of task 1= Task Fit + Task Fot / Memory * cost of Memory= 2200 + 300 / 113 * 0.2= 4.42

Gain Storage Cost of task 1= Task Fit + Task Fot / Storage * cost of Storage= 2200 + 300 / 250 * 0.02= 0.2

EGC of task 1 in VM1 = 0.02 + 0.041 + 4.42 + 0.2= 4.681

VM1 is tagged =1 because the EGC of task 1 in VM1 is 4.681 10 and the TB of task 1 → 4.681 10.

- **EGC of task 1 in VM2:**

Gain CPU Cost of task 1= length of task / MIPS of VM2 * cost of CPU= 1000 / 2000 * 0. 04= 0.02

Gain Bandwidth Cost of task 1= Task Fit + task Fot / Bandwidth * cost of Bandwidth= 2200 + 300 / 80 * 0.001= 0.031

Gain Memory Cost of task 1= Task Fit + task Fot / Memory * cost of Memory= 2200 + 300 / 210 * 0.3= 3.57

Gain Storage Cost of task 1= Task Fit + task Fot / Storage * cost of Storage= 2200 + 300 / 250 * 0.02= 0.2

EGC of task 1 in VM2 = 0.02 + 0.031 + 3.57 + 0.2= 3.821

VM2 is tagged =2 because the EGC of task 1 in VM2 is 4.681 10 and the TB of task 2 → 4.681 10.

- **EGC of task 1 in VM3:**

Gain CPU Cost of task 1= length of task / MIPS of VM3 * cost of CPU= 1000 / 1000 * 0. 02= 0.02

Gain Bandwidth Cost of task 1= Task Fit + task Fot / Bandwidth * cost of Bandwidth= 2200 + 300 / 40 * 0.001= 0.062

Gain Memory Cost of task 1= Task Fit + task Fot / Memory * cost of Memory= 2200 + 300 / 170 * 0.1= 1.47

Gain Storage Cost of task 1= Task Fit + task Fot / Storage * cost of Storage= 2200 + 300 / 350 * 0.04= 0.285

EGC of task 1 in VM3 = 0.02 + 0.062 + 1.47 + 0.285= 1.837

VM3 is tagged =1 because the EGC of task 1 in VM3 is 4.681 10 and the TB of task 1 → 1.1837.

- **EGC of task 2 in VM1:**

Gain CPU Cost of task 2= length of task / MIPS of VM1 * cost of CPU= 2000 / 500 * 0.01= 0.04

Gain Bandwidth Cost of task 2= Task Fit + task Fot / Bandwidth * cost of Bandwidth= 2750 + 100 / 60 * 0.001= 0.047

Gain Memory Cost of task 2= Task Fit + task Fot / Memory * cost of Memory= 2750 + 100 / 113 * 0.2= 5.04

Gain Storage Cost of task 2= Task Fit + task Fot / Storage * cost of Storage= 2750 + 100 / 250 * 0.02= 0.228

EGC of task 2 in VM1 = 0.04 + 0.047 + 5.04 + 0.228= 5.355

VM1 is designated =0 because the EGC of task 2 in VM1 > TB of task 2→ 5.355 > 4.

- **EGC of task 2 in VM2:**

Gain CPU Cost of task 2= length of task / MIPS of VM2 * cost of CPU= 2000 / 2000 * 0.04=0.04

Gain Bandwidth Cost of task 2= Task Fit + task Fot / Bandwidth * cost of Bandwidth= 2750 + 100 / 80 * 0.001= 0.035

Gain Memory Cost of task 2= Task Fit + task Fot / Memory * cost of Memory= 2750 + 100 / 210 * 0.3= 4.07

Gain Storage Cost of task 2= Task Fit + task Fot / Storage * cost of Storage= 2750 + 100 / 250 * 0.02= 0.228

EGC of task 2 in VM2 = 0.04 + 0.035 + 4.07 + 0.228= 4.373

VM2 is designated =0 because the EGC of task 2 in VM2 > TB of task 2→ 4.373 > 4.

- **EGC of task 2 in VM3:**

Gain CPU Cost of task 2= length of task / MIPS of VM3 * cost of CPU= 2000/ 1000* 0.02=0.04

Gain Bandwidth Cost of task 2= Task Fit + task Fot / Bandwidth * cost of Bandwidth= 2750+100 / 40* 0.001=0.071

Gain Memory Cost of task 2= Task Fit + task Fot / Memory * cost of Memory= 2750+100 / 170* 0.1=1.67

Gain Storage Cost of task 2= Task Fit + task Fot / Storage * cost of Storage= 2750+100 / 350* 0.04=0.325

EGC of task 2 in VM3 = 0.04 + 0.071 + 1.67 + 0.325= 2.106

VM3 is tagged =1 because the EGC of task 2 in VM3 is TB of task 2→ 2.106 4.

- **EGC of task 3 in VM1:**

Gain CPU Cost of task 3= length of task / MIPS of VM1 * cost of CPU= 5000/ 500 * 0.01=0.1

Gain Bandwidth Cost of task 3= Task Fit + task Fot / Bandwidth * cost of Bandwidth= 1500+200/ 60 * 0.001=0.028

Gain Memory Cost of task 3= Task Fit + task Fot / Memory * cost of Memory= 1500+200 / 113 * 0.2=3.00

Gain Storage Cost of task 3= Task Fit + task Fot / Storage * cost of Storage= 1500+200 / 250 * 0.02=0.136

EGC of task 3 in VM1 = 0.1 + 0.028 + 3.00 + 0.136= 3.264

VM1 is designated =0 because the EGC of task 3 in VM1 > TB of task 3 →3.264 > 3.

- **EGC of task 3 in VM2:**

Gain CPU Cost of task 3= length of task / MIPS of VM2 * cost of CPU= 5000/ 2000* 0.04=0.1

Gain Bandwidth Cost of task 3= Task Fit + task Fot / Bandwidth * cost of Bandwidth= 1500+200 / 80 * 0.001=0.021

Gain Memory Cost of task 3= Task Fit + task Fot / Memory * cost of Memory= 1500+200 / 210* 0.3=2.42

Gain Storage Cost of task 3= Task Fit + task Fot / Storage * cost of Storage= 1500+200 / 250 * 0.02=0.136

EGC of task 3 in VM2 = 0.1 + 0.021 + 2.42 + 0.136= 2.677

The EGC of task 3 in VM2 < TB of task 3 → 2.677 < 3 means VM2 is labeled =1.

- **EGC of task 3 in VM3:**

Gain CPU Cost of task 3= length of task / MIPS of VM3 * cost of CPU= 5000 / 1000 * 0. 02=0.1

Gain Bandwidth Cost of task 3= Task Fit + task Fot / Bandwidth * cost of Bandwidth= 1500+200 / 40* 0.001=0.042

Gain Memory Cost of task 3= Task Fit + task Fot / Memory * cost of Memory= 1500+200 / 170* 0.1=1

Gain Storage Cost of task 3= Task Fit + task Fot / Storage * cost of Storage= 1500+200 / 350* 0.04=0.194

EGC of task 3 in VM3 = 0.1+0.042+1+0.194=1.336

The EGC of task 3 in VM3 < TB of task 3 → 1.336 < 3 means VM3 is labeled =1.

- **EGC of task 4 in VM1:**

Gain CPU Cost of task 4= length of task / MIPS of VM1 * cost of CPU= 4000/ 500 * 0.01=0.08

Gain Bandwidth Cost of task 4= Task Fit + task Fot / Bandwidth * cost of Bandwidth= 2000+500/ 60 * 0.001=0.041

Gain Memory Cost of task 4= Task Fit + task Fot / Memory * cost of Memory= 2000+500 / 113 * 0.2=4.42

Gain Storage Cost of task 4= Task Fit + task Fot / Storage * cost of Storage= 2000+500 / 250 * 0.02=0.2

EGC of task 4 in VM1 = 0.08 + 0.041 + 4.42 + 0.2= 4.741

The EGC of task 4 in VM1 < TB of task 4 → 4.741 < 7 means VM1 is labeled =1.

- **EGC of task 4 in VM2:**

Gain CPU Cost of task 4= length of task / MIPS of VM2 * cost of CPU= 4000/ 2000* 0.04=0.08

Gain Bandwidth Cost of task 4= Task Fit + task Fot / Bandwidth * cost of Bandwidth= 2000+500 / 80 * 0.001=0.031

Gain Memory Cost of task 4= Task Fit + task Fot / Memory * cost of Memory= 2000+500 / 210* 0.3=3.57

Gain Storage Cost of task 4= Task Fit + task Fot / Storage * cost of Storage= 2000+500 / 250 * 0.02=0.2

EGC of task 4 in VM2 = 0.08+0.031+3.57+0.2=3.881

The EGC of task 4 in VM2 < TB of task 4 → 3.881 < 7 means VM2 is labeled =1.

- **EGC of task 4 in VM3:**

Gain CPU Cost of task 4= length of task / MIPS of VM3 * cost of CPU= 4000 / 1000 * 0.02=0.08

Gain Bandwidth Cost of task 4= Task Fit + task Fot / Bandwidth * cost of Bandwidth= 2000+500 / 40* 0.001=0.062

Gain Memory Cost of task 4= Task Fit + task Fot / Memory * cost of Memory= 2000+500 / 170* 0.1=1.47

Gain Storage Cost of task 4= Task Fit + task Fot / Storage * cost of Storage= 2000+500 / 350* 0.04=0.285

EGC of task 4 in VM3 = 0.08 + 0.062 + 1.47 + 0.285= 1.897

VM3 is classified =1 because the EGC of task 4→ VM3 is 1.897 7 and the TB of task 4 is 1.897 7.

Then, as indicated in Table 3.16, our model BAS will examine and compare the length and file size information from Table 3.13 for each task to determine how the tasks should be allocated among all available VMs:

Table 3.16: Task priority and comparison of length and file size for each task

Task_ ID	Priority	Len > Fit	Len < Fit	Procedure
1	Time		√	EDT
2	Cost		√	EDG
3	Cost	√		ECG
4	Time	√		ECT

The procedure for obtaining the table values in Table 3.16 is as follows:

Because task 1's TB > Avg.TB →10 >, task 1's priority is time. 6. The EDT was determined using the Len of Task 1→ input file 1000 <2200.

- ✓ The priority of Task 2 is cost, because the TB of Task 2 is an average of 4 TB. 6. The EDG was determined using the Len of Task 2 input file of Task 2→ 2000 < 2750.

Because the TB of task 3 Avg.TB 3 6. Len of task 3 > input file of task 3 →5000 > 1500, the ECG was calculated.

- ✓ Because task 4's TB > Avg.TB 7 >, task 4's priority is time. 6. Len of task 4 > input file of task 4 >→ 4000 > 2000, resulting in ECT calculation.

The task priority, the process, and the expected gain cost should all be applied after computing the expected gain cost and identifying the VM state. We calculated all the

procedures for all the tasks on the VMs based on the findings from Table 3.16, and each task will be scheduled to the appropriate VM as indicated in Table 3.17:

Table 3.17: Performance of each VM

Task_ID	Procedure	VM1	VM2	VM3
1	EDT	41.67	**31.25**	62.5
2	EDG	Not meet TB	Not meet TB	**0.071**
3	ECG	Not meet TB	**0.1**	0.1
4	ECT	8	**5**	6

The following are the computations for the Table 3.17 results:

Task 1:

- Task 1 (EDT) in VM1=DT of pervious task + DT of current task= 0 + 2500/60= 41.66
- Task 1 (EDT) in VM2=DT of pervious task + DT of current task= 0 + 2500/80= 31.25
- Task 1 (EDT) in VM3=DT of pervious task + DT of current task= 0+ 2500/40= 62.5

Task 2:

We did not calculate the predicted gain cost in VM1 and VM2 based on the findings from Table 3.15 because task 2 in the first and second virtual machines did not fulfil the task budget.

- Task 2 (EDG) in VM3= Fit + Fot / Bandwidth * cost of Bandwidth = 2850/40*0.001= 0.071

Task 3:

We do not calculate the predicted gain cost in VM1 based on the data from Table 3.15 since task 3 in the first virtual machine did not fulfill the task budget.

- Task 3 (ECG) in VM2= length of task / CPU * cost of CPU=5000/2000*0.04 =0.1
- Task 3 (ECG) in VM3= length of task / CPU *cost of CPU=5000/1000* 0.02 =0.1

Task 4:

- Task 4 (ECT) in VM1= Ex of pervious task + EX of current task= 4000 / 500 = 8 + 0= 8
- Task 4 (ECT) in VM2= Ex of pervious task (Task 1, Task 3) + EX of current task= ((1000/2000) + (5000/2000)) + (4000 / 2000)= 0.5 + 2.5 + 2 = 5
- Task 4 (ECT) in VM3= Ex of pervious task (Task 2) + EX of current task= (2000 / 1000) + (4000 / 1000) = 2 + 4 = 6
    - Task 1 will be assigned to VM2, which returns less EDT, as shown in Table 3.15.
    - Task 2 will be mapped to VM3, which, among other things, meets the budget constraint.
    - Task 3 will be assigned to VM2, which produces less ECG than the first.
    - Task 4 will be assigned to VM2, which produces less ECT.

3.9.3 Steps of Budget-Aware Scheduling Model

The following are the steps in the BAS model:

Input

List of tasks, list of VMs

Output

Assigning the tasks on available VMs based on the budget constraint

1. The tasks are submitted with their TB

2. Calculate Avg.TB

3. For each task in the task list

Choose the suitable VMs that meet TB

For each VM in list of VMs

Calculate EGC

If the EGC <= TB then VM state = 1

Else: VM state =0

End for

Check the task priority and its requirements

If TB >Avg.TB, the task has time priority:

If Len > Fit, find lower Completion Time (ECT) for the task

else, lower Data Transfer (EDT) time is found

else, if TB ≤ Avg.TB , the task has cost priority:

If Len > Fit, lower ECG is found for the task

else, lower EDG is found for the task

End for

4. Computation performance estimates the metrics:

i. Makespan average based on Equation 3.31.

ii. Based on Equation 3.35, the mean of the whole response time.

iii. Profit for the provider, calculated using Equation 3.36.

iv. Total gain cost based on Equation 3.38.

v. Resource utilization based on Equation 3.37.

vi. Improvement of the cost ratio using Equation 3.39.

3.10 METRICS OF PERFORMANCE

The performance indicators used to evaluate the proposed BAS model are presented in this section. The following measures are used in task scheduling to assess the algorithms' efficiency:

- **Makespan:** The time it takes the available VM to perform all of the jobs is called Makespan. It is defined using Equation 3.28.

$$\textbf{Makespan} = \sum_{i=1}^{n} (ECT[i,j] * A[i,j]) \tag{3.28}$$

In Equation 3.29, where n is the total number of tasks and $1 \leq j \leq m$, A[i, j], a Boolean variable, can be defined as follows:

$$\textbf{A [i,j]} = \begin{cases} 1 & if\ T\ assign\ to\ VM_j \\ 0 & Otherwise \end{cases} \tag{3.29}$$

Then, as shown in Equation 3.30, we calculate the average makespan of all VMs.

$$\textbf{Avg.Makespan} = \frac{\sum_{j=1}^{m} Makespan_j}{m} \tag{3.30}$$

The number of virtual machines is denoted by the letter m.

As a result, Equation 3.31 presents the overall makespan as follows:

Max_span = max (Makespanj) (3.31)

- **Response Time:**

In cloud computing, the reaction time is the time it takes to locate a process, which includes the time it takes to complete the task. Equation 3.32 is used to compute the response time.

RT = FTi - SBi (3.32)

The finish time is FT, while the submission time is SB.

The average reaction time and the mean of all VMs' total response time are then determined using Equations 3.33 and 3.34, respectively.

$$\textbf{Avg.RT}=\frac{\sum_{i=1}^{N} RT_i}{N} \quad (3.33)$$

where N is the number of tasks in specific VM.

$$\textbf{M.Total Avg.RT} = \frac{\sum_{j=1}^{m} Avg.RT_j}{m} \quad (3.34)$$

- **Service Provider Profit:**

The cloud provider is concerned with optimizing profit and resource use, whereas the user is looking for good performance at a low cost. The provider will carry out the tasks in such a way that the user is satisfied and profits are increased. Equation 3.36 will be used to compute the profit.

$$\textbf{Provider profit} = \sum_{i=1}^{n} (TB_i - GC_i) \quad (3.35)$$

For instance, if we suppose:

T1: (TB=120), (GC=135),

T2: (TB=190), and (GC=100),

T3: (TB=20), and (GC=15) then

Profit = (120 – 135) + (190-100) + (20-15) = 80.

Provider profit = 80.

- **Average Resource Utilization:**

The average resource usage rate should be as high as possible. As a result, a better scheduling algorithm is one that distributes jobs to resources that are used efficiently. Equation 3.36 calculates the average resource consumption.

$$\mathbf{RU} = \frac{\sum_{j=1}^{m} Makespan_j}{m * Max\_span} \quad (3.36)$$

- **Total Gain Cost:**

The total gain is the entire cost paid to the service provider by the user for all jobs completed. Equation 3.37 is used to compute the total gain cost.

$$\mathbf{Total\ Gain\ Cost} = \sum_{i=1}^{n} GC_i \quad (3.37)$$

- **Improvement of Cost Ratio:**

The cost is a critical statistic that we consider while evaluating our model's performance. Equation 3.38 is used to calculate the cost ratio improvement.

$$\mathbf{Cost\ ratio} = \left(1 - \frac{\sum_{i=1}^{e} \text{Cost BAS}}{\sum_{i=1}^{e} \text{Cost of other algorithm}}\right) * 100 \quad (3.38)$$

Where e is the number of experimentations

3.11 RESULTS OF EXPERIMENTS AND ANALYSIS

3.11.1 Implementation Environment

We used the Cloud Sim toolbox to simulate our suggested BAS model in this chapter as well. This is due to the fact that the CloudSim toolkit is the most ideal toolkit for efficiently simulating the cloud environment, as well as its modeling behavior. It simulates data centers, hosts, cloudlets, virtual machines, and resource provisioning policies in the cloud. It also allows for the modeling of multiple data centers for the study of resource allocation policies, dependability, and scale.

3.11.2 Experiments Configuration

Table 3.18 explains how the trials will be set up:

Table 3.18: Configuration requirement for the experiments

Configurations	**Datacenter**	**Server**	**Virtual machine**
Numbers	**1**	2	10
Core		Quad-Core and Dual-Core	1 processing element
Memory (RAM)		16 GB	0.5 GB
Storage		1 TB	10 GB image size
Bandwidth		100 GB/s	1 GB/s
VM scheduling algorithm		Time-Shared	Time-Shared
Architecture		X86 Architecture	Xen

Operating system		Linux	
Virtual machine monitor		Xen	
Speed		10,000 MIPS	500, 1000, 2000 and 3000MIPS

3.11.3 Dataset

The dataset is referred to as a cloudlet in this chapter, and the cloudlet represents a task in the CloudSim simulator. In other words, the user's application is the CloudSim toolkit's cloudlet class. As a result, the size of an application is determined by the computing needs, which are expressed in terms of instruction length and data transfer. The cloudlet has the following characteristics:

- ✓ Length.
- ✓ Million Instructions (MI).
- ✓ Input files size.
- ✓ Output files size.

250, 500, 750, 1000, 1500, and 2000 jobs were generated and dispersed. We evaluated the measured metrics (average makespan, mean of total average response time, resources consumption, number of task violations in VMs, provider profit, total gain cost, and VM usage time (hour)) to examine and evaluate our BAS model, as shown in Table 3.19:

Table 3.19: Experiments conducted for BAS model for tasks with 10 VMs

Expts.	No. of VMs	No. of tasks	Average makespan	Mean of total average RT	Resource utilization	No. of violations	Provider profit	Total gain cost	VMs usage time(hour)
1	10	250	781	344	870	34	1605287	657827	2.1
2	10	500	1512	649	1716	70	3291402	1247651	4.2
3	10	750	2251	959	2557	110	4920359	1825919	6.2
4	10	1000	2982	1265	3475	144	6591310	2415496	8.2
5	10	1500	4429	1872	5129	222	9952866	3594399	12.3
6	10	2000	5905	2491	6853	296	13316125	4770199	16.4

3.12 PERFORMANCE EVALUATION

We ran six experiments using ten virtual computers to host increasing quantities of jobs to assess the performance and efficiency of the proposed BAS model. The following experiments were carried out:

i. 250 tasks with ten VMs.

ii. 500 tasks with ten VMs.

iii. 750 tasks with ten VMs.

iv. 1000 tasks with ten VMs.

v. 1500 tasks with ten VMs.

vi. 2000 tasks with ten VMs.

By comparing our suggested BAS model to state-of-the-art scheduling algorithms such as Max-Min, Round Robin, and SJF algorithms, we can demonstrate its efficiency level. Each of these algorithms has a unique technique for allocating jobs to available resources. Table 3.20 presents the results of testing with various numbers of tasks hosted on ten virtual machines (VMs), comparing the average makespan of our proposed BAS model to that of alternative scheduling algorithms. The average makespan for each task conducted on VM in the BAS model has smaller value than the average makespan for tasks run in other scheduling algorithms, as shown in the data.

Table 3.20: Average of makespan of BAS, Max-Min, Round Robin and SJF algorithms

Experiments		**Average makespan**			
No. of tasks	**No. of VMs**	**BAS**	**Max-Min**	**Round Robin**	**SJF**
250	10	**781**	1020	1604	1551
500	10	**1512**	2013	3097	3071
750	10	**2251**	3024	4633	4633
1000	10	**2982**	4031	6159	6170
1500	10	**4429**	6044	9263	9247
2000	10	**5905**	8061	12348	12337

When comparing the outcomes of raising the makespan for increasing the number of tasks while utilizing the same number of VMs, our proposed BAS model performed better, however in all other scenarios, the makespan grows with increasing the number of tasks. Consider experiments 1 and 2, in which 250 and 500 jobs are done on 10 virtual machines, respectively. When compared to algorithms such as Max-Min (achieves 993), Round Robin (1493), and SJF (1520), our proposed BAS model has the smallest makespan increment (1512-781= 731). As a result, our proposed BAS model performs best when the number of jobs is increased. Figure 3.16 shows the graph of these results. In comparison to Max-Min, which has the next greatest makespan values, and Round Robin and SJF, which have larger makespan values, our BAS model has the smallest makespan values for the jobs hosted on the VMs.

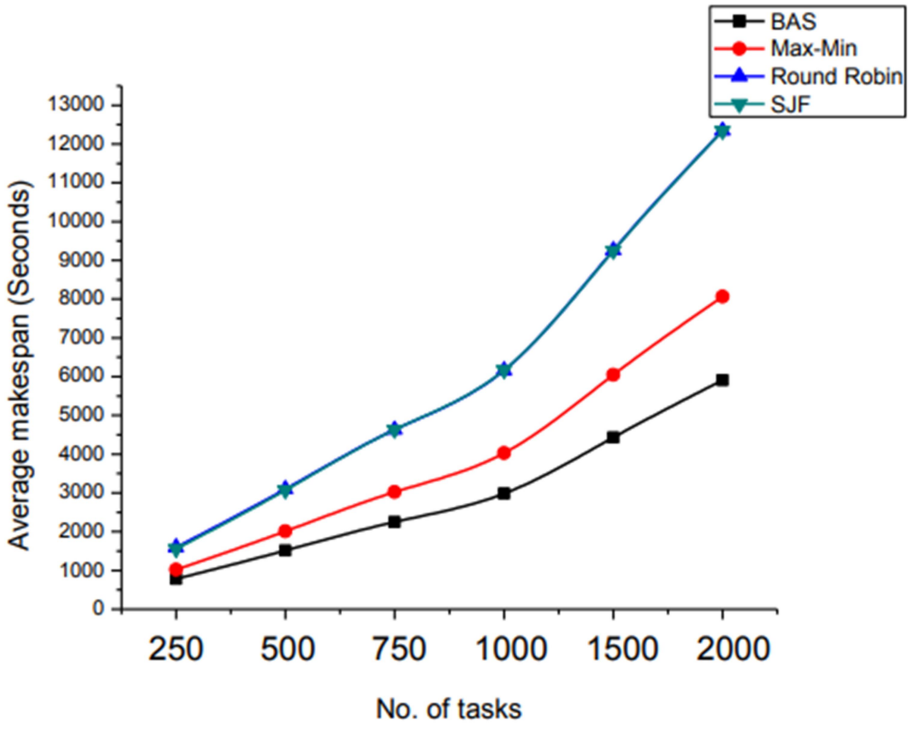


Figure 3.16: Comparison of average makespan

Table 3.21 presents the results of testing performed on various numbers of tasks hosted on ten virtual machines (VMs), comparing the mean of total average response time to execute tasks on VMs for our proposed BAS model to alternative scheduling methods. The results show that the mean of total average reaction time resulting from

the BAS model for each task run on VM is lower than the mean of total average response time arising from other scheduling methods for the same workloads.

