An Efficient QoS oriented Multi-Objective Ranking Algorithm for Cloud Computing Services

A thesis submitted to the

University of Petroleum and Energy Studies

For the Award of

Doctor of Philosophy

in

Computer Science and Engineering

by

Preeti Sirohi

November 2020

Supervisor(s)

Dr. Amit Agarwal

Dr. Piyush Maheshwari

UPES

UNIVERSITY WITH A PURPOSE

School of Computer Science

University of Petroleum and Energy Studies

Dehradun, Uttarakhand-248007, India

An Efficient QoS oriented Multi-Objective Ranking Algorithm for Cloud Computing Services

A thesis submitted to the

University of Petroleum and Energy Studies

For the Award of

Doctor of Philosophy

in

Computer Science and Engineering

by

Preeti Sirohi

(SAP ID 500042705)

November 2020

Supervisor(s)

Dr. Amit Agarwal

Professor (On Leave), SoCS, UPES, Dehradun

External Supervisor

Dr. Piyush Maheshwari

Faculty of Engineering & IT

The British University in Dubai (BUiD)

Dubai International Academic City, UAE

UPES

UNIVERSITY WITH A PURPOSE

School of Computer Science

University of Petroleum and Energy Studies

Dehradun, Uttarakhand-248007, India

.

I dedicate my Ph.D. Thesis to

My loving Parents, In-Laws, and my Supervisors

Dr. Amit Agarwal,

Dr. Piyush Maheshwari

for their endless support, blessings and guidance.

DECLARATION

I declare that the thesis entitled **"An Efficient QoS oriented Multi-Objective Ranking Algorithm for Cloud Computing Services"** has been prepared by me under the guidance of Dr. Amit Agarwal, Professor of School of Computer Science, University of Petroleum and Energy Studies and Dr. Piyush Maheshwari, Faculty of Engineering & IT, The British University in Dubai (BUiD), Dubai International Academic City, UAE. No part of this thesis has formed the basis for the award of any degree or fellowship previously.

Preeti

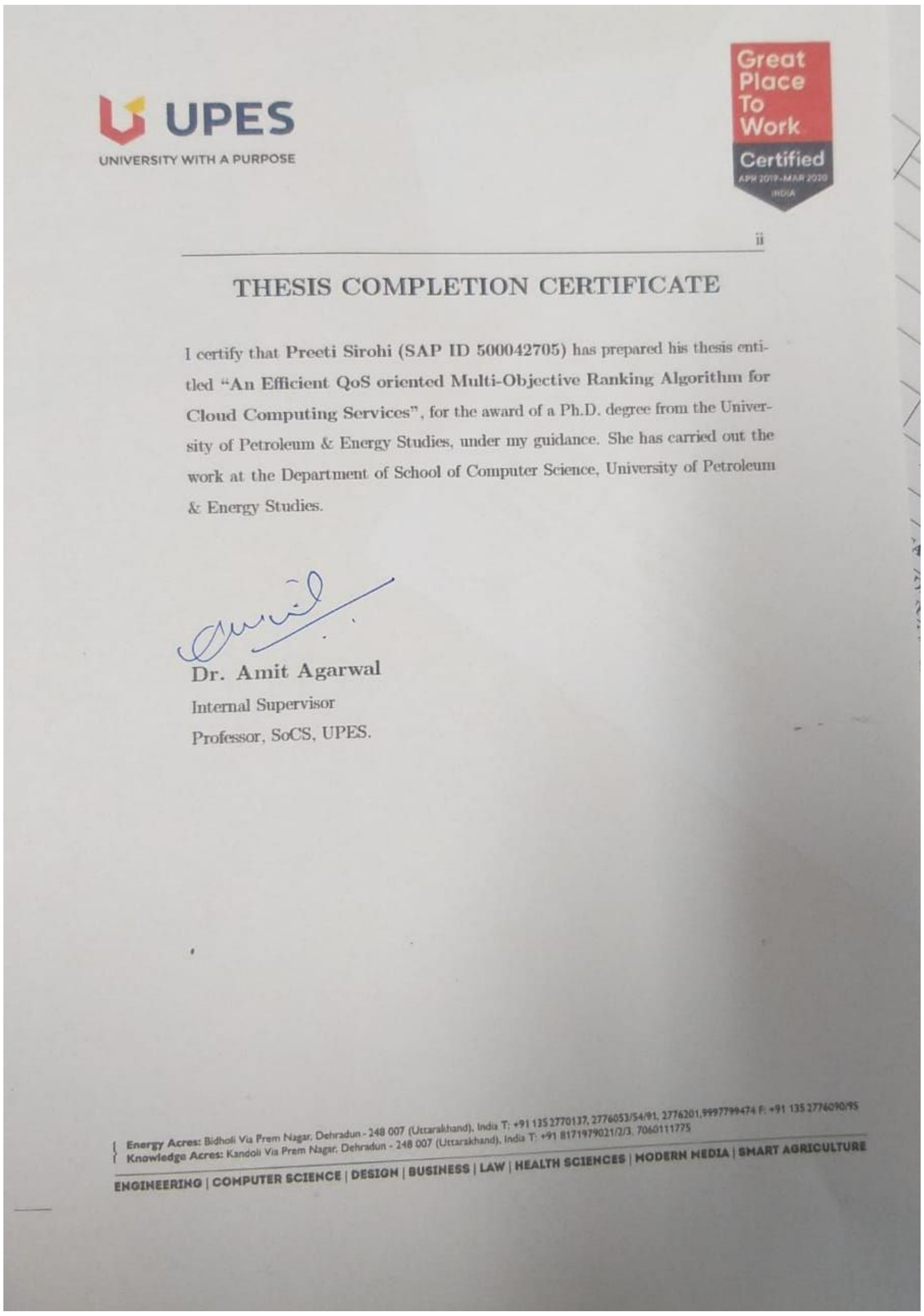
Signature of the Candidate

SAP ID: 500042705

Date: 23 / Nov / 2020

THESIS COMPLETION CERTIFICATE

Internal Supervisor



UPES
UNIVERSITY WITH A PURPOSE

Great Place To Work Certified APR 2019-MAR 2020 INDIA

ii

THESIS COMPLETION CERTIFICATE

I certify that **Preeti Sirohi (SAP ID 500042705)** has prepared his thesis entitled **"An Efficient QoS oriented Multi-Objective Ranking Algorithm for Cloud Computing Services"**, for the award of a Ph.D. degree from the University of Petroleum & Energy Studies, under my guidance. She has carried out the work at the Department of School of Computer Science, University of Petroleum & Energy Studies.

[illegible]

Dr. Amit Agarwal
Internal Supervisor
Professor, SoCS, UPES.

Energy Acres: Bidholi Via Prem Nagar, Dehradun - 248 007 (Uttarakhand), India T: +91 135 2770137, 2776053/54/91, 2776201,9997799474 F: +91 135 2776090/95
Knowledge Acres: Kandoli Via Prem Nagar, Dehradun - 248 007 (Uttarakhand), India T: +91 8171979021/2/3, 7060111775

ENGINEERING | COMPUTER SCIENCE | DESIGN | BUSINESS | LAW | HEALTH SCIENCES | MODERN MEDIA | SMART AGRICULTURE

THESIS COMPLETION CERTIFICATE

External Supervisor

THESIS COMPLETION CERTIFICATE

I certify that **Preeti Sirohi** (SAP ID 500042705) has prepared her thesis entitled "**An Efficient QoS oriented Multi-Objective Ranking Algorithm for Cloud Computing Services**" for the award of Ph.D. degree from the University of Petroleum & Energy Studies, under my guidance. She has carried out the work at the School of Computer Science, University of Petroleum & Energy Studies, Dehradun.

Piyush Maheshwari

Dr. Piyush Maheshwari, *Ph.D. Manchester, Senior Member IEEE*
External Guide
Professor – Computer Science
The British University *in* Dubai (BUiD),
PO Box 345015, Dubai International Academic City, United Arab Emirates
Email: piyush.maheshwari@buid.ac.ae; Website: www.buid.ac.ae

Abstract

Cloud technology has become a paradigm for providing on-demand cloud resources and services on a subscription basis. Several cloud players in the market offer various cloud services with different quality of service (QoS) attributes. In a cloud environment, the users have diverse requirements that should be fulfilled through the specified applications and services offered. One of the biggest challenges in front of the customer is selecting an appropriate cloud service to meet their customized demand. The second challenge is how to perform cloud service selection by simultaneously considering multiple objectives of the customer. The third challenge is how to complete the process of service selection with high effectiveness and efficiency. The fourth challenge is that the existing work in the area of cloud service selection for multi-objective optimization problem first convert the multi-objective problem into single objective and then find the best service to meet the customer requirement. Therefore the optimal service which should include all the objectives is not taken rather the focus goes to find the best service. Therefore a judicious decision is needed for a thorough evaluation of the cloud services from the customer perspective. The non-dominated sorting and ranking of services (NDS-ROS) algorithm proposed to overcome the above challenges and to address the above issues for cloud service selection. The NDS-ROS algorithm is efficient in reducing complexity, lowering down execution time, and the number of comparisons required for finding an optimal solution in the multi-objective optimization problem.

Keywords: Cloud Computing, Multi-Objective Optimization, Optimal Service

Acknowlegement

I earnestly take this opportunity to thank everybody who has contributed directly or indirectly in realizing this thesis. First and foremost, I have to bow my head modestly to pay my deepest regards to the almighty for looking at me with the benevolent intentions and showering his blessings on me to complete this thesis.
Having an opportunity to acknowledge the help I received in completing my thesis, and here the names first come to my mind are my learned supervisors Prof. Amit Agarwal and Prof. Piyush Maheshwari, who has guided me in real sense. I wish to express my deep feeling of gratitude and appreciation to them for their inspiring supervision and invaluable suggestions, without whom this thesis would not have taken shape.
I wholeheartedly acknowledge their full cooperation that I received from the very beginning of this work up to the completion in the form of this thesis. My advisors have given me complete freedom to explore the ways cloud computing has drifted over these years and always guided me with new solutions to the problems. Their encouragement meant a lot to me. I learned a lot from them in the academic area and other spheres of life.
Besides my advisors, I would like to thank Chancellor Dr. S.J. Chopra, Vice Chancellor Dr. Sunil Rai, and Dean SoCS Dr. Manish Prateek, Dean SoCS R&D Dr. Kiran Kumar Ravulakellu at the University of Petroleum and Energy Studies for their continuous encouragement, valuable suggestions, and all-time support.
I want to express my special thanks to Dr. J.K Pandey, R & D Director, and Dr. Rakhi Ruhal, Program Manager Ph. D at the University of Petroleum and Energy Studies (UPES) assistance during the research work. I am also grateful to the UPES for providing me all the necessary support in smooth conduction of my research work and allowing me to be the part of this university I would like to express my sincere thanks to Prof. Alok Pandey, Director, IMS Ghaziabad. Dr. Tapan Kumar Nayak, Dean Academics, Dr. Sachin Malhotra, Dean MCA, and all my faculty colleagues provide me with enormous support to complete my research

work.

It was a wonderful moment when I choose to proceed ahead with pursuing my Ph.D. even when I was entering the busiest phase of my personal and professional life. I am eternally indebted to have a wonderful and loving family and express my gratitude to my parents, my in-laws, sisters, and all the sacrifices and support throughout my Ph.D. journey. I want to express my appreciation for my all-time supporting husband, Sameer, and my daughter Jia, Saara, Saee for unconditional love, encouragement during the tough time, and believing in me. I am incredibly thankful to all my family and friends for making my research journey memorable.

Contents

.

List of Abbreviations

QoS	Quality of Service	2
EA	Evolutionary Algorithm	2
NDS-ROS	Non-dominated sorting and ranking of Services	5
SOA	Service Oriented Architecture	10
IaaS	Infrastructure as a Service	13
PaaS	Platform as a Service	13
SaaS	Software as a Service	13
NIST	National Institute of Standards and Technology	13
SMI	Service Measurement Index	15
SOAP	Simple Object Access Protocol	18
WDSL	Web Service Description Level	18
MCDM	Multi Criteria Decision Making	18
AHP	Analytical Hierarchy Process	18
ANP	Analytical Network Process	19
SAW	Simple Additive Weight	23
SLA	Service Level Agreement	24
HEIM	Hypothetical Equivalent and Inequivalent Methods	26
SOP	Simple Objective Optimization	29
MOOP	Multi-Objective Optimization Problem	29
BOS	Best Oder Sort	40
GBOS	Generalized Best Order Sort	40
CSP	Cloud Service Provider	43
ENS-SS	Efficient Non-Dominated Sorting –Sequential Search	54
ANOVA	Analysis of Variance	74

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background of the Research

Cloud technology enables an on-demand service delivery and also offer virtual computing resources (Stienhans & Klimentiev, 2011) to the cloud users. Cloud is a cost-effective model because it helps organizations to prevent spending money on buying resources such as hardware and software licenses (Ferris & Riveros, 2018). The cloud model offers a shared pool of computing resources amid several tenants simultaneously and which shows the benefits of the economies of scale (Wang et al., 2011) in the cloud. The resources are assigned dynamically to the cloud users, and these resources can be upgraded or downgraded as per the user's requirement (Gao et al., 2011).

In the current scenario, more and more organizations are shifting their day to day activities into the cloud due to flexibility. Different customers have diverse requirements, and therefore cloud-based customized service selection approaches are scheduled and applied to fulfil users' demands. The cloud model offers its customer the cloud resources in the form of infrastructures, platforms, and applications through a pay-per-usage basis (Ibrahim et al., 2011), which enormously cuts the upfront cost of buying the cloud resources. In the traditional computing model, the cloud user has to arrange enormous upfront costs to buy and use resources. In the cloud, the customer is not concerned about the initial investment in purchasing computing hardware, software, licenses, etc. before starting any business. The cloud user instead has the facility to access the required shared computing resources and pay only for the consumed resources. Several cloud service providers offer their cloud services to their customers over the internet. Some of the popular cloud service providers are cisco Webex, salesforce, google. Code, force.com, cloud works, amazon web services, etc.

Cloud technology has played an important role in dispensing its services through the internet. The computing model facilitates computing resources' to access them whenever required (Buyya et al., 2009). Each cloud service has a quality of service (QoS) attributes (Ding et al., 2014) that distinguish one service from another. To select the appropriate cloud service, the user should consider these QoS attributes. The accuracy and promptness involved in selecting service from the enormous list of available services is a real task. It is also challenging to choose solely individual service from a vast list of available services that meet the user's requirement. Therefore, to obtain high performance in efficiency and accuracy, there is a necessity to develop a proficient decision system for choosing and ranking the services. The decision system will identify candidate services built on the user's needs and then allow the services (C.-T. Chen & Lin, 2010) (Hao et al., 2010). The optimal service is one of the ranked services which fits best into the user's customized requirements.

The QoS attributes are classified into qualitative or quantitative features (Mabrouk et al., 2009). The proposed NDS-ROS algorithm will use quantitative QoS attributes to select, rank, and compare the services until the optimal service is identified (Stojanovic et al., 2010) (L. Chen et al., 2011). The assessment and evaluation of numerous cloud services are possible only by measuring the services' quality attributes. In the traditional approaches, the QoS-based service selection and ranking are not considered essential, though the QoS attributes dependency in cloud service selection can not be ignored. Therefore recently, QoS-based service selection and ranking have taken massive attention recently in research due to vast business immigration on the cloud.

Real-life engineering problems shows that the user requirements, most of the time, be fulfilled through multiple objectives (Dhiman & Kumar, 2018). At the time of presenting the cloud services, these objectives are considered by the provider. Therefore a cloud service selection and ranking are formulated as a multi-objective optimization problem (Ding et al., 2017). The optimal solution considers the trade-off between multiple different objectives specified by the user (Jannat et al., 2010). The existing work on multi-objective optimization problems mounts the numerous objective problems into a single-objective optimization problem and then finds the best solution using traditional techniques (Zeng et al., 2004). The single-objective optimization problem find the best solution, and this solution has considered priority objective function to identify the service which best suits the objective (Knowles et al., 2001).

The Evolutionary algorithms (EAs) (Van Veldhuizen & Lamont, 2000) use a heuristic approach (Gabbani & Magazine, 1986) to find optimal solutions in a multi-objective problem. The critical aspect of the EAs approach deals with discovering the list of all the potential solutions that meet the multi-objective criteria. The genetic algorithms are the common EAs, which includes various steps of searching, selecting, mutation, and recombinations of available solutions (Sardinas et al., 2006). The vital step in the evolutionary algorithm lies in choosing the solution in the next generation. Therefore, it is essential to identify some criteria that will decide if one solution is better. Generally, solutions that have better QoS values than others are considered superior to the rest. The non-dominated sorting strategy (Peng et al., 2010) is the primary technology behind the evolutionary approach for comparing two solutions. Two solutions are said to be a non-dominated solution (Kukkonen & Deb, 2006) when one solution is found better in at least one of the objectives functions from the other solution. The dominant solution (Özlen & Azizoğlu, 2009) has one solution better than the other solution's objectives.