Table 3.21: Mean of total average response time of BAS, Max-Min, Round Robin and SJF algorithms

Experiments		Mean of total average response time			
No. of tasks	**No. of VMs**	**BAS**	**Max-Min**	**Round Robin**	**SJF**
250	10	**344**	513	693	654
500	10	**649**	859	1312	1289
750	10	**959**	1276	1948	1945
1000	10	**1265**	1701	2584	2592
1500	10	**1872**	2535	3894	3880
2000	10	**2491**	3389	5188	5178

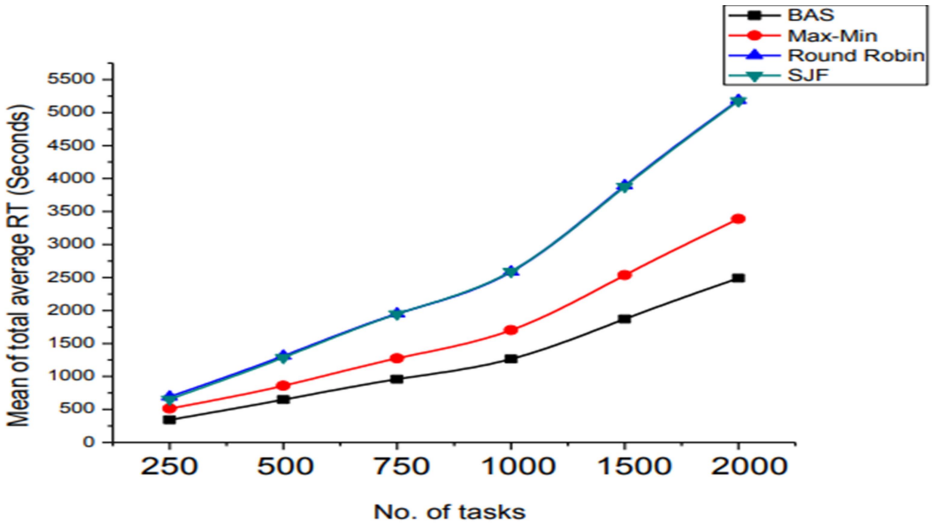


Figure 3.17: Comparison of mean of total average response time

These results, as shown in Figure 3.17, show that our BAS model has the lowest response time in the graph, indicating that the tasks were completed faster than Max-Min. Round Robin and SJF have a faster task execution reaction time. Table 3.22 shows that the scheduling algorithms Max-Min, Round Robin, and SJF were unable to efficiently utilize resources for the execution of the users' activities, whereas the BAS model, which had the highest resource utilization values, demonstrated the novelty of our suggested model.

Table 3.22: Resource utilization of BAS, Max-Min, Round Robin and SJF algorithms

Experiments		**Resource utilization**			
No. of tasks	**No. of VMs**	**BAS**	**Max-Min**	**Round Robin**	**SJF**
250	10	**870**	547	702	694
500	10	**1716**	1130	1383	1361
750	10	**2557**	1537	2080	1982
1000	10	**3475**	2152	2752	2582
1500	10	**5129**	3149	4136	4049
2000	10	**6853**	4070	5498	5387

Figure 3.18 depicts the resource utilization metric's comparing results. When we run different tasks (250, 500, 750, 1000, 1500, and 2000) on ten VMs, the Max-Min algorithm has the smallest utilization graph, indicating that this algorithm creates a lot of load and complexity when scheduling tasks on cloud resources; next comes SJF, which has a smaller utilization factor than Round Robin; and finally our BAS model, which has the highest utilization graph line, indicating that our model creates no overload or complexity when scheduling tasks on cloud resources.

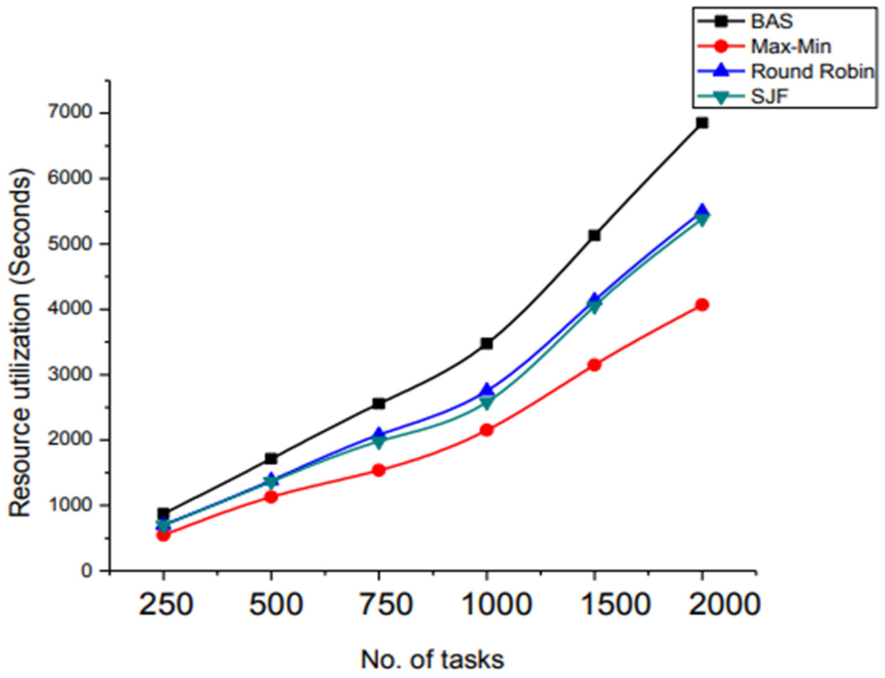


Figure 3.18: Comparison of resource utilization

Table 3.23 presents the results of testing with various numbers of tasks hosted on ten virtual machines (VMs), in which we compare the number of violated tasks for our proposed BAS model to alternative scheduling algorithms. The number of violated tasks for each task conducted on VM in the BAS model has less value than the number of violated tasks executed in other scheduling algorithms, as shown by the findings. As a result of the lesser number of violated tasks in the BAS model, we may predict that tasks will be completed quickly and efficiently.

Table 3.23: Number of task violations of BAS, Max-Min, Round Robin and SJF algorithms

Experiments		**No. of violations**			
No. of tasks	**No. of VMs**	**BAS**	**Max-Min**	**Round Robin**	**SJF**
250	10	**34**	40	41	51
500	10	**70**	75	84	101

750	10	**110**	127	127	158
1000	10	**144**	160	170	227
1500	10	**222**	247	256	313
2000	10	**296**	341	343	418

The results graphed in Figure 3.19 show that the SJF algorithm has many violated tasks; the next algorithm is Round Robin, which is followed by Max- Min; and the BAS model has the lowest values of violation in VM, implying that our BAS model is capable of accessing tasks efficiently on a large number of VMs with fewer violations. When compared to state-of-the-art scheduling algorithms, this demonstrates the originality of our suggested BAS model.

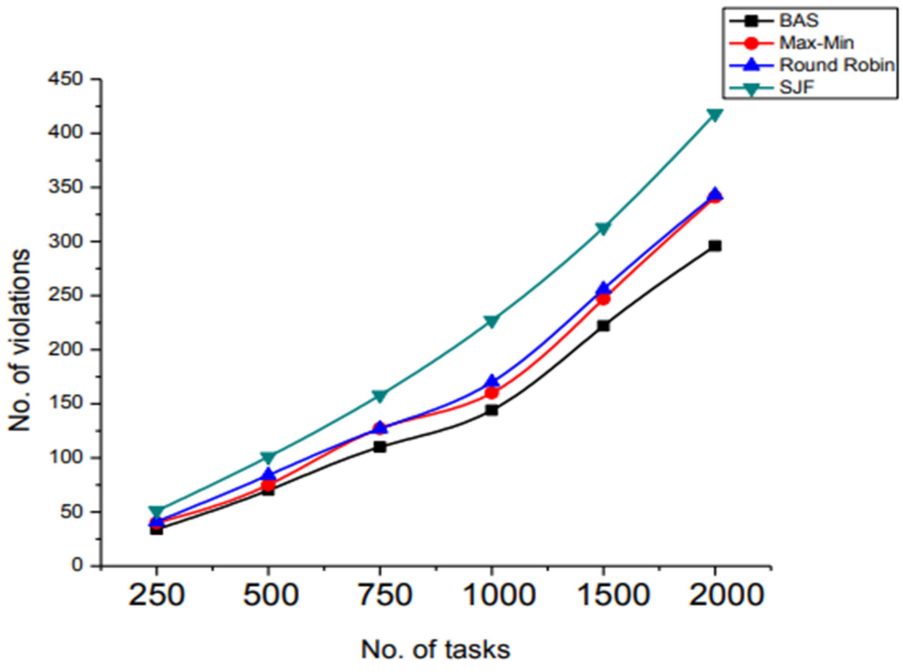


Figure 3.19: Number of violations

The difference between the task budgets and the gain cost after execution is measured as provider profit, indicating how our proposal can produce a gain in modifying task scheduling expenses over cloud resources and the user's budget. When comparing the

provider profit for our proposed BAS model to existing scheduling methods, Table 3.24 summarizes the trials done on varied numbers of jobs hosted on 10 VMs. The provider profit for each task conducted on VM in the BAS model is higher than the provider profit for tasks executed in other scheduling algorithms, as shown in the findings.

Table 3.24: Provider profit of BAS, Max-Min, Round Robin and SJF algorithms

Experiments		**Provider profit**			
No. of tasks	**No. of VMs**	**BAS**	**Max-Min**	**Round Robin**	**SJF**
250	10	**1605287**	1487018	1344750	1367344
500	10	**3291402**	3153615	2695663	2711696
750	10	**4920359**	4569718	4006838	4136371
1000	10	**6591310**	6063645	5349034	5510228
1500	10	**9952866**	9338029	8068379	8279185
2000	10	**13316125**	12150021	10764050	11003674

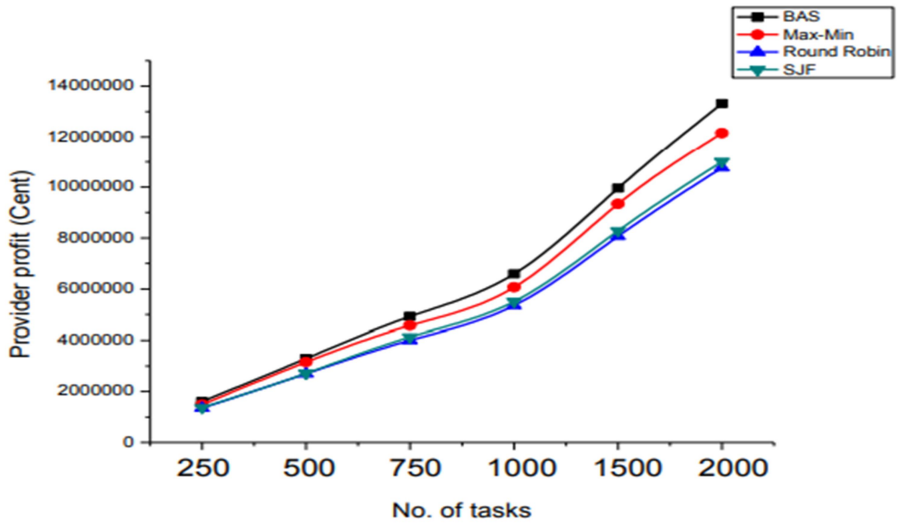


Figure 3.20: Provider profit

We can see from the graphed data in Figure 3.20 that the Round Robin algorithm has the lowest provider profit, indicating that it does not achieve compatibility between user budgets and provider expenses for resource utilization. This is followed by SJF and Max-Min, both of which have a somewhat greater provider profit value. When compared to the others, the BAS model has the greatest values, as seen by its graph, leading us to conclude the uniqueness and accuracy of our BAS model in meeting the user's needs while adhering to the provider's QoS.

The overall gain cost of the scheduling methods is an extremely important parameter to track. Table 3.25 shows how the suggested model BAS achieves its goal of lowering overall gain costs by utilizing resources to complete tasks at lower costs than state-of-the-art scheduling algorithms.

Table 3.25: Total gain cost of BAS, Max-Min, Round Robin and SJF algorithms

Experiments		**Total gain cost**			
No. of tasks	**No. of VMs**	**BAS**	**Max-Min**	**Round Robin**	**SJF**
250	10	**657827**	776106	918366	895773

500	10	**1247651**	1385444	1843389	1827357
750	10	**1825919**	2176551	2739433	2609909
1000	10	**2415496**	2943171	3657772	3496595
1500	10	**3594399**	4209240	5478870	5268092
2000	10	**4770199**	5936315	7322249	7082671

Even with a larger number of tasks on the same number of VMs, the BAS model schedules tasks with the lowest cost increment when compared to other algorithms. For example, in experiments 1 and 2, the BAS model returns the lowest cost increment (1247651-657827= 589824), whereas the algorithms Max-Min (achieves 609338), Round Robin (925023), and SJF have higher cost increments (931584). As a result, we can infer that our proposed BAS model was successful in satisfying users and completing tasks at a lower cost. Round Robin has a larger cost than the SJF method, which is followed by the Max-Min algorithm, as seen in Figure 3.21. All of this suggests that our proposed BAS model, which has the lowest total gain cost, has precisely and effectively achieved its objectives in lowering resource execution costs and achieving budgetary stability between these costs and the users' budgets.

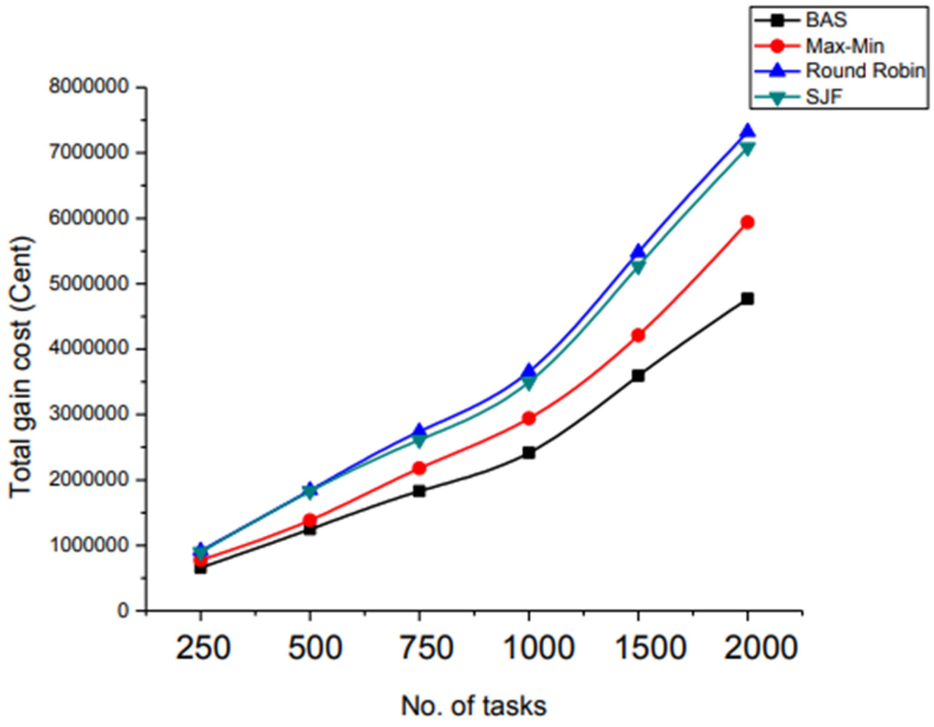


Figure 3.21: Total gain cost

Table 3.26 shows how we calculate our VM consumption per time as allowed by the provider. As a result, we can see that all of the state-of-the-art scheduling algorithms had greater VM utilization per time values. In comparison to the others, our proposed BAS model has the lowest values of VM usage per time, indicating that our model is efficient and capable of delivering good VM per time execution.

Table 3.26: VMs usage time (hour) of BAS, Max-Min, Round Robin and SJF algorithms

Experiments		VMs usage time (hour)			
No. of tasks	**No. of VMs**	**BAS**	**Max-Min**	**Round Robin**	**SJF**

250	10	**2.1**	2.8	4.4	4.3
500	10	**4.2**	5.5	8.6	8.5
750	10	**6.2**	8.4	12.8	12.8
1000	10	**8.2**	11.1	17.1	17.1
1500	10	**12.3**	16.7	25.7	25.6
2000	10	**16.4**	22.3	34.3	34.2

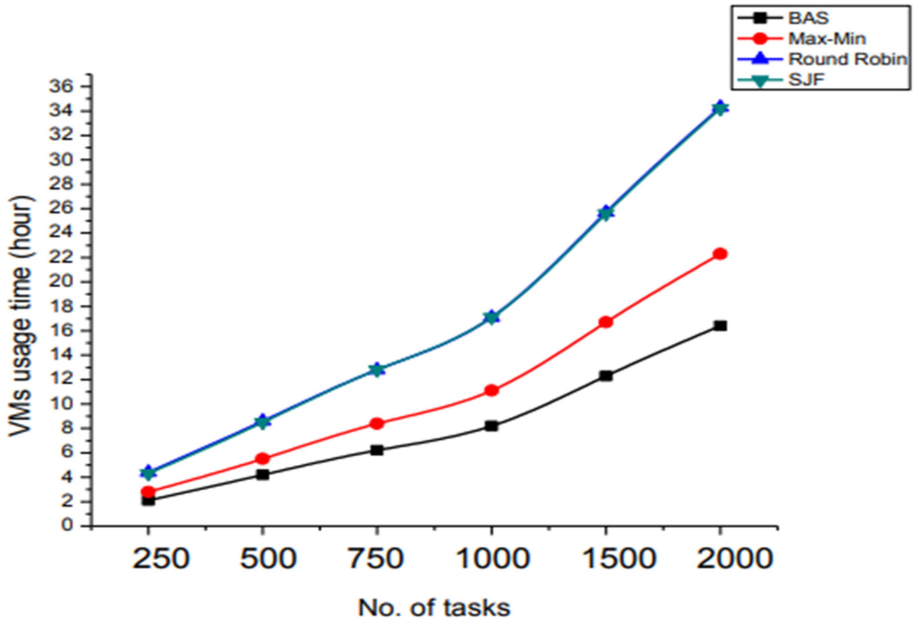


Figure 3.22: Virtual machines usage time (hour)

Figure 3.22 shows that state-of-the-art scheduling strategies result in increased VM use per time (Round Robin, SJF, and Max-Min). Our proposed model BAS, on the other hand, has the lowest graph line, indicating that our model's VM use to execute users' activities per time is efficient and meets the primary goal of scheduling algorithms. Finally, Table 3.27 shows that our suggested BAS model with Max-Min, Round Robin, and SJF algorithms improved the cost ratio by 17 percent, 34 percent, and 31 percent, respectively.

Table 3.27: Improvement of cost ratio for BAS compared with Max-Min, Round Robin, and SJF algorithms

Improvement of cost ratio				
	BAS	**MAX-Min**	**Round Robin**	**SJF**
Experiments	**657827**	776106	918366	895773
	1247651	1385444	1843389	1827357
	1825919	2176551	2739433	2609909
	2415496	2943171	3657772	3496595
	3594399	4209240	5478870	5268092
	4770199	5936315	7322249	7082671
Sum of Expts.	**14511491**	17426827	21960079	21180397
Improvement BAS vs other algorithm		**17%**	**34%**	**31%**
Sum of all cost ratio		82		
Average		**27%**		

Our proposed BAS model was statistically analyzed using the T-test. The T-test is a useful tool for calculating the mean of a BAS model and demonstrating the use of normality. The T-test results of the BAS model vs other algorithms such as Max-Min, Round Robin, and SJF are tabulated in Table 3.28.

Table 3.28: T-test of BAS model compared to Max-Min, Round Robin and SJF algorithms

Metrics	BAS model vs other algorithms		
	Max-Min	**Round Robin**	**SJF**
Average makespan	0.2	0.05	0.05
Mean of total average response time	0.2	0.05	0.05
Resource utilization	0.1	0.2	0.2
No. of violations	0.3	0.3	0.1
Provider profit	0.4	0.2	0.3
Total gain cost	0.3	0.1	0.1
VMs usage time (hour)	0.2	0.05	0.05

We presented a budget model that divides tasks into categories based on their characteristics to make resource allocation decisions easier. This method is carried out in the following manner:

- First, the budget-compliant VMs are labeled, and the task priority is established.
- Task properties (length and file size) are identified.
- The task is assigned to resources that meet the budget limit and keep the makespan as short as possible while incurring the least amount of resource use cost.

The second question is answered by looking at the summary of this chapter: How to create a model that reduces the entire cost of task scheduling in a cloud computing

environment, which includes the cost of processor, memory, bandwidth, and storage, all of which are constrained by a budget. We presented a Budget-Aware Scheduling model, in which we were concerned about the user's budget for their tasks when they were run on virtual machines. Experiments on our proposed BAS model were conducted and compared to state-of-the-art scheduling algorithms, demonstrating that the BAS reduces the makespan, response time, and number of violations for task execution on VMs, as well as increasing resource utilization and profit for the provider, and achieving an acceptable total gain cost for any user. The proposed BAS model improves the cost ratio by 17 percent, 34 percent, and 31 percent over the Max-Min, Round Robin, and SJF algorithms, respectively. In addition, the T-test was used to compare our suggested BAS model to other algorithms such as Max-Min, Round Robin, and SJF statistically.

CHAPTER 4

DEADLINE BUDGET SCHEDULING FOR VIRTUAL CLOUD ENVIRONMENT

In comparison to grid computing, cloud computing has become the most appealing platform, offering a variety of services such as infrastructure, platform, and software as a service, where users can use these services on the cloud and pay based on their usage and the fulfillment of quality-of-service constraints such as deadlines and budgets. Cost monetary metrics must be considered when improving execution time performance within users' stated limits in order to schedule jobs efficiently. The deadline and budget limitations are discussed in this chapter, and the proposed model seeks to meet the goal of assigning jobs to the most appropriate VM based on the deadline and budget restrictions.

4.1 INTRODUCTION

Cloud computing has changed the face of all computing paradigms in recent years by making a wide range of services available over the internet. The cloud's resources, such as infrastructure, platform, and software, are offered as services, with a pay-per-use pricing model. Users would refuse to pay if the required performance is not met, so it is vital to use efficient scheduling in the cloud computing environment to achieve this goal. Furthermore, the inherent heterogeneity, highly dynamic nature, and multidomain characteristics lead to variances in resource costs, as well as computing, communication, and storage capabilities, all of which challenge resource management. The heterogeneous resources, which are used to run heavy computing applications, might be local or distributed geographically. Before beginning any service use, the cloud user always negotiates with the cloud service provider to sign on service level agreements. In the computing model, a user only pays for the services and resources that he requires. The two most essential criteria for a user are money and time. Task scheduling is tough due to cost and time constraints.

In cloud computing, task scheduling is critical since it is responsible for assigning tasks to the appropriate resources. Task limitations are used to submit user jobs to various cloud resources based on computing time and cost (deadline and budget). A

provider can reduce operating expenses while also assuring a high quality of service by ensuring that all users have equal access to shared resources. As a result, task execution must account for scheduling in a heterogeneous environment in order to meet the user-defined deadline, and the monetary costs of completed tasks must not exceed the user-defined budget. Due to rising resource leasing expenses, the cost of execution and resource utilization leased has become increasingly important. In general, task scheduling should meet the following scheduling objectives:

- **Service Level Agreement/Quality of Service Constraint:** The service level agreement should explicitly outline the quality-of-service requirements for scheduling tasks, including a deadline, task scheduling budget, service security, and system reliability. As a result, task scheduling limitations that affect quality of service should be considered in order to meet the quality-of-service requirement.

- **Service Revenue:** Because cloud computing uses a large number of servers, the input cost is higher. As a result, various economic ideas and methods are applied in task scheduling in cloud computing, which appear to be more successful in performance and appear more rational. So, in addition to meeting quality of service limits and resource needs, increasing service revenue has become another essential goal of task scheduling for cloud service providers.

Thus, how can jobs be planned with low payment prices and shorter completion times (makespan) while simultaneously maximizing resource usage in a cloud computing environment has become a growing problem, and it is also considered one of the technical difficulties in the academic sector.

4.2 BUDGET AND DEADLINE CONSTRAINTS

For meaningful scheduling solutions, the deadline and budget values must be discussed between users and cloud service providers so that the agreed values are fair and mutually accepted. The time of task execution under the deadline constraint, as well as the cost, which must not exceed the budget constraint supplied by the user, are guaranteed according to the quality-of-service requirements. As illustrated in Figure 4.1, there are certain measures that can work and fall within the deadline and budget

limits, such as makespan, number of violations, improvement of makespan ratio, improvement of cost ratio, total gain cost, provider profit, and resource usage.

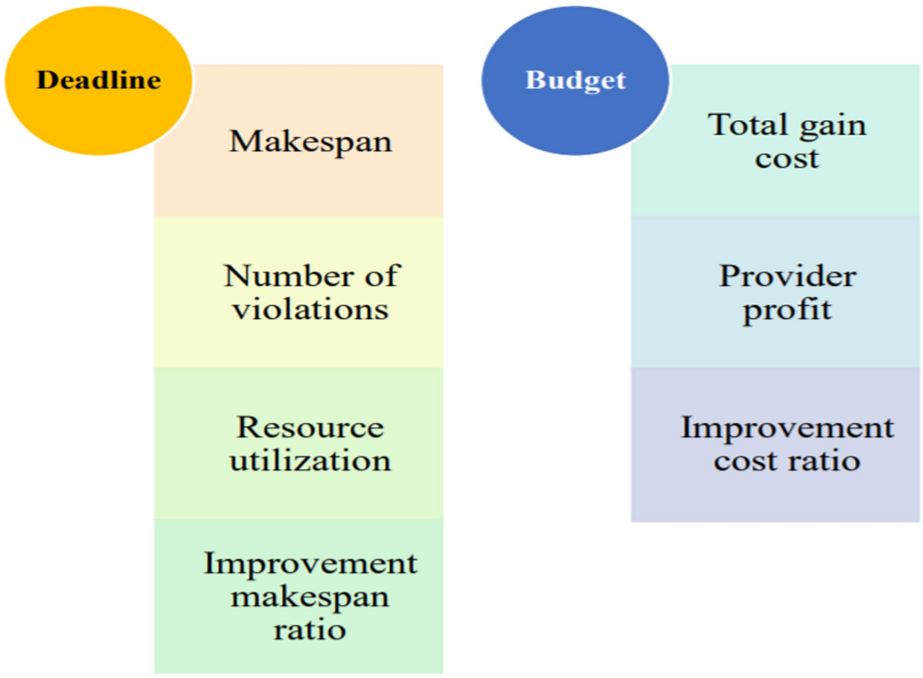


Figure 4.1: Metrics under deadline and budget constraints

4.3 TASK SCHEDULING BASED ON DEADLINE BUDGET MODEL

This contribution emphasizes deadline and budget limitations in task scheduling to improve QoS in conjunction with two basic metrics: minimize execution time and cost of executed activities, and eventually improve service income as well as VM and host resource utilization (s). As a result, we introduced the Deadline Budget Scheduling (DBS) model for executing users' tasks on VMs while adhering to QoS requirements in order to reduce execution time. The DBS model's methodology assumes that the cloud environment is hosted in a data center with a mix of servers, each of which supports multiple virtual machines. Varying VM configurations may have different processor capacities, memory sizes, and communication lines with various bandwidths and storage capacities. The DBS model aims to reduce execution time and cost while staying within a user-defined budget and deadline.

The following scenarios are considered while task scheduling: The virtual machines (VMs) are heterogeneous, with varied performance efficiencies based on the resources available to them. To define the DBS model, we define Dc as a data center made up of many H hosts, each of which holds a set of virtual machines (VMs). VM resources are denoted as RVM=RC, RB, RM, RS, where RC is the CPU capability expressed in Million Instructions Per Second (MIPS). RB stands for bandwidth, RM for virtual machine memory, and RS for virtual machine storage. The processing cost of a task varies based on whose cost VMs it is allocated to. On the other hand, due to bandwidth variability between two separate VMs, the communication cost between two VMs is shifting. Furthermore, the cost of memory and storage varies from one VM to the next, hence each RVM's cost is represented as CR = CRC, CRB, CRM, CRS, accordingly. Each VM implements individual tasks represented by T, which have numerous parameters such as a deadline, a budget, a length, an input file size, and an output file size, and are denoted by T=TD, TB, Len, Fit, Fot. Our goal is to reduce the time to market and the cost of execution while maintaining user happiness.

4.3.1 Type of Task Constraint

Based on user satisfaction, each task has a constraint type. As shown in Equation 4.1 there are three sorts of constraint types in our suggested DBS model.

$$\textbf{Task Constraint Type}=\begin{cases} D \text{ and } B & \text{if the constraints are Deadline and Budget} \\ D & \text{if the constraint is Deadline} \\ B & \text{if the constraint is Budget} \end{cases} \tag{4.1}$$

4.3.2 Resources Clustering

The available resources are clustered in the following fashion based on the user's satisfaction:

i. The first cluster consists of a collection of virtual machines (VMs) that adhere to the budget and deadline requirements.

ii. The second cluster is made up of VMs that adhere to a strict deadline.

iii. The third cluster is made up of VMs that are simply constrained by a budget.

The scheduling DBS model, as shown in Figure 4.2, comprises of cloud users that send their tasks, including the limitations (deadline, budget). Based on user satisfaction, each task is examined for recognizing its constraint type. If the task's constraint type is both deadline and budget, it will be implemented in cluster one; if the constraint type is only deadline, it will be implemented in cluster two; and if the constraint type is only budget, it will be implemented in cluster three. Other task attributes (length, file size) are also taken into account in the selection.

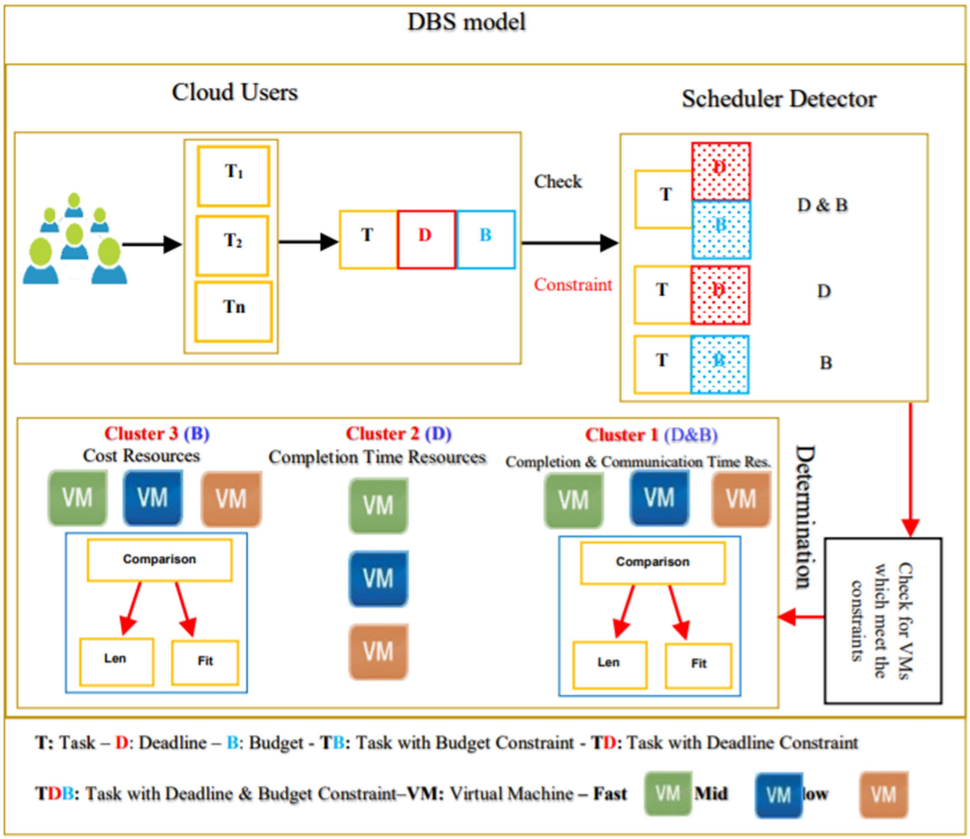


Figure 4.2: Proposed Deadline Budget Scheduling model

4.3.3 Scheduling Strategy

Level one involves determining the task constraint type, whereas level two involves the DBS model determining the task's needs based on the task attributes, as shown in Figure 4.3. The deadline and budget limitations should be met when the constraint type is D & B. For task implementation, our suggested DBS model will find the right VM in the first cluster. The task length (Len) and input file size (Fit) are then compared in stage two, as shown in Equation 4.2. When the task's length exceeds the

input file size, the task's completion time must be considered; otherwise, the data transfer time must be considered.

$$\text{D\&B Constraint Type} = \begin{cases} \text{completion time} & \text{if } Len > Fit \\ \text{data transfer time} & \text{if } Len \leq Fit \end{cases} \quad (4.2)$$

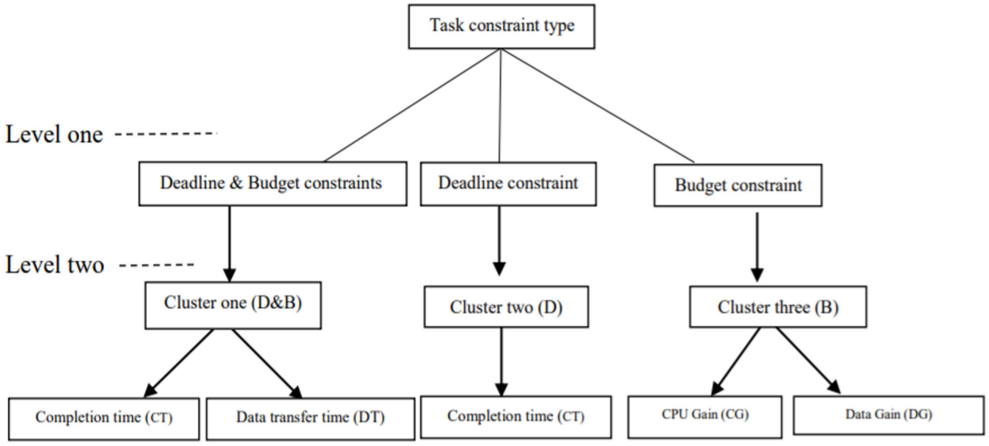


Figure 4.3: Organogram of the DBS model

The VM with the shortest predicted completion time will be chosen in the first cluster. Equation 4.3 is used to compute the Expected Completion Time (ECT). Where p is the number of prior tasks in a given VMj, and EX is the task's execution time, which is computed using Equation 4.4. When the task's length is smaller than the size of the input file, the task will be assigned to a VM with a faster data transmission time. The data transfer time is computed using Equation 4.5.

$$\text{ECT} = \sum_{i=1}^{p} EX_i + \text{EX of current task} \quad (4.3)$$

$$\text{EX} = \frac{Len}{R_C} \quad (4.4)$$

$$\text{DT} = \frac{Fit + Fot}{R_B} \quad (4.5)$$

The task's estimated data transfer time in each VM is then determined using Equation 4.6.

$$\mathbf{EDT} = \sum_{i=1}^{p} DT_i + \text{DT of current task} \quad (4.6)$$

The second constraint type in our proposed DBS model is D, which focuses solely on the deadline constraint; the task completion time is calculated in the second cluster, and the task is assigned to the VM with the shortest predicted completion time. Equation 4.3 specifies the estimated completion time. Our proposed DBS model's final constraint type is B. The task requirement is to calculate the cost of VMs in the third cluster using Equation 4.7, which compares the task length and input file size as follows:

$$\mathbf{B\ Constraint\ Type} = \begin{cases} \text{CPU Gain} & \text{if} \quad \text{Len} > \text{Fit} \\ \text{Data Transfer Gain} & \text{if} \quad \text{Len} \leq \text{Fit} \end{cases} \quad (4.7)$$

When the task is the longest, it is computed using Equation 4.8, assuming that a faster machine is used by selecting a VM with lower Expected CPU Gain (ECG). When the file size is the highest, however, the data transfer cost is remained high. The task will be translated to a faster virtual machine (VM), and the Expected Data Gain (EDG) will be determined using Equation 4.9. Finally, using Equation 4.10, the task's Gain Cost (GC) will be determined. The scheduling model computes the optimal solution to determine the suitable scheduling decision after updating the state of all VMs.

$$\mathbf{ECG} = \frac{Len_i}{R_C} * C_{RC} \quad (4.8)$$

$$\mathbf{EDG} = \frac{Fit_i + Fot_i}{R_B} * C_{RB} \quad (4.9)$$

$$\mathbf{GC} = \left(\frac{Len_i}{R_C} * C_{RC} + \frac{Fit_i + Fot_i}{R_B} * C_{RB} + \frac{Fit_i + Fot_i}{R_M} * C_{RM} + \frac{Fit_i + Fot_i}{R_S} * C_{RS}\right) \quad (4.10)$$

4.3.4 Case Study

To demonstrate the mechanism of our proposed DBS model, we use Tables 4.1 and 4.2 to provide task attributes and virtual machine configurations, respectively:

Table 4.1: Task attributes

Task _ID	Length	Input file size	Output file size	TB	TD	Constraint type
1	1000	1200	300	3	1	D & B
2	2000	2750	100	3	-	B
3	5000	1500	200	-	10	D
4	4000	2000	500	4	-	B

Table 4.2: VM configurations

VM_Id	VM_MIPS	VM-BW	VM-RM	VM-ST	Cost-MIPS	Cost-BW	Cost-RM	Cost-ST
1	500	60	113	250	0.01	0.001	0.2	0.02
2	2000	80	210	250	0.04	0.001	0.3	0.02
3	1000	40	170	350	0.02	0.001	0.1	0.04

The three types of constraint types used in our proposed DBS model, as shown in Equation 4.1, are used to detect the constraint type of each task. The tasks are divided among the available VMs based on the identified constraint type; when the constraint type is D & B, the expected deadline and estimated gain cost are calculated to see which VMs meet the two constraints: deadline and budget. If the constraint type is D, the expected deadline is checked to see if each VM has met the deadline constraint; if not, the expected gain cost is calculated to find the VM that has met the budget constraint; and each VM is then classified into cluster one, cluster two, and cluster three, as shown in Table 4.3.

Table 4.3 Expected deadline and gain cost of each task into each VM

Task _ID	Expected deadline			Gain Cost (GC)			Cluster of VM		
	VM1	VM2	VM3	VM1	VM2	VM3	VM1	VM2	VM3
1	2	0.5	1	2.815	2.298	1.107	--	C1	C1
2	---	---	---	5.355	4.373	2.036	--	--	C3
3	10	2.5	5	---	---	---	C2	C2	C2
4	---	---	---	4.741	3.881	1.897	--	C2	C3

The expected deadline is not calculated in tasks 2 and 4 because the user has already defined the constraint as budget; also, the gain cost is not calculated in task 3 because the user has already defined the constraint as the deadline.

The methods for obtaining the values in Table 4.3 are as follows:

The deadline and budget constraint types are used in task 1. For task 1, the projected deadline and gain cost are computed as follows:

- ✓ **Task 1 in VM1:**

Expected deadline of task 1= length of task / MIPS of VM1= 1000/ 500 = 2

Gain CPU Cost of task 1= length of task / MIPS of VM1 * cost of CPU= 1000 / 500 *

0.01= 0.02

Gain Bandwidth Cost of task 1= Fit + Fot / Bandwidth * cost of Bandwidth= 1200 + 300 / 60 * 0.001= 0.025

Gain Memory Cost of task 1= Fit + Fot / Memory * cost of Memory= 1200 + 300 / 113 * 0.2= 2.65

Gain Storage Cost of task 1 = Fit + Fot / Storage * cost of Storage= 1200 + 300 / 250 * 0.02= 0.12

EGC of task 1 in VM1 = 0.02 + 0.025 + 2.65 + 0.12= 2.815

Expected deadline of task 1 in VM1> TD of task 1 □ 2 > 1.

Expected GC of task 1 in VM1 < TB of task 1 □ 2.815 < 3.

As a result, VM1 is tagged as 0 because the Expected deadline = 2 (beyond the constraint defined) and Expected GC=2.815. The constraints are not met by VM1.

- ✓ **Task 1 in VM2:**

Expected deadline of task 1 = length of task / MIPS of VM2= 1000/ 2000=0.5

Gain CPU Cost of task 1= length of task / MIPS of VM2 * cost of CPU = 1000 / 2000 * 0.04= 0.02

Gain Bandwidth Cost of task 1 = Fit + Fot / Bandwidth * cost of Bandwidth= 1200 + 300 / 80 * 0.001= 0.018

Gain Memory Cost of task 1= Fit+ Fot / Memory * cost of Memory= 1200 + 300 / 210 * 0.3= 2.14

Gain Storage Cost of task 1 = Fit + Fot / Storage * cost of Storage= 1200 + 300 / 250 * 0.02= 0.12

EGC of task 1 in VM2 = 0.02 + 0.018 + 2.14 + 0.12= 2.298

supposed deadline of task 1 in VM2 < TD of task 1 → 0.5 < 1

Expected GC of task 1 in VM2 < TB of task 1 → 2.298 < 3.

As a result, VM2 is designated as meeting both conditions, with an expected deadline of 0.5 and an expected GC of 2.298.

- ✓ **Task 1 in VM3:**

Expected deadline of task 1= length of task / MIPS of VM3= 1000/ 1000= 1

Expected GC of task 1= length of task / MIPS of VM3 * cost of CPU= 1000 / 1000 * 0.02= 0.02

Gain Bandwidth Cost of task 1= Fit + Fot / Bandwidth * cost of Bandwidth= 1200 + 300 / 40 * 0.001= 0.037

Gain Memory Cost of task 1= Fit + Fot / Memory * cost of Memory= 1200 + 300 / 170 * 0.1= 0.88

Gain Storage Cost of task 1= Fit + Fot / Storage * cost of Storage= 1200 + 300 / 350 * 0.04= 0.17

EGC of task 1 in VM3 = 0.02 + 0.037 + 0.88 + 0.17= 1.107

Expected deadline of task 1 in VM3= TD of task 1 → 1 = 1.

Expected GC of task 1 in VM3 < TB of task 1 → 1.107 < 3.

As a result, VM3 is tagged as 1 with an Expected deadline of 1 and an Expected GC of 1.107. VM3 satisfies both requirements.

Because the constraint type for task 2 is budget, you just need to calculate the gain cost as follows:

- ✓ **Task 2 in VM1:**

Gain CPU Cost of task 2= length of task / MIPS of VM1 * cost of CPU= 2000 / 500 * 0.01= 0.04

Gain Bandwidth Cost of task 2= Fit + Fot / Bandwidth * cost of Bandwidth= 2750 + 100 / 60 * 0.001= 0.047

Gain Memory Cost of task 2= Fit + Fot / Memory * cost of Memory= 2750 + 100 / 113 * 0.2= 5.04

Gain Storage Cost of task 2= Fit + Fot / Storage * cost of Storage= 2750 + 100/250* 0.02= 0.228

EGC of task 2 in VM1 = 0.04 + 0.047 + 5.04 + 0.228 = 5.355

VM1 is designated =0 because expected GC of task 2 in VM1 > TB of task 2→5.355 > 3.

- ✓ **Task 2 in VM2:**

Gain CPU Cost of task 2= length of task / MIPS of VM2 * cost of CPU= 2000 / 2000 * 0.04= 0.04

Gain Bandwidth Cost of task 2= Fit + Fot / Bandwidth * cost of Bandwidth= 2750 + 100 / 80 * 0.001= 0.035

Gain Memory Cost of task 2= Fit + Fot / Memory * cost of Memory= 2750 + 100 / 210* 0.3= 4.07

Gain Storage Cost of task 2= Fit + Fot / Storage * cost of Storage= 2750 + 100 / 250 * 0.02= 0.228

EGC of task 2 in VM2 = 0.04 + 0.035 + 4.07 + 0.228 = 4.373

VM2 is designated =0 because expected GC of task 2 in VM2 > TB of task 2 → 4.373 > 3.

- ✓ **Task 2 in VM3:**

Gain CPU Cost of task 2= length of task / MIPS of VM3 * cost of CPU= 2000 / 1000 * 0.02= 0.04

Gain Bandwidth Cost of task 2= Fit + Fot / Bandwidth * cost of Bandwidth= 2750 + 100 / 40 * 0.001= 0.071

Gain Memory Cost of task 2= Fit + Fot / Memory * cost of Memory= 2750 + 100 / 170 * 0.1= 1.6

Gain Storage Cost of task 2= Fit + Fot / Storage * cost of Storage= 2750 + 100 / 350 * 0.04= 0.325

EGC of task 2 in VM3 = 0.04 + 0.071 + 1.6 + 0.325 = 2.036

Expected GC of task 2 in VM3 < TB of task 2 → 2.036 < 3 means VM3 is labeled =1.

Because the constraint type for task 3 is deadline, you must compute the projected deadline as follows:

- ✓ **Task 3 in VM1:**

Expected deadline of task 3= length of task / MIPS of VM1= 5000/ 500 = 10

Expected deadline of task 3 in VM1 = TD of task 3→10 = 10 means VM1 is labeled =1.

- ✓ **Task 3 in VM2:**

Expected deadline of task 3= length of task / MIPS of VM1= 5000/ 2000 = 2.5

Task 3's expected deadline in VM2 equals TD of task 3→2.5, which indicates VM2 is designated =1.

- ✓ **Task 3 in VM3:**

Expected deadline of task 3= length of task / MIPS of VM1= 5000/ 1000 = 5

Expected deadline of task 3 in VM3 < TD of task 3→5 < 10 means VM3 is labeled =1.

Because the constraint type in task 4 is budget, you must calculate the gain cost as follows:

- ✓ **Task 4 in VM1:**

Gain CPU Cost of task 4= length of task / MIPS of VM1 * cost of CPU= 4000 / 500 * 0.01= 0.08

Gain Bandwidth Cost of task 4= Fit + Fot / Bandwidth * cost of Bandwidth= 2000+500 / 60 * 0.001= 0.041

Gain Memory Cost of task 4= Fit + Fot / Memory * cost of Memory= 2000 + 500 / 113 * 0.2= 4.42

Gain Storage Cost of task 4= Fit + Fot / Storage * cost of Storage= 2000 + 500 / 250 * 0.02= 0.2

EGC of task 4 in VM1 = 0.08 + 0.041 + 4.42 + 0.2 = 4.741

Expected GC of task 4 in VM1 > TB of task 4 → 4.741 > 4 means VM1 is labeled =0.

- ✓ **Task 4 in VM2:**

Gain CPU Cost of task 4= length of task / MIPS of VM2 * cost of CPU=4000/ 2000 * 0.04= 0.08

Gain Bandwidth Cost of task 4= Fit + Fot / Bandwidth * cost of Bandwidth= 2000+500 / 80 * 0.001= 0.031

Gain Memory Cost of task 4= Fit + Fot / Memory * cost of Memory= 2000 + 500 / 210 * 0.3= 3.57

Gain Storage Cost of task 4= Fit + Fot / Storage * cost of Storage= 2000+500 / 250 * 0.02= 0.2

EGC of task 4 in VM2 = 0.08 + 0.031 + 3.57 + 0.2 = 3.881

TB of task 4 3.881 4 implies VM2 is labelled =1. Expected GC of task 4 in VM2 3.881 4 means VM2 is labelled =1.

- ✓ **Task 4 in VM3:**

Gain CPU Cost of task 4= length of task / MIPS of VM3*cost of CPU= 4000 / 1000* 0.02= 0.08

Gain Bandwidth Cost of task 4= Fit + Fot / Bandwidth * cost of Bandwidth= 2000+500 / 40 * 0.001= 0.062

Gain Memory Cost of task 4= Fit + Fot / Memory * cost of Memory= 2000 + 500 / 170 * 0.1= 1.47

Gain Storage Cost of task 4= Fit + Fot / Storage * cost of Storage= 2000 + 500 / 350 * 0.04= 0.285

EGC of task 4 in VM3 = 0.08 + 0.062 + 1.47 + 0.285 = 1.897

VM3 is tagged =1 because the expected GC of task 4 in VM3 is 1.897 and the TB of task 4 is 1.897.

Following that, as indicated in Table 4.4, our suggested DBS model will check and compare the length and file size for each task.

Table 4.4: Task constraint type and comparison of length and file size for each task

Task_ ID	Constraint type	Len > Fit	Len < Fit	Procedure
1	D & B		√	EDT
2	B		√	EDG
3	D	√		ECT
4	B	√		ECG

The procedure for obtaining the table values in Table 4.4 is as follows:

The time and cost constraint is the first sort of constraint encountered in this task:

- ✓ The EDT was computed by multiplying the Len of task 1 by the input file of task 1 → 1000 and 1200.

The cost constraint is the second sort of restriction in this task:

- ✓ The EDG was determined using the Len of task 2 input file of task 2→ 2000 2750.

Time is the third type of restriction in Task 3:

As a result, the ECT was calculated using the following formula:

- ✓ Len of task 3 > input file of task 3 → 5000 > 1500

The cost constraint is the fourth type of constraint:

As a result, the ECG was calculated using the following formula: Len of task 4 > input file of task 4 →4000 > 2000.

Each task should be assigned to a VM that provides efficient performance, after which we calculate the EDT, EDG, ECT, and ECG using Equations 4.6, 4.9, 4.3, and 4.8 as shown in Table 4.5:

Table 4.5: Performance of each VM

Task_ID	Procedure	VM1	VM2	VM3
1	EDT	Not meet TD & TB	18.75	37.5
2	EDG	Not meet TB	Not meet TB	0.07
3	ECT	10	3	5
4	ECG	Not meet TB	0.08	0.08

We calculated all of the procedures for all of the tasks on the VMs using the results from Table 4.4. The following are the computations for the Table 4.5 results:

- **Task 1:**

Because task 1 in the first virtual machine failed to meet the task deadline and budget limits, we did not calculate the expected deadline and expected gain cost in VM1 using Table 4.3's values.

Task 1 EDT VM2 = DT of pervious task + DT of task = 0 + 1500 / 80 = 18.75

Task 1 EDT VM3 = DT of pervious task + DT of task = 0 + 1500 / 40 = 37.5

- **Task 2:**

We did not compute the predicted gain cost in VM1 and VM2 based on the data from Table 4.3 since task 2 in the first and second virtual machines did not match the task budget.