The software as a service (SaaS) model (Coello et al., 2007) is the most service model in the market; therefore, research emphasis on the selection and ranking of SaaS cloud (Benlian et al., 2009)(Wei-Wen, 2011)(Seethamraju, 2015). The SaaS model helps the user quickly deploy SaaS applications and start using it, thereby eliminating the upfront cost of purchase, installations, and the resources' ongoing maintenance cost. The users of SaaS applications also can download the application and use it at their convenience. Thus the user of SaaS always has a competitive edge and can speed up the business benefits faster. Therefore SaaS is the most widely used service model compared to other services models. Thus, SaaS's demand has increased in recent years and gained popularity among users, making a customized service selection approach in the SaaS cloud (Tsai & Sun, 2013).

The research aims to design and develop an efficient selection and ranking algorithm for the cloud by optimizing the QoS attributes for SaaS user's customized requirements. The overall idea behind our proposed approach is to decrease the methodological gap in the literature and improve the performance of the selection procedure involved in SaaS cloud selection (Cusumano, 2010). The performance parameter in the research work will reduce the complexity, the total number of comparisons required and, also the execution time taken to find optimal service. The remainder of this chapter elaborates on the problem statement, research objectives, and the main contribution of the research and thesis roadmap.

1.2 Problem Statement

The swift upsurge in the cloud services makes the service selection and ranking process strenuous for the customers as they have to choose from the diverse list of available services. The SaaS applications are gaining popularity due to the ease it offers to its customer in accessing the applications through the web interface. Several SaaS providers in the market offer diversity of services in terms of portfolios, wide pricing range, and different QoS attributes put a challenge for the customer to select SaaS cloud is very intricate and challenging (Hussain et al., 2020).

The researchers have proposed a few service selection approaches, but these are a generalized approach that fits in different cloud service models (Yadav et al., 2014). The current study shows that each service models have their own important QoS attributes which should be considered at the time of service selection and ranking. The generalized model will not help the customer opt for any specific cloud service model (IaaS, SaaS, or PaaS) (Kavis, 2014)because the QoS requirements vary from one service model to another. In case if the generalized selection model is used, then it will not produce an efficient result. Different users have different QoS attributes requirements, and therefore, it is challenging for the decision-makers to identify the appropriate services model that fits into their customized needs.

The service selection and ranking consider multiple objectives simultaneously for identifying and selecting appropriate services. Therefore, choosing the best possible services is essential, which considers all the user's goals. The existing techniques proposed in cloud service selection by assessing multiple objectives and evaluating the optimal solution are not sufficiently studied in the literature. The variety of cloud services leads to some research questions: How to identify the optimal service by considering multiple customer objectives. For example, the decision-maker will match the available services for numerous conflicting goals; it is not easy for the decision-maker to generalize the customer's necessities and find the service. For instance, one service's cost is low, but the response time is 30% high compared to other services. Similarly, the decision-maker might choose one service by considering its price and another for its response time.

Evolutionary algorithms are considered efficient to find an optimal solution, especially in optimization problems (Coello et al., 2007). The technique's efficiency lies in lowering the search space by reducing the number of comparisons required to find an optimal solution. The existing literature suggests a massive scope in reducing the search space for comparison by removing un-necessary and duplicate comparisons. The design and development of an efficient approach are required, which reduces the overall search space and provides a fast and accurate optimal service to the customer.

1.3 Research Objectives

The research contributes to cloud service selection and ranking by developing ”an efficient QoS oriented multi-objective ranking algorithm for service selection in SaaS cloud model.”

1.3.1 Sub-Objectives

1. Study and review of various cloud service selection and ranking approaches, framework, and algorithm to identify existing cloud service selection issues and ranking issues.
2. Development of QoS oriented multi-objective non-dominated sorting approach for selection and ranking SaaS cloud (NDS-ROS).
3. Implementation of the NDS-ROS using simulator- Jmetal.
4. Experiments were steered to show the competence of the NDS-ROS with three current algorithms.
5. The validation of the NDS-ROS approach by ANOVA using the SPSS tool.

1.4 Contribution of the Thesis

The research in the thesis considers the services selection and ranking. The thesis contributes to a new methodology for SaaS cloud service identification as per customized requirements. The proposed NDS-ROS algorithm will be efficient in reducing the search space by lowering the number of comparisons and execution time taken to find the optimal service as per the customized requirements of the customer. The research contributes to cloud service selection by developing an effective and scalable algorithm for addressing the QoS-based service selection for the multi-objective problems in the SaaS environment. Specifically, the thesis makes the following contributions.

1. The comprehensive study is done related to cloud service selection, multi-objective optimization problem, evolutionary algorithms, cloud service selection, and ranking approach. The research develops an outline with future research directions into cloud service selection based on the survey findings.
2. The QoS-based novel NDS-ROS algorithm is proposed for service selection and ranking. The NDS-ROS algorithm will meet the research objectives and overcome the research problem. Therefore, the main research contribution

lies in evolutionary computation by considering multiple objectives simultaneously and selecting the cloud's software (SaaS). The novice NDS-ROS includes four steps (i) filtration, (ii) sorting, (iii) ranking (iv) dominance comparisons. The NDS-ROS algorithm reduces the methodological gap of the literature and offers SaaS service according to user-customized requirements The NDS-ROS algorithm uses QoS attributes, facilitating optimal service delivery to its cloud customers responsively and efficiently. During the NDS-ROS algorithm development, the research introduced different parameters for comparing cloud services, which will improve the search space and help the user find customized optimal cloud service. The study provides a comprehensive evaluation of the NDS-ROS algorithm, which adds substantially to the understanding of using an evolutionary algorithm to address the QoS aware cloud service selection.

3. The research also incorporates various QoS attribute for the assessment of cloud services. Including the QoS attributes in research will measure service performance. The study utilizes a dataset from Kaggle for decision-making. The contribution is the design and development of the proposed approach in the form of pseudocode and implementing the simulation environment approach.

4. The proposed NDS-ROS approach is further compared with three existing methods to determine NDS-ROS's efficiency. The research evaluates the NDS-ROS approach is efficient on the following parameters, number of comparisons, execution time, and computational complexity.

5. The outcome of the proposed NDS-ROS is validated through ANOVA using the SPSS tool. The results favor the NDS-ROS approach in the number of comparisons.

The research outcome will be beneficial to the cloud customers, cloud providers, SaaS vendors involved in SaaS services. The NDS-ROS algorithm is helpful in a cloud selection system to provide efficient selection and ranking processes without compromising the performance.

1.5 Road Map of the Thesis

The thesis organized as follows:

Chapter 2 defines this research background and introduces basic cloud computing concepts, QoS qualities, service selection, multi-objective problems, and different approaches developed in the SaaS model for selection. Evolutionary algorithms and non-dominated sorting techniques proposed by authors have been discussed in detail. Comparing various methods based on a parameter such as their QoS values, publications, model/ framework used discussed in detail in tabular format.

Chapter 3 The NDS-ROS approach proposed cloud service selection architecture. The different stages involved in NDS-ROS are described, and pseudocode is written for the proposed system, and the dominance comparison rules are defined.

Chapter 4 defines the experimental environment, and tools such as JMetal and ANOVA discussed in detail. The performance parameters considered and the relevance of these parameters in the research are discussed in detail.

Chapter 5 describes the NDS-ROS algorithm's experimental implementation with three existing algorithms, and performance evaluated for the NDS-ROS on fixed objectives or fixed candidate services. The results were calculated for the comparison required and the execution time represented in tabular format and graphical representation.

Chapter 6 discusses the conclusion and future work according to the evaluation of the results achieved in chapter 5.

Chapter 2

Cloud Computing

The Cloud environment provides an ideal platform that allows users to access services distributed globally and provides the platform to accesses these services via the internet using cloud technology. The chapter talks about cloud concepts, cloud characteristics, deployment models (Savu, 2011), and service models (Gibson et al., 2012). The quality of services (QoS) attributes (Cao et al., 2009) and the prominence of these parameters supporting the service selection have discussed in detail. The existing literature and techniques proposed by researchers in cloud services thoroughly studied the set of pre-defined parameters. The comparative analysis of existing work helps identify open research issues in service selection and ranking.

Background of Cloud Computing

The cloud technology gaining is popularity in the market due to its features and the wide range of services presented by service providers. The cloud model eliminates planning requirements ahead of the resource provisioning and permits an enterprise to begin from small infrastructure and grow its resources at the time of service demand. Cloud computing is not a new technology; the concept is built on existing technologies that help run a business differently. The technologies used by the cloud, such as virtualization (Xing & Zhan, 2012) and utility computing (Nickolov et al., 2013) for the payment model on resource pricing perspective, where a customer will pay for the services used, that of grid computing (Hey & Trefethen, 2003). The outline of the cloud computing paradigm is shown in FIGURE 2.1.

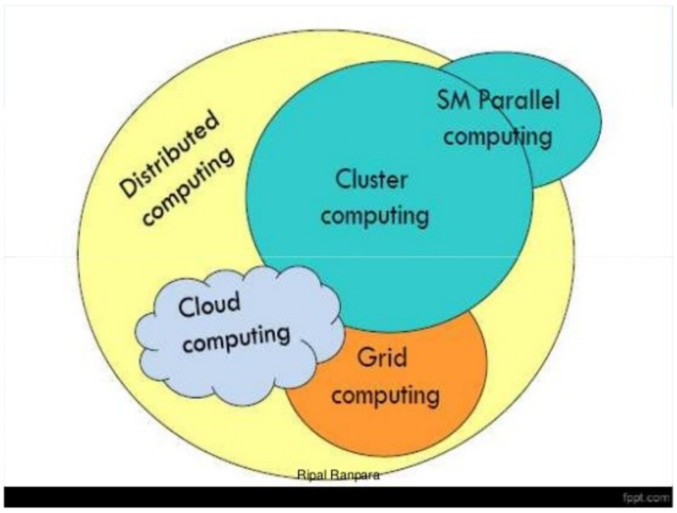


FIGURE 2.1: Outline of Cloud Computing Paradigm

The traditional business model engages the purchase of both hardware and software before beginning its day-to-day activities, thereby increasing the upfront cost of business. Cloud technology offers a customer-friendly start or shift to the cloud and pays the provider only for the services taken and used (Binnig et al., 2009). The Cloud model has gained popularity due to its features like up-front investment required, broad network access, highly scalable, resource pooling, rapid elasticity, easy access, low business risks, management costs (Lewis, 2010), etc. The increasing demand for cloud services among the customers shows growth in providers' numbers and their varied cloud services list. FIGURE 2.2 displays some service providers' prevalent services widespread among its customer.

FIGURE 2.2: Cloud Service Providers

The emergence of several cloud providers and their offered services in the market drove a high competition among the providers and challenges the cloud customers to identify the most suitable service from the massive list of available services. The National Institute of Standards and Technology (NIST) explains significant elements broadly used in the cloud community (Mell et al., 2011). A cloud model facilitates a user by offering an on-demand pool of services that can be rapidly increased or decreased as per user requirements. Cloud theory is grounded on two core technologies; the first is Service Oriented Architecture (SOA) (Erl, 1900), which divides and integrates small tasks known as services offered to the customers. A wide range of services is shortlisted and provided to the customer in a cloud-based SOA. The second important technology is virtualization (Lombardi & Di Pietro, 2011), which offers different virtual applications and machines. The computing devices and resources virtually represented also help the user carry out computational tasks without worrying about the hardware deployment.

2.1 Cloud Components

A Cloud computing environment works on three major elements: clients, distributed servers, datacenter, and all three components contributing to cloud technology in one way or another (Velte et al., 2009). The clients provide the platform through which the user cooperates with the cloud environment and is distinguished as mobile clients, thin clients, and thick clients (Dikaiakos et al., 2009). The datacenter is an assembly of servers and might be situated far from the clients. Distributed Servers (Chan & Tobagi, 2001) also participates in the cloud by providing access to the user for using the applications through the internet (Siegel & Perdue, 2012). FIGURE 2.3 depicts various components involved in a cloud environment.

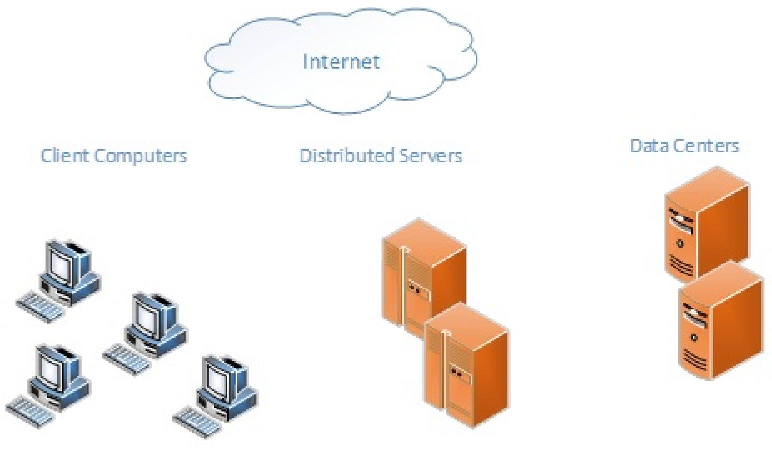


FIGURE 2.3: Cloud Components (Dikaiakos et al., 2009)

2.1.1 Characteristics of Cloud Computing

The definition of cloud includes architectures (Varia, 2008), security issues, deployment strategies, and characteristics (Gong et al., 2010), which are the backbone of cloud technology. The key features that have attracted both providers and consumers towards the cloud are discussed as below (Mell et al., 2011).

1. On-demand self-service: The objective of on-demand self-service (Malathi, 2011) is to reduce the delivery time with minimal service provider intervention. The cloud services such as network storage, CPU time, etc. can be taken instantaneously or for a particular time slot.

2. Broad network access: Customers can use the virtualized services for their applications and are made available to the customers through heterogeneous devices.

3. Resource Pooling (Wischik et al., 2008), The cloud services are offered and used by multiple customers simultaneously using a cloud multi-tenancy concept (Goyal et al., 2019). The resources are dynamically assigned and reassigned or taken back as per customer requirements.

4. Rapid elasticity: The cloud technology makes computing resources available to the customer as per their choice (Shawky & Ali, 2012). Customer's complete discretion is to decide the type of services they would like to take from the provider and its duration. There is no upfront cost or commitment from either side, and the customer can scale up or scale down resources as per their usage.

5. Measured services: The cloud provider creates the pool of virtual resources over the physical resources. Although the cloud customers are using resources through a pooled or shared approach, the cloud service provider can easily calculate the usage through its metering applications.

6. Effortless Maintenance: The servers are maintained easily with significantly less downtime. The regular update feature adds more efficiency in terms of compatibility and improved performance of cloud computing services.

7. Availability: The cloud competencies ensure that data and services will be available whenever required. The provider also guarantees that the data and services can be modified and extended as per customer need. The cloud service is accessible anytime and from anywhere.

8. . Automation Support: The cloud automatically analyses the user requirements and offers services to the user as per their requirements. The data and service users metered, monitored, and controlled to provide transparency to the provider and user.

9. Cost-effective: The cloud is an economic model as the customer does not have to spend money buying hardware and resources. The company can rent the hardware, software, and resources as per their requirements, which saves lots of their money from buying resources and later on maintenance.

10. Security: The cloud provides a trustworthy storage service and has the best security features attached. The data will not be hacked or lost, even if any server damages.

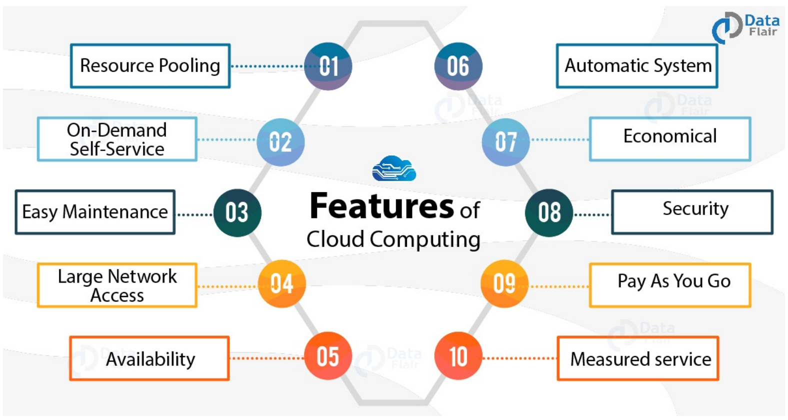


FIGURE 2.4: Cloud Computing Characteristics

2.2 Cloud Deployment Models

The selection of a deployment model (Savu, 2011) depends on that model's practice in the organization.

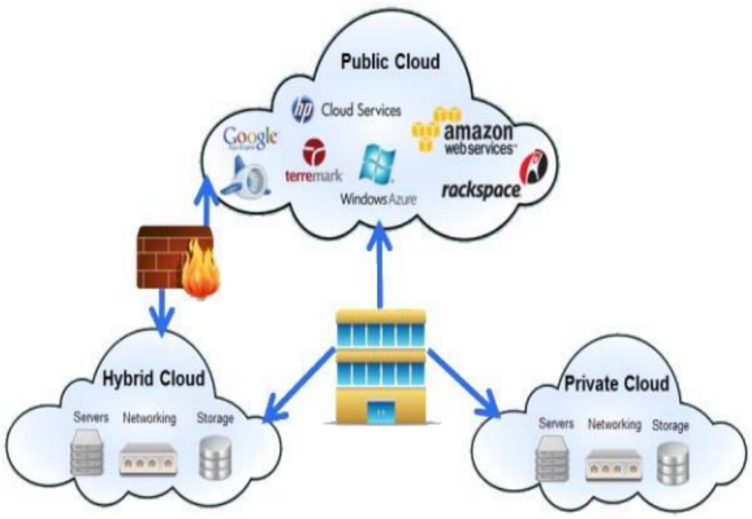


FIGURE 2.5: Cloud Deployment Model (Buyya et al., 2009)

NIST classifies the deployment models into Public Cloud, Private Cloud, Hybrid Cloud, and Community Cloud (Mell et al., 2011). The deployment models access the cloud resources from within the organization or outside the organizations and shown in FIGURE 2.5.