Task 2 EDG VM3=Fit + Fot / Bandwidth * cost Bandwidth=2850/40*0.001=0.07

- **Task 3:**

Task 3 ECT VM1=Ex of pervious task +EX of current task= 5000 / 500=10+0=10

Task 3 ECT VM2 = Ex of pervious task (task 1) + EX of current task= (1000/2000) + (5000/2000)= 0.5 + 2.5= 3

Task 3 ECT VM3 = Ex of pervious task (task 2) + EX of current task= (2000/1000) + (5000 / 1000)= 2+5= 7

- **Task 4:**

We do not calculate the predicted gain cost in VM1 based on the data from Table 4.3 since task 4 in the first virtual machine did not fulfill the task budget.

Task 4 ECG VM2 =length task / CPU * cost CPU = 4000 / 2000 * 0.04 = 0.08

Task 4 ECG VM3 =length task / CPU * cost CPU = 4000 / 1000 * 0.02 = 0.08

- ✓ Task 1 will be assigned to VM2, which has a lower EDT.
- ✓ Task 2 will be assigned to VM3 if it matches the task's requirements.
- ✓ Task 3 will be assigned to VM2, which will produce less ECT.
- ✓ Task 4 will be assigned to VM2, which produces less ECG than the first.

Deadline Procedures Model for Budget Scheduling

The DBS model has the following steps:

Input	List of unmapped tasks which have user-defined constraints (deadline,

	budget)
Output	Minimizing the makespan and expenditure cost for these tasks

1	For each task check the user-defined constraint where:
	
	If a user-defined constraint is TD & TB:
	
	a. Create cluster one from a set of VMs which meet the two constraints.
	
	b. Check the task requirements
	
	For each VM in cluster one
	
	If Len > Fit then
	
	Calculate ECT
	
	else
	
	Calculate EDT
	
	End for
	
	c. Assign the task to VM that returns less ECT
	
	Else, if a user-defined constraint is TB:
	
	Create cluster three of VMs which meet the budget constraints.
	
	Check the task requirements
	
	For each VM in cluster three
	
	If Len > Fit then
	
	Calculate ECG
	
	else
	
	Calculate EDG

	End for Assign the task to fastest VM. End for
2.	**Computation performance** Calculate the metrics: o Average makespan based on the Equation 4.13. o Total cost based on the Equation 4.14. o Number of violations based on the Equation 4.15. o Profit of provider based on the Equation 4.16. o Resource utilization based on the Equation 4.17. o Improvement of makespan ratio based on the Equation 4.18. o Improvement of cost ratio based on the Equation 4.19.

4.4 PERFORMANCE METRICS

The performance indicators used to evaluate the suggested DBS model are presented in this section. The following measures are used in task scheduling to assess the algorithms' efficiency:

- **Makespan:** In terms of the user, the amount of time it takes to accomplish the tasks must be lowered. As specified in Equation 4.11, the makespan is the time it takes for all jobs conducted by a single VM to complete.

$$\textbf{Makespan} = \sum_{i=1}^{n} (ECT_i * A[i,j]) \quad (4.11)$$

In Equation 4.12, where n is the total number of tasks and $1 \leq j \leq m$, A[i, j], a, Boolean variable can be defined as follows:

$$\mathbf{A\ [i,j]} = \begin{cases} 1 & if\ T\ is\ assigned\ to\ VM_j \\ 0 & Otherwise \end{cases} \quad (4.12)$$

The average makespan is then determined for all VMs, as shown in Equation 4.13.

$$\mathbf{Avg.Makespan} = \frac{\sum_{j=1}^{m} Makespan_j}{m} \quad (4.13)$$

- **Cost of Gain in Total:** Another performance indicator utilised to evaluate our suggested DBS model is cost, which is defined by an algorithm's capacity to execute tasks within a given budget limit, and is calculated by comparing the task's cost to the budget. Equation 4.14 is used to compute the total gain cost.

$$\mathbf{Total\ Gain\ Cost} = \sum_{i=1}^{n} GC_i \quad (4.14)$$

- **NoV (Number of Violations):** The total number of tasks that miss their deadline is the number of violations. In Equation 4.15, the NoV is defined.

$$\mathbf{Number\ of\ Violations} = \sum_{i=1}^{n} TvD \quad (4.15)$$

- **Profit for the provider:** Profit, which is computed by subtracting the actual implemented cost of the task from the budget of the completed task, is the most important indicator for service providers. Equation 4.16 will be used to compute the profit.

$$\mathbf{Provider\ Profit} = \sum_{i=1}^{n} (TB_i - GC_i) \quad (4.16)$$

- **Resource Utilization on the Average:** Another important measure that the

service provider is concerned about is resource consumption. In comparison to existing algorithms where resource utilization should be maximized, our proposed DBS model was evaluated by efficient resource use by achieving the users' limitations. Equation 4.17 defines the average resource consumption.

$$\mathbf{RU}=\frac{\sum_{j=1}^{n} \text{number of successful tasks}}{\sum_{k=1}^{m} R_C} \quad (4.17)$$

- **Improvement of Makespan Ratio:** Because makespan is such an important metric for our model's performance, we need to calculate the improvement ratio of this metric. Equation 4.18 is used to compute the improvement in the makespan ratio.

$$\textbf{Makespan ratio} = \left(1 - \frac{\sum_{i=1}^{e} \text{Avg.Makespan DBS}}{\sum_{i=1}^{e} \text{Avg.Makespan of other algorithm}}\right) * 100 \quad (4.18)$$

where e is the number of experiments

- **Cost Ratio Improvement:** The cost ratio is a key parameter for evaluating our DBS model's performance. As a result, we used Equation 4.19 to compute the cost ratio improvement.

$$\textbf{Cost ratio} = \left(1 - \frac{\sum_{i=1}^{e} \text{Cost DBS}}{\sum_{i=1}^{e} \text{Cost of other algorithm}}\right) * 100 \quad (4.19)$$

4.5 EXPERIMENTAL RESULTS AND ANALYSIS

4.5.1 Environment for Implementation

The DBS model experiments were implemented using the CloudSim toolkit simulator. The CloudSim is distinguished by its ability to model behavior for cloud system components such as data centers, processing elements, virtual machines, and so on. Researchers and industry developers can focus on specific and essential system design concerns without having to worry about infrastructure or cloud-based services.

Because of its flexibility and simplicity, the CloudSim toolkit is a widely used simulator that runs on the Java programming language. The CloudSim's benefits include the ability to create policies that connect tasks to VMs, allocate VMs for hosting in data centers, scheduling VMs, and track energy use.

4.5.2 Experiments Configuration

The Table 4.6 clarifies the configuration for the tests:

Table 4.6: Configuration requirement for the experiments

Configurations	Data center	Server	Virtual machine
Numbers	1	2	5, 7, 9, and 11
Core		Quad-Core and Dual-Core	1 processing element
Memory (RAM)		16 GB	0.5 GB
Storage		1 TB	10 GB image size
Bandwidth		100 GB/s	1 GB/s
VM scheduling algorithm		Time-Shared	Time-Shared
Architecture		X86 Architecture	Xen
Operating system		Linux	
Virtual machine monitor		Xen	
Speed		10,000 MIPS	500, 1000, 2000 and 3000 MIPS

4.5.3 Dataset

The CloudSim allows you to model the task using a cloudlet, which is a programming structure. The dataset in this section of the research is a cloudlet, and the cloudlet represents a task in the CloudSim simulator. The number of instructions that must be implemented is listed in the cloudlet. A cloudlet has the following characteristics:

1. Length.
2. Million Instructions (MI).
3. Input file size.
4. Output file size

There is also the option to expand and add more attributes such as deadlines and budget limits. Tasks were developed and distributed as 250, 500, 750, and 1000 tasks to analyze the results of our DBS model. Table 4.7 shows the results of the calculated metrics (average makespan, total gain cost, number of infractions, provider profit, and average resource utilization):

Table 4.7: Experiments conducted for tasks with different VMs of the DBS model

Expts.	No. of VMs	No. of tasks	Average makespan	Total gain cost	Number of violations	Provider profit	Average resource utilization
1	5	250	1883	576158	10	1686740	1827
2	7	500	2879	1200682	18	3337891	2927
3	9	750	3246	1770457	45	4975104	3174
4	11	1000	3659	2359776	45	6646103	3719

4.6 PERFORMANCE EVALUATION

We ran four tests to test the performance and efficiency of the proposed DBS model, in which we used different numbers of VMs to host different numbers of jobs. The following experiments were carried out:

1. 250 tasks with five VMs.
2. 500 tasks with seven VMs.
3. 750 tasks with nine VMs.
4. 1000 tasks with eleven VMs.

Comparing our suggested DBS model to state-of-the-art scheduling algorithms such as Genetic (GA), Max-Min, Round Robin, and SJF algorithms demonstrates its efficiency level. Each of these algorithms has a unique technique for allocating jobs to available resources.

One of the main goals of this study is to evaluate the performance of our suggested DBS model. Key performance measures are used to assess the efficacy of our model's performance from both the users' and service providers' viewpoints. As a result, we define makespan as the task of time it takes to accomplish all tasks as well as the expense of doing so. The remaining budget and deadline metrics show if the suggested DBS model is capable of completing the job within its restrictions.

Table 4.8 shows the results of testing with various numbers of tasks hosted on varying numbers of VMs, comparing the average makespan of our proposed DBS model to that of existing scheduling algorithms. The average makespan for each task conducted on VM using the DBS model is less than the average makespan for tasks executed using the other scheduling algorithms, as shown in the results.

Table 4.8: Average of makespan of DBS, GA, Max-Min, Round Robin and SJF algorithms

Experiments		Average makespan				
No. of tasks	**No. of VMs**	**DBS**	**GA**	**Max-Min**	**Round Robin**	**SJF**
250	5	1883	2922	1926	3114	3052
500	7	2879	4909	3020	5015	4933
750	9	3246	5362	3402	5374	5383
1000	11	3659	6166	3922	6325	6284

As shown in Figure 4.4, our suggested DBS model is capable of running tasks according to user satisfaction in a heterogeneous environment and minimizes the average makespan in all trials when compared to other algorithms such as GA, Max-Min, Round Robin, and SJF. By running the jobs (250, 500, 750, 1000) on VMs (5, 7, 9, 11), we can see that the SJF method has the longest average makespan, followed by the Round Robin algorithm, the GA, and finally the Max-Min. Finally, the least makespan is represented in the DBS model, which shows that the tasks' execution time is as short as feasible given the deadline and budget limitations, maximizing the DBS model's performance level.

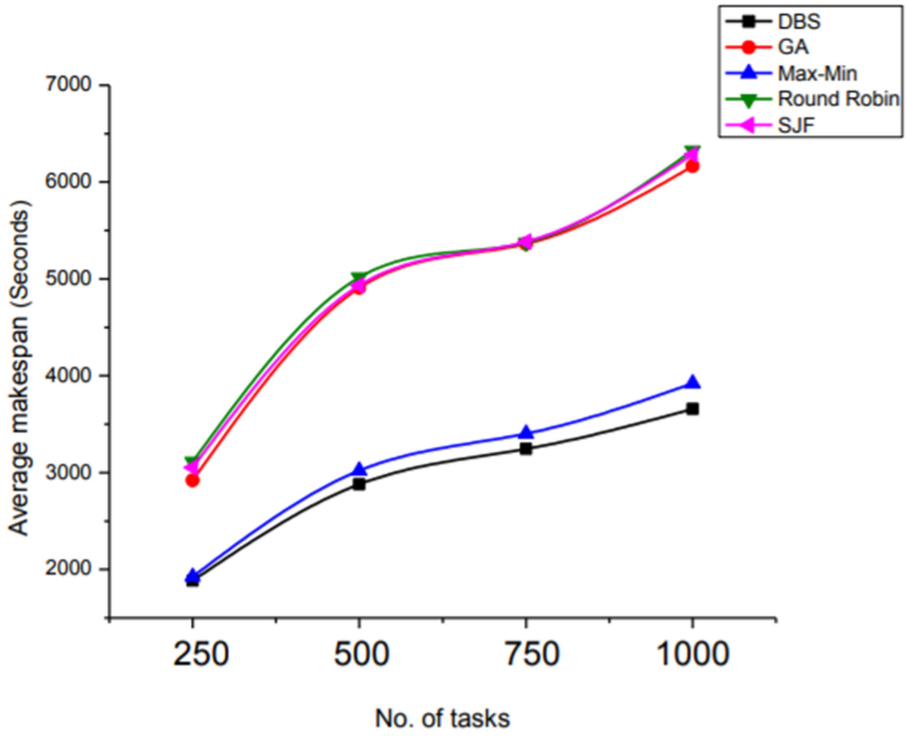


Figure 4.4: Comparison of average makespan

Table 4.9 shows how the proposed DBS model achieves its goal of improving overall gain costs by utilizing resources to perform activities at lower costs than state-of-the-art scheduling algorithms.

Table 4.9: Total gain cost of DBS, GA, Max-Min, Round Robin and SJF algorithms

Experiments		**Total gain cost**				
No. of tasks	**No. of VMs**	**DBS**	**GA**	**Max-Min**	**Round Robin**	**SJF**
250	5	576158	848033	660214	859654	921143
500	7	1200682	1910206	1418749	1893837	1894094
750	9	1770457	2728512	2050902	3103795	2878976
1000	11	2359776	3744648	2797295	3811976	3811306

The lowest gain cost was for tasks done with the DBS model, whereas the highest cost was for activities executed with the SJF method, as shown in Figure 4.5.

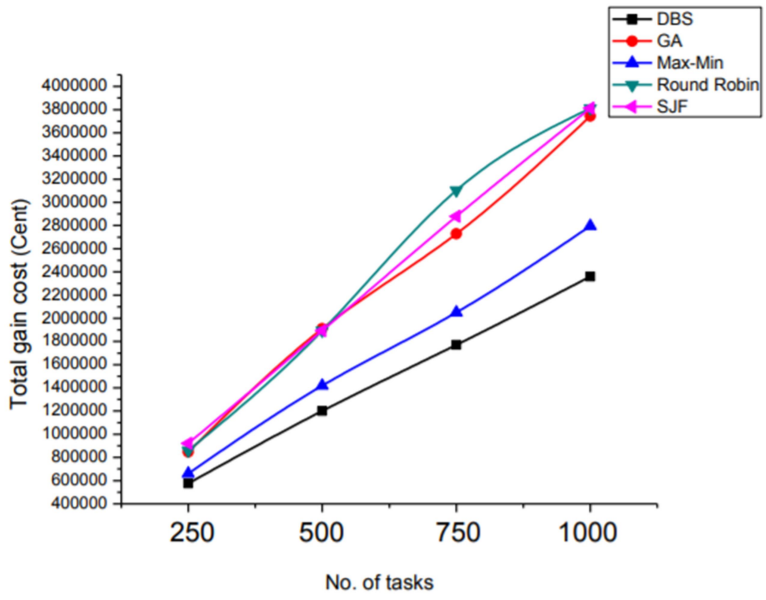


Figure 4.5: Total gain cost

Another indicator for evaluating the algorithm's performance is the number of violations. When the method obtains a lower number of violations, it implies that the algorithm is efficient. This is accomplished in our suggested DBS model, which achieves a lower number of violated tasks than the other existing algorithms shown in Table 4.10.

Table 4.10: Number of violations of DBS, GA, Max-Min, Round Robin and SJF algorithms

Experiments		**Number of violations**				
No. of tasks	**No. of VMs**	**DBS**	**GA**	**Max-Min**	**Round Robin**	**SJF**
250	5	10	29	19	28	28

500	7	18	63	42	58	52
750	9	45	96	59	84	97
1000	11	45	124	70	129	124

The amount of violations in our suggested DBS model is shown in Figure 4.6. When compared to GA, Max-Min, Round Robin, and SJF algorithms, the DBS model has less task violations due to deadline and budget limitations.

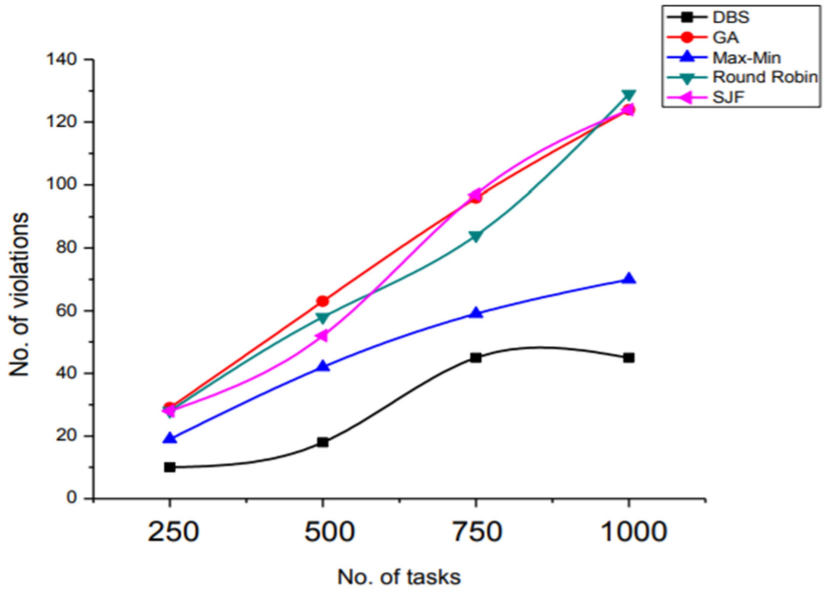


Figure 4.6: Number of violations

When comparing the provider profit for our proposed DBS model to existing scheduling algorithms, Table 4.11 summarizes the trials done on varied numbers of tasks hosted on varying numbers of VMs. The provider profit in the DBS model is always higher than the provider profit for the same tasks completed in other scheduling algorithms, as shown by the findings.

Table 4.11: Provider profit of DBS, GA, Max-Min, Round Robin and SJF algorithms

Experiments		Provider profit				
No. of tasks	**No. of VMs**	**DBS**	**GA**	**Max-Min**	**Round Robin**	**SJF**
250	5	1686740	1414874	1602697	1403249	1341766
500	7	3337891	2628381	3119848	2644751	2644497
750	9	4975104	4017073	4694679	3641805	3866607
1000	11	6646103	5261244	6208620	5193870	5194580

Figure 4.7 graphically depicts all of the differences between state-of-the-art scheduling methods and the proposed DBS model, demonstrating that the DBS model achieves a higher provider profit while executing tasks on VMs than other algorithms.

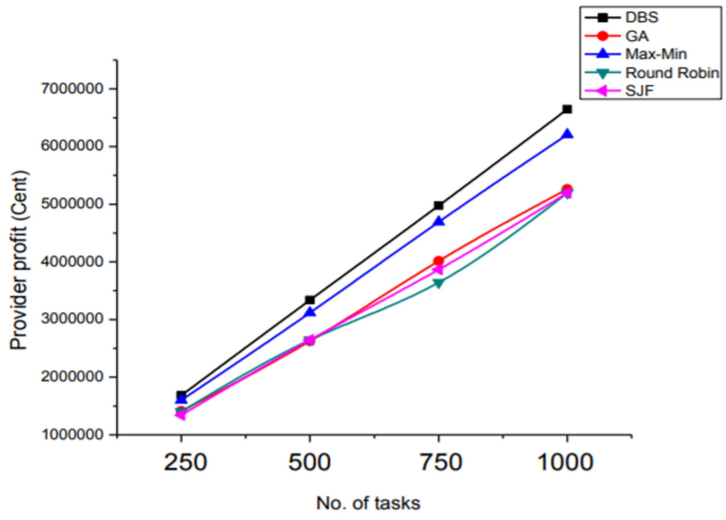


Figure 4.7: Provider profit

The task scheduling method should maximize resource usage to improve the performance of the cloud computing system. When compared to other algorithms, the results of the proposed DBS model in Table 4.12 show that our model is capable of optimal resource use based on deadline and budget limitations.

Table 4.12: Resource utilization of DBS, GA, Max-Min, Round Robin and SJF algorithms

Experiments		Resource utilization				
No. of tasks	**No. of VMs**	**DBS**	**GA**	**Max-Min**	**Round Robin**	**SJF**
250	5	1827	1698	1486	1705	1700
500	7	2927	2653	2250	2686	2714
750	9	3174	2946	2591	3019	2934
1000	11	3719	3452	3090	3403	3439

In comparison to GA, Max-Min, Round Robin, and SJF algorithms, Figure 4.8 shows that the suggested DBS model has the highest resource usage for all jobs based on user satisfaction.

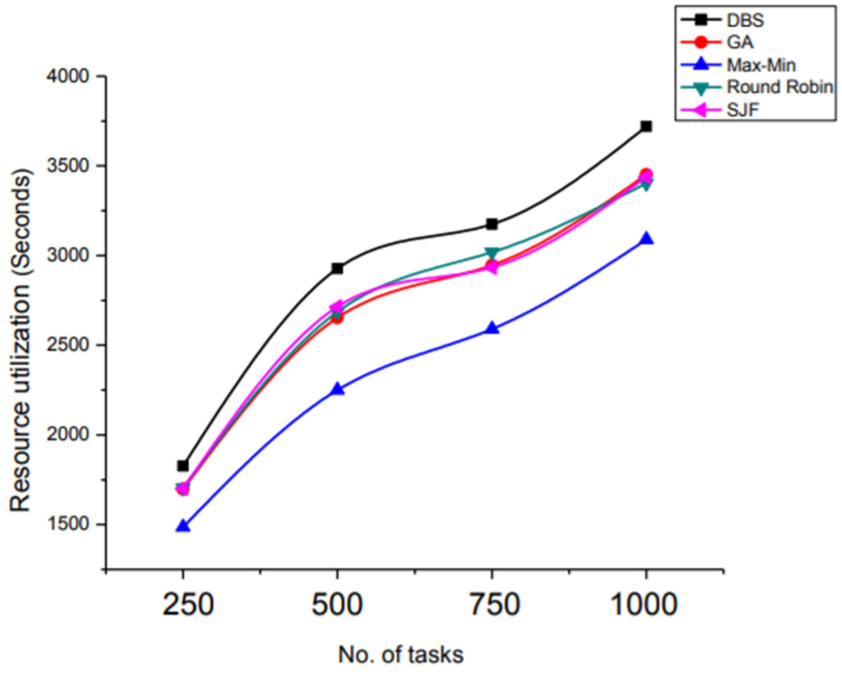


Figure 4.8: Comparison of resource utilization

Table 4.13 shows that our suggested DBS model outperforms GA, Max-Min, Round Robin, and SJF algorithms by 39 percent, 5 percent, 41 percent, and 41 percent, respectively, in terms of makespan ratio.

Table 4.13: Improvement in makespan ratio for DBS compared with GA, Max-Min, Round Robin, and SJF algorithms

Improvement in makespan ratio					
	DBS	**GA**	**Max-Min**	**Round Robin**	**SJF**
Experiments	1883	2922	1926	3114	3052
	2879	4909	3020	5015	4933
	3246	5362	3402	5374	5383
	3659	6166	3922	6325	6284
Sum of Expts.	11667	19359	12270	19828	19652
Improvement DBS vs algorithm		39%	5%	41%	41%
Sum of all makespan ratio		126			
Average		31.5 %			

Table 4.14 shows that our suggested DBS model outperforms GA, Max-Min, Round Robin, and SJF algorithms by 36 percent, 14 percent, 38 percent, and 38 percent, respectively, in terms of cost ratio.

Table 4.14: Improvement of cost ratio for DBS compared with GA, Max-Min, Round Robin, and SJF algorithms

Improvement of cost ratio					
	DBS	**GA**	**Max-Min**	**Round Robin**	**SJF**
Experiments	576158	848033	660214	859654	921143
	1200682	1910206	1418749	1893837	1894094
	1770457	2728512	2050902	3103795	2878976
	2359776	3744648	2797295	3811976	3811306
Sum of Expts.	5907073	9231399	6927160	9669262	9505519
Improvement DBS vs other algorithm		36%	14%	38%	38%
Sum of all cost ratio		126			
Average		31.5 %			

To perform statistical analysis on our suggested DBS model, we employed the T-test. The T-test is a useful function for calculating the mean of a DBS model and demonstrating the use of normality. The T-test results of the DBS model vs other algorithms such as GA, Max-Min, Round Robin, and SJF are tabulated in Table 4.15.

Table 4.15: T-test of DBS model compared to GA, Max-Min, Round Robin, and SJF algorithms

Metrics	**DBS model vs other algorithms**			
	GA	**Max-Min**	**Round Robin**	**SJF**
Average makespan	0.02	0.3	0.01	0.02
Total gain cost	0.1	0.3	0.1	0.1

Number of violations	0.03	0.1	0.04	0.04
Provider profit	0.2	0.4	0.2	0.2
Resource utilization	0.3	0.1	0.3	0.3

We've suggested a novel model in which the user specifies two constraints: deadline and budget:

- ✓ Tasks will be assigned to relevant VMs based on the constraints set by the user.
- ✓ User satisfaction is achieved by minimizing the process's completion time and cost by performing the tasks on VMs and adhering to the QoS requirements.
- ✓ Maximizing profit income and resource usage leads to provider satisfaction.

4.7 AUTO-SCALING TO MINIMIZE COST AND MEET APPLICATION DEADLINES IN CLOUD WORKFLOWS

The cloud application accepts jobs supplied by service customers on a continuous basis, and the workload is constantly changing. As a result, the auto-scaling mechanism must continue to monitor dynamic workload information as well as the progress of submitted jobs, and then respond quickly in terms of scaling and scheduling. This is a continuous process rather than a one-time event. As a result, the method employs a monitor-control loop. A scaling decision and a scheduling decision are made every time inside the loop based on the most recent updated information. Because cloud VMs are now invoiced by instance hours (rather than by the precise amount of time consumed), scaling and scheduling decisions should avoid wasting partial instance hours. Furthermore, unlike in a fixed-size resource environment, the auto-scaling method can acquire a VM instance and assign a task to it as long as there is unhandled workload. The scaling and scheduling decisions are made step by step in this dissertation, as indicated in the remainder of this section. Based on the workload,

it first calculates the number of VMs required for each VM type. It then determines if two or more existing virtual machines can be merged. Finally, it uses the Earliest Deadline First (EDF) algorithm to schedule jobs on the active VMs.

4.7.1 Preprocessing

Step 1- The auto-scaling system pre-analyzes the job classes and determines deadlines for each sub task using the following strategies to reduce runtime overhead and accelerate dynamic scaling plan development. Task bundling is the first step. Task bundling combines tasks that prefer the same instance type into a single task and requires them to run on the same instance. As a result, it can save data transfers by relying on locally stored temporary findings. One example of task bundling is shown in Figure 4.9. Because Task 6 and Task 8 are most cost-effective on computers with several CPUs, the auto-scaling system will treat them as a single task, Task 6'. Only jobs that prefer the same sort of instance and have one-to-one task dependencies are grouped together in this chapter.

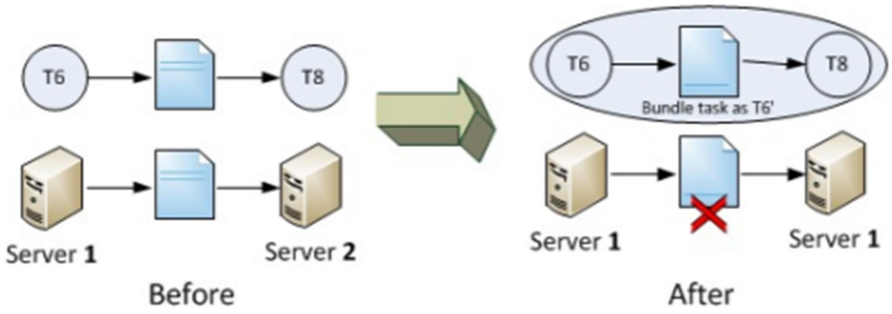


Figure 4.9: Task bundling

Step 2 – There is a deadline for this task. Jobs, not tasks, are related with deadlines. When a job is submitted, separate subtasks are given their own deadlines. If each task is completed by the deadline allocated to it, the job will be completed on deadline. Introduces and describes the concept of deadline assignment in a DAG. The shortest job makespan can be calculated by assuming that all tasks do not wait for resources, and then extending the task execution time by the ratio of deadline to job makespan. They allocate deadlines according to the quickest task execution time, and then look for the cheapest provider for each task. Individual deadlines are set in this dissertation based on task processing time on the most cost-effective equipment. The reason for

this is that the price of a cloud VM is not always proportional to its processing capacity, and a more expensive machine does not necessarily guarantee a speedier computer. If the initial deadline assignment fails to complete the project on time, this dissertation employs a heuristic proposed to find a feasible plan. The goal is to compute the job makespan for the initial deadline assignment, and then halt the process if the project can be completed inside the deadline. If not, it tries to schedule each task on a faster but more expensive machine and calculate the new job makespan to see which task upgrade cuts the makespan the most for the same money. In other words, it assigns a faster machine to the task with the higher cost-efficiency rank (equation 4.20). It repeats this process until the project is completed within the specified deadline frame. The auto-scaling technique breaks the task precedence limitations and treats each task independently throughout the deadline assignment process. One example of a deadline assignment is shown in Figure 4.10.

$$Cost\_Efficiency\_Rank = \frac{makespan_{before} - makespan_{after}}{cost_{after} - cost_{before}} \tag{4.20}$$

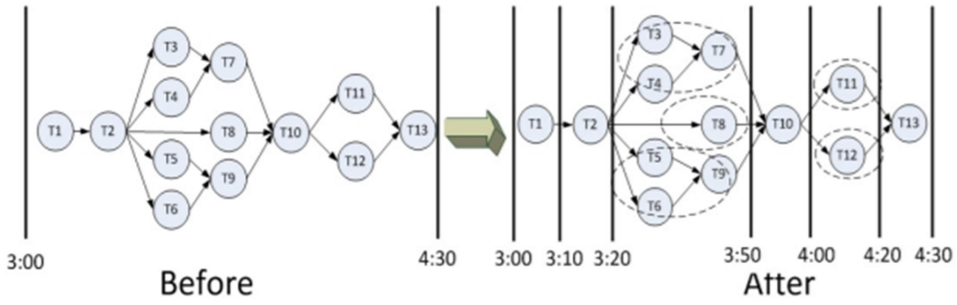


Figure 4.10: Deadline assignment

Other strategies can be utilized at this point to further customize the approach to the cloud features. Unlike in a utility computing environment, the scheduling algorithm can reserve a time slot and place the task on the service as long as services are available. Although there are unlimited resources in the cloud and a service provider can purchase an instance at any time, it is not always prudent to do so every time a task needs to be completed, especially if the task will use a major chunk of a

purchased instance hour. As a result, limiting task concurrency is a cost-effective strategy to enhance instance usage. This concept is depicted in Figure 4.11. If tasks T3, T4, and T5 are all less than one hour, three instance hours can be saved by processing them sequentially rather than in simultaneously. It combines parallel jobs using breadth-first search, and the search terminates when a task has to change its initially scheduled machine type to finish before the deadline. It stops looking because this method is exclusively for job-level optimization and should have no bearing on global scheduling decisions. Algorithm 1 depicts the total deadline assignment algorithm.

Figure 4.11: Parallelism reduction

Algorithm 1: Deadline assignment

Result: Schedule plan $S = task_i \rightarrow VM_v$

Generate the cheapest schedule $S = task_i \rightarrow VM_{cheapest}$;

while *true* **do**

  if $makespan(S) < deadline$ **then**

    return S;

  else

    for *each* $task_i$ **do**

      $S_i = S - (task_i \rightarrow VM_v) + (task_i \rightarrow nextFasterVM(task_i))$;

      $SpeedUp_i = (makespan(S) - makespan(S_i))/(cost(S_i) - cost(S))$;

    end

    $index = subscript(max(SpeedUp_i))$;

    $S = S_{index}$;

  end

end

4.7.2 Dynamic Scaling-consolidation-scheduling

The most up-to-date information is used to make dynamic scheduling and scaling decisions inside each monitor-control loop. The auto-scaling mechanism uses the earliest-deadline-first technique to recalculate task deadlines in order to determine the instance number, consolidate incomplete instance hours, and schedule jobs (EDF).

Step 3 - Scaling. This dissertation introduces a metric called load vector to determine the number of virtual machines. Load Vector (definition 7) Each task has its own load vector (LV). Following the deadline assignment, each task is assigned an execution interval [T0, T1], and the task execution time on V Mv is tv. As a result, the task's load vector LVv is defined as [tv/(T1 T0)], with T0 being the starting point and T1 being the end point. The vector denotes the number of machines required to complete the task on V Mv within the time interval. If the ratio is bigger than 1, the task cannot be completed in time since it cannot be further divided and executed in parallel. Assume, as shown in figure 4.12, that the execution interval for task T1 is between 3:00 and 4:00 p.m., and that it will take 15 minutes to complete on V M1. It signifies that the task must be completed between 3:00 and 4:00 p.m. on V M1. Another task, T2, must be completed between 3:15 and 3:45 p.m. and takes 15 minutes on V M1. In all, the auto-scaling system can process both T1 and T2 on a single instance. When defining the load vector, the finest granularity is 1 minute.

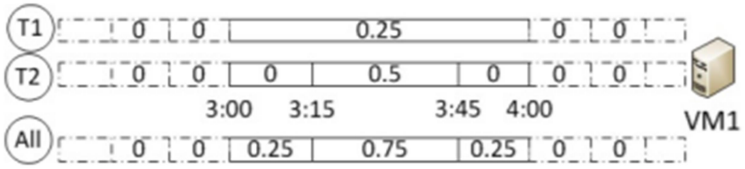


Figure 4.12: Load vector

The auto-scaling mechanism calculates and adds the load vectors for each task. Each VM type has its own load vector, and each task can only add to the load vector of the scheduled machine type. All tasks will be completed within the allotted execution interval if the auto-scaling system can ensure that the number of current machines is always larger than or equal to the load vector at any moment. Because cloud VMs may take some time to boot up, task load vectors are generated between the intervals [T0 + lagv,, T1] rather than [T0, T1] for scaling-up scenarios. The load vector is also

used to make scaling-down decisions. Every instance's acquisition time is known to the auto-scaling algorithm. When the number of instances exceeds the load vector and one instance is approaching full hour operation, it might be shut down. Another aspect that may influence the decision to shut down an instance is VM churn. If the instance acquisition lag cannot be precisely calculated, too many VM acquisitions/releases will damage the auto-scaling mechanism's performance. In the assessment section, the impacts of erroneous parameter estimation are shown. Note that deadline assignments must be adjusted on a regular basis because certain tasks may be completed before their initial deadlines, and reassignment may allow later jobs to execute on less expensive machines.

Consolidation of instances is the fourth step. If all jobs can be completed on the most cost-effective instances and all instances are completely utilized, that is ideal. When considering the arrival timings and execution times of the jobs, it is not always possible to ensure that no partial instance hours are lost. Consolidating partial instance hours can assist customers lower overall cloud costs, thus it's often necessary to conduct jobs on non-cost-effective machines to consolidate partial instance hours. Instance consolidation is the term for this procedure. This is seen in Figure 4.13. T11 and T12 were initially planned on a high-CPU and a normal CPU instance, respectively. Because both jobs only require a fraction of an instance hour and no other processes are using the instances at the same time, consolidating the two tasks on the same standard machine and saving one high-CPU instance hour is a prudent scheduling option (although T11 runs slower and costs more on a standard machine). A "consolidated task" must, of course, be completed by the original deadline. Algorithm 2 describes the procedure.

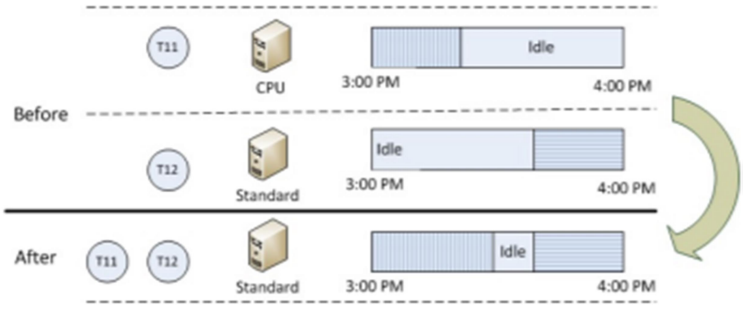


Figure 4.13: Instance consolidation

Algorithm 2: Instance consolidation

Data: LoadVector LV_v and exiting instance number N_v for each VM type VM_v

Result: Updated LoadVector LV_v after instance consolidation

for *each* VM_v *whose* $LV_v > N_v$ **do**

  for *each* VM_n *whose* $LV_n <= N_n$ **do**

    if *InstanceNum*(LV_n + LV_n(*TopTask*[VM_v])) $\leq N_n$ *AND tasks following TopTask*[VM_v] *do not change scheduled VM* **then**

      LV_v - LV_v(TopTask[VM_v]);

      LV_n - LV_n(TopTask[VM_v]);

      schedule TopTask[VM_v] to VM_n instances;

    end

  end

end

Step 5 - Scheduling that is flexible. After identifying the number of instances for each VM type, the auto-scaling system schedules tasks on each VM type using the Earliest Deadline First (EDF) algorithm. Each task is assigned to a VM type when the deadline is set and the instances are consolidated. For each VM type, tasks are classified according to their deadlines, and the task with the earliest deadline is scheduled when an instance becomes available. The task facing deadline misses can be detected in time using dynamic scaling, and the auto-scaling system can quickly acquire instances to complete the task. In other words, dynamic scaling assures that the load vector for each instance type is always less than 1 (equation 4.21). In such instances, EDF is known to be the best scheduling approach. As a result, all of the tasks, as well as the entire workflow project, will be completed ahead of schedule. The overall auto-scaling solution is described in Algorithm 3.

$$\sum_i \frac{t_i}{T_{end\_i} - T_{start\_i}} < 1 \qquad (4.21)$$

Algorithm 3: Auto-scaling

```
while true do
    update LV_v for each task t_i;
    get existing N_v for each VM type VM_v;
    LV_v ← InstanceConsolidation(LV_v , N_v);
    for each VM_v where NumNeeded(LV_v) > N_v do
        acquire(VM_v , NumNeeded(LV_v) − N_v);
    end
    for each VM_v where NumNeeded(LV_v) < N_v do
        for each instance I_i of type VM_v do
            if I_i approaches full hour operation  AND  NumShutdown < N_v − NumNeeded(LV_v)
            then
                shutdown(I_i);
                NumShutdown + +;
            end
        end
    end
    schedule the tasks based on EDF;
    wait for the next monitor interval;
end
```

4.8 EVALUATION

The provided auto-scaling technique is evaluated using three types of applications and four workload patterns in this dissertation. Simulating the input parameters and locating the key factors in the auto-scaling mechanism aids in controlling the input parameters and locating the key factors in the auto-scaling mechanism. It also aids in the analysis of the relationship between the performance of the mechanism and the budget restrictions. It also reduces the cost of evaluation by speeding up the procedure. The first part of this dissertation analyses two baseline techniques in terms of cost and instance utilization. It then considers the impact of workload volume and the mechanism's capacity to deal with erroneous input parameters (e.g. estimated task running time and the instance acquisition lag). Finally, it calculates the overhead of the mechanism.

4.8.1 Application, Workload, and Virtual Machine

Pipeline, Parallel, and Hybrid are the three types of representative applications. Pipeline applications are simple multi-stage applications in which tasks must be processed one at a time while adhering to strict precedence rules. Because there are few precedence constraints, parallel programs have a large degree of potential concurrency. Hybrid applications combine the benefits of both pipeline and parallel applications. Hybrid programs might have a lot of task dependencies. The pipeline application, parallel application, and hybrid application used in the evaluation are depicted in Figure 4.14 is where they came from.

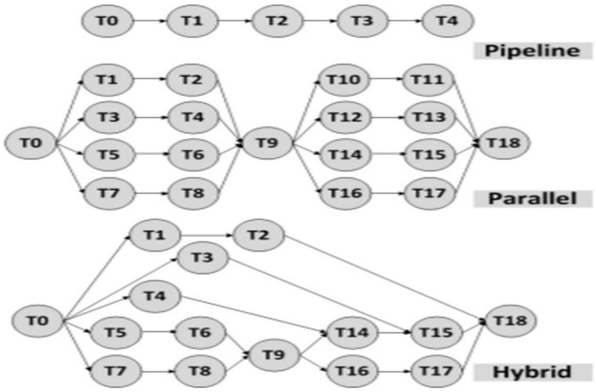


Figure 4.14: Application models

Stable, Growing, Cycle/Bursting, and On-and-Off are the four typical workload patterns in the cloud (figure 4.15). Each workload is an example of a common application or scenario. The Growing Workload pattern, for example, depicts a scenario in which a story or video gets famous overnight and attracts an increasing number of users to press the button. The workload is rapidly increasing. The workload pattern of an online store is represented by the Cyclic/Bursting workload. The daytime has a higher burden than the nighttime, and holiday shopping seasons may see more traffic. The On-and-Off workload pattern represents work that is processed on a regular or irregular basis, such as batch processing and data analysis in a research department that is done daily or weekly. These programs have a short active duration, after which the service can be turned off or kept at the lowest level of service. The task execution time on different types of VMs is randomly generated in the evaluation (the task execution time and distribution are produced based on), and 200

permutations for each workload pattern are generated. All of the test results showed a similar level of performance. As a result, there is no information about the task's completion time provided here. This dissertation mimics a 72-hour timeframe for each test with four different deadlines - 0.5-hour, 1 hour, 1.5 hour, and 2 hour - for each application and workload pattern. During the observation time, the total cost is recorded.

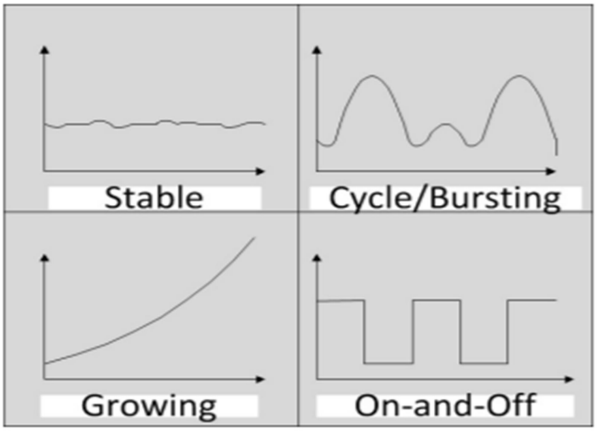


Figure 4.15: Workload patterns

This dissertation replicates four types of VMs: Micro, Standard, high-CPU, and high-I/O instances, in addition to the three application models and four workload patterns. These VMs' costs (table 4.16) are taken from Amazon EC2.

Table 4.16: VM types and prices

VM Type	Price
Micro	$0.02/hour
Standard	$0.085/hour
High-CPU	$0.68/hour
High-I/O	$0.50/hour

4.8.2 Cost and Resource Utilization

It is difficult to construct a benchmark against which this dissertation may be measured. Previous research has focused on either a batch-queue model (which ignores task relationships) or a single workflow instance (not a stream of submitted jobs). Furthermore, cost-effectiveness (deadline and cost) is not always a top priority. The rule-based trigger system is a good starting point. However, determining the scaling indicators and thresholds using this dissertation is really difficult. The trigger mechanism, as discussed in Chapter 2, does not truly solve the performance-resource mapping dilemma, and any rules chosen in this dissertation run the risk of creating an "unfair game."

As a result, the baseline for this dissertation is to extend two current techniques, Greedy and GAIN. This section compares the Scaling Consolidation-Scheduling (SCS) strategy from the preceding part with these two alternatives. These two methodologies were created for the cost-conscious execution of a single workflow in a utility computing environment. This dissertation expands on these two ideas to support continuous workflow submissions and to make them aware of the cloud's instance hour charging model. To begin, they treat each work individually using their original algorithms. They set aside instances for each job and add more when the number of active instances becomes insufficient. Second, acquired instances can only be released once they have reached full hour operation and no longer have any duties that require them. As a result, all job requests are fulfilled on time, and the number of unused partial instance hours is decreased. Note that the original GAIN method begins with the lowest option and improves iteratively based on cost-efficiency rank until the budget cap is achieved. The break condition is changed in this dissertation such that the predicted job completion time is shorter than the deadline. The Greedy algorithm always tries to discover the cheapest instance first, out of all the live instances. If there aren't enough instances, it chooses the cheapest one from among the possible instance kinds. Figures 4.16, 4.17, and 4.18 demonstrate the results of the tests with various workload patterns. This dissertation displays the overall running cost and average instance utilization for each application model and deadline. It is assumed that the task running duration and instance acquisition lag (6.5 minutes) can be reliably approximated in all circumstances. As a result, SCS, Greedy, and GAIN can all complete jobs ahead of schedule. When compared to the Greedy and GAIN

techniques, SCS incurs the least cost and has the highest instance utilization (greater utilization indicates fewer idle instance hours), as demonstrated in the data. The cost savings vary from 9.8% to 40.4 percent. Because all jobs are required to run on their fastest computers to finish the project within the deadline, these three techniques tend to perform similarly when the deadline is short.

To put it another way, all three systems provide extremely similar scheduling/scaling plans, leaving little room for cost-cutting optimization. When the deadline is longer (the "scheduling slack time"), SCS saves the most money, Greedy performs the poorest, and GAIN performs somewhere in the middle Instead of evaluating cost-efficiency for each task, the Greedy strategy always chooses the cheapest machine. In this situation, running numerous jobs on the cheapest equipment really costs more than running them on their preferred machines. As a result, the overall cost is significant. This result demonstrates the necessity of selecting appropriate VM types for various workloads. The GAIN strategy, on the other hand, always schedules tasks at their most cost-effective times. In this way, it accomplishes cost optimization for each work and saves more money than the Greedy technique, but it has a lower instance utilization rate, implying that more partial instance hours are lost. Through instance consolidation, the SCS strategy not only takes advantage of task-level cost-efficiency, but also makes better use of incomplete instance hours. As a result, it saves more money and has a higher instance utilization rate than the GAIN technique. The Growing example has the highest utilization of the four workload patterns, since as more and more jobs are submitted swiftly, all instances are loaded with tasks, and partial instance hours grow fewer and fewer. In general, if the workload volume is big enough, task-level cost-efficiency may outweigh overall cost. As a result, the Greedy strategy is always defeated by the SCS and GAIN approaches. When the deadline is longer, the cost-saving benefits become more apparent, because the Greedy strategy will assign the majority of work to the cheapest machines, incurring higher costs than shorter deadlines.

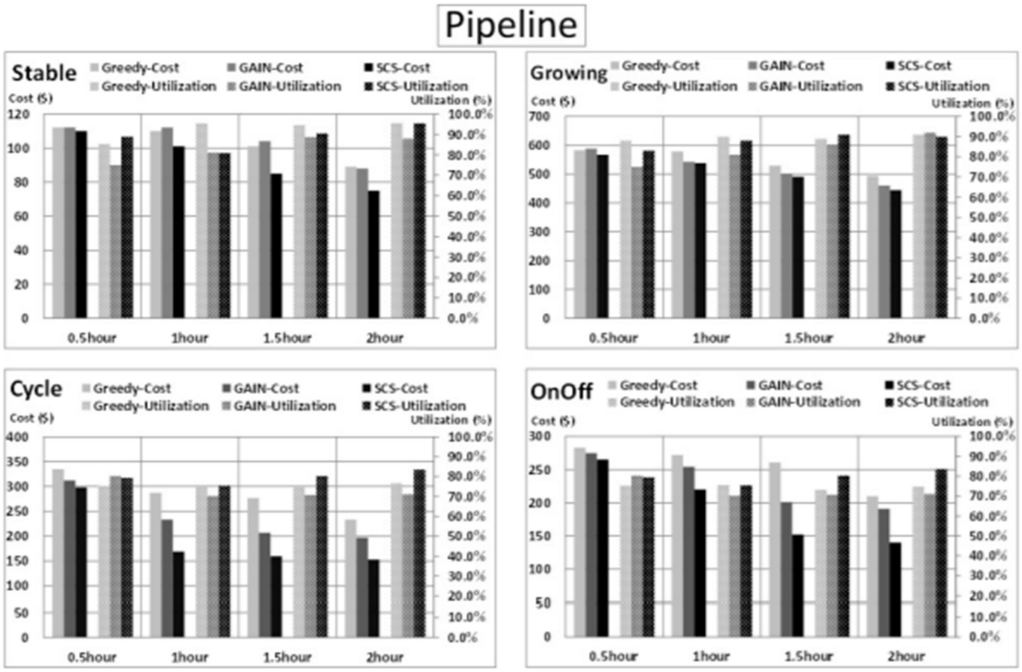


Figure 4.16: The performance for pipeline applications

4.8.3 Heavy Workload vs Light Workload

In extreme instances, such as a single workflow instance, the task volume can be quite minimal. In such scenarios, the idea of assigning tasks to the most cost-effective instances may not always save money, because instances are not always fully utilized, and there may be a significant amount of unutilized instance hours. In other words, the cost-effectiveness of individual tasks is outweighed by the advantage of instance consolidation. When the workload volume is high enough to occupy all of the instances, however, cost-effective task placement also implies global cost-effectiveness. This is due to the fact that all of the jobs can be completed at a low cost, and there are relatively few idle instances. The evaluation results of the pipeline application are shown in this dissertation to demonstrate this argument.