1. Public Clouds The most common way of deploying cloud services is through the public. The model allows any individual organizations to use these models. The resources are open for the people and retrieved at a low cost. The services' controls are with the service providers rather than the cloud user, and therefore security threats associated with the model.

2. Private Cloud The private cloud deployment model is more safe and secure because the individual or organization exclusively uses it. The resource availability and access control are within the network and protected by the firewall (Latif et al., 2014). A private cloud can be taken as per the need of the user and is costlier as compared to a public cloud

3. Community Cloud The community cloud includes both public and private clouds; the resources are shared and accessed among individuals with similar requirements. The community cloud provides a perfect balance between the resources, security, and privacy risk associated with the cloud.

4. Hybrid Cloud The hybrid cloud involves the grouping of two or more cloud deployment models. The model is beneficial for the customer who wants to deploy public cloud services and private cloud services. Therefore services that involve more security are executed in a private cloud, which does not have any security threats performed in a public cloud.

2.3 Cloud Service Models

The cloud offers a self-driven business model and provides various flexible services to satisfy the user's requests. The service model (Rani & Ranjan, 2014) offers appropriate and usage-based network access to configurable computing resources (Bohn et al., 2011) (Wang et al., 2011). The cloud services are categorized into three main types: Infrastructure as a Service (IaaS) (Bhardwaj et al., 2010), Platform as a Service (PaaS) (Beimborn et al., 2011), and Software as a Service (SaaS) (Dubey & Wagle, 2007) and is illustrated in FIGURE 2.6.

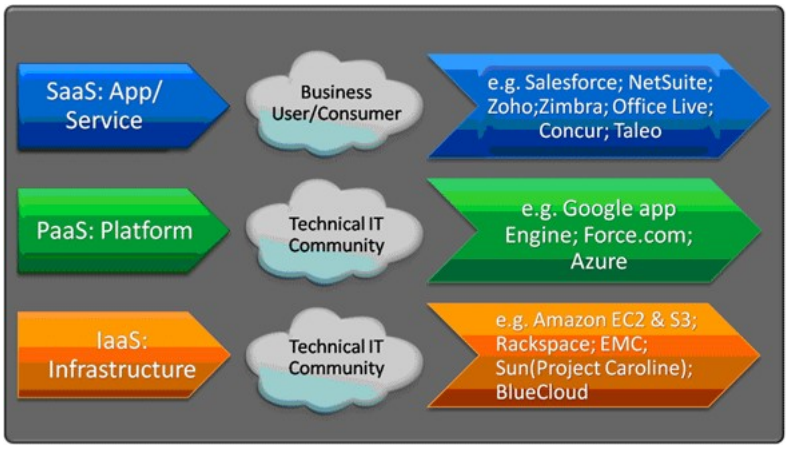

SaaS: App/ Service
Business User/Consumer
e.g. Salesforce; NetSuite; Zoho;Zimbra; Office Live; Concur; Taleo
PaaS: Platform
Technical IT Community
e.g. Google app Engine; Force.com; Azure
IaaS: Infrastructure
Technical IT Community
e.g. Amazon EC2 & S3; Rackspace; EMC; Sun(Project Caroline); BlueCloud

FIGURE 2.6: Cloud Service Delivery Models

1. Infrastructure as a Service (IaaS) The provider gives different types of hardware-related services (Gates III et al., 2011), such as virtual machines (Smith & Nair, 2005), networks, storage, processing to the user. In turn, the user can submit a task or run his applications on cloud infrastructure, where the user has limited capabilities to control the rented infrastructure. The significant computing resources are offered through a virtual metered basis. The Different IaaS cloud service providers in the market are Amazon EC2 (Ostermann et al., 2009), Rackspace are a few examples of IaaS.

2. Platform as a Service (PaaS) The providers offer the platform to the user to develop user-customized applications without worrying about operating systems, pre-defined tools, editor tools, programming environments, and other computing resources. Based on the application needs, the resources are scaled up and down automatically by the provider. The PaaS as a service model is a cost-effective, efficient model for developing customized applications. The Google App Engine (Zahariev, 2009), Microsoft Azure are a few PaaS examples.

3. Software as a Service (SaaS) The SaaS models are the highest delivered service model, and the service provider allows its users to access the existing applications on the cloud. The user uses present software or customizes applications without worrying about the hardware and other software products. The virtualized SaaS services can be accessed simultaneously by multiple users. Google Docs, etc. are a few examples of the SaaS model. The research aims to develop a proficient service selection and ranking system to optimizes the QoS and meets the customer requirements.

The complete cloud computing architecture is shown in FIGURE 2.7, describes the relationship among the deployment models, users, service models, and cloud providers.

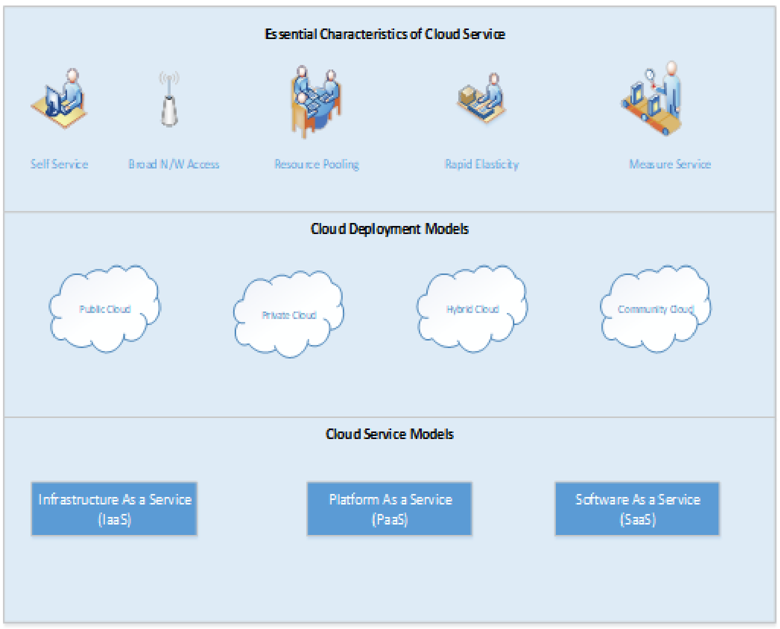


FIGURE 2.7: Relationship User, Service Models, Deployment Models and Service Providers (Luo et al., 2011)

2.4 Quality of Service

The QoS model (G. Chen et al., 2011) describes QoS attributes and their various cloud computing application requirements. The challenges faced by cloud applications is the identification of the right QoS and their management. Each quality attributes have their pre-defined property with a specific measurement metric to assess its value. The QoS attribute defines the service-related characteristics that provide insights into its performance, availability, reliability, etc. (Salama et al., 2013).

2.4.1 QoS Parameters

The associated QoS attributes of the cloud play an essential role in selecting services to meet user-customized requirements. The understanding of QoS parameters of the cloud service depends on identifying QoS parameters for cloud services. The most accepted models include the cloud service measurement index (SMI) (Arab, 2010). The SMI model has identified seven significant attributes and many sub-attributes.

TABLE 2.1: SMI framework for QoS Parameters.

Category	**Attributes**
Accountability	SLA verification
	Compliance Ease of doing business
	Providers certifications
Agility	Scalability
	Portability
	Elasticity
Assurance	Availability
	Reliability
	Fault Tolerance
Financials	Ongoing Cost
	Acquisition and Transition Cost
Performance	Service Response Time
	Functionality
	Interoperability
Security and Privacy	Access Control
	Data Privacy and Data Loss
	Data Integrity
Usability	Accessibility
	Learnability
	Suitability

The exponential growth of cloud service providers leads to publishing many similar services with diverse QoS. The identification of the nuclear service from a similar service group is an NP-hard optimization problem (Milan et al., 2017). The cloud services are deployed in various clouds geographically; therefore, network QoS parameters are essential such as network delay, and network reliability also plays a vital role in service selection. If the customer uses some cloud-based applications, then the mostly adopted QoS parameters (Dong et al., 2013) are response time and efficiency. Therefore, as per the user's requirement, QoS is essential for the selection process. FIGURE 2.8 shows the most adopted QoS parameters for different cloud service models.

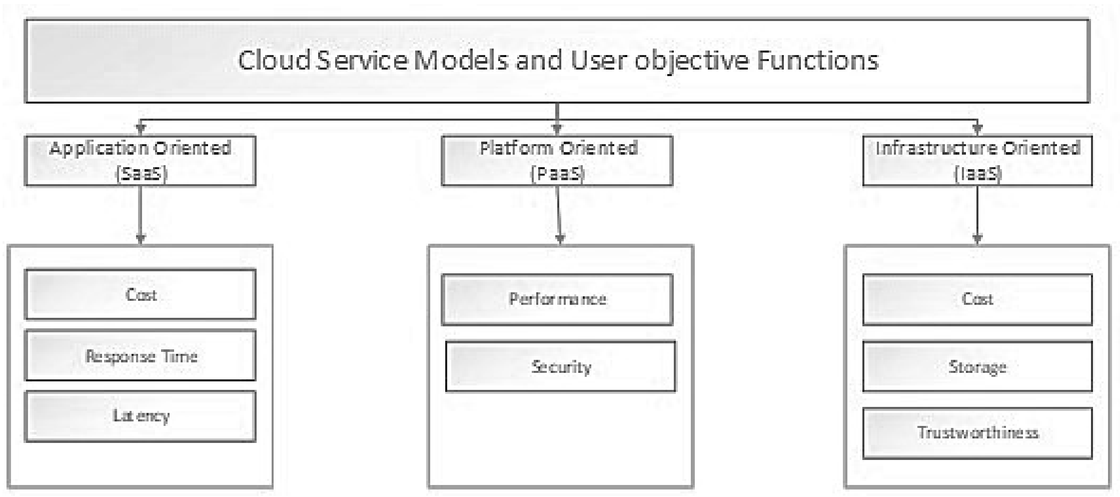

Cloud Service Models and User objective Functions
Application Oriented (SaaS)
Platform Oriented (PaaS)
Infrastructure Oriented (IaaS)
Cost
Response Time
Latency
Performance
Security
Cost
Storage
Trustworthiness

FIGURE 2.8: Relevant QoS for Cloud Service Model (Dong et al., 2013)

2.4.2 QoS Monitoring

QoS monitoring (Dong et al., 2013) is a mechanism to measure cloud services' QoS values. It is classified into three types depending upon who is counting the QoS parameters, i.e., at the client-side, the server-side, and third party side.

1. **Client-Side Monitoring-** The QoS measurement at the client-side rests on the user involvement with the service. The response time is a QoS metric measured at the client-side by evaluating the total time taken to receive the receiver's response.

2. **Server-Side Monitoring-** The QoS measurement at the server-side is done by accessing the server. The availability of service is an excellent example of server-side monitoring. The technique that is to monitor server-side QoS is Windows performance counters (WPC) (Arab, 2010) of the windows communication foundations (WCF) (Lin et al., 2007).

3. **Automation through third-party Monitoring-** The quality is measured by outsiders as per the requirement to evaluate the service's performance.

2.4.3 QoS Normalization

The QoS values of non-functional parameters have different dimensions, and to quantify them to a uniform distribution, the normalization (Raj & Sasipraba, 2012) is applied to original QoS values. As per the user's objective function requirements, the quality parameters are divided into positive and negative criteria. With the increase in positive measures, the objective function will increase, and with an increase in negative parameters, the objective function will decrease.

2.5 Cloud Service Selection

The cloud environment connects a pool of dynamically provisioned virtualized computers for offering different computing resources per the service agreements (Keller & Ludwig, 2003) signed by the providers and users. The diversity of the provider's services brings a challenge in the cloud service selection and ranking process. There are a set of pre-defined standards such as SOAP and WSDL (Ardagna & Pernici, 2005), which assist in service selection and allow an elastic way for applications to communicate over the internet.

Cloud services' spread increases the demand for a service discovery approach considering customized user requirements (Garg et al., 2011). The service selection issue has, therefore, attracted considerable attention for research in this area. Existing literature talks about different techniques proposed to help decision systems for assisting in service selection. To solve the above problem, researchers have worked a lot in designing processes that provide a common platform for showcasing the services and helping the user pick up the required services. Also, cloud ontologies (Moscato et al., 2011) developed for aiding a standard procedure of service specification and matchmaking.

2.5.1 Generalized Process Involved Cloud Service Selection

A generalized model defines the aim of cloud service selection, the role of people involved in the selection procedure, modelling various criteria of discussed services, and evaluating the requirements. The generalized model is shown in FIGURE 2.9 also describes how to choose the best service and the people involved in the service selection procedure.

1. **Selecting the best service-** The approaches and techniques use multi-criteria decision making (MCDM) (Lai et al., 1994) for service selection based on the customers' requirements. The choice builds on the priority level given to the criteria by the customers. Each criterion's priority is taken from the customer and using that, the best service selected, and therefore, the chosen service will meet the customer's requirements. The standard techniques for MCDM include Analytical Hierarchy Process (AHP), Analytical

Network Process ANP Görener (2012), weighted sum approach (Marler & Arora, 2010), TOPSIS (Lai et al., 1994), and fuzzy techniques for decision making (Hong & Choi, 2000).

2. **People in service selection-** The people involved in selection include the cloud providers, cloud consumers, cloud customers, brokers, and each one has a specific role in service selection.

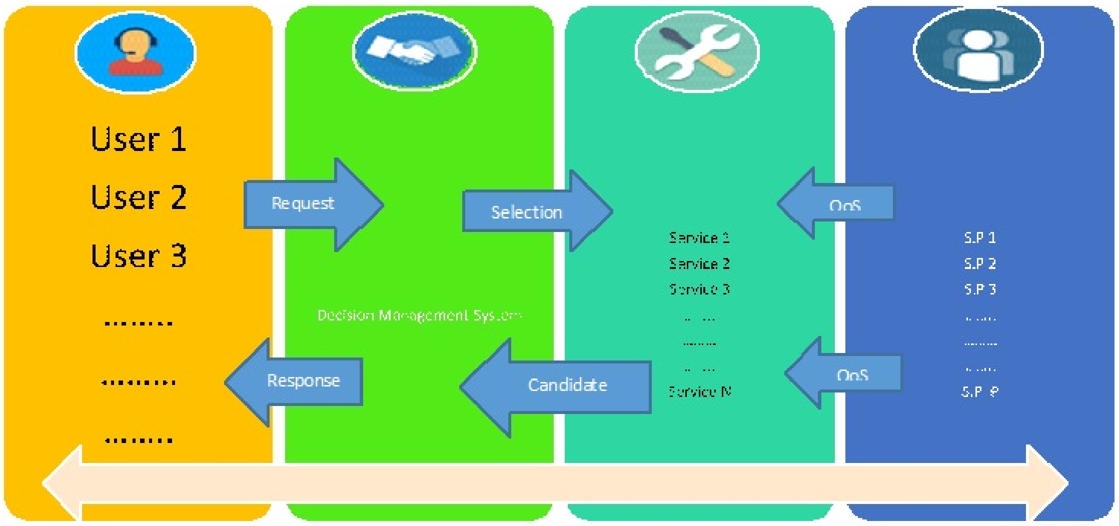


FIGURE 2.9: Generalized model for Cloud Service Selection

2.6 Traditional Techniques of Cloud Service Selection

The section discusses the existing techniques used for finding the service that fits in the requirements. The categorization of methodologies involved in the area of service selection is explored. The work by various researchers and their proposed approaches consider service assessment parameters, service choice techniques, the purpose of service selection, and service criterion evaluation are discussed in detail. Based on the service assortment process, the approach characterizes into six major groups: Multi-Criteria Decision Making (MCDM) (Mousavi-Nasab & Sotoudeh-Anvari, 2017) service selection approaches, trust models for service selection (Ali et al., 2005), fuzzy service selection (Lin et al., 2007), and broker service selection, QoS based service selectionv(Huang et al., 2009) and evolutionary algorithms (Coello et al., 2007). The different techniques, algorithms, methods used in the above mentioned approaches are discussed in detail.

2.6.1 MCDM approach for service selection

This area's decision-making strategy depends on identifying the services fit appropriately into the customer requirements' demand. The MCDM is the most used approach when the decision criteria and the alternatives are finite in number. The MCDM involves different methods for selecting, evaluating, and ranking services based on QoS values. The MCDM selection approach's challenge consists of evaluating each service's performance and for every single attribute as shown in Fig 2.10. The MCDM approaches use techniques like Analytic Hierarchy Process (Golden et al., 1989), Analytic Network Process (Saaty, 1988), TOPSIS, ELECTRE Methods (Figueira et al., 2005), and Simple additive weighting methods.