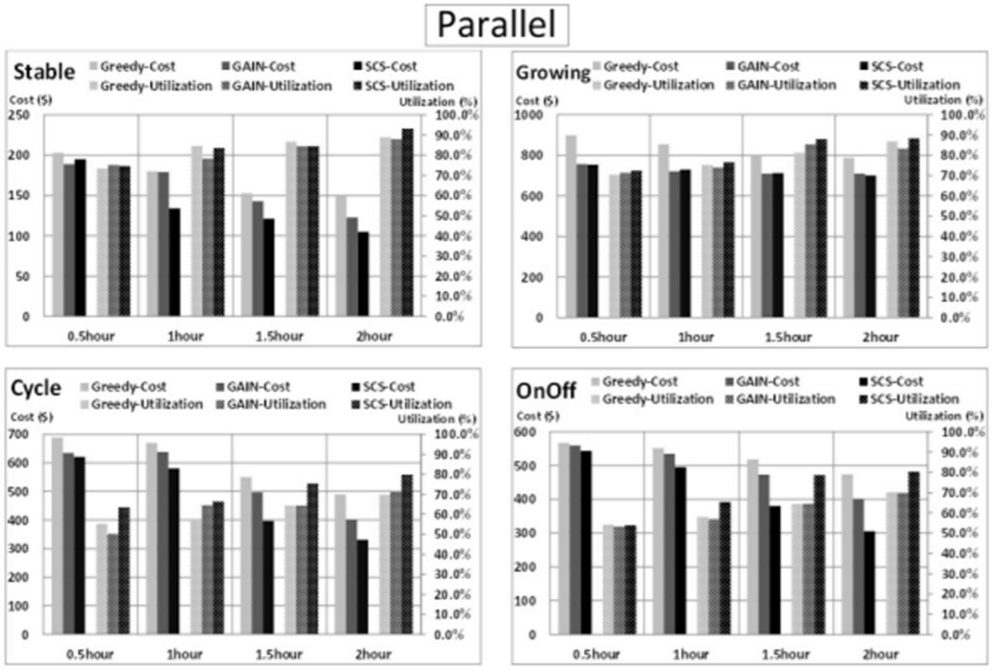
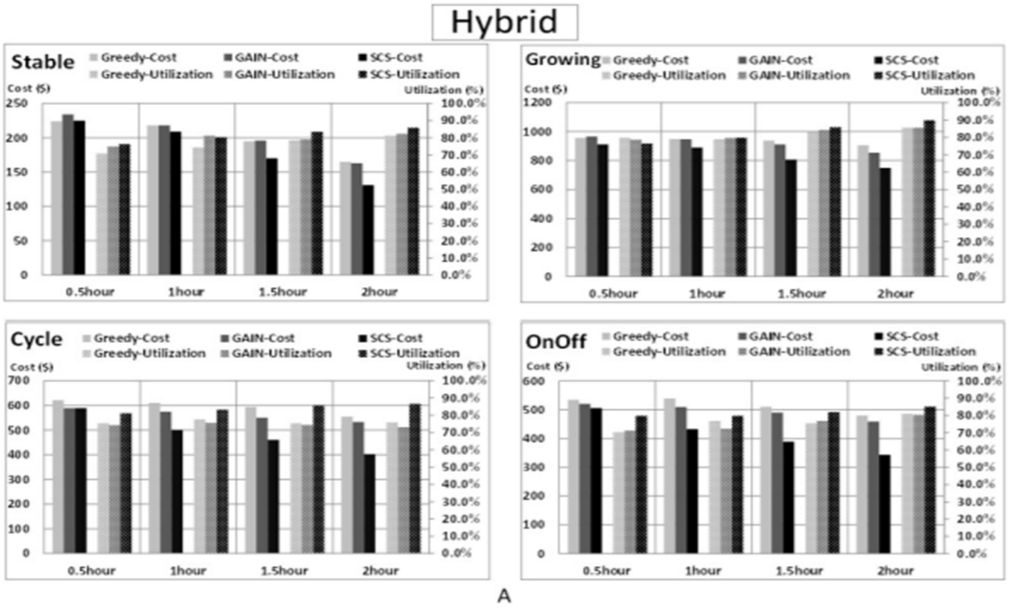


Figure 4.17: The performance for parallel applications



A

Figure 4.18: The performance for hybrid applications

Using a one-hour deadline, with both low (X) and high (10X) workload volumes. Figure 4.19 shows that when the workload volume is low, the Greedy technique outperforms GAIN, but when the workload volume is high, it outperforms GAIN.

This is because it schedules as many tasks as possible on the cheapest machines, which is an example of consolidation technique. In both low and high-volume workload environments, the SCS technique outperforms the Greedy and GAIN approaches. When the workload volume grows high, it takes advantage of task level cost-efficiency and tackles low volume cases through instance consolidation. Because parallel and hybrid applications perform similarly, their outcomes are not explored here.

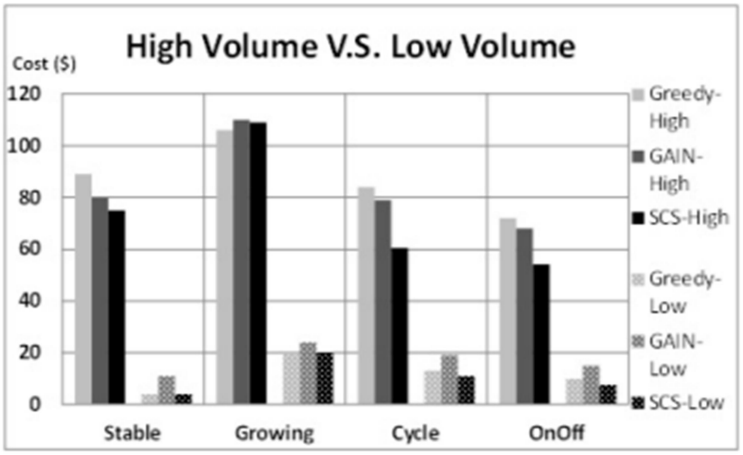


Figure 4.19: Heavy workload and light workload

4.8.4 Sensitivity to Inaccurate Parameters

The task execution time and the instance acquisition lag can be estimated in this dissertation. Under these conditions, it is demonstrated that the dynamic scaling and EDF scheduling technique can meet all deadlines. In practice, however, reliable task execution time estimates are not always available, and the latency in instance acquisition is out of the user's control. The ability of the auto-scaling system to handle erroneous input parameters is assessed in this section. It first allows the real task running time to be calculated with a 20% inaccuracy and then checks the pipeline application's deadline non-miss rate with a 0.5-hour deadline (figure 4.20). The dynamic scaling characteristic of SCS copes admirably with erroneous task runtime estimation. It can accomplish more than 90% of jobs on time, which is significantly better than the other two methods. In reality, because customers can set longer dates, SCS can achieve a higher non-miss rate. Following that, this dissertation allows for a 20% inaccuracy in the expected instance acquisition lag and examines the pipeline

application's deadline non-miss rate using a one-hour deadline (figure 4.21). Because the auto-scaling mechanism reacts to dynamic changes through instance acquisition, which is the core function of the auto-scaling mechanism, the instance acquisition lag can have a greater impact on the mechanism's performance than the task running time estimation (deadline non-miss rate is below 80%). In the most severe scenarios, where the acquired instance is never ready to handle user tasks, all jobs will be incomplete and the deadline would be missed.

The user's performance requirement is another consideration to consider when determining whether or not the instance acquisition lag is acceptable. When the requirement is no deadline misses, a 1-minute or 10-minute lag makes no difference if the user has an extremely tight deadline and wants to acquire additional instances right away. Even if the instance acquisition latency is 30 minutes, if the deadline is 2 hours for a brief job, the job can still be completed on time. The Growing workload pattern has the worst performance of all the workload patterns. This is due to the fact that it gets more instances than the other test cases, and hence is more affected by improper lag estimation. This test case demonstrates that the instance acquisition lag has a significant impact on an auto-scaling mechanism's performance. To plan for instance acquisitions ahead of time, workload prediction techniques are required. It's also useful to reduce the number of operations required for instance acquisition and release (for example, parallelism reduction).

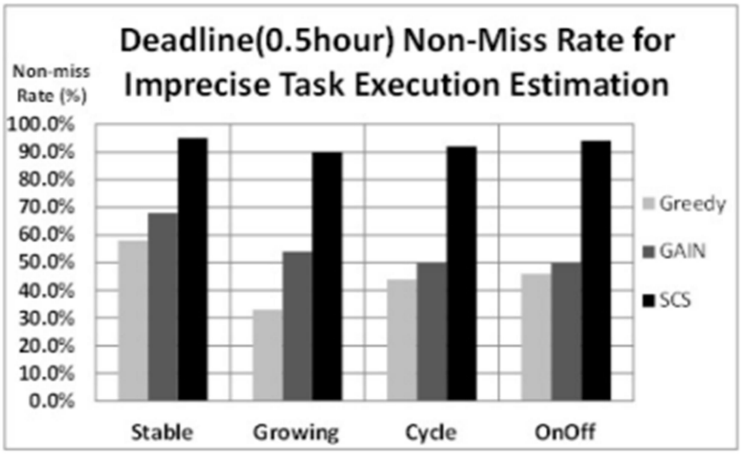


Figure 4.20: Inaccurate task execution time

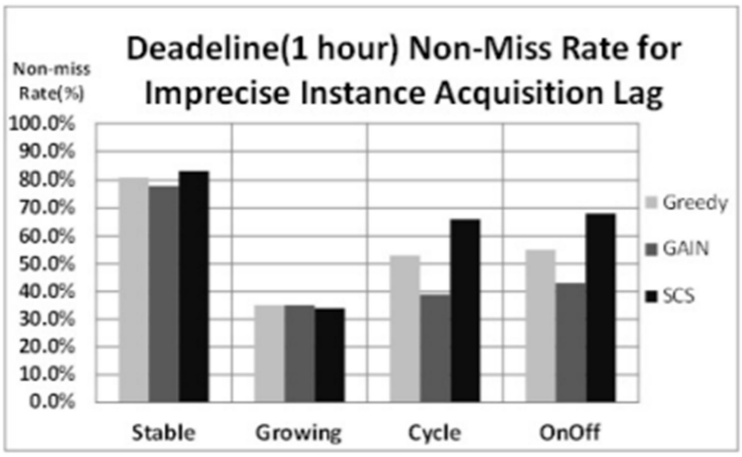


Figure 4.21: Inaccurate instance acquisition lag

4.8.5 Mechanism Overhead

Finally, the mechanism overhead is assessed in this dissertation. It ignores the overhead of monitoring and updating runtime information, such as the number of freshly submitted jobs and the status of running processes, in this test. In practice, cloud providers such as Windows Azure Diagnostics and AWS Cloud Watch [90] may provide such information monitoring functionality, or the application itself may incorporate it. As a result, the overhead is very dependent on how it is implemented. Instead, then focusing on the entire monitor-control loop, this dissertation just looks at the performance of the basic scheduling and scaling mechanism. The test is performed on a desktop computer with an Intel P4 2.4GHz processor, 4GB of memory, and 512GB of storage. It calculates the time it takes to update load vectors, consolidate partial instance hours, and make scaling/scheduling decisions for various job sizes (from 10 to 100000). The test employs 100 hybrid job classes and 16 virtual machine kinds. The overhead is negligible, and the performance scales linearly with the work number, as illustrated in figure 4.22. The following two strategies are used to produce a reduced mechanism overhead. Preprocessing can handle the most time-consuming part, deadline assignment, and the result can be cached after the first computation. Later jobs in the same class do not need to recalculate deadline assignments on a regular basis. Second, instead of requiring a large array for each task, the other time-consuming part, updating the Load Vector, can be implemented using pair-wise data structures, which reduces both memory and computation time.

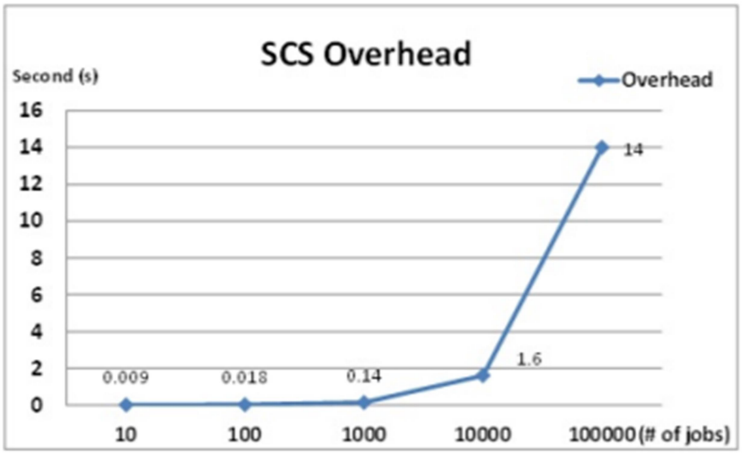


Figure 4.22: SCS overhead

It is not the same as the batch-queue model described in the preceding chapter. Instead of bags of chores, the workload is in the form of workflows. It enables the service provider to complete projects in a cost-effective manner before the user-specified timeframes. The system is built around a monitor-control loop that can respond to dynamic changes like workload bursting and delayed instance acquisitions. The results of the evaluation suggest that it can assist in cost reduction for a variety of application architectures and workload patterns. When compared to the two baseline procedures, cost savings range from 9.8% to 40.4 percent. Not only does the instance consolidation procedure increase instance utilization, but it also lowers partial instance hour waste. It effectively manages both high and low workload volumes, and it analyses both job-level and global-level cost-efficiency. Furthermore, the system has a high tolerance for incorrect settings. The monitor-control loop can assist with task execution time and instance acquisition latency assumptions that are erroneous. It reacts quickly to dynamic changes. The overhead of the approach is also significantly reduced because to preprocessing and caching. This paper was presented at the 2011 international conference on high-performance computing, networking, storage, and analysis (SC 2011). In the following chapter, this dissertation looks at the flip side of the optimization problem: minimizing task turnaround time while staying within budget limitations.

In conclusion, we can claim that this contribution will address the third question: How can we create a task scheduling model that will execute tasks while meeting user-

defined constraints? We suggested a Deadline Budget Scheduling (DBS) model for scheduling jobs in a heterogeneous cloud environment with two competing QoS requirements: time and cost, all while maintaining user happiness. The suggested DBS model's most important KPIs are to reduce monetary expenses while staying within the user-defined budget and to minimize the makespan under a user-defined deadline. In numerous different conditions, such as limited resources or high resources ability, variable number of tasks and VMs, simulation results show that the proposed DBS model achieves greater performance in decreasing the makespan and cost when compared to state-of-the-art techniques. The number of violations is minimized in the DBS model to meet user requirements while maximizing the provider's profit and resource utilization. As a result, our model outperforms state-of-the-art algorithms such as GA, Max-Min, Round Robin, and SJF. The improvement in makespan ratio for our proposed DBS model over GA, Max-Min, Round Robin, and SJF algorithms is 39 percent, 5 percent, 41 percent, and 41 percent, respectively, as well as a 36 percent, 14 percent, 38 percent, and 38 percent improvement in cost ratio for our proposed DBS model over GA, Max-Min, Round Robin, and SJF algorithms. In addition, the T-test was used to compare our suggested DBS model to other algorithms such as GA, Max-Min, Round Robin, and SJF statistically.

4.8.6 Proposed Additions in Thesis

- **Alternative approaches or variations that could offer additional insights or improvements**

Deadline-Aware Priority Scheduling

In the cloud, where users and applications share resources, different versions and extensions of time-sensitive priority scheduling can provide additional insights and enhancements to optimize resource usage and meet deadlines efficiently. Below are some variations unique to cloud:

1.Cost-Aware Deadline Scheduling: Integrate cost into deadline-aware planning so that you can allocate resources based on both the time of the task and the cost of the resource. This way, you can ensure that your high-priority tasks are completed on time while reducing resource costs, which is in line with your budget.

2.Multi-Tenant Scheduling: Design scheduling algorithms that take into account the fact that cloud environments have multiple tenants, where users and applications share resources. By taking into account the different workloads and priorities of tenants, a scheduling algorithm can optimize resource allocation and accommodate diverse deadline requirements.

3.Service-Level Agreement (SLA): Develop scheduling strategies that give priority to tasks according to their SLA commitments and the penalties incurred for failing to meet deadlines. By taking into account the SLA requirements and associated penalties, the scheduling algorithm can efficiently assign resources to tasks, ensuring maximum adherence to SLA agreements while minimizing expenses.

4.Data Locality-Aware Scheduling: Consider data locality when scheduling tasks in cloud environments with distributed data storage. By placing tasks near the data they need, this method can lower data transfer expenses and enhance system performance, especially for workloads that heavily rely on data and have tight deadlines.

5.Hybrid Cloud Scheduling: Enhance deadline-aware scheduling in hybrid cloud environments that combine resources from both public and private clouds. By taking

into account the unique characteristics and costs of resources from different clouds, the scheduling algorithm can optimize the allocation of resources and meet deadlines while minimizing costs across hybrid environments.

6.QoS-Driven Scheduling: Prioritize tasks based on their quality-of-service requirements, such as response time, throughput, or availability. By considering QoS metrics alongside deadlines, the scheduling algorithm can ensure that tasks meet performance objectives while also meeting their deadlines within budget constraints.

7.Dynamic Pricing and Resource Negotiation: Integrate dynamic pricing mechanisms and resource negotiation protocols into the scheduling process to optimize resource allocation and efficiently meet deadlines. By leveraging real-time pricing information and negotiation strategies, the scheduling algorithm can dynamically adjust resource allocations to meet deadline requirements while minimizing costs.

8.Elastic Scheduling and Auto-scaling: Implement elastic scheduling and auto-scaling mechanisms that dynamically adjust resource allocations based on workload fluctuations and deadline requirements. By scaling resources up or down in response to changing demand, this approach ensures the timely completion of tasks while optimizing resource utilization and minimizing costs.

9.Geographical and Regulatory Constraints: Take into consideration geographical and regulatory constraints when scheduling tasks in distributed cloud environments that span multiple regions or jurisdictions. By adhering to legal and regulatory requirements, as well as optimizing resource allocation based on geographical proximity, the scheduling algorithm can ensure compliance and optimize performance.

10.Feedback-Driven Optimization: Incorporate feedback mechanisms to collect performance data and user feedback, which can be utilized to continuously optimize scheduling decisions.

By investigating these different possibilities and expansions, deadline-conscious priority scheduling in cloud computing can be customized to meet specific application

needs, enhance the efficiency of resource allocation, and guarantee the punctual accomplishment of tasks while minimizing expenses and fulfilling SLA obligations.

Budget-Aware Scheduling

In the realm of cloud computing, effective resource provisioning and allocation play a critical role in cost management and meeting performance standards. Various strategies have been devised to address budget-conscious scheduling, all with the goal of enhancing resource utilization while ensuring compliance with service-level agreements (SLAs) and financial limitations. Here are a few of these strategies:

1.Adaptive Scheduling: Adaptive scheduling dynamically modifies resource allocations in response to workload fluctuations and budget constraints. It adjusts resource levels up or down as needed to accommodate changes in demand, thereby preventing budget overruns while upholding performance and availability.

2.Utilization of Spot Instances: Spot instances refer to unused cloud resources available at a reduced cost. Leveraging spot instances opportunistically helps minimize expenses while meeting workload demands. Tasks are assigned to spot instances when they fall within budget constraints.

3.Time-Conscious Scheduling: Time-conscious scheduling prioritizes tasks based on their deadlines and allocates resources accordingly to ensure timely completion without exceeding budget limits. This approach optimizes resource usage by adjusting priorities and allocations dynamically based on task deadlines and budget availability.

4.Serverless Architecture: Serverless architecture abstracts infrastructure management from users, enabling them to concentrate on application development and execution without the burden of resource provisioning. In serverless environments, users are charged based on resource consumption and execution duration, facilitating budget-conscious scheduling through automated resource scaling in response to workload requirements and budget restrictions.

5.Strategies for Cost-Effective Task Placement: The objective of cost-aware task placement strategies is to minimize expenses by strategically assigning tasks to cloud resources that offer the best balance between cost and performance. These strategies

take into account various factors, including resource prices, performance characteristics, and task requirements, in order to optimize resource allocations within the given budget constraints.

6.Enhancing Resource Utilization through Workload Consolidation: Workload consolidation involves the consolidation of multiple tasks or workloads onto a smaller number of resources, resulting in improved resource utilization and reduced costs. By dynamically adjusting resource allocations and consolidating workloads based on workload characteristics and budget limitations, this approach optimizes the utilization of resources while still meeting performance requirements.

7.Efficient Resource Scaling with Budget Considerations: Budget-aware autoscaling mechanisms dynamically adapt resource allocations based on the demand of the workload and budget constraints. This ensures that resources are scaled up or down in a cost-effective manner, maximizing resource utilization while staying within the defined budget limits.

8.Anticipating Resource Needs with Predictive Resource Allocation: Predictive resource allocation utilizes predictive analytics and machine learning algorithms to forecast future workload demands and resource requirements. By anticipating resource needs, this approach enables proactive resource allocation decisions that optimize resource usage and effectively utilize the allocated budget.

9.Coordinated Resource Allocation with Federated Cloud Scheduling: Federated cloud scheduling coordinates resource allocation and scheduling across multiple cloud providers or data centers. By leveraging resources from multiple providers based on pricing and availability, federated cloud scheduling optimizes resource usage and cost-effectiveness while ensuring workload and budget constraints are met.

These alternative methods for budget-aware scheduling in cloud computing present diverse strategies to optimize resource utilization, reduce costs, and meet performance requirements within budget constraints. The selection of a particular approach depends on factors such as workload characteristics, budget considerations, and the specific demands of the cloud environment.

Deadline budget scheduling

Deadline budget scheduling in cloud computing involves the management of tasks based on their deadlines and allocated budgets. There are several variations and extensions that can provide additional insights and improvements in this area:

1.Dynamic Budget Adjustment: Algorithms can be developed to dynamically adjust task budgets based on fluctuations in workload, resource availability, and deadline requirements. This adaptive approach ensures that tasks have enough budget allocations to meet their deadlines while optimizing the utilization of resources.

2.Budget-Aware Elastic Scaling: Elastic scaling mechanisms can be integrated with budget-awareness to dynamically adjust resource allocations based on budget constraints. This ensures that resource provisioning aligns with budget allocations and deadline requirements, maximizing cost-effectiveness and meeting deadlines.

3.Budget-Based Resource Reservation: Resource reservation mechanisms can be implemented based on task budgets to ensure that tasks have access to the necessary resources within their allocated budgets. By reserving resources in advance, this approach guarantees that tasks can meet their deadlines without exceeding budget constraints.

4.Cost-Optimized Deadline Scheduling: Deadline-aware scheduling can be combined with cost optimization techniques to schedule tasks in a way that minimizes costs while meeting deadlines. This involves considering both task deadlines and budget constraints when making scheduling decisions to achieve cost-effectiveness.

5.Budget-Driven Task Migration: Algorithms can be developed for migrating tasks between different cloud resources based on their budget allocations and deadline requirements. By migrating tasks to resources with lower costs or better performance characteristics, this approach optimizes resource utilization while meeting deadlines within budget constraints.

6.Budget-Aware QoS Differentiation: By incorporating task budgets into the prioritization and resource allocation process, the QoS differentiation mechanisms can be extended. This approach guarantees that tasks with higher budgets are given

priority, allowing for optimal resource allocation and ensuring that deadlines are met while maximizing the return on investment.

7.Budget-Optimized Multi-Objective Scheduling: Discover multi-objective scheduling algorithms that enhance resource allocation across various dimensions, such as task deadlines, budgets, and performance goals. Through effectively balancing these conflicting objectives, the scheduling algorithm is able to make well-informed choices that optimize cost-effectiveness and ensure timely completion.

8.Budget-Conscious Workload Consolidation: Implement workload consolidation techniques that take into account task budgets when consolidating workloads onto a reduced number of resources. By optimizing the utilization of resources according to budget allocations, this strategy effectively reduces costs while ensuring that deadline requirements are met.

9.Budget-Aware Fault Tolerance: Improve fault tolerance mechanisms by incorporating budget considerations to guarantee task completion despite failures. By setting aside funds for fault tolerance strategies, this method enhances dependability without exceeding financial limits.

By exploring these variations and extensions, deadline budget scheduling in cloud computing can be enhanced to optimize resource utilization, minimize costs, and ensure timely completion of tasks while meeting budget constraints.

- ## Potential Challenges or methods for assisting users in Specifying Constraints

It is of utmost importance to effectively specify constraints in a DAPS, BAS, and DBS model in order to achieve the desired performance outcomes. Nevertheless, users might encounter difficulties in accurately defining these constraints due to various factors, including complexity, uncertainty, and diverse requirements. Presented below are several potential challenges and approaches to aid users in effectively specifying constraints.

1.Understanding Requirements: Users might not have a complete grasp of their application's requirements or the consequences of different constraints on scheduling results. By offering educational materials, tutorials, and documentation, users can gain a better understanding of the significance of various constraints and make well-informed choices.

2.Complexity of Constraints: The complexity of DAPS, BAS, and DBS lies in the various parameters it encompasses, including task deadlines, priorities, resource requirements, and dependencies. By providing user-friendly interfaces and visualization tools, the process can be simplified as users can easily specify constraints in a clear and comprehensible manner.

3.Uncertainty and Variability: Variations in task characteristics and workload conditions can result in uncertainty when specifying constraints. By implementing adaptive algorithms that can adjust constraints in real-time based on feedback and workload changes, it is possible to mitigate uncertainty and enhance scheduling efficiency.

4.Trade-offs and Prioritization: Users might require help in comprehending the compromises associated with establishing various limitations and arranging them according to their significance. Offering guidance via interactive decision support systems or recommendation engines can aid users in making well-informed choices regarding constraint specification.

5.Automated Constraint Inference: Users can effectively specify constraints by implementing algorithms that automatically infer constraints based on historical data, workload patterns, and system characteristics. By utilizing machine learning techniques, past scheduling decisions can be analyzed to derive appropriate constraints for similar scenarios.