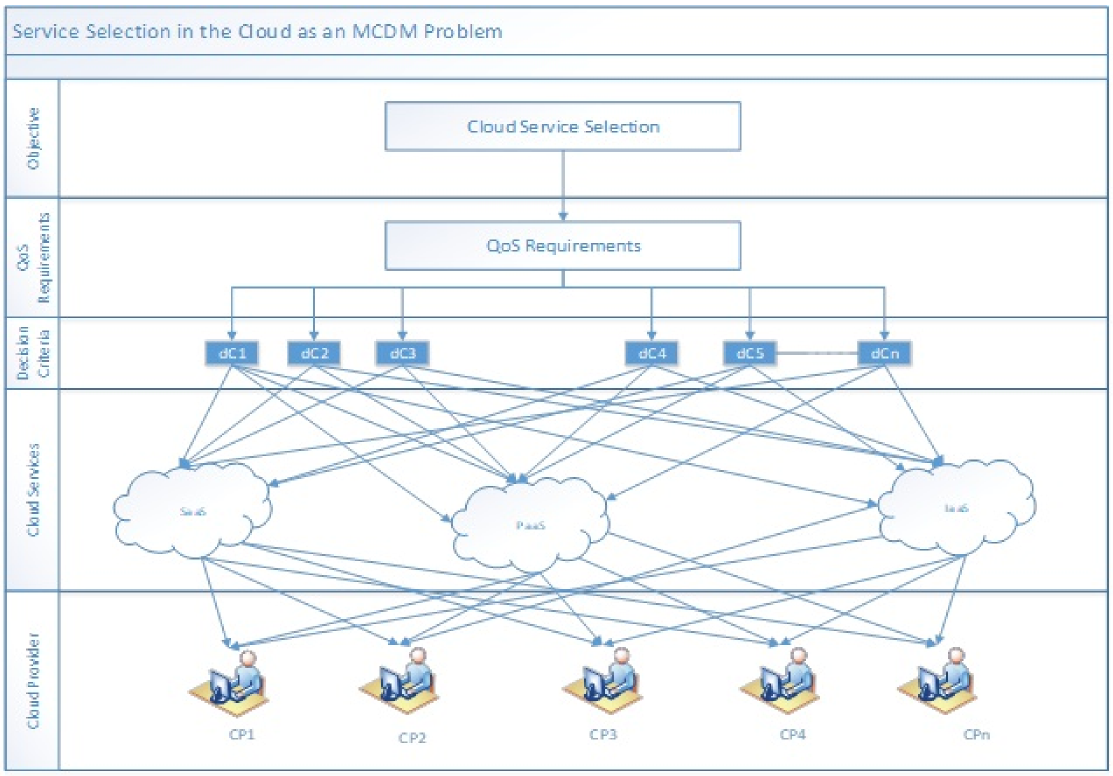


FIGURE 2.10: MCDM Model for Cloud Service Selection

AHP is a decision-making technique that helps the decision-maker to discover the best-suited alternative from a list of available options considering user requirements as a priority. An AHP ranking process works on decomposition, comparative judgment, and synthesis (Görener, 2012). The hierarchical designing of the framework in AHP helps the decision-maker systematically evaluate the various alternatives and compare them for finding the best option. FIGURE 2.11 shows the AHP hierarchy, aiming to achieve the goal based on different criteria 1, 2, 3, and 4.

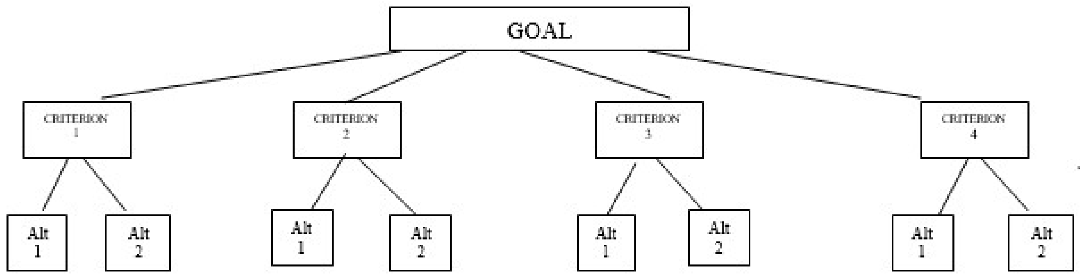


FIGURE 2.11: Analytic Hierarchical Process (AHP) (Görener, 2012)

The service selection and ranking techniques proposed by researchers using AHP are discussed in detail :

In (Godse & Mulik, 2009), Godse and Mulik suggested a SaaS service selection technique through AHP considering attributes and sub-attributes. The features, along with sub-attributes, are used to compute the individual performance of service. The services are pairwise compared gives the scoring to each attribute scoring, which helps rank the services.

In (Karim et al., 2013), Raed and co-authors discovered the selection by considering the QoS of services and assigns ranks to IaaS and PaaS services according to the user requirements. The author proposed the AHP method, which will match the QoS for the offered services to that of QoS requirement by the customer, the services that matched taken as optimal service with QoS guarantees.

Gonclaves and co-authors (Gonçalves Junior et al., 2015) consider an architectural design in cloud selection. The author proposed an AHP based cloud selection that relies on non-functional parameters, including efficiency, cost, and scalability based on the multi-criteria optimization method. The authors implement the proposed approach on word-press and deploy it in the Amazon cloud.

In (Menzel & Ranjan, 2012), Menzel and co-authors proposed the Multi-Criteria Comparison (MC2) framework for IaaS selection. The framework helps the user to identify which infrastructure would be best suited for their requirements. The (MC2)2 distinguishes the infrastructure alternatives regarding cost-effectiveness, advantages, prospects, and risk. The framework (MC2)2 is used in various decision-making scenarios in information technology infrastructures. Menzel and co-authors (Menzel et al., 2013) further proposed a framework called CloudGenius to help dynamic decision process and assist him in choosing cloud infrastructure. CloudGenius utilize an AHP technique to automate the decision-making process based on QoS attributes.

In (Sun et al., 2016), Sun and co-authors presented a new framework built on a fuzzy decision approach, which improves the service selection process. Fuzzy Ontology-based model developed to reduce the uncertain relations among various objects defined in the database, and an AHP based multi-criteria technique helps ranking cloud services.

Boutkhoum in (Boutkhoum et al., 2016) suggested a methodology using the Fuzzy Analytic Hierarchy Process (FAHP) and Preference Ranking Organization Method for Enrichment Evaluations (PROMETHEE) to help in analyzing and ranking the best cloud which removes the issues related to the big data and cloud technology. The Fuzzy AHP technique helps assign weights for evaluating the criteria, while the PROMETHEE method will evaluate different value given and rank the decision alternatives.

Khowfa and Silasai in(Khowfa & Silasai, 2017) proposed a novel approach using the Association Rules technique and the AHP method. Association Rules are utilized in data mining for finding out relations among various items in the database. AHP compares and ranks the services; the combination of association technique with AHP will evaluate and select appropriate services.

Hwang and Yoon developed the TOPSIS technique in 1981 (Hwang & Yoon, 1981), which considers the ideal solutions and finds the best alternative near the perfect result and farthest from the non-ideal outcome (Mohammadshahi, 2013). Vector Normalization is used to normalize the decision matrix, and then perfect and non-ideal solutions are recognized, keeping in view the normalized decision matrix (Zavadskas et al., 2006) for cloud service selection. The TOPSIS method does not consider the attributes' importance but only finds the ideal solution and the non-ideal solution. In (Lo et al., 2010), Lo and co-authors discussed the fuzzy TOPSIS method in which fuzzy triangular numbers parameterize pre-determined linguistic variables for calculating the weights of different measures, and the rating of alternative service is calculated, and best service is identified.

Rai and Kumar in (Rai & Kumar, 2016) use the traditional service selection method based on the criteria weights according to user demand. Different data centers are used to provide computing environments, referred to as the cloud service providers, and are responsible for publishing services stored in the database. The broker is responsible for identifying appropriate services according to users' requirements.

The MCDM approach used in the study is the outranking approach (De Boer et al., 1998). Several versions of ELECTRE methods were released, including ELECTRE I –VI, ELECTRE TRI (Mousseau & Slowinski, 1998). There are two sets of parameters in ELECTRE one is the coefficient, and the other is veto fresh (Mohammadshahi, 2013). ELECTRE performs a pairwise evaluation between alternatives and eliminates alternatives that less fit the criteria, leaving a small set of alternatives. The ELECTRE technique can be useful for both subjective and objective parameters. Concordance and discordance indices are applied (Hiessl et al., 1985) to the partial raking on the set of alternatives left after elimination.

In (Yazdani-Chamzini et al., 2013), Abdolreza Yazdani and co-authors used the ELECTRE technique for identifying the risk components in the tunnelling project. The research happens in two phases; in phase one, all the potential risky parameters are defined, and in step two, an order is established among them. As the tunnelling system is complicated and involves many uncertainties; therefore, the Fuzzy ELECTRE (Sevkli, 2010) approach is taken. Parameters such as damages, accidents of machinery, unsafe working conditions put on priority, and at high risk, and delays in project completion and equipment failure are ranked lower.

Silas and co-authors (Silas et al., 2012) offered a middleware for selection known as SSM_EC. The user preference related to QoS parameters and service description information collected from the provider, the concordance index, and the discordance index evaluated along with the credibility degree. The concordance index will give the truthfulness result according to the criteria set to calculate the candidates; the discordance will provide the QoS with the difference between alternatives. Finally, the credibility is calculated by aggregating the concordance and discordance index.

Chou and co-authors (Chou et al., 2008) gave novel fuzzy decision-making (FMADM) considering multiple attributes. The fuzzy process, in coordination with the Simple Additive Weighting (SAW) method, is known as (FSAWS), which involves the qualitative and quantitative attributes to identify the facility location's problem in group judgment criteria. The FSAWS gives the final score of each alternative after evaluating the scores derived from different decision-makers.

In (Saripalli & Pingali, 2011), Saripalli and Pingali use the SAW method to rank alternatives by the values assigned to them and find the best option. The author presented a MADMAC framework for the adoption of cloud services. The framework has three functions, which will act as the decision areas known as cloud switch, cloud type, and vendor choice. Upadhyay in (Upadhyay, 2017) talked about various challenges faced at evaluating cloud service performance. The paper also discussed cloud computing concepts along with the significance of QoS in cloud computing. The framework proposed to assess the services built on QoS parameters.

2.6.2 Trust Model for Cloud Service Selection

The service providers offer numerous services with different performance parameters, keeping customers confused about whether the provider will deliver the promised services. The cloud user trusts that whatever the cloud provider has mentioned in SLA will be given to him without fail. Therefore the main challenge is the selection of trustworthy cloud providers. Several models studied in the literature show a different approach proposed in this area, which helps select the best services from a reliable cloud provider. In (Pan et al., 2015), Pan and co-authors suggested a service selection model known as a trust-enhanced similarity model, which will improve trust between the provider and the consumer. This model will combine QoS of the cloud service, fills all the absent QoS, and finally assign rank the cloud services, eventually building trust in the cloud users. The cloud service selection model is classified into four parts. In the first part, the author proposes a trust modelling method in which a mutual trust degree is built between two users for general cases and is assumed to be equivalent. The second part stores experience user feedback, and numerical values are assigned based on their experience of QoS values. The third part includes utilizing the trust-enhanced similarity model to identify all the similar trusted neighbors and, finally, rank the service. After experimental analysis, this model is found better than other approaches in cloud service selection and order of cloud services.

Josang and co-authors (Jøsang et al., 2007) presented trust from a different perspective. The author talked about the trust-based selection in cloud computing as the trustworthiness between cloud providers and cloud users, i.e., both the parties can provide some rating to each other to derive trust and reputation. Various trust-based approaches discussed in the paper have taken both the qualitative and quantitative attributes in drawing out the conclusion of trust and reputation (Pawar et al., 2012) between the cloud provider and cloud consumer. Qualitative parameters evaluated using user feedback, expert judgment, and Quantitative parameters evaluated through some QoS performance evaluation approach.

Wu and co-authors (Wei-Wen, 2011) talked about reliability as an essential parameter for ensuring trust. The novel unfair rating filtering method proposed for revising the already existing reputation system. The proposed method filters all the unjust and unimportant ratings, thereby increasing the method's overall performance and assisting the service providers and consumers in opting for appropriate service. The proposed method shows an improved reputation revision system.

In (Ghosh et al., 2014), Ghosh and co-authors recommended a SELCSP trust framework for computing the interaction risk between the provider and consumer. The author discussed the importance of service level agreement (SLA) to ensure guaranteed service quality and ensure the cloud provider's reliability. The SELCSP framework includes trustworthiness, reliability of estimating risk involved in interaction among the provider and the consumer.

Hu Ma and co-authors (Huang et al., 2009) gave a novel cloud service interval neutrosophic set (CINS) using a time-aware approach (M. Liu et al., 2017) to evaluate the trustworthiness in service selection. The time-aware problem consists of multiple criteria for decision problems, and then the solution is considered, and the proposed CINS method is used in ranking cloud services. The experimental evaluation of CINS done using real data sets and the efficiency assessed for the proposed approach.

Supriya and co-authors (Supriya et al., 2016) discuss the importance of the trust parameter among the cloud consumer and provider. The growth in cloud services poses a dare in front of the consumer to select trustworthy providers. The author discussed various multi-criteria decision-making techniques to help providers to rank services. Combining the fuzzy approach and the analytic method (Talja, 1999) proposed evaluating the trust parameter in identifying from different cloud providers. N. Sasikaladevi, in (Sasikaladevi, 2016), talked about how trust plays a vital role in service selection. The author raises his concern regarding the trustworthy service for cloud service composition, which is not a simple task but includes lots of complexity. The author recommended a service trust estimation technique based on the Beta distribution in which a trust-based cloud service selection structure. The experimental outcomes prove that the offered framework performs enhanced on execution time and optimality value than other approaches.

Li and co-authors (X. Li et al., 2016) proposed a trust-based system for cloud service selection. The method includes the chosen services. These services are also delivered by determining the service's trust value, satisfying customer requirements by meeting safety parameters, and mark trust-based decisions to attain service controllability. Instance results show that the scheme is effective and can promise service selection under a safe and controlled scenario.

2.6.3 Fuzzy- Based Service Selection

Achar and Thilagam in (Achar & Thilagam, 2014) carried out the cloud service selection to support three steps. The first step is the identification of the criteria according to Service Measurement Index (SMI) (Siegel & Perdue, 2012), the second step is the determination of the priority of the involved measures, the last stage involves the TOPSIS technique (Mahmoodzadeh et al., 2007) for ranking alternative services. The result of the study evaluated using the CloudSim toolkit (Calheiros et al., 2011).

In (Srivastava & Sorenson, 2010), Srivastava and Sorenson discussed all the related literature about QoS and proposed a technique that compares the user ratings and real QoS performance. The method used in the process is the mid-level splitting method. The author in the research only took subjective parameters for the study. According to Hypothetical Equivalents and Inequivalents (HEIM) methods, the weights are assigned to the customer preference attributes. In (Pandey & Daniel, 2017), Pandey and Daniel proposed a cost-based framework named QoCS, which helps identify cloud environment services. The framework is based on trustworthiness and uses fuzzy logic, which allows the user to analyze the cloud services on multidimensional perspectives. Numerous QoS parameters are used in which cost is one of the main criteria for service selection. Experimental results also show a better performance of the framework under various cost constraints.

Sun and co-authors (Sun et al., 2016) presented a fuzzy-based decision framework for improving the overall service selection approach. A fuzzy ontology (Tho et al., 2006) models to match the services stored in the database and built the relationship among the objects. The novel Analytic hierarchy process approach is proposed to calculate the semantic similarity among parameters. The framework uses an MCDM technique for ranking services; the experimental outcomes show that the suggested method is efficient.

Kumar and co-authors (R. R. Kumar et al., 2017) suggested and designed a novel approach for the service selection using the AHP (Lai et al., 1994) and fuzzyTOPSIS (Saaty & Vargas, 2013). The method uses the weights criteria for comparing solutions pairwise, and the TOPSIS technique assigns the rank to the solutions.

The model considers the non-functional QoS requirements for selecting appropriate service and for evaluating each criteria weights. Tajvidi and co-authors (Tajvidi et al., 2014) discussed the cloud service selection issue due to several providers and service lists, several selection norms, and customer inclinations. The new fuzzy logic framework proposed considers individual QoS criteria of customers for cloud service selection. The framework also includes the essential services with their QoS data, monitors services, evaluates customers' feedback, and takes information from certified cloud providers. Hussain and co-authors (Hussain et al., 2020) talk about numerous factors considered for making a decision and removing the uncertainty in the whole process of making a decision. The complexity involved in the selection procedure's existing approaches makes the overall process challenging and less trustworthy. The paper proposed the novel framework to make room for cloud Service Selection. The ranking system under a fuzzy environment enhances the procedure of cloud service selection.

2.6.4 Broker-Based Service Selection

The cloud service provider available in the market assures that the best cloud service will be provided. Selecting an appropriate service that can meet the user requirement involves lots of decision-making skills. The emergence of cloud service brokers resolves the issue of identifying services efficiently. A broker (Guzek et al., 2015) gathers information about providers' offered list of services by understanding the customers' requirements, intending to find the best match. The cloud broker is responsible for coordinating with CSPs and the customer by ensuring that SLA-based services are maintained. The involvement of CSBs is assisting both the parties by building the trust that appropriate services provided, which match customers' needs. The Cloud Broker also manages cloud services' performance, delivery, and exchanges relationships between cloud providers and cloud consumers.

Aazam and Huh (Aazam & Huh, 2017) predict user behavior based on relinquish probability by recommending a broker who ensures the ease of selecting an appropriate service that meets the requested services. The study also includes a refund mechanism if the customer is not happy with the services or wants to shift to another provider.

Kanagasaba and Rajaraman in (Kanagasabai et al., 2012) talk about the growing demand for cloud technology adoption and services. The author proposed a framework for SaaS provisioning, which depends on SaaS providers and customers' cloud brokerage. The CSB help consumers to select the SaaS provider that understands his requirements and fulfil them. The CSB responsibility is to match the QoS offering with the needs and also ranks the services. Lucas-Simarro and co-authors (Lucas-Simarro et al., 2013) talk about the features and other essential components of cloud brokerage architecture. The author identifies the cloud broker administrator and the cloud broker user as prominent leaders in the decision-making process. The cloud broker's role is to identify the cloud provider's complete details regarding the services and the accounting information. The broker administrator decides on the available information provided by the broker. Annette and Banu in (Annette & Banu, 2015) proposed a service broker framework to select and provide the cloud-based render farm and converse the 3D Animation industry cloud to represent the 3D images. The cloud-based render farms help in scaling up and down as per the demand. One of the vital challenges is that the 3D studios face used for comparing and selecting the cloud-based render farm service provider. The provider will consider the quality of service QoS requirements and ensure that the condition is met. The CSB helps them identify the service provider based on the QoS. The CSB also simplifies the service level agreement (SLA) negotiation and third-party monitoring services. Sundareswaran and co-authors (Sundareswaran et al., 2012) discuss the challenges faced in selecting the appropriate service from the vast pool of offered services. Identifying cloud services is a time-consuming process because customers have to gather all the crucial evidence and then analyze all service providers to decide. As multiple customers have similar requirements, the same computation is repeatedly required. The author proposed a novel brokerage-based architecture that uses a unique indexing technique for managing the information and ranking potential service providers.

Wu and co-authors (Wu et al., 2014) proposed the QoS-based service composition to meet the customers' customized requirements at the service composition time. The research focused on understanding individual users' demand and applying a QoS- strategy replaces the traditional service composition strategy. The service broker facilitates implementing an on-demand system by purchasing several service instances from the providers and then using these instances to provide the set of composite services with different QoS parameters, which will vary from consumer to consumer.

Prasad and co-authors (Prasad et al., 2016) suggested a cloud-based algorithm, CLOUD-CABOB, to solve the optimization problem. The user requirements are taken in advance, and vendors also submit their offers for the services, including price, QoS for services, and other sets of resources. The SMIcloud framework [169] substantially creates vigorous competition among providers. The services that fit in the user requirements satisfies are taken further for service level agreement (SLA) and Sundareswaran and co-authors (Sundareswaran et al., 2012) offered a novel indexing technique using cloud brokerage architecture. The indexing will assist in supervising and dealing with the available information of the service provider. B+ tree method is used as an indexing factor to understand the provider list and encodes the information about all the user-related services requirements. The indexing architecture helps store the available cloud service properties and helps the cloud user match and facilitate fast information retrieval. A greedy algorithm is also designed based on the CSP index, which will rank the various service providers and improve cloud service selection. Multiple parameters and criteria are taken to mend the parameters, such as reliability and availability.

2.6.5 QoS Attributes for Service Selection

The cloud service selection and ranking are time-consuming and costly because the services filtered from a massive list of available services. The quantitative and qualitative assessments are done to find the optimal service. The QoS is essential in inspecting, evaluating, and selecting the service as necessary in the searching process. The QoS criteria, along with the weights, are used for ranking. The cloud vendors use their experience and QoS attributes to find the optimal service.

Saurabh Kumar Garg and co-authors (Garg et al., 2013) recommend a ranking framework for finding the services that consider a method to find and measure the service quality and rank the cloud services. The proposed framework will help analyze the benefits and generate healthy competition in the process of service selection. The SLA requirements are fulfilled through the metrics designed at various dimensions to every single attribute for each provider. Mojtaba Khezrian and co-authors (Khezrian et al., 2012) applied an AHP along with VIKOR technique to propose a hybrid approach for assisting the web service selection process. The AHP technique will estimate each criterion mentioned by the customer, and VIKOR applied to identify and rank the suitable candidate services. The proposed approach will make use of four different QoS criteria and five services. The result shows that this technique is helpful and selects the best service.

Dinesh Kumar and co-authors (R. D. Kumar & Zayaraz, 2011) recommend a model for web service selection based on using the AHP technique and QoS parameters. The model assists in selecting the most appropriate service instance using the QoS. The broker in this approach is the QoS manager, the middle person between providers and customers Dou Wanchun and co-authors (Wanchun et al., 2011) discusses that the service selection depends on service selection's personal preferences in qualitative attributes. The novel approach based on QoS- Aware Service Evaluation is proposed. The model inspects a service and assessment technique to encourage and evaluate the service and later get shared among the users. The AHP technique is used to deploy the QoS model and convert user preference by prioritizing the best service.

A Kumar N. and co-authors (N. Kumar & Agarwal, 2014) presented the framework by considering QoS for the service selection procedure. The framework assists the user in finding the appropriate service provider from the repository. The AHP approach understands the user's requirements and multiple criteria decision-making using QoS that smoothens the overall selection process. The framework shows that the results help users quickly select the best choice based on their tailored user needs.

2.7 Cloud Service Selection and Ranking using Evolutionary Algorithm

Evolutionary algorithms are considered an appropriate method for multi-objective optimization (MOOP). These algorithms are used in many real-world challenges to find an optimal solution that can be solved using two methodologies. Firstly, the MOOP gets distributed to a set of different single-objective problems (SOP) by considering the priority of the user and then solving the issue as a single-objective problem. Secondly, all the objectives are taken concurrently, and the optimal solution is assessed. A non-dominated sorting approach is the primary technology behind the evolutionary algorithm. It is beneficial to compare various solutions and find optimal solutions. Therefore, the research will consider a non-dominated sorting-based approach to find an optimal solution.

2.7.1 Optimization Problem

The optimization problem is a computational problem, aiming to identify the feasible solution from the available solutions list. The possible solution must consider maximization/minimization objective functions (Q. Zhang & Li, 2007) for finding optimal service. The user required objectives to assist in comparing the different choices for determining the "optimal solution." Therefore, optimization is a quantitative analysis to find optima using techniques, methods, procedures, and algorithms. Optimization problems are visible in real-life applications and find solutions to those problems, various modelling techniques, frameworks, and algorithms identifying optimal solutions. The optimization problem is shown in equations 2.1 and 2.2.

Mathematically, we can represent the optimization problem as(Huang et al., 2009).

$$\textbf{Optimize} \quad y = f(x_1, x_2, ...x_p.....x_n) \tag{2.1}$$

$$\textbf{Subjected to} \quad gj(x_1, x_2, ...x_p.....x_n) \begin{pmatrix} \leq \\ \geq \\ = \end{pmatrix} bj \tag{2.2}$$

Where j = 1 to k.

Equation 2.1 and 2.2 represents the MOOP mathematically. The xp variables, lies in the set of $\{x_p \mid x_1, x_2, ..., x_n\}$, represented by decision variables y. represents the $f(x_1, x_2, ..., x_n)$. The problem will decide whether the objective functions are to be maximized or minimized. Equation 2.2 shows the constraint which represented in any form of $(\leq, =, \geq)$ relationship.

Different reasons that increase the multi-objective problem's complexity include the multiple decision variables considered in a single situation and their complex relationships. The constraints handled at the time of the decision problem and techniques can identify the feasible solution.

2.7.1.1 Constrained Vs. Unconstrained Optimization

The mathematical techniques for solving optimization problems depend on particular criterion and constraint functions, but some straightforward situation is an unconstrained optimization problem.

1. **Constrained Optimisation-** Constrained optimization has some of the constraints imposed on the attributes. In general, all the constraints specified should be considered as they define the dependencies among variables and parameters of the given problem . The optimization problem express equality and inequality expressions as constraints and is shown mathematically in equations 2.3 and 2.4.

   $$gi(x) \leq 0 \tag{2.3}$$

   Where i = 1,2,3,...,n

   or

   equalities:

   $$hj(x) \leq 0 \tag{2.4}$$

   Where j = 1,2,3,...,q

2. **Unconstrained Optimisation-** The optimization problems having unconstrained minimization/minimization objective function. Many of the practical applications consist of unconstrained optimization. Some of the research proposed that constrained optimization is replaceable by a penalty function in the objective function. The constrained problems are considered when the applications have an explicit constraint on the variables.

2.7.1.2 Single objective optimization

The real-world applications involve different objectives to be considered in making a decision and to meet the QoS, such as minimizing risks and cost and maximizing reliability and security. A single-objective optimization problem finds the best solution using two ways. Firstly the priority related to objective function (either minimum or maximum) value is taken. Secondly, the different objectives are combined to form a single goal, and then the user requirements are integrated as an SOP. Therefore the best solution is recognized based on the priority of the objective function.

A single-objective algorithm finds the best solution considering a specific criterion such as execution time, cost, energy consumption, or power dissipation metrics. Also, if there are multiple criteria, then they are combined into a priority objective based on cost function as a weighted sum of the normalized costs associated with each of the metrics as given in equation (2.5) and (2.6).

Generally, a single-objective optimization problem is defines as:

$$\textbf{optimize} \quad x = [x_1, x_2, \ldots, x_n]^T \tag{2.5}$$

$$\textbf{subject to} \quad gi(x) \leq 0, \quad hj(x) = 0 \tag{2.6}$$

Where i= 1. . . . . . . m and i= 1. . . . . p

where gi(x) is the ith inequality constraint and equality constraints is hj(x).

2.7.1.3 Multi-objective optimization

The Multiobjective Optimization Problem (MOOP) involves multiple objectives that the decision-maker should consider when selecting services. These objectives are often conflicting in nature. Therefore an optimized solution is one that thinks all objective functions before making any decision. A list of feasible solutions is identified in a multiobjective optimization problem, which considers all the conflicting objectives. The collaboration among conflicting objectives provides agreed-upon solutions, known as the Pareto-optimal solutions. In MOOP, to identify the optimal service, the decision-maker has to make the pairwise comparison by considering all the objectives, and this process is known as Pareto domination. Two solutions are non-dominating to each other when it is impossible to improve one solution on some objective without making other objectives worse. However, there is a need to identify feasible solutions within the Pareto-optimal range to certify that an acceptable solution is selected. The mathematical representation of (MOOP) is defined below shown in equation (2.7) and (2.8).

$$\textbf{minimizing/maximizing} \quad f(x) = (x_1, x_2, x_3 \ldots, fk(x)) \tag{2.7}$$

$$\textbf{subject to} \quad gi(x) \leq 0hj(x) = 0 \tag{2.8}$$

Where i = 1,2.3....,m, and j = 1,2,3,...,p x ∈ Ω. and f(x) represents the n-dimensional decision variable for $x = (x_1, ..., x_n)$ from search space Ω. The $gi(x) \leq$

0 and hj (x)=0 shows the constraints and Ω contains possible solution to satisfy objectives. Single-objective optimization identifies a single optimal solution that satisfies the user's priority objective and involves aggregating different objectives combined into one objective function and find the best solution. In multi-objective optimization, the goals are expressed as constraints and to attain the respective objective functions. Several runs applied to obtain solutions corresponding to different satisfaction of conditions. In a multiobjective problem, the methodologies involved mainly consider multiple alternatives simultaneously and find the optimal solution. Therefore in single-objective functions, all the defined objectives to be measured as a single function, and in multi-objective a trade-off curve considering multiple objectives are considered where the result is the optimal service.

2.7.2 Cloud Service Selection as MOOP

The procedure for selecting service uses QoS values to compare the services' performance. QoS attributes taken through boolean, numerical value, or range. For example, response time can be in the field of (20, 40) sec. The cloud user requirements are taken as objectives; if the customer has one objective, it is a single-objective optimization problem. If there are two or more objectives that should be considered simultaneously for decision making, then the problem is a bi-objective or multi-objective problem. Therefore, in MOOP, two or more objectives and constraints included in the problem domain and service selection simultaneously consider all objectives. The comparison between single objective and multiple objectives on the basis of their approaches is shown in FIGURE 2.12. These objectives need to be optimized at the same time to find optimal service. Figure 2.12 below shows the user requirements are taken as either a single objective in which the user gets the best solution. The user gets an optimal solution in multiple objectives, also known as a Pareto-optimal solution.

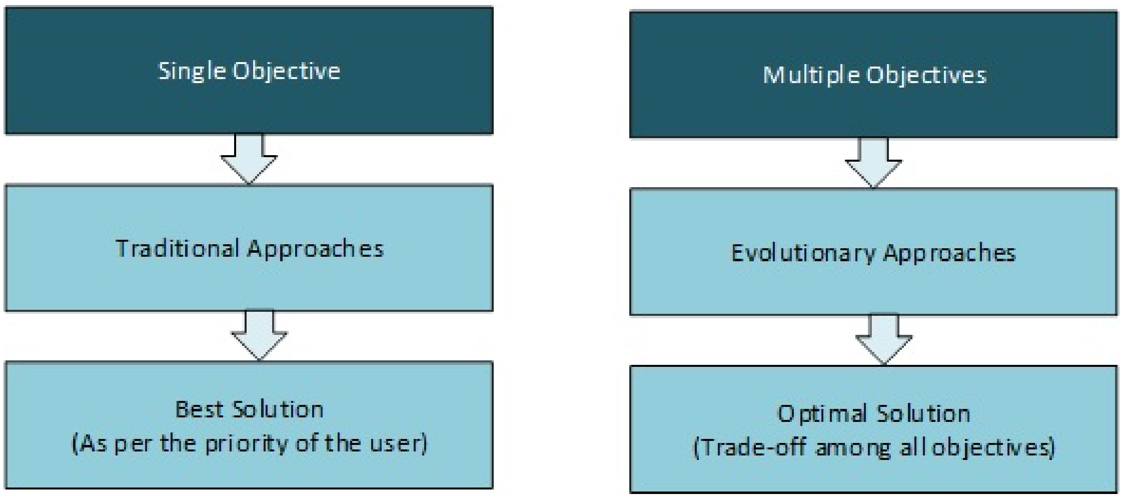
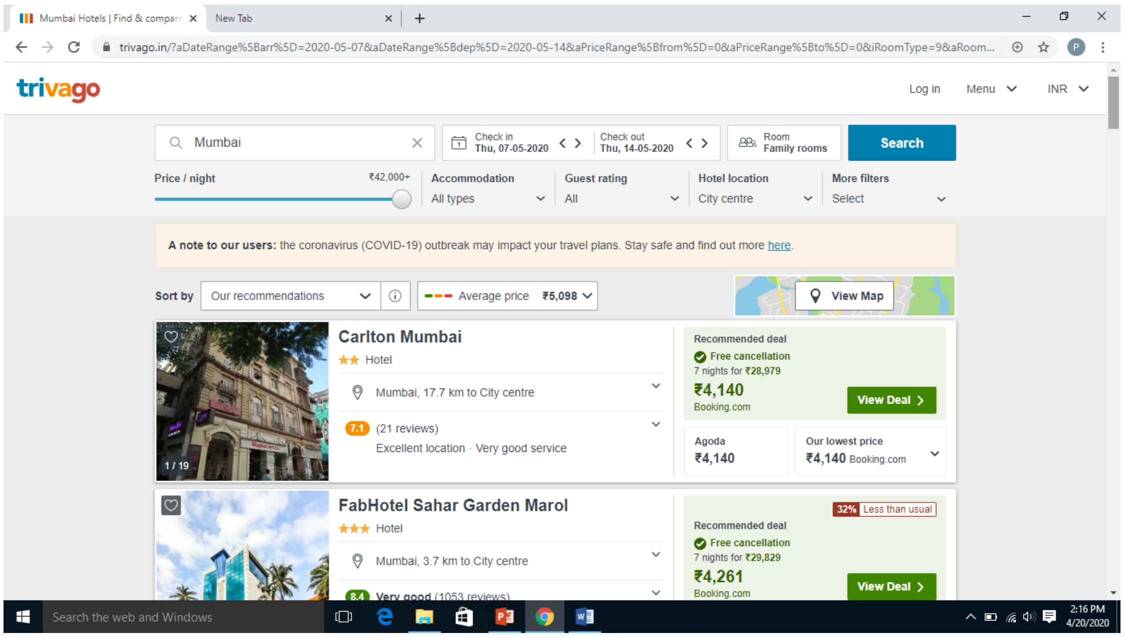


FIGURE 2.12: Single- Objective Vs. Multi-Objective

Example: Cloud SaaS Model as a Multi-Objective Optimisation Problem

To show service selection in the cloud as multiple objective problems, The example of a SaaS application, namely ”TRIVAGO- Compare hotel price worldwide.” Trivago is a global search platform that helps customers compare hotel prices from thousands of travel sites and give customers the best price. Each customer has different requirements, and therefore the customized service needs to be given according to the condition. Also, customer requirements cannot fit into the single objective, and therefore multiple objectives are considered in finding the best deal for the customer. Trivago uses a metasearch engine to understand the customer needs to provide an appropriate solution. Therefore, the above example shows that cloud service selection can be said as a multi-Objective Optimisation problem.