6.Constraint Validation and Feedback: Implementing validation mechanisms to verify the coherence and practicality of defined constraints can assist users in detecting mistakes or discrepancies in their specifications. Offering guidance on constraint breaches and proposing remedial measures can aid users in enhancing their constraints.

7.Template-Based Constraint Specification: Providing pre-designed templates or templates tailored to common scenarios can assist users in easily defining limitations without the need for extensive understanding of scheduling algorithms or parameters. These templates can be personalized by users to align with their individual needs.

By addressing these challenges and implementing methods to assist users in specifying constraints effectively, the usability and effectiveness of DAPS, BAS, and DBS can be enhanced, leading to improved scheduling outcomes and user satisfaction.

- **Trade-offs between different metrics or consider additional factors such as scalability and adaptability**

Consider some trade-offs between different metrics, focusing on scalability and adaptability of DAPS, BAS, and DBS in a cloud computing environment:

1.Makespan vs. Resource Utilization:

•**Trade-off:** As the makespan decreases (tasks are completed at a faster rate), there is a tendency for resource utilization to increase. This occurs because tasks may be scheduled more assertively in order to meet deadlines, resulting in greater resource competition and potentially reduced overall system efficiency.

•**Scalability Implication:** In systems with a high number of concurrent tasks, prioritizing the reduction of makespan can lead to problems with resource contention and scalability. This occurs when the system faces difficulties in efficiently allocating resources to an ever-growing number of tasks.

2.Cost of Execution vs. Deadline Adherence:

•**Trade-off:** Reducing the expense of execution frequently entails enhancing resource allocation and workload scheduling to minimize idle time and maximize resource utilization. Nevertheless, this approach may result in compromising task deadlines, as tasks could be postponed or interrupted to minimize costs.

•**Adaptability Implication:** In environments that are constantly changing, with prices of resources fluctuating and priorities of workloads varying, it becomes difficult to

find a balance between optimizing costs and meeting deadlines. Scheduling strategies that can adapt are necessary to constantly reassess task priorities and allocate resources in order to ensure timely completion without exceeding budget constraints.

3.Scalability vs. Adaptability:

•**Trade-off:** Scalability pertains to the system's capacity to effectively manage growing workloads and resource requirements as the system expands in size. Nevertheless, as the system expands, preserving flexibility becomes increasingly difficult due to the rising complexity and diversity of the environment.

•**Scalability Implication:** Scheduling algorithms that are scalable must have the capability to allocate resources dynamically and adapt scheduling policies to support increasing workloads and resource requirements without causing a notable impact on performance or adding to the overhead.

•**Adaptability Implication:** Adaptable scheduling algorithms need to be capable of adjusting to fluctuations in workload attributes, availability of resources, and system behavior instantly, all while guaranteeing scalability. This necessitates strong monitoring, forecasting, and decision-making mechanisms to uphold peak performance in diverse circumstances.

4.Resource Utilization vs. Energy Efficiency:

•**Trade-off:** Enhancing resource allocation frequently leads to increased energy usage, since additional resources are maintained in an active state to fulfill workload requirements. Conversely, prioritizing energy efficiency may require combining tasks onto a smaller number of resources or enforcing stringent power management strategies, potentially resulting in decreased resource utilization.

•**Scalability Implication:** Achieving energy efficiency in extensive cloud environments with numerous servers can lead to substantial cost savings and environmental advantages. Nevertheless, maintaining energy efficiency on a large scale involves managing resource utilization alongside power management overhead, while also guaranteeing that performance remains unaffected.

- **Application in real cloud computing environment**

Conducting simulations in a real cloud computing environment offers several advantages over using simulation toolboxes like CloudSim. Here's how applying DAPS, BAS, and DBS in a real cloud environment can enhance research:

1.Real-world Validity: Simulations frequently rely on assumptions or simplifications that may not accurately capture the intricacies of actual cloud environments. By performing experiments in an authentic cloud setting, researchers can verify the efficacy of DAPS, BAS, and DBS under genuine conditions, guaranteeing the applicability and reliability of the findings.

2.Scalability Testing: Real cloud environments enable researchers to assess the scalability of DAPS across various workloads, resource configurations, and system sizes. This grants valuable insights into the performance of the scheduling algorithm as both the workload and system complexity grow.

3.Dynamic Adaptability: In an actual cloud setting, researchers have the opportunity to witness the dynamic adaptation of DAPS, BAS, and DBS to variations in workload patterns, resource availability, and system dynamics. This encompasses situations like abrupt surges in demand, resource malfunctions, or alterations in task priorities that might not be completely replicated in simulations.

4.Resource Cost Analysis: Researchers have the ability to evaluate the cost-effectiveness of DAPS, BAS, and DBS by taking into account various factors including the efficiency of resource usage, operational expenses, and potential savings in both time and money. The utilization of real cloud experiments offers precise data for conducting a thorough cost-benefit analysis, thereby supporting the rationale behind implementing DAPS, BAS, and DBS in real-world cloud deployments.

5.Industry Relevance: Showing the efficacy of DAPS, BAS, and DBS in an actual cloud setting enhances its significance for industry players, including cloud service providers, IT experts, and software developers. This may result in wider acceptance and utilization of the scheduling algorithm in practical cloud scenarios.

6.Benchmarking Against Alternatives: Researchers have the opportunity to evaluate DAPS, BAS, and DBS by comparing it with various scheduling algorithms or methods in an actual cloud setting, allowing for a thorough assessment of their efficiency, adaptability, and applicability to diverse scenarios in real-world cloud applications.

Overall, applying DAPS, BAS, and DBS in a real cloud computing environment enhances the credibility, relevance, and practicality of the research, providing valuable insights into its effectiveness and potential impact on real-world cloud deployments.

- **Comparative analysis for insights into relative strengths and weaknesses of proposed models compared to alternative solutions**

Deadline-Aware Priority Scheduling (DAPS)

Compare Deadline-Aware Priority Scheduling (DAPS) with another common scheduling algorithm in cloud computing, such as Round Robin (RR) scheduling:

Strengths of Deadline-Aware Priority Scheduling (DAPS) in Cloud Computing:

1.Deadline Compliance: DAPS prioritizes tasks with approaching deadlines, a critical aspect in cloud computing settings where tasks frequently come with time-sensitive demands. Timely completion not only boosts user satisfaction but also helps avoid penalties for failing to meet deadlines.

2.Resource Optimization: DAPS enhances resource utilization in cloud environments by prioritizing tasks according to their deadlines. The allocation of resources is initially focused on critical tasks, guaranteeing the timely completion of significant workloads.

3.Efficiency in Resource Allocation: DAPS optimally distributes resources among tasks according to their significance, resulting in enhanced system performance and throughput. This advantage is especially valuable in multi-tenant cloud environments where resource conflicts are prevalent.

4.Dynamic Adaptability: DAPS has the capability to flexibly modify task priorities in response to evolving workload conditions and deadline constraints. This flexibility enables it to adeptly manage variations in workload levels and optimize resource allocation.

Weaknesses of Deadline-Aware Priority Scheduling (DAPS) in Cloud Computing:

1.Complexity: Deploying DAPS in cloud environments necessitates an advanced scheduling algorithm that takes into account task deadlines, introducing intricacy to the scheduling procedure. This intricacy could potentially raise the costs associated with implementation and upkeep.

2.Overhead: DAPS might result in increased overhead as a result of the requirement to consistently monitor task deadlines and adapt priorities as necessary. This additional workload has the potential to affect system performance, particularly in extensive cloud deployments with a significant volume of tasks.

3.Fairness: DAPS focuses on tasks according to their deadlines, potentially resulting in unequal treatment of low-priority tasks that have longer deadlines. This lack of equity could pose challenges in cloud environments where equal treatment of all users or tasks is essential.

4.Dependency on Deadline Accuracy: The efficiency of DAPS greatly depends on precise deadline information. Incorrect or vague deadline estimates may result in less than optimal scheduling choices, which can have a negative impact on system performance and meeting deadlines.

Comparative Analysis with Round Robin (RR) Scheduling:

1.Deadline Compliance: DAPS surpasses RR scheduling in meeting task deadlines by prioritizing tasks according to their deadlines, unlike RR scheduling which treats all tasks equally regardless of their urgency. This characteristic renders DAPS more appropriate for time-sensitive workloads.

2.Resource Utilization: DAPS generally leads to improved resource utilization in contrast to RR scheduling by giving priority to critical tasks, guaranteeing that

resources are assigned to the most crucial workloads initially. RR scheduling could potentially result in resource underutilization, particularly when significant tasks are postponed because of equal treatment.

3.Adaptability: Although RR scheduling is straightforward and convenient to execute, it does not possess the flexibility of DAPS in managing changing workload patterns and organizing tasks according to their level of importance. DAPS has the capability to adaptively modify task priorities for efficient resource distribution, rendering it better suited for dynamic cloud settings.

4.Fairness: RR scheduling is commonly regarded as more equitable than DAPS as it provides equal treatment to all tasks. Nevertheless, this impartiality may result in the possibility of failing to meet deadlines for crucial tasks in time-sensitive cloud environments.

In conclusion, Deadline-Aware Priority Scheduling presents significant advantages in meeting task deadlines and optimizing resource utilization in cloud computing environments. However, it introduces added complexity and overhead, and its effectiveness relies on accurate deadline information. In contrast, Round Robin scheduling is simpler and fairer but may result in missed deadlines for critical tasks and underutilization of resources in time-sensitive cloud environments.

Budget-Aware Scheduling (BAS)

Compare Budget-Aware Scheduling (BAS) with another common scheduling algorithm in cloud computing, such as Proportional-Share Scheduling:

Strengths of Budget-Aware Scheduling (BAS) in Cloud Computing:

1.Cost Optimization: BAS strives to enhance cost-effectiveness by taking into account budget limitations when organizing tasks in cloud settings. This guarantees that resource distribution is in line with the financial boundaries of users, avoiding excessive spending and facilitating improved resource management.

2.Resource Efficiency: BAS optimizes the allocation of resources by considering users' budgets, guaranteeing that resources are distributed in a way that maximizes

cost-effectiveness. As a result, resource utilization is enhanced and wastage is minimized, benefiting both users and cloud providers.

3.Fairness: BAS aims to uphold equity by distributing resources in proportion to users' budgets. This guarantees that users with higher budgets are allocated a fair share of resources, thus averting resource domination by those with larger budgets.

4.Adaptability: BAS possesses the capability to flexibly modify resource allocations in response to evolving budgetary limitations and workload requirements. This flexibility enables it to efficiently manage resource utilization while staying within the confines of users' budgetary restrictions, thereby optimizing cost effectiveness.

Weaknesses of Budget-Aware Scheduling (BAS) in Cloud Computing:

1.Complexity: The integration of BAS necessitates a complex scheduling algorithm that considers budget limitations, thereby complicating the scheduling procedure. This complication could potentially raise the costs associated with implementing and maintaining cloud services.

2.Overhead: BAS might result in increased overhead as a result of the continuous monitoring of users' budget usage and the subsequent adjustment of resource allocations. This additional workload has the potential to affect system performance, particularly in large-scale cloud deployments with a significant number of users.

3.Budget Accuracy: The efficiency of BAS relies on precise evaluation and monitoring of users' budgets. Incorrect or vague budget estimations may result in less than optimal allocation of resources, affecting cost effectiveness and user contentment.

4.Fairness Trade-offs: While the primary goal of BAS is to ensure fairness by distributing resources in proportion to users' budgets, it can unintentionally put users with smaller budgets at a disadvantage in fiercely competitive settings. Achieving a balance between fairness and cost optimization objectives can be a complex task that necessitates making compromises.

Comparative Analysis with Proportional-Share Scheduling:

1.Cost Optimization: BAS prioritizes cost optimization by taking into account the budget limitations of users, while Proportional-Share Scheduling assigns resources based on predetermined shares, without explicitly considering budget constraints. BAS is more suitable for users or applications that are conscious of costs or have strict budget limitations.

2.Resource Efficiency: BAS and Proportional-Share Scheduling have a common goal of enhancing resource utilization. However, BAS accomplishes this objective by adaptively modifying resource allocations according to users' budgets, while Proportional-Share Scheduling allocates resources based on predetermined shares. By aligning resource allocations with users' budgetary needs, BAS has the potential to optimize resource utilization more effectively.

3.Fairness: Both Budget-Aware Scheduling (BAS) and Proportional-Share Scheduling aim to ensure equity in distributing resources. BAS accomplishes this by distributing resources in proportion to users' budgets, while Proportional-Share Scheduling assigns resources according to predetermined shares that may not necessarily align with users' specific needs. BAS could potentially offer a more equitable allocation method by taking into account users' budget limitations explicitly.

4.Adaptability: BAS and Proportional-Share Scheduling exhibit contrasting characteristics when it comes to accommodating fluctuating workload demands and budget limitations. BAS possesses the ability to dynamically modify resource allocations by considering users' budget utilization, enabling it to effectively respond to changing conditions. On the other hand, Proportional-Share Scheduling may have limited adaptability since it relies on predetermined shares that may not accurately align with users' evolving requirements.

To summarize, Budget-Aware Scheduling provides benefits in terms of cost optimization, resource efficiency, fairness, and adaptability in cloud computing environments. Nevertheless, it introduces additional complexity and overhead, and its effectiveness relies on precise budget estimation and tracking. On the other hand, Proportional-Share Scheduling offers a simpler resource allocation approach but may

not deliver the same degree of cost optimization and adaptability as BAS, particularly in environments with varying budget constraints and fluctuating workload demands.

Deadline Budget Scheduling (DBS)

Compare Deadline Budget Scheduling (DBS) with another scheduling algorithm commonly used in cloud computing, such as Priority-based Scheduling:

Strengths of Deadline Budget Scheduling (DBS) in Cloud Computing:

1.Deadline and Budget Compliance: DBS guarantees that tasks are arranged to adhere to both their deadlines and budget restrictions. This is beneficial in cloud settings where users face budget constraints and time-critical tasks. DBS ensures that tasks are finished within their designated deadlines while also staying within budget boundaries.

2.Cost Optimization: DBS enhances cost efficiency by taking into account deadline and budget limitations during task scheduling. Through strategic resource allocation according to task deadlines and user budgets, DBS reduces resource wastage and avoids exceeding budget limits, ultimately resulting in enhanced cost-effectiveness.

3.Resource Utilization: DBS enhances resource allocation efficiency by dynamically assigning resources according to task deadlines and budget limitations. This guarantees that resources are distributed to tasks in a way that optimizes both meeting deadlines and utilizing the budget effectively, resulting in improved resource utilization within the cloud environment.

4.Fairness: DBS ensures fairness by taking into consideration the deadlines of tasks as well as the budgets of users during the task scheduling process. This approach guarantees that resources are distributed equitably among users, while also accounting for their specific needs and limitations.

Weaknesses of Deadline Budget Scheduling (DBS) in Cloud Computing:

1.Complexity: Implementing DBS necessitates a refined scheduling algorithm that takes into account task deadlines as well as user budgets. This introduces intricacy to

the scheduling procedure and could potentially amplify the expenses associated with implementation and maintenance for cloud providers.

2.Overhead: DBS could potentially affect system performance, particularly in large-scale cloud deployments with a high volume of tasks and users, as it requires constant monitoring of task deadlines and user budgets, along with adjusting resource allocations accordingly. This additional overhead may have an impact on the overall efficiency of the system.

3.Accuracy Requirements: The efficiency of DBS relies on precise evaluation and monitoring of task deadlines and user budgets. Incorrect or vague estimations may result in less than optimal scheduling choices, affecting both adherence to deadlines and utilization of budgets.

4.Trade-offs between Deadline and Budget: DBS might encounter difficulties in achieving a balance between meeting task deadlines and adhering to budget constraints. In situations where task deadlines clash with budget limitations, DBS may have to make compromises that prioritize either one, potentially resulting in less than optimal results.

Comparative Analysis with Priority-based Scheduling:

1.Deadline and Budget Consideration: DBS takes into account task deadlines and user budgets when organizing tasks, guaranteeing that resources are distributed efficiently to satisfy both limitations. On the other hand, Priority-based Scheduling might prioritize tasks solely on their significance or urgency, neglecting to explicitly factor in budget constraints.

2.Cost Optimization: DBS effectively manages expenses by taking budget restrictions into consideration, unlike Priority-based Scheduling which may overlook financial limitations. DBS guarantees that resources are distributed in a way that reduces costs and meets deadlines, resulting in improved cost efficiency within budget-constrained settings.

3.Resource Utilization: DBS optimizes resource allocation by dynamically adjusting based on task deadlines and budget constraints, resulting in enhanced resource

utilization. Priority-based Scheduling, on the other hand, may not be as efficient in maximizing resource utilization since it focuses on task importance or urgency without taking budget restrictions into account.

4.Fairness: DBS ensures equity by taking into account both task deadlines and user budgets during resource allocation. In contrast, Priority-based Scheduling may prioritize tasks solely based on their priority, potentially resulting in uneven resource allocation, particularly in multi-tenant cloud environments.

In essence, Deadline Budget Scheduling presents advantages in meeting task deadlines and adhering to budget constraints in cloud computing environments. However, it introduces added complexity and overhead, and its effectiveness relies on accurate estimation and tracking of task deadlines and user budgets. Conversely, Priority-based Scheduling may not explicitly account for budget constraints and may not optimize resource allocation as effectively in budget-constrained environments.

CHAPTER 5

CONCLUSION

5.1 CONCLUSION

We provided three models in this study to address the task scheduling problem in a cloud computing environment, with the models focusing on the following: Tasks were planned in the first model based on a deadline restriction. The tasks were scheduled based on the budget constraint in the second model, and the tasks were scheduled based on the deadline and budget constraints in the third model, with the user defining all of the restrictions. In addition, we developed a set of criteria for evaluating the efficacy of our suggested models for solving the task scheduling problem.

Cloud computing has transformed IT by allowing users and consumers to access services via the Internet. These services range from hardware to software, saving money on both the expense of setting up physical resources and the cost of obtaining the appropriate software licenses. The task scheduling problem is one of the most important and prominent challenges facing the cloud computing system. Task Scheduling is concerned with assigning user-generated tasks to the appropriate resources given by the service provider. Some restrictions such as deadline and budget are the major constraints in QoS applied on the users' requirements of cloud resources, according to the Quality of Service (QoS) agreed in the Service Level Agreement (SLA), which means the service provider commits to perform not only the tasks within a specified time but also to ensure that the cost of execution does not exceed the user's budget. As a result, we strive to solve the problem of task scheduling in many ways in this thesis, beginning with improving task scheduling within a deadline restriction by presenting the Deadline-Aware Priority Scheduling (DAPS) model.

The DAPS model's method is that tasks are ranked in ascending order based on length priority, and the Virtual Machine's (VM's) state is labeled as successful when the deadline restriction is met. The jobs are then assigned to the appropriate VM while keeping the make span and completion time to a minimum. The DAPS model aims to achieve the optimum performance by lowering metrics like average makespan, mean

of total average reaction time, number of violations, and violation ratio in order to achieve user satisfaction. In addition, the DAPS model aims to maximize resource usage and guarantee ratio when the user is primarily concerned with the cost of completing tasks, which includes the cost of processing, memory, bandwidth, and storage depending on the budget constraint. The Budget-Aware Scheduling (BAS) model was designed to address this issue. The BAS model's mechanism is that tasks are scheduled depending on a budget limitation. The task priority is established first, and then the VM that fits the budget is labeled. The task attributes are then checked and assigned to the resources that fulfill the budget limit, in order to minimize the resource use cost and makespan to a minimum. The BAS model schedules work by accepting user submission requests, and task priority is determined by computing the average of task budget constraints and comparing them to the overall task budget constraints. The VMs that meet the budget constraint are next examined. The task is then assigned to the appropriate VM, and the cost of resources used is determined. The users have a lot of chores that they need to schedule on the cloud resources. The task has a number of critical properties, such as length and file size, that must not be overlooked, and each resource has a different cost than the others.

Some task performance indicators (gain cost, profit, resource usage, violations number, makespan, and response time) can influence resource cost, as these metrics are critical for the suggested BAS model. In addition to the obstacles that the user and the service provider confront, the user must complete the task while adhering to many constraints, such as a deadline and a budget. The Deadline Budget Scheduling (DBS) model was introduced to execute users' tasks on virtual machines under QoS restrictions with the shortest feasible execution time and the lowest possible cost. The suggested DBS model works by having the user define two constraints: a deadline and a budget, and then assigning a constraint type to each task based on user satisfaction. Task constraint type is D & B if the constraints are deadline and budget, D if the constraint is a deadline, and B if the constraint is a budget in the DBS model. The following is how the available resources are clustered based on customer satisfaction: The first cluster is made up of VMs that adhere to the budget and deadline requirements. The second cluster is made up of virtual machines that must meet a deadline. The third cluster is made up of VMs that only meet the budget constraint. Cloud customers send their jobs to the service provider, along with any deadlines or

budget limits. Based on user satisfaction, each task is examined for recognizing its constraint type. If the task constraint type is both deadline and budget, it will be implemented in cluster one; if the constraint type is only deadline, it will be implemented in cluster two; and if the constraint type is only budget, it will be implemented in cluster three (length and file size). Finally, under the DBS model, customer satisfaction is accomplished through reducing the process's completion time and cost. Provider satisfaction, on the other side, is attained through optimizing profit revenue and resource usage. The CloudSim toolbox is used in this thesis to simulate cloud computing system investigations.

In conclusion, this study provides answers to the following questions:

i. **Q1-** How do you create a scheduling model that makes efficient use of resources during load management and reduces total completion time while meeting a deadline? In Chapter 3, the Deadline-Aware Priority Scheduling (DAPS) model was proposed to plan tasks and assign them to available resources while minimizing makespan based on a deadline constraint and a minimal completion time.

ii. **Q2-** How to develop a model which reduces the total cost for task scheduling problem in a cloud computing environment, which comprises of the cost of processor, memory, bandwidth, and storage based on a budget constraint? In Chapter 4, the answer was found by presenting the Budget-Aware Scheduling (BAS) model for user budget when performing tasks on VMs in order to keep resource use costs to a minimum.

iii. **Q3-** How can we create a task scheduling model that allows us to complete activities while adhering to user-defined deadlines and budgets? In Chapter 5, the answer was found by introducing a Deadline Budget Scheduling (DBS) model capable of scheduling jobs in a heterogeneous cloud environment to minimize the makespan under a user-defined deadline while lowering monetary expenses while staying within the user-defined budget.

5.2 CONTRIBUTIONS

The following are the contributions to improving task scheduling algorithms in the cloud computing environment offered in this study: Based on the deadline constraint, we suggested a Deadline-Aware Priority Scheduling (DAPS) model capable of assigning jobs while responding to user and provider satisfaction. When compared to GA, Min-Min, SJF, and Round Robin algorithms, the DAPS model guarantees an increase in both makespan and resource utilization under deadline constraints. Furthermore, the DAPS model's performance evaluation shows that our model outperforms the competition by reducing metrics like average makespan, mean total average reaction time, number of violations, and violation ratio while maximizing resource utilization and guarantee ratio. Our proposed DAPS model improves the makespan ratio by 10%, 6%, 12%, and 12.7 percent when compared to GA, Min-Min, SJF, and Round Robin algorithms, respectively. In addition, the T-test was used to compare our suggested DAPS model to other algorithms such as GA, Min-Min, SJF, and Round Robin in a statistical study. A budget distribution technique to allocate jobs according to their attributes in order to assist resource selection decisions is proposed in the Budget-Aware Scheduling (BAS) model.

The experiments were carried out on the BAS model and compared to state-of-the-art scheduling algorithms, demonstrating that the BAS minimizes the makespan, response time, and number of violations for task execution on VMs, as well as increasing resource utilization and provider profit, and achieving an acceptable total gain cost for any user. The proposed BAS model outperforms the Max-Min, Round Robin, and SJF algorithms by 17 percent, 34 percent, and 31 percent, respectively, in terms of cost ratio. In addition, the T-test was used to compare our suggested BAS model to other algorithms such as Max-Min, Round Robin, and SJF in a statistical study. The user must complete the task while adhering to other constraints, such as the deadline and budget. As a result, we've devised a novel model in which the user sets two constraints: a deadline and a budget. Simulation results indicate that the Deadline Budget Scheduling (DBS) model outperformed state-of-the-art algorithms in lowering the makespan and cost in a variety of configurations, including low resources or high resources model, varied number of jobs and virtual machines (VMs). The violation ratio is lowered in the DBS model to meet user requirements while enhancing the

provider's profit and resource utilization. As a result, the DBS model is preferred over state-of-the-art algorithms such as GA, Max-Min, Round Robin, and SJF. The DBS model outperforms the GA, Max-Min, Round Robin, and SJF algorithms in terms of makespan ratio by 39 percent, 5 percent, 41 percent, and 41 percent, respectively, as well as cost ratio by 36 percent, 14 percent, 38 percent, and 38 percent, respectively. In addition, the T-test was used to compare our suggested DBS model to other algorithms such as GA, Max-Min, Round Robin, and SJF in a statistical study.