Mumbai Hotels | Find & compare
New Tab
trivago.in/?aDateRange%5Barr%5D=2020-05-07&aDateRange%5Bdep%5D=2020-05-14&aPriceRange%5Bfrom%5D=0&aPriceRange%5Bto%5D=0&iRoomType=9&aRoom...
trivago
Log in
Menu
INR
Search
View Map
View Deal
View Deal
Search the web and Windows
2:16 PM
4/20/2020

Mumbai | Check in Thu, 07-05-2020 | Check out Thu, 14-05-2020 | Room Family rooms

Price / night ₹42,000+ | Accommodation All types | Guest rating All | Hotel location City centre | More filters Select

A note to our users: the coronavirus (COVID-19) outbreak may impact your travel plans. Stay safe and find out more here.

Sort by Our recommendations | Average price ₹5,098

Carlton Mumbai
Hotel
Mumbai, 17.7 km to City centre
7.1 (21 reviews)
Excellent location · Very good service
Recommended deal
Free cancellation
7 nights for ₹28,979
₹4,140
Booking.com
Agoda ₹4,140
Our lowest price ₹4,140 Booking.com

FabHotel Sahar Garden Marol
Hotel
Mumbai, 3.7 km to City centre
8.4 Very good (1053 reviews)
32% Less than usual
Recommended deal
Free cancellation
7 nights for ₹29,829
₹4,261
Booking.com

FIGURE 2.13: Trivago – Search Engine

The growing demand for cloud services provides excellent opportunities for customers to find the appropriate service according to their customized requirements, which further raises the challenge of picking the vast list service. The service identification is a time-consuming and tedious process for the customer as they have to collect necessary information and analyze all the available services. The above Figure 2.13 shows the real-world example of selecting SaaS through a service selection algorithm to get for faster and efficient results. The service selection approach is generally used by the cloud service provider or the brokers to understand the customer's needs and to find the optimal service as per the customized requirements.

The Cloudcmp, Salesforce, AWS, IBM Cloud, etc. use cloud service selection and ranking algorithms to meet customer demand.

2.7.3 Multi-Objective Optimisation using Evolutionary Algorithm

The two or more objectives are considered simultaneously to identify the feasible solution. The objective functions are either be minimization/maximization or both. The optimal solution is the one that has the best tradeoff between competing objectives. The user requirements are not reasonable if the multi-objective problem gets converted to a single-objective problem.

In multi-objective optimization, the traditional optimization approaches operate on candidate solutions do not yield efficient results. Simultaneously, evolutionary algorithms are more efficient in efficiently identifying optimal service as they recognize the set of promising solutions. A generalized of the multi-objective problem is given by Edgeworth and Pareto (Abbas et al., 2017).

The challenge faced by the researchers in finding appropriate answers to problem impacts in most of the disciplines. However, various techniques developed to handle such issues, but the complexities arise due to the alternative approaches associated with a particular problem. The evolutionary algorithms (EAs) are found to solve MOOP and have motivated the researchers to use EAs, allowing generating Pareto optimal solution in a single execution. An EA involves the user's requirements and evaluates the fitness to find a feasible solution. Simultaneously, the fitness function measures the efficiency of any solution that satisfies the condition and gives corresponding real-value to that optimal solution Evolutionary algorithms are a heuristic approach to resolving difficult problems in the polynomial time frame and classified as NP-Hard problems. The evolutionary process is considered efficient for solving NP-Hard problems. The EA works on natural selection, which has four steps: initialization, assignment, genetic operators, and termination. EAs, in most cases, use multiple fitness functions and, instead of identifying a single optimal point to identify optimal solutions. The solution lies on Pareto front, and no solution dominates any other solution by reducing the answers based on some problem context. The evolutionary algorithms use population and allow the generations of possible solutions at a single execution.

The objective of an EA algorithm is identifying solutions to achieve the following:

1. Identification of the fitness function for the solution

2. The best-found Pareto front should come near to the Pareto front.

3. Maintain diversity by preventing convergence from achieving a well-distributed trade-off front.

4. All feasible solutions in the front should be uniformly distributed to give the real picture of the decision-makers trade-offs.

5. Provide a possible number of answers to contribute to choosing the most suitable solution.

The MOEA approaches designed for different application areas are classified as decomposition, weight-based/ preference, Indicator based, and Non-dominated sorting and ranking (Rai & Kumar, 2016).

2.7.3.1 Decomposition based MOEA

The decomposition process involves dividing the objective space into sub-objectives, which individually has a Pareto optimal solution. The list of solutions helps in maintaining the diversity of obtained solutions and assist in solving MOP problems. Suppose one solution dominates the other in Pareto front even then, the solution can produce new solutions, making each sub-objective solution to assemble to reach appropriate solutions.

The multiobjective evolutionary algorithm based on decomposition (MOEA/D) uses traditional aggregation methods and alters MOOP into a single-objective problem. MOEA/D is efficient for multiobjective problems and discrete decision variables. MOEA/D uses a decomposition approach of mathematical programming, can be easily incorporated into EAs. MOEA/D helps optimize scalar problems and does not solve fitness function assignment, diversity, and convergence. MOEA/D is less complicated, and corporate normalization techniques for dealing with objectives. The most adopted method in many MOEA/D is the weight vector generation, which uses a uniform random sampling approach. The benefit of using this approach is that the sample size is flexible. Qi and co-authors (Qi et al., 2014) talked about the uniform distribution of Pareto optimal solutions using distributed weight vectors in MOEA/D. Furthermore, the authors stressed the initialization technique for the transformation weight vector would not work efficiently in elaborate Pareto fronts.

Li and co-authors gave a stable matching model (STM) (K. Li et al., 2013) known as MOEA/D-STM to select promising solutions for problems in MOEA/D. The proposed approach ranks the solutions present in the list using aggregation function, and the solutions having better aggregation function values are preferred. The solution having a smaller distance value to the direction vectors is the preferred solution. Li and co-authors extended MOEA/D-STM by presenting MOEA/D-IR, using incorporation based on selecting MOEA/D. The proposed approach also defines the mutual preferences between solutions. The experimental study showed that MOEA/D-IR is better than MOEA/D-STM on several test-suites. Pilat and Neruda (Pilat & Neruda, 2015) proposed a user preferences approach named cw-MOEA/D, an extension of MOEA/D. In cwMOEA/D, user requirements are takes as preference and recorded as a function and evaluate all the solutions at each iteration. The negative number represents the preferred solutions, and positive numbers represent the less preferred numbers.

Li and co-authors (Lin et al., 2015) proposed C-MOEA/DD an extended of MOEA/DD to solve constrained MaOPsm. In C-MOEA/DD, the tournament selection selects feasible solutions and also, the steady-state update process of MOEA/DD is improved to get possible results.

2.7.4 Weight based MOEA

The multi-objective optimization consists of many conflicting objectives that should consider for finding an optimal solution. The priority for the objective functions makes it easy to identify the preferred solution according to the priority objective. The existing preference-based optimization is a priori, interactive, and a posteriori methods. A priori method, the input conditions are put in advance before starting the optimization process. The limitation lies here due to the limited knowledge to the DMs related to the problem. A posteriori method obtains a set of Pareto optimal solutions to choose a different trivial number of explanations as per their user's preferences. Interactive methods helps the decision-maker to infuse their preferences between the optimization. In interactive processes, the decision can modify their choices depending on the area information acquired during the optimization. Sakawa and Kato in (Sakawa et al., 2003) used a fuzzy-based approach and presented the requirements in the form of reference points created until a satisfactory result was is achieved. Phelps and koksalan in (Phelps & Köksalan, 2003) compared individual solutions for their fitness value while considering each iteration's decision maker's preferences. A weighted sum of objectives uses a single substitute to get some generations.

Jin and Sendhoff proposed a methodology and converted fuzzy preferences to the weight intervals to calculate dynamic weighted aggregation EA and find equivalent solutions . The method transforms the multiobjective to a single objective problem using a weighted accumulation methodology to find a solution.

Fonseca and Fleming (Fonseca et al., 1993) and Deb (Deb, 1999) use goal programming helps the DM to identify a goal. Fonseca and Fleming remove all those objectives that do not find a solution according to the purposes specified. Deb also used the goal strategy and modified the optimization criteria. The plan decided should neither be too high or too low; otherwise, the solutions will not reach the goal. Zibin Zheng and co-authors (Zheng et al., 2012) talked about cloud services' performance and how it helps users select the right services. The author proposed a cloud ranking framework that improves the selection efficiency and reduces the cost and time required to find the best services. The extensive real-time experiments were conducted to check the cloud ranking algorithm's accuracy and compared it with other existing ranking algorithms. The proposed algorithm was found more efficient as compared to the existing ranking algorithm.

Yang and co-authors (Yang et al., 2014) explored different service selection techniques in the cloud environment and concluded that it is a challenging and complicated process. A service selection approach is proposed to find services from different perspectives: functional dimensions, non-functional dimensions, and transactional dimensions. The proposed work considers the user's simultaneous service request and designed a novel strategy based on hybrid particle swarm optimization that uses a genetic algorithm to find a solution. Ye and co-authors (Ye et al., 2011) talked about two cloud categories: application services and the other is utility services. The author proposed a genetic algorithm-based service composition approach and considers four attributes, response time, price, availability, and reputation. QoS values and constraints to find cloud services.

Qiang He and co-authors (He et al., 2012) proposed a novel MSSOptimiser (multi-tenant SaaS optimizer) approach for making SaaS service selection. The process assists the SaaS developers to use QoS requirements and use them in achieving various optimization goals. The method is efficient even if users or services is increased. The experimental study shows that MSSOptimizer is more efficient as compared to other techniques.

2.7.4.1 Indicators based MOEA

The indicator MOEAs used the indicators to measure solution value and use the criteria for selecting cloud environment services. The indicator-based EA (IBEA) (Zitzler & Künzli, 2004) uses an evolutionary algorithm (SMS-EMOA) (Beume et al., 2007) for service selection. There are also generational distance MOEA

(Menchaca-Mendez & Coello, 2015) indicator for many objetive, and metaheuristic-II (MOMBIII) (Hernández Gómez & Coello Coello, 2015). The hypervolume indicator based MOEA is proposed in (Zitzler & Künzli, 2004) to identify the candidate solutions. The proposed Pareto compliant Sharpe-Ratio indicator is an alternative way for generalizing the hypervolume to many-objective optimization (Fonseca et al., 1993). Zitzler and Kunzli in (Zitzler & Künzli, 2004) gave the first indicator-based evolutionary algorithm [IBEA], which uses a genetic algorithm based on a predefined binary indicator and find the solution. The results show IBEA is better among MOEAs, NSGA-II, and SPEA2.

2.7.4.2 Non-Dominated Sorting based MOEA

The genetic algorithm for solving MOOP generates a solution set that undergoes various operations selection, crossover, and mutation. During the process, the non-dominance of the final solution set is obtained retrived all the solution, whereas no solution is considered to be strictly superior to other solution in the set. Non-dominated sorting for a maximization problem with m objectives, where two solutions x and y are solutions is mathematically defined in equation (2.9):

$$x < y \mid \forall i : f_i(x) \geq f_i(y) and \exists j : f_j(x) > f_j(y) \tag{2.9}$$

where fi(x) and fi(y) are the i-th objective values for function x and y. The above equation (2.9) shows that objectives in correspondence to solution x are either greater or equal to the goals of solution y, and there is at least one objective value for x, which is greater than y. The Pareto front includes a set of all the non-dominated solutions, and therefore in a given population, there may be several fronts. Each of the fronts represented by a unique front number such as k in this case, where k: k¿1. The thesis research considers all the fronts, and the solutions smaller front numbers represent higher ranks. The non-dominated fronts are said to consist of properties mentioned below:

1. The solution in front k must dominate at least one solution from front k+1.
2. The solution in front k+1 may or may not dominate solutions from front k+2.

Therefore the solutions lying in front of having low rank are preferred than those having a high rank. According to the Pareto dominance principle, the non-dominated sorting approach sorts the population's solutions, which plays a vital part in the selection procedure. The Non-dominated Sorting Genetic Algorithm (NSGA) is a technique of evolutionary algorithm and is an extension of the Genetic Algorithm. NSGA also falls into the category of algorithms such as (EMOO) . The NSGA algorithm's foremost objective is to increase the overall searching process to identify the feasible solution from the list of candidate solutions constrained by a set of objective functions.

Deb and Jain (Deb, 1999) suggested a non-dominated sorting genetic algorithm, considering a reference point-based to solve optimization problems. The proposed algorithm originated from the NSGA-II algorithm to improve its capability by changing its predecessors during the selection process. The difference between them lies in the way of selecting cloud services. Fang and co-authors proposed a non-dominated sorting method using a divide-and-conquer technique, where several redundant comparisons are removed. Drozdík M and co-authors (Drozdik et al., 2014) gave a non-dominated sorting technique called M-front; this method's key objective lies in utilizing the existing information the population used in the dominant relationship. In M-front, the mathematical properties are used for Pareto dominance by increasing the insertion speed of feasible solutions and remove non-dominated solutions.

Zhang and co-authors (X. Zhang et al., 2016) suggested a professional ENS-SS approach to sort solutions suited for MOPs even when the number of objectives is less. ENS-SS is considered efficient due to the method used here; the pre-sorting done in which sorted population is put in the first front, and the rest of the solutions are left behind. The pre-sorting sorts the candidate solutions according to the first objective function. The ENS-SS algorithm practices the strategy, which ensures that the latter solutions cannot lead the former.

Jensen (Jensen, 2003) proposed a sorting technique, called Jensens sort, an enhanced form of the non-dominated sorting approach for two objectives, using the divide-and-conquer strategy. The Jensen algorithm presorted all the solutions and stored them in ascending order.

Roy and co-authors (Roy et al., 2016) offered the Best Order Sort (BOS) method's whose primary objective is to find the 'not-worse' solutions. Suppose there are M sets of an objective function, then the algorithm searches the smallest group to compare, and once the solutions are compared, then these solutions are picked one by one, and rank is applied. Sumit Mishra and co-authors (Mishra et al., 2018) used the BOS algorithm and generalized the complete process of BOS, known as Generalised Best Order Sort (GBOS). The approach reduces the comparisons required to identify an optimal solution. The GBOS is the generalized approach of the BOS method and helps to handle the limitation of BOS without negotiating on time and space complexity.

2.7.4.3 Pareto Solutions and Pareto Front

A solution is called Pareto optimal solution when one solutions objective function is said not to be improved without debasing other solutions objective values. The Pareto approach handles the MOOP and provides optimal solutions considering trade offs among various objectives. Figure 2.14 shows the Pareto solution for a minimization bi objective problem.

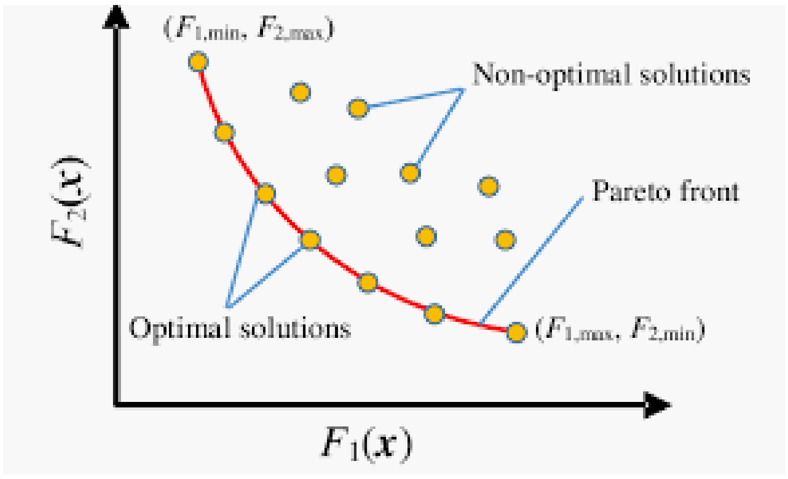


Figure 2.14: Pareto Front and Pareto Solutions

Evolutionary algorithms are appropriate for managing and controlling solutions by finding all the feasible solutions in a single execution. Deb showed if the population's size increases, then the number of required comparisons automatically increases. The Pareto concept for evolutionary algorithms is practical; otherwise, it is computationally expensive to identify Pareto solutions.

2.7.4.4 Non- Dominated Sorting for Cloud Service Selection and Ranking

Jahani A and Khanli LM in (Jahani et al., 2017) offered the NSGA_SR algorithm, exploits both objective and subjective parameters, and uses a non-dominated sorting method to rank the solutions. The technique allows the customer to enter the requirements in either Boolean, numerical, and range. Also, the approach makes use of feedback collected from an experienced user. The experiments conducted assure that this approach is better in flexibility and scalability than the existing approach.

Zhang and co-authors (M. Zhang et al., 2012) suggested a declarative approach based on the logic that covers transactional aspects, and SQL semantics to manage the transactional parts. According to the author, this approach is better than the traditional sorting and selection algorithm. The process also helps the user to optimize the query and find the result using the Cartesian product.

Hussain and co-authors (Hussain et al., 2020) proposed a Methodology for Optimal Service Selection (MOSS), which includes different stages such as prequel, calculation, ranking, combination, and selection. The MOSS considers both QoS and QoE parameters to find a feasible solution. All the stages work in coordination with each other to find optimal results. Fletcher and co-authors (Fletcher & Liu, 2015), the CF-based service commendation system proposed, have steps like retrieving service QoS values, generating missing QoS values, and selecting the suitable weights. The missing QoS values are anticipated based on the historical data and customer preference of the past services. CF-based service selection has become the most popular algorithm for personalized product ranking.

Ding Shuai and co-authors (Ding et al., 2017) proposed the novel service recommendation system two rank prediction algorithms. This recommendation system's main objective is to mend ranking accuracy and meet the range of requirements of cloud users. The idea behind this recommendation system is to provide user satisfaction by formulating cloud service selection and ranking as MOOP. The approach identifies the best solution, which is a trade-off between the competing objective functions. The experimental outcomes specify that the suggested service recommendation system is efficient and the results achieved are also accurate.

Jahani and co-authors (Jahani et al., 2014), the new W_SR (Weight Service Rank) approach provides ranks to the cloud service. The author discusses the different users have different QoS requirements leading to exploring cloud Service providers that can meet their customized demand. The author determines the best service with a specific user's requirement. The W_SR compares and ranks the services by quality of services (QoS) value to select a suitable service.

Jahani and co-authors in (Jahani et al., 2017) proposed another approach called a multi-agent-based method (ARank) system. The author discusses the user's challenges in selecting an appropriate service from the vast available services list. Therefore to choose a suitable service that matches the user requirements is challenging. The ARank technique uses intelligent agents from the list of candidate services and ranks these services. The experimental results show users' waiting time is reduced in comparison with other algorithms.

In (Urena et al., 2019), the author proposed the Fuzzy MOOP based on the trust framework. The author discussed cloud computing technology, along with different cloud service models. The author talks about the drastic increase in Cloud Service Providers (CSPs), and the challenge user has to face. These providers offer a diversity of services in front of the customer, making it difficult for them to choose the appropriate service to meet their customized requirements. The author's Fuzzy approach will simplify the process of identifying a suitable service to meet user-specified needs. In (Abdel-Basset et al., 2019), the decision support system selects an infrastructure service that will help determine the appropriate cloud infrastructure and help choose the best services.

2.8 Comparative analysis and Research Issues in Cloud Service Selection Technique

TABLE 2.2 below gives an overview of research done in the area of cloud service selection and ranking. The proposed work, techniques, etc. are discussed on a few parameters such as technique used or algorithm designed, QoS parameter, brief overview of working, and the approach.

Technique	Framework	QoS Parameters	Cloud	Approach
AHP	SFA (Sales Force Automation) to identify the services	Integration, Scalability Security, Usability, Vendor Reputation, Cost	SaaS	MCDM
AHP	Case Study Based Approach (a set of rules)	QoS Attributes	SaaS, IaaS	MCDM
AHP	AHP based framework is proposed, and the QoS parameters are used to select service	Non Functional QoS attributes (Efficiency- Time Behaviour and Resource utilization, Response Time – CPU utilization and Memory Utilisation, Cost, Scalability)	SaaS	MCDM
AHP	Framework (MC2)2 which considers QoS parameters to identify the solution	Cost, Benefits, opportunities, and Risk	IaaS	MCDM

AHP	Framework (Cloud Genius)	Cost, Low Latency, performance, Uptime	IaaS	MCDM
TOPSIS	The TOPSIS technique is considered to evaluate the solution	Scalability, Capability, Performance, Reliability, and Availability		MCDM
Fuzzy ELECTRE	Tunnel construction of projects	Risk (Delay and Failure) are at the lowest rank Damages, unsafe working, accidents of machinery	Other	MCDM
ELECTRE	SSM_EC middleware	Subjective (turnaround time, Service Cost, Trust, Reliability)	IaaS	MCDM
Fuzzy Simple Additive Weighting	location selection problem	Subjective and Objective attributes	Other	MCDM
MADMAC framework	Works on three decision areas cloud switch, cloud type	Subjective and Objective	PaaS, IaaS, SaaS	MCDM
Mid-level-splitting, Hypothetical equivalents and Inequivalents	The technique compares functionally equivalent services based on customer perception	Accuracy, Response-time, and Security	SaaS	MCDM
trust-enhanced similarity model	This model will combine QoS of the cloud service, fills all the missing QoS values, and rank the services.	QoS Parameters	PaaS, IaaS, SaaS	Trust Model

trust-based selection approach	The author talked about the trust-based selection in cloud computing as the trustworthiness between cloud providers and cloud users	trust and reputation	PaaS, IaaS, SaaS	Trust Model
reputation	the proposed method is filtering all the ratings, which are unfair and unnecessary, thereby increasing overall performance through the market mechanism, helping the service providers and the service consumers go for their choice	reliability	PaaS, IaaS, SaaS	Trust Model
SELCSP framework	SELCSP for computing the interaction risk between the cloud provider and cloud consumer	trustworthiness and reliability	PaaS, IaaS, SaaS	Trust Model
CINS approach	(CINS) based on the time-aware approach to evaluate the trustworthiness in the cloud service selection	QoS	PaaS, IaaS, SaaS	Trust Model
service trust estimation method	The author proposed a trust estimation method for cloud service selection based on trust.	trust	PaaS, IaaS, SaaS	Trust Model

Trust Scheme model	The model divided the selection into parts services chosen, and the services are delivered, making sure that trust decisions are used to achieve service.	trust	PaaS, IaaS, SaaS	Trust Model
TOPSIS technique	The first step is identifying the criteria according to the Service Measurement Index (SMI); the second step is the determination of assigned weights.	trust	PaaS, IaaS, SaaS	Fuzzy-Based
mid-level splitting method	The method used in the technique is the mid-level splitting method. The author in the research only took subjective parameters for the study	trust	PaaS, IaaS, SaaS	Fuzzy-Based
QoCS and cost-based framework	The framework is based on trustworthiness and uses fuzzy logic, which helps the user in analyzing the cloud services from multidimensional perspectives	trust	PaaS, IaaS, SaaS	Fuzzy-Based
fuzzy decision-making framework	The ranking framework is designed which helps in ranking of the services for all service models	trust	PaaS, IaaS, SaaS	Fuzzy-Based

AHP and Fuzzy	The author considers a fuzzy-based approach and ahp method to assign weights, and later, both are integrated to rank the services.	trust	PaaS, IaaS, SaaS	Fuzzy-Based
dynamic broker	The approach predicts the behaviour of user based on relinquish probability, the likelihood that the user will cease to use the requested services	Multi-criteria	PaaS, IaaS, SaaS	Broker Based
Broker based framework	A Cloud Service Broker (CSB) framework is responsible for selecting and providing the cloud-based render farm.	QoS	PaaS, IaaS, SaaS	Broker Based
Indexing	An indexing technique proposed to manage extensive information helpful in making a decision	QoS	PaaS, IaaS, SaaS	Broker Based
QoS based ranking Framework	The framework proposed and which identifies the user requirements and finds the solution	QoS	PaaS, IaaS, SaaS	QoS Based
VIKOR and AHP	The hybrid approach using the priority in the form of weights and assigns ranks	Response time, security, cost, and reliability.	PaaS, IaaS, SaaS	QoS Based
AHP	The service selection is made using the AHP approach and assists the user by identifying requirements.	throughput, availability, security	PaaS, IaaS, SaaS	QoS Based

AHP and SAW	The proposed method uses an AHP and SAW form and identify the preference of the user in finding a solution	QoS	PaaS, IaaS, SaaS	QoS Based
AHP	The framework selects and ranks services by using the AHP model and multi-criteria	cost, response time, throughput, availability, and consistency	PaaS, IaaS, SaaS	QoS Based
MOEA/D-STM	The approach ranks all the solutions in the solution list by using its aggregation function values.	QoS	PaaS, IaaS, SaaS	Decomposition based MOEA
fuzzy approach	The weighted based approach used to identify the preference in the form of reference points and compared a pair of individuals for their fitness value	QoS	PaaS, IaaS, SaaS	Weight-based/ preference-based MOEA
Goal Programming	The strategy is to modify the optimization criteria. The plan decided should neither be too high or too low; otherwise, the solutions will not reach the goal.	QoS	PaaS, IaaS, SaaS	Weight-based/ preference-based MOEA
Hybrid particle swarm optimization	the model considers the concurrent request of service from the user and uses a genetic algorithm to find a solution	QoS	PaaS, IaaS, SaaS	Weight-based/ preference-based MOEA

genetic algorithm	genetic algorithm-based service composition approach. QoS values and constraints are considered to find cloud services.	QoS	PaaS, IaaS, SaaS	Weight-based/ preference-based MOEA
MSSOptimiser	The process assists the SaaS developers to use QoS requirements and use them in achieving various optimization goals.	QoS	PaaS, IaaS, SaaS	Weight-based/ preference-based MOEA
M-front	The aim of the work lies in utilizing the existing information of the population used in the dominance relationship	QoS	PaaS, IaaS, SaaS	Non-Dominated Sorting based MOEA
ENS	ENS's is considered efficient due to the approach used here; the pre-sorting has done in which sorted population put in the first front, and the rest of the solutions are left behind.	QoS	PaaS, IaaS, SaaS	Non-Dominated Sorting based MOEA
T-ENS	T-ENS uses a tree-based structure to find the dominant relationship between solutions.	QoS	PaaS, IaaS, SaaS	Non-Dominated Sorting based MOEA
Best Order Sort BOS	The BOS method's primary objective is to differentiate between meaningful and un-important solutions and then find solutions	QoS	PaaS, IaaS, SaaS	Non-Dominated Sorting based MOEA

Generalized binary order sort GBOS	The GBOS is the generalized form of BOS to handle the limitation of this approach and reduce complexity	QoS	PaaS, IaaS, SaaS	Non-Dominated Sorting based MOEA

Table 2.2: Summary of Selection Approaches to Cloud Services

2.8.1 Analysis based on Techniques and QoS parameters

The proposed approaches used in service selection are divided into five categories: the MCDM approach, the Trust approach, the Fuzzy methodology, cloud brokerage, and QoS parameters for service selection. All techniques are discussed in detail and summarized in Table 2.2. Figure 2.10 shows that in the MCDM approach, AHP is the most commonly proposed approach used by the researchers in their work. In the MOOP, evolutionary algorithms are found efficient and quickly adopted algorithm by the researchers.

The quality parameters are essential for the service selection process and based on the current review process. The QoS is considered in all the recent work as the parameters are significant in deciding for the decision-maker's service. The authors have considered specific QoS parameters in their study, but some authors have taken all the QoS parameters. The section includes the importance of using different QoS parameters in research is discussed. FIGURE 2.15 shows various QoS parameters like trust, Throughput, Availability, Uptime, Risk, Efficiency, Reputation, Security, and Scalability. The frequency of these parameters in the literature is calculated and plotted in the bar graph to obtain the essential and significant QoS parameters. The map shows that some of the attributes are treated as more important than others by most researchers at the time of service selection in a cloud-like Cost, Response time, and availability. While the parameters, including Capability, correctness, etc. are not studied much in the literature.

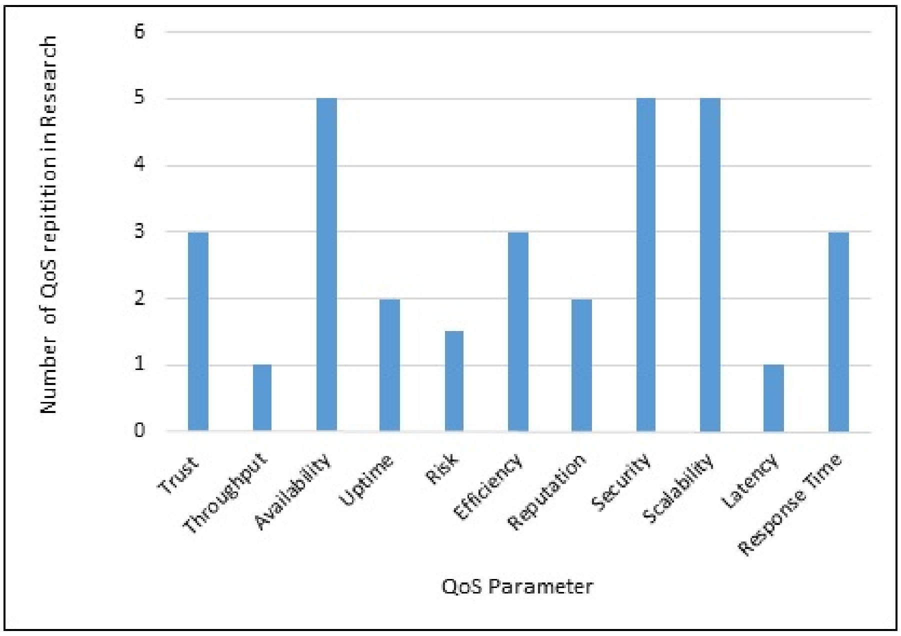


FIGURE 2.15: QoS parameters according to their relevance in Research

TABLE 2.3: QoS Parameters about their importance in the selection of services.

QoS	No of Repetitions in QoS
Scalability	4
Reliability	3
Security	3
Usability	1
Reputation	2
Cost	8
Efficiency	1
Response Time	6
Risk	3
Latency	1
Uptime	1
Capability	1
Availability	5
Accuracy	1
Throughput	1
Correctness	1
Trust	3

2.8.2 Identified Research Gaps in Cloud Service Selection

A comprehensive study and examination of the existing literature using different techniques and algorithms suggested by various authors applied for service selection are described in detail. The literature survey done in the research paper is through the existing proposed work and comparing different techniques and parameters used for service selection. The taxonomy of cloud service selection studied considering various approaches in this area, such as multi-criteria decision making, trust models, fuzzy approach, brokerage model, QoS based techniques, Non-dominated service selection techniques. The existing literature includes both the traditional and evolutionary methods, and from the study, it is evaluated that the conventional algorithms are well suited in a single-objective problem. In contrast, evolutionary algorithms are considered more efficient in the multi-objective optimization problem. The current research work discusses various algorithms, frameworks, approaches proposed in area cloud services. The non-dominated sorting approach sorts the population by pairwise comparing the available solution and finding an optimal solution. The efficiency of the evolutionary process depends on the number of comparisons required for finding the optimal solution. The challenges users face when selecting the appropriate services according to their requirements are discussed below.

1. **Generic approaches proposed for different cloud service model-** SaaS is the most extensively used service model as compared to other services models. The existing literature discussed a generalized approach for all cloud service selection and ranking for all service models. The research shows that each cloud service model has its own important QoS parameter, which helps evaluate that service model's efficiency. Therefore, a generalized approach for finding optimal service by considering the QoS parameter will not produce an efficient result.

2. **Insufficient approach considering customized requirements in cloud service selection-** The existing literature proposed a generalized approach for cloud services selection and ranking without considering an individual's requirements. Every customer has a different set of requirements for selecting a service in real life. Therefore a generalized approach will not produce the customized results.

3. **Cloud service selection as multi-objective Optimization Problem-** The literature review shows various techniques proposed in MOOP applied in diverse application areas. The insufficient approaches are seen in cloud service selection and ranking as a multi-objective optimization problem. Some of the techniques developed for cloud service selection use traditional strategies that are considered efficient for single-objective problems.

4. **Increased number of services or objectives or both-** The optimal solution is found by comparing the list of available solutions. At the time of the comparison, there are many redundant comparisons and un-necessary comparisons that increase the search space and decrease the search efficiency. An efficient method should be developed, which selects and ranks the services and find an optimal solution by reducing the total number of comparisons.

5. **Cloud Service Selection and Ranking-** The rapid increase in the number of service providers in the market and the diversity of existing services challenge identifying the appropriate service to fulfill its customized requirements.

2.9 Summary

The chapter discussed the literature review and different approaches and techniques proposed by various researchers in cloud services. The QoS attributes, framework, and methodology in the existing work are represented in table 2.2. The detailed study on the literature helps understand the working and the procedure followed by authors in the cloud service selection. The generalized service selection model is conferred in the chapter, and the literature review identifies limitations and issues in the literature and helps recognize the research gaps. Although the industry has widely adopted cloud computing, still new problems are seen in its industrial applications. These gaps make the user not very comfortable in the adoption of the cloud for their business activity. The current study done on the cloud model identified a few of the challenges which can be used in future research. Identifying limitations and issues in the existing literature is discussed, which laid a foundation for future research directions. In the future author plans to explore the multi-objective optimization approaches for service selection as the user has various objectives at the time of service selection. Therefore, traditional methods will not be an effective way in the area of service selection. Therefore the research emphasizes SaaS as the problem domain, leading to challenges in providing a better SaaS to meet the demands. The SaaS cloud services' selecting and ranking requires an efficient approach to improve efficiency and overcome the weakness of the existing strategies used in SaaS cloud service selection. Customer confidence is gained through an efficient cloud service selection is used to select the services that can meet customized requirements.