5.3 FUTURE SCOPE OF THE WORKS

While wrapping up this study, we'd want to highlight some of the research's future directions, which are still open and potential to pursue in order to address the following issues:

- Allocating virtual machines at the physical machine level that is, reallocating them to the needs of the user while also achieving high performance in resource exploitation for the service provider.
- As virtual machines are relocated from one host to another in the cloud datacenter, load balancing in the datacenter is achieved.
- Reducing the amount of energy utilized by the datacenter in a cloud computing environment while scheduling as many operations or processes as possible in the datacenter.

REFERENCES

1. Gagandeep Kaur (2021). Framework for Resource Management in Cloud Computing. 10.1007/978-981-15-7062-9_3.

2. Shaw, Rachael (2021),” Applying machine learning towards automating resource management in cloud computing environments”,

3. Harvinder Singh et al (2020),” Cloud Resource Management: Comparative Analysis and Research Issues”, International Journal of Scientific & Technology Research Volume 9, Issue 06

4. Mohd Ameen Imran et al (2020),” Log as a Secure Service Scheme (LASS) for Cloud”, Journal of Scientific Research

5. Singh, Saurabh, Young-Sik Jeong, and Jong Hyuk Park. "A survey on cloud computing security: Issues, threats, and solutions." Journal of Network and Computer Applications 75 (2016): 200-222.

6. Gao, Yue, et al. "An energy and deadline aware resource provisioning, scheduling and optimization framework for cloud systems." Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2013 International Conference on. IEEE, 2013:1-10.

7. Dong, Ziqian, Ning Liu, and Roberto Rojas-Cessa. "Greedy scheduling of tasks with time constraints for energy-efficient cloud-computing data centers." Journal of Cloud Computing 4.1 (2015): 1-14.

8. Xu, Minxian, Wenhong Tian, and Rajkumar Buyya. "A survey on load balancing algorithms for virtual machines placement in cloud computing." Concurrency and Computation: Practice and Experience 29.12 (2017):1-20.

9. Deldari, Arash, Mahmoud Naghibzadeh, and Saeid Abrishami. "CCA: a deadline-constrained workflow scheduling algorithm for multicore resources on the cloud." The journal of Supercomputing 73.2 (2017): 756-781.

10. Singh, Sukhpal, Inderveer Chana, and Maninder Singh. "The Journey of QoSAware Autonomic Cloud Computing." IT Professional 19.2 (2017): 42-49.

11. Panda, Sanjaya K., Indrajeet Gupta, and Prasanta K. Jana. "Task scheduling algorithms for multi-cloud systems: allocation-aware approach." Information Systems Frontiers (2017): 1-19.

12. Shawish, Ahmed, and Maria Salama., 2014. "Cloud computing: paradigms and technologies." Inter-cooperative Collective Intelligence: Techniques and Applications. Springer Berlin Heidelberg, pp. 39-67.

13. Gill, Sukhpal Singh, and Rajkumar Buyya. "A Taxonomy and Future Directions for Sustainable Cloud Computing: 360 Degree View." arXiv preprint arXiv: 1712.02899 (2017).

14. Kratzke, Nane, and Peter-Christian Quint. "Understanding cloud-native applications after 10 years of cloud computing-A systematic mapping study." Journal of Systems and Software126 (2017): 1-16.

15. Yousafzai, Abdullah, et al. "Cloud resource allocation schemes: review, taxonomy, and opportunities." Knowledge and Information Systems 50.2 (2017): 347-381.

16. Shah, Manan D., and Harshad B. Prajapati. "Reallocation and allocation of virtual machines in cloud computing." arXiv preprint arXiv: 1304.3978 (2013).

17. Masdari, Mohammad, et al. "A Survey of PSO-based scheduling algorithms in cloud computing." Journal of Network and Systems Management 25.1 (2017): 122-158.

18. Deka, Rup Kumar, Dhruba Kumar Bhattacharyya, and Jugal Kumar Kalita. "DDoS Attacks: Tools, Mitigation Approaches, and Probable Impact on Private Cloud Environment." arXiv preprint arXiv: 1710.08628 (2017).

19. Pandey, Asmita. "Virtual machine performance measurement." Recent Advances in Engineering and Computational Sciences (RAECS), IEEE, 2014:1-3.

20. García-Valls, Marisol, Tommaso Cucinotta, and Chenyang Lu. "Challenges in real-time virtualization and predictable cloud computing." Journal of Systems Architecture 60.9 (2014): 726-740.

21. Gill, Sukhpal Singh, et al. "CHOPPER: an intelligent QoS-aware autonomic resource management approach for cloud computing." Cluster Computing (2017): 1-39.

22. Sahal, Radhya, Mohamed H. Khafagy, and Fatma A. Omara. "A Survey on SLA Management for Cloud Computing and Cloud-Hosted Big Data Analytic Applications." International Journal of Database Theory and Application 9.4 (2016): 107-118.

23. Singh, Sukhpal, and Inderveer Chana. "QRSF: QoS-aware resource scheduling framework in cloud computing." The Journal of Supercomputing 71.1 (2015): 241-292.

24. Buyya, Rajkumar, Rajiv Ranjan, and Rodrigo N. Calheiros. "Intercloud: Utilityoriented federation of cloud computing environments for scaling of application services." International Conference on Algorithms and Architectures for Parallel Processing. Springer, Berlin, Heidelberg, 2010: 13-31.

25. Kaur, Er Amanpreet, Bikrampal Kaur, and Dheerendra Singh. "CHALLENGES TO TASK AND WORKFLOW SCHEDULING IN CLOUD ENVIRONMENT." International Journal 8.8 (2017): 412-415.

26. Buyya, Rajkumar, Rodrigo N. Calheiros, and Xiaorong Li. "Autonomic cloud computing: Open challenges and architectural elements." Emerging Applications of Information Technology (EAIT), 2012 Third International Conference on. IEEE, 2012: 3-10.

27. JongBeom Lim et al (2019),” Intelligent Resource Management Schemes for Systems, Services, and Applications of Cloud Computing Based on Artificial Intelligence”, J Inf Process Syst, Vol.15

28. J. Antony John Prabu (2019),” Performance Analysis of Proposed D1FTBC Approach for Improving Consistency in Cloud Data Transactions”, International Journal of Scientific & Technology Research Volume 8, Issue 08

29. Dr. Diwakar Ramanuj Tripathi (2019),” Analytical Study of Cloud Computing Databases Performance on the Basis of Expenditure and Implementation Time”, International Journal of Science and Research (IJSR)

30. Gawali, M.B., Shinde, S.K. Task scheduling and resource allocation in cloud computing using a heuristic approach. J Cloud Comp 7, 4 (2018). https://doi.org/10.1186/s13677-018-0105-8

31. Mitrevski, Filip & Pajkovski, Darko & Dimovski, Tome. (2017). Transaction Processing Applications in Cloud Computing.

32. Qusay Kanaan Kadhim et al (2017),” A Review Study on Cloud Computing Issues”,

33. Sultan Aldossary (2016),” Data Security, Privacy, Availability and Integrity in Cloud Computing: Issues and Current Solutions”, International Journal of Advanced Computer Science and Applications, Vol. 7, No. 4,

34. Artan Mazreka et al (2016),” Pricing Schemes in Cloud Computing: An Overview”, International Journal of Advanced Computer Science and Applications, Vol. 7, No. 2

35. Joundy, Manar. (2016). Improving Tasks Scheduling In Cloud Computing.

36. J. Antony John Prabu (2015),” Issues and Challenges of Data Transaction Management in Cloud Environment”, International Research Journal of Engineering and Technology

37. D. S. B. R. Kumar (2015),” Issues and Challenges of Data Transaction Management in Cloud Environment”,

38. R. Velumadhava Rao (2015),” Data Security Challenges and It’s Solutions in Cloud Computing”, Procedia Computer Science 48 (2015) 204 – 209

39. Usman Namadi Inuwa (2015),” The Risk and Challenges of Cloud Computing”, Int. Journal of Engineering Research and Applications

40. Azim, Nesrine & El-Bastawissy, Ali. (2014). Transactions Management in Cloud Computing. Egyptian Computer Science Journal. 38.

41. Nesrine Ali Abd-El Azim (2014),” Transactions Management in Cloud Computing”, Egyptian Computer Science Journal Vol. 38 No. 1

42. Petter Svard (2014),” Dynamic Cloud Resource Management”,

43. Linlin Wu (2014),” SLA-based Resource Provisioning for Management of Cloud-based Software-as-a-Service Applications”,

44. Shri V D Garde (2013),” Concurrency Lock Issues in Relational Cloud Computing”, Network and Complex Systems

45. Pranita P. Khairnar (2013),” Cloud Computing Security Issues and Challenges”, International Refereed Journal of Engineering and Science

46. Kashif Munir (2013),” Framework for Secure Cloud Computing”, International Journal on Cloud Computing: Services and Architecture

47. Sindhu S (2011),” Efficient Task Scheduling Algorithms for Cloud Computing Environment”,

48. Muhammad Usman Sana and Zhanli Li (2021),” Efficiency aware scheduling techniques in cloud computing: a descriptive literature review”, PeerJ Comput. Sci.

49. Fahd Alhaidari (2021),” Enhanced Round-Robin Algorithm in the Cloud Computing Environment for Optimal Task Scheduling”, *Computers*

50. J. Kok Konjaang (2021),” Multi-objective workflow optimization strategy (MOWOS) for cloud computing”, Journal of Cloud Computing: Advances, Systems and Applications

51. Farooq Hoseiny et al (2021),” Joint QoS-aware and Cost-efficient Task Scheduling for Fog-Cloud Resources in a Volunteer Computing System”, ACM Trans. Internet Technol., Vol. 1, No. 1

52. Honglin Zhang (2021),” EHEFT-R: multi-objective task scheduling scheme in cloud computing”,

53. Zeinab Shahbazi (2021),” Improving Transactional Data System Based on an Edge Computing–Blockchain–Machine Learning Integrated Framework”, Processes 2021, 9, 92. https://doi.org/10.3390/ pr9010092

54. Dinh C. Nguyen (2020),” Integration of Blockchain and Cloud of Things: Architecture, Applications and Challenges”, IEEE Communications Surveys & Tutorials

55. Rasha Makhlouf (2020),” Cloudy transaction costs: a dive into cloud computing economics”, Journal of Cloud Computing: Advances, Systems and Applications

56. Tahani Aladwani (2020),” Types of Task Scheduling Algorithms in Cloud Computing Environment”,

57. Aida Amini Motlagh (2019),” Task scheduling mechanisms in cloud computing: A systematic review”, International Journal of Advanced Computer Science and Applications, Vol. 10, No. 5,

58. Preethi Sheba Hepsiba (2019),” Intelligent Scheduling of Bag-of-Tasks Applications in the Cloud”,

59. Danlami Gabi et al (2019),” Minimized Makespan Based Improved Cat Swarm Optimization for Efficient Task Scheduling in Cloud Datacenter”,

60. Simanta Shekhar Sarmah (2019),” Application of Blockchain in Cloud Computing”, International Journal of Innovative Technology and Exploring Engineering

61. Abhishek A. Singh et al (2019),” WedgeDB: Transaction Processing for Edge Databases”,

62. J. Antony John Prabu (2019),” Performance Analysis of Proposed D1FTBC Approach for Improving Consistency in Cloud Data Transactions”, International Journal Of Scientific & Technology Research Volume 8, Issue 08

63. Gui Huang et al (2019),” X-Engine: An Optimized Storage Engine for Large-scale E-commerce Transaction Processing”, SIGMOD ’19

64. Dr. Diwakar Ramanuj Tripathi (2019),” Analytical Study of Cloud Computing Databases Performance on the Basis of Expenditure and Implementation Time”, International Journal of Science and Research (IJSR)

65. Dileep Mardham (2018),” Cloud tr Cloud transactions and caching for impr ansactions and caching for improved performance in formance in clouds and DTNs”,

66. Aasha Begum (2018),” Database Transaction Processing with Effective Locking Mechanism for Consistency in Cloud Database”, International Journal of Computer Sciences and Engineering

67. Alejandro Zlatko Tomsic. Exploring the design space of highly-available distributed transactions. Databases [cs.DB]. Sorbonne Université, 2018. English.

68. AR. Arunarani (2018),” Task scheduling techniques in cloud computing: A literature survey”,

69. Fatema Akbar Lokhandwala (2018),” A Heuristic Approach to Improve Task Scheduling in Cloud Computing using Blockchain technology”,

70. Ankit Patel et al (2017),” Power-Aware Scheduling for Urgent Tasks in Cloud Environment”, International Journal of Computational Intelligence Research

71. Mokhtar A. Alworafi et al (2017),” Cost-Aware Task Scheduling in Cloud Computing Environment”, I. J. Computer Network and Information Security

72. Qiang Guo (2017),” Task Scheduling Based on Ant Colony Optimization in Cloud Environment”,

73. Zhihao Peng (2017),” Energy-Aware Scheduling of Workflow Using a Heuristic Method on Green Cloud”,

74. Mitrevski, Filip & Pajkovski, Darko & Dimovski, Tome. (2017). Transaction Processing Applications in Cloud Computing.

75. Qusay Kanaan Kadhim et al (2017),” A Review Study on Cloud Computing Issues”,

76. Jin Ho Park et al (2017),” Blockchain Security in Cloud Computing: Use Cases, Challenges, and Solutions”, Symmetry

77. Yousri Mhedheb (2016),” Energy-efficient Task Scheduling in Data Centers”,

78. Sultan Aldossary (2016),” Data Security, Privacy, Availability and Integrity in Cloud Computing: Issues and Current Solutions”, International Journal of Advanced Computer Science and Applications, Vol. 7, No. 4,

79. Ali Gholami (2016),” Security and Privacy of Sensitive Data in Cloud Computing”,4

80. Akon Samir Dey (2016),” Transactions across Heterogeneous Data Stores”,

81. Artan Mazreka et al (2016),” Pricing Schemes in Cloud Computing: An Overview”, International Journal of Advanced Computer Science and Applications, Vol. 7, No. 2

82. Chaowei Yang et al (2016),” Big Data and cloud computing: innovation opportunities and challenges”,

83. D. S. B. R. Kumar (2015),” Issues and Challenges of Data Transaction Management in Cloud Environment”,

84. III Albert Horvath (2015),” An Analytical Study Of Consumer Trust In Cloud Computing”,

85. R. Velumadhava Rao (2015),” Data Security Challenges and Its Solutions in Cloud Computing”, Procedia Computer Science 48 (2015) 204 – 209

86. Usman Namadi Inuwa (2015),” The Risk and Challenges of Cloud Computing”, Int. Journal of Engineering Research and Applications

87. Ettazi, Widad and Hafiddi, Hatim and Nassar, Mahmoud and Ebersold, Sophie A cloud-based architecture for transactional services adaptation. (2015) In: International Conference onTechnologies and Applications (CloudTech 2015),

88. Azim, Nesrine & El-Bastawissy, Ali. (2014). Transactions Management in Cloud Computing. Egyptian Computer Science Journal. 38

89. Vinit A Padhye (2014),””, Transaction and Data Consistency Models for Cloud Applications

90. Padhye, Vinit A. (2014),” Transaction and data consistency models for cloud applications”,

91. Shri V D Garde (2013),” Concurrency Lock Issues in Relational Cloud Computing”, Network and Complex Systems

92. Jagirdar, Srinivas & Venkata, K & Reddy, Subba & Qyser, Dr. (2013). Cloud Computing BASICS. International Journal of Advanced Research in Computer and Communication Engineering. 1. 343.

93. Waleed Al Shehri (2013),” Cloud Database Database as a Service”, International Journal of Database Management Systems (IJDMS) Vol.5, No.2

94. Pranita P. Khairnar (2013),” Cloud Computing Security Issues and Challenges”, International Refereed Journal of Engineering and Science

95. Katarina Grolinger et al (2013),” Data management in cloud environments: NoSQL and NewSQL data stores”, Journal of Cloud Computing: Advances, Systems and Applications

96. Jichao Hu (2015),” Task Scheduling Model of Cloud Computing based on Firefly Algorithm”, International Journal of Hybrid Information Technology

97. Lizheng Guo (2012),” Task Scheduling Optimization in Cloud Computing Based on Heuristic Algorithm”, Journal of Networks, VOL. 7, NO. 3

98. Xiaoli Wang et al (2012),” Energy-efficient task scheduling model based on MapReduce for cloud computing using genetic algorithm”, Journal Of Computers

99. Dillon, Tharam, Chen Wu, and Elizabeth Chang. "Cloud computing: issues and challenges." Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on. Ieee, 2010: 27-33.

100. Qu, Chenhao, Rodrigo N. Calheiros, and Rajkumar Buyya. "A reliable and costefficient auto-scaling system for web applications using heterogeneous spot instances." Journal of Network and Computer Applications 65 (2016): 167-180.

101. Singh, Poonam, Maitreyee Dutta, and Naveen Aggarwal. "A review of task scheduling based on meta-heuristics approach in cloud computing." Knowledge and Information Systems 52.1 (2017): 1-51.

102. Jintao, Jiao, Yu Wensen, and Guo Lei. "Research on Batch Scheduling in Cloud Computing." TELKOMNIKA (Telecommunication Computing Electronics and Control) 14.4 (2016): 1454-1461.

103. Patel, Swachil, and Upendra Bhoi. "Priority based job scheduling techniques in cloud computing: a systematic review." International journal of scientific & technology research 2.11 (2013): 147-152.

104. Liu, Zhongli, et al. "Algorithm optimization of resources scheduling based on cloud computing." Journal of Multimedia 9.7 (2014): 977-985.

105. Atiewi, Saleh, et al. "A review energy-efficient task scheduling algorithms in cloud computing." Long Island Systems, Applications and Technology Conference (LISAT), 2016 IEEE. IEEE, 2016: 1-6.

106. Alworafi, Mokhtar A., and Suresha Mallappa. "An Enhanced Task Scheduling in Cloud Computing Based on Deadline-Aware Model." International Journal of Grid and High Performance Computing (IJGHPC) 10.1 (2018): 31-53.

107. Nehru, E. Iniya, Saswati Mukherjee, and Abhishek Kumar. "Deadline-based Priority Management in Cloud." Artificial Intelligence and Evolutionary Algorithms in Engineering Systems. Springer, New Delhi, 2015. 745-751.

108. Dhari, Atyaf, and Khaldun I. Arif. "An Efficient Load Balancing Scheme for Cloud Computing." Indian Journal of Science and Technology 10.11 (2017).

109. Zhang, Yi, and Baomin Xu. "Task Scheduling Algorithm based-on QoS Constrains in Cloud Computing." International Journal of Grid and Distributed Computing 8.6 (2015): 269-280.

110. Pop, Florin, et al. "Deadline scheduling for aperiodic tasks in inter-Cloud environments: a new approach to resource management." The Journal of Supercomputing 71.5 (2015): 1754-1765.

111. Shin, SaeMi, Yena Kim, and SuKyoung Lee. "Deadline-guaranteed scheduling algorithm with improved resource utilization for cloud computing." Consumer Communications and Networking Conference (CCNC), 2015 12th Annual IEEE. IEEE, 2015: 814-819.

112. Peng, Zhiping, et al. "A Reinforcement Learning-Based Mixed Job Scheduler Scheme for Cloud Computing under SLA Constraint." Cyber Security and Cloud Computing (CSCloud), 2016 IEEE 3rd International Conference on. IEEE, 2016: 142-147.

113. Khorsand, Reihaneh, et al. "ATSDS: adaptive two-stage deadline-constrained workflow scheduling considering run-time circumstances in cloud computing environments." The Journal of Supercomputing 73.6 (2017): 2430-2455.

114. Arabnejad, Vahid, Kris Bubendorfer, and Bryan Ng. "Scheduling deadline constrained scientific workflows on dynamically provisioned cloud resources." Future Generation Computer Systems 75 (2017): 348-364.

115. Toosi, Adel Nadjaran, Richard O. Sinnott, and Rajkumar Buyya. "Resource provisioning for data-intensive applications with deadline constraints on hybrid clouds using Aneka." Future Generation Computer Systems 79 (2018): 765- 775.

116. Arabnejad, Hamid, and Jorge G. Barbosa. "A budget constrained scheduling algorithm for workflow applications." Journal of grid computing 12.4 (2014): 665-679.

117. Thanasias, Vasileios, et al. "VM capacity-aware scheduling within budget constraints in IaaS clouds." PloS one 11.8 (2016): e0160456.

118. Rodriguez, Maria A., and Rajkumar Buyya. "Budget-driven scheduling of scientific workflows in IaaS clouds with fine-grained billing periods." ACM Transactions on Autonomous and Adaptive Systems (TAAS) 12.2 (2017): 5.

119. Chen, Weihong, et al. "Efficient task scheduling for budget constrained parallel applications on heterogeneous cloud computing systems." Future Generation Computer Systems 74 (2017): 1-11.

120. Shi, Jiyuan, et al. "Elastic resource provisioning for scientific workflow scheduling in cloud under budget and deadline constraints." Cluster Computing 19.1 (2016): 167-182.

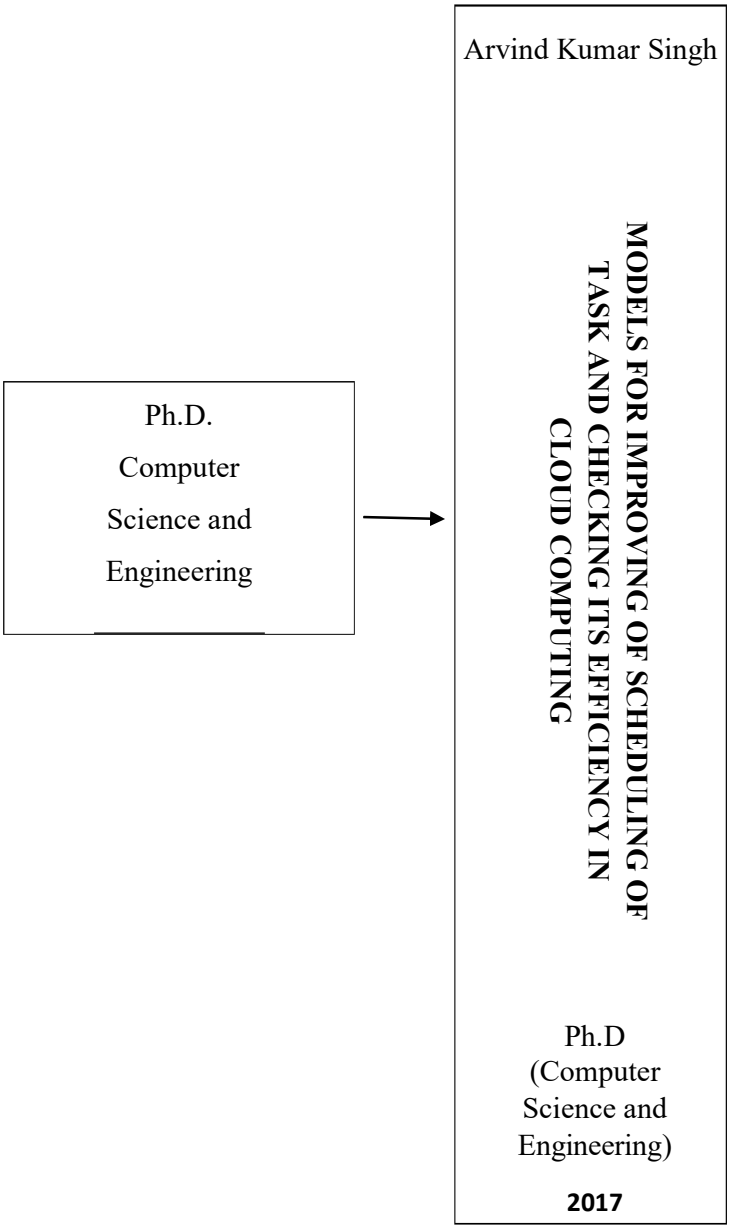
121. Arabnejad, Hamid, Jorge G. Barbosa, and Radu Prodan. "Low-time complexity budget–deadline constrained workflow scheduling on heterogeneous resources." Future Generation Computer Systems 55 (2016): 29-40.

122. Verma, Amandeep, and Sakshi Kaushal. "A hybrid multi-objective Particle Swarm Optimization for scientific workflow scheduling." Parallel Computing 62 (2017): 1-19.

List of Publication

1) **"Tasks Scheduling with Virtual Machines of the Deadline-Aware Priority Scheduling Model in Cloud Computing"** Arvind Kumar Singh, Dr. Hitendra Singh and Dr. Manish Varshney in International Journal of INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING, ISSN: 2147-6799, IJISAE, 2024, 12(8s), 123–127.

2) **"The Significance and Diversity of Task Scheduling Methods in Cloud Computing Platforms"** by Arvind Kumar Singh, Dr. Hitendra Singh and Dr. Manish Varshney in International Journal of INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING, ISSN:2147-6799, IJISAE, 2024, 12(14s), 644–649.


Arvind Kumar Singh
MODELS FOR IMPROVING OF SCHEDULING OF TASK AND CHECKING ITS EFFICIENCY IN CLOUD COMPUTING
Ph.D
(Computer Science and Engineering)
2017
Ph.D.
Computer
Science and
Engineering