Chapter 3

Non-Dominated Sorting and Ranking of Services (NDS-ROS) Algorithm

This chapter investigates the research issues mentioned in chapter 2 and proposed a novel approach, a non-dominated sorting algorithm for ranking of services (NDS-ROS) algorithm to find the feasible service according to the user requirement. The NDS-ROS considers cloud service selection and ranking as a MOOP and use QoS attributes to meet the tailored requests of the users. The NDS-ROS architecture is also discussed in detail, which shows the overall working of the approach. FIGURE 3.1 illustrates the architecture, which consists of four stages of the NDS-ROS, namely (1) filtration, (2) Sorting, (3) ranking, and (4) Dominance comparison. The NDS-ROS algorithm helps in identifying optimal service as per the user requirements. The NDS-ROS specifically designed and developed to make the service selection for SaaS cloud; also, the QoS parameters considered for finding optimal service are the SaaS QoS parameters (He et al., 2012). The NDS-ROS architecture is shown in fIGURE 3.1, which covers all the four stages and also different conditions involved in the whole process. The NDS-ROS algorithm's illustration is also demonstrated with a working example to clarify the process flow of the stages involved in service selection. The assumptions used in NDS-ROS represent the number of users, objectives, and services. These assumptions were later used for mathematical modeling of NDS-ROS. The chapter also covers the proposed dominance comparisons rules to reduce the duplicacy in comparing the services. The fitness function is also given to the non-dominated services to identify the user's customized optimal service.

The NDS-ROS algorithm compared with three existing algorithms, including deductive sort, efficient non-dominated sorting algorithm (ENS-SS), and generalized binary order sort (GBOS-SS), to check the algorithm's efficiency. The NDS-ROS shows improved results on the increased services or objectives size. The NDS-ROS methodology involves steps that increase the complete performance of the searching and ranking the cloud services. The filtration step filters out relevant services from services that are not required by the customer. The sorting technique used in the algorithm will help sort the cloud services according to the objective functions. The ranking step will assign a rank to the services. Finally, the dominance comparison will compare the services until an optimal service is identified. The NDS-ROS algorithm's objective is to reduce the search space to minimize the customer's waiting time.

3.1 Overview of NDS-ROS

The multi-objective evolutionary algorithms rank the population-based solutions based on their dominance factor, which is received in each iteration process. The process is considered a costly affair in finding the multi-objective problem (L. Liu & Zhang, 2015) or when the number of services is vast. The existing algorithms like NSGA II (Deb, 1999), SPEA2 (Zitzler & Künzli, 2004), NPGA (Erickson et al., 2002), MOMGA (Zydallis et al., 2001), and others rank the population according to the dominance principle, which again increases the overall computational complexity. Therefore, efficient non-dominated sorting and ranking of service (NDS-ROS) is designed and developed, lowering down the complexity of existing algorithms and is cost-effective by improving the approach's overall performance and finding optimal SaaS service cloud model.

The NDS-ROS algorithm finds an optimal service using QoS attributes, the range, and priority of QoS value taken from the user's. The NDS-ROS aims:

1. Reduce the procedural gap in the existing work and
2. Increase the performance of the selection procedure involved in the SaaS cloud.

The NDS-ROS consist of four stages, includes filtration, sorting, ranking, and dominance comparison. The techniques are applied to support the NDS-ROS algorithm. The QoS range and its priority are taken from the user to filter out the candidate services. Merge sort will sort the candidate services in objective functions individually either in ascending or descending order. The ranking is done on the sorted services for each candidate service, and dominance comparison is made between the same ranked candidate services taken from the different objective functions. The dominance comparison will continue unless optimal service is identified. The NDS-ROS algorithm reduces th comparisons required to find optimal service(s).

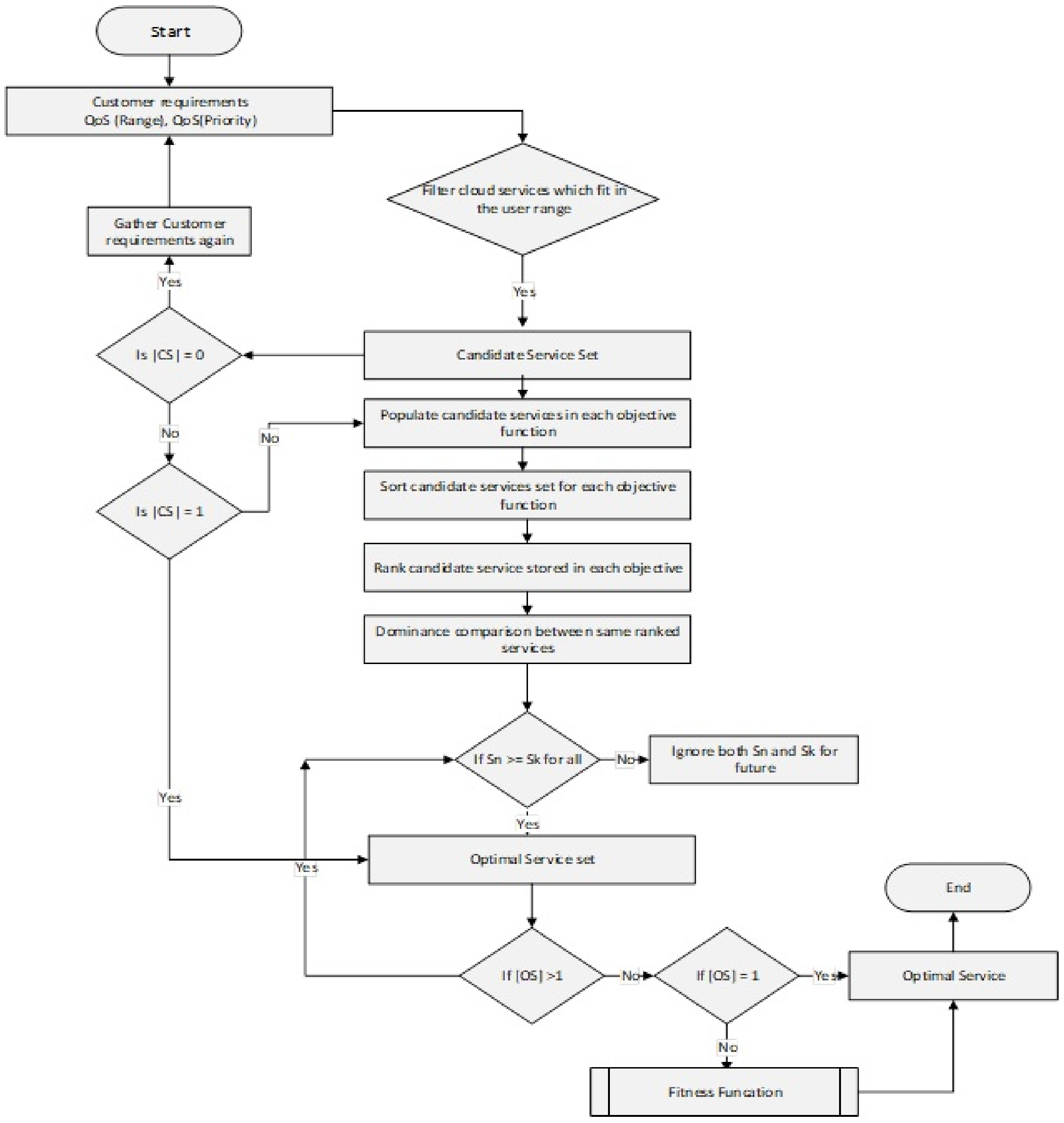

Start
Customer requirements QoS (Range), QoS(Priority)
Filter cloud services which fit in the user range
Gather Customer requirements again
Yes
Yes
Is |CS| = 0
Candidate Service Set
Populate candidate services in each objective function
No
No
Is |CS| = 1
Sort candidate services set for each objective function
Rank candidate service stored in each objective
Dominance comparison between same ranked services
If Sn >= Sk for all
No
Ignore both Sn and Sk for future
Yes
Yes
Optimal Service set
Yes
End
If |OS| >1
No
If |OS| = 1
Yes
Optimal Service
No
Fitness Funcation

FIGURE 3.1: Architecture of Non-Dominated Sorting and Ranking of Service (NDS-ROS)

3.2 Architecture of NDS-ROS

The Non-dominated sorting and ranking of services (NDS-ROS) algorithm's main objective is to find the optimal results from the diverse list of available services as per the user requirements. In most of the existing approaches, several unnecessary or duplicate comparisons raise the overall time. The NDS-ROS algorithm does not alter the MOOP to an SOP; instead, it finds the optimal service by simultaneously considering multiple objectives. The optimal service evaluated considering the tradeoffs of numerous objectives. NDS-ROS algorithm focus on finding optimal service by simultaneously lowering the duplicate or un-necessary comparisons between the services. The proposed dominance rules applied for comparing services that improve the searching process faster and the overall search process's efficiency.

The NDS-ROS architecture in FIGURE 3.1 shows the working of the algorithm.

3.2.1 Candidate Service Filtration

The filtration step takes the preliminary processing of requests from potential customers who submit their QoS attributes' requirements and priorities. When a SaaS consumer submits a request, the filtration process will filter out the cloud services that meet its needs. i.e., all the cloud services which fall into the range of user requirements are filtered out and stored in the candidate service set. E.g., the customer requirements are taken in the pre-defined range [90%-100%] for availability, [200ms- 250ms] for response time for the bi-objective optimization problem. The objective function's priority is taken in numeric order, which will later be used to calculate the fitness function.

There are two cases in Candidate Service Filteration:

1. **Case 1:** The user requirements do not match the QoS value of the service; therefore, the filtration step will not filter out any candidate service, i.e., the candidate set will be empty, then either we can retake the requirements from the user, or we can include all the available service in the candidate service set for further process.
2. **Case 2:** In the filtration step, the candidate service set filter our singleton service, an optimal service. Therefore no further process or action is required.

3.2.2 Candidate Service Sorting

The candidate services stored in the candidate service set are picked one by one, and merge sort is applied. The sorting (ascending /descending) order is done using the QoS values of each candidate service. Once the services are sorted, these services are stored in each of the objective function set.

3.2.3 Candidate Service Ranking

The sorted candidate services from each objective function set are taken and ranked in ascending order if the services sorted for minimization objective; otherwise, these services ranked in descending order. The ranking process is repeated for all objective function set and also for all the candidate services.

3.2.4 Dominance Comparison

The same ranked services from different objective functions are taken for pairwise dominance comparisons. The service which dominates another service in all the objectives is stored in the optimal solution set. Two services are non-dominated to each other if one of the services is non-dominating to another service in at least one of the objectives, and both the services are removed from the comparison list. There are two cases in Dominance Comparison:

1. **Case 1:** If only one service is found in the optimal service set after dominance comparison, then the candidate service will be considered an optimal service.
2. **Case 2:** If there are two or more than two services in the optimal service set, then these services are sent again for pairwise dominance comparison, and the process continues till the optimal service set has a single service. Suppose in case the optimal service set has two or more non-dominating services. In that case, the fitness function is calculated using the priority objective taken from the user in the filtration step. The fitness value for each non-dominating service is calculated. In the case of minimization objective function, the service whose fitness value is least is selected as optimal service; otherwise, the service whose fitness value is highest is the optimal service.

3.3 Illustration of Proposed NDS-ROS approach through a working example

The population S comprises ten services and a bi-objective minimization function (Cost and Response Time) shown in TABLE 3.1. The first column shows the service list and the values of each objective. The following four tables (filtration, sorting, ranking, and dominance comparison) show the optimal service evaluation. The following steps previously described in reaching the optimal service are as follows:

Table 3.1: Illustrative Example of NDS-ROS approach

Available Services	**Cost (Rs/Hr)**	**Response Time(no of services request/sec)**
S1	52	3.5
S2	60	3
S3	70	2
S4	10	6
S5	25	2
S6	30	1
S7	90	4
S8	110	2
S9	56	3
S10	80	4

TABLE 3.1 shows the list of available services S1, S2.......S10, the cost, and response time offered for each service mentioned here.

Table 3.2: Filtration step for candidate services

Available Services	**Cost (Rs/Hr)**	**Response Time(no of services request/sec)**
User Requirements	Cost [40 – 90]	RT [2 – 5]
S1	50	3.5
S2	60	3
S3	70	2
S7	90	4
S9	56	3
S10	80	4
Candidate Service Set | CS|	{ s1, s2,s3,s7,s9, and s10}	

The user requirements are taken, and candidate services are taken out of the table.

Step 1: The user requirements are taken in range, the cost of the service mentioned by user ranges between [40-90] rs/ hr, and the response time range is between [2-5] seconds. The provider's QoS attributes values for the services are matched with the QoS attributes mentioned by the user. The services which fall into the user-specified range in all the objective functions are filtered out, and they will go in the next round. Therefore the services fall in the range for both the objectives are stored in the candidate service set { s1, s2,s3,s7,s9, and s10}.

Table 3.3: Sorting for candidate services

Available Services	**Cost (Rs/Hr)**	**Response Time(no of services request/sec)**
S1	S1(50)	S3(2)
S2	S9(56)	S2(3)
S3	S2(60)	S9(3)
S7	S3(73)	S1(3.5)
S9	S10(80)	S7(4)
S10	S9(90)	S10(4)

Step 2: The services are sorted for cost and later for response time in ascending order as the objectives are minimization objective function. The sorted services in objective set 1 are { s1,s9,s2,s3,s10,s9} and the sorted services in objective set 2 are { s3,s2,s9,s1,s7,s10}.

Table 3.4: A ranking step for candidate services

Available Services	**Cost (Rs/Hr)**	**Response Time(no of services request/sec)**
R1	S1(50)	S3(2)
R2	S9(56)	S2(3)
R3	S2(60)	S9(3)
R4	S3(73)	S1(3.5)
R5	S10(80)	S7(4)
R6	S9(90)	S10(4)

Step 3: The sorted services are ranked in the ascending order in objective set 1 { r1-s1, r2-s9, r3-s2,r4-s3,r5-s10,r6-s9} and the ranked services in objective set 2 are { r1- s3,r2- s2,r3-s9, r4- s1,r5-s7, r6- s10}.

Table 3.5: Non-dominated - Dominance Comparison

Rank [Candidate Services]	**Dominance Comparison**	**Optimal Service Set**
R1	(S1,S3)	Non-Dominated [Ignore both services]
R2	(S9, S2)	S9 Dominates [store for future comparison]
R3	(S2, S9)	Already compared [Ignore]
R4	(S3,S1)	Already compared [Ignore]
R5	(S10, S7)	Non-Dominated [Ignore both services]
R6	(S9,S10)	S9 Dominates [store for future comparison]
Optimal Service Set | OS |	S9 is the optimal service	

Step 4: The same ranked services are compared { s1,s3},{s9,s2},{s7,s10},{s9,s10} and then optimal service is evaluated using dominance relationship between the compared service. The dominance rules mentioned in section 4.3 are applied to do the comparison. The table [1-5] shows that the non-dominated services not considered for further comparison; therefore, {s1,s2,s3,s7, and s10} are ignored to be sent in the optimal set, but {s9} is the dominating service. Therefore it is the optimal service and stored in the optimal service set.

3.4 Assumptions in NDS-ROS algorithm

The NDS-ROS algorithm works on discovering and ranking appropriate services considering the user's necessities and finding optimal services from available services by considering multiple objectives simultaneously. The proposed approach aims to maximize the user's objective and minimize the deviation from the requirement to less essential services. The cloud service ranking is modelled as a MOOP and ranks cloud services on objective parameters. The assumptions of the proposed approach NDS-ROS shown in the section. The objective assessments are modelled in the sub-sections below, based on the service modelling, objective functions modelling, and QoS values modelling.

The assumptions proposed in NDS-ROS are mentioned here to select and find optimal quantitative parameters for users to enter the range requirements. The NDS-ROS algorithm reduces the number of duplicate and un-necessary comparisons involved to find optimal service. Cloud service selection and ranking are modelled as a multi-objective optimization problem. The entities involved in NDS-ROS below include users, cloud service, QoS attributes, and Objective functions.

3.4.1 Assumptions on Number of Users

The User Set U represents the total number of the user requesting the services. The process of modelling the users is shown in equation (3.1), (3.2), (3.3), and (3.4). The modelling process of users is depicted below:

$$U = \{u_1, u_2, u_3, u_4, , \dots\dots\dots u_j, \dots\dots\dots J\} \tag{3.1}$$

$$U_j = \{S_n, R_j[min - max], P_j\} \tag{3.2}$$

$$R_j = \{R_{j1}, R_{j2}, R_{j3} \dots\dots\dots, R_{jm}, \dots\dots\dots, R_{jM}\} \tag{3.3}$$

$$P_j = \{P_{j1}, P_{j2}, P_{j3} \dots\dots\dots, P_{jm}, \dots\dots\dots, P_{jM}\} \tag{3.4}$$

U is the set that includes one of the user uj, where j lies between 1 and J if we have total J users. Rj is an ordered pair, including QoS requirement for the jth user. Each Rjm has a range from (min Rjm) to (max Rjm). Pj refers to the user's priority and showing the mth QoS coefficient of the jth user.

3.4.2 Assumptions on the number of Services

The service set is represented by S, and the total number of services is N, where Sn, lies between 1 and N. Every service has some pre-defined QoS attributes. The cloud service modelling is shown in equation (3.5) and (3.6).

$$S = \{s_1, s_2, s_3, u_4, , \dots\dots\dots s_n, \dots\dots\dots N\} \tag{3.5}$$

$$S_n = \{q_{n1}, q_{n2}, q_{n3} \dots\dots\dots, q_{nm}, \dots\dots\dots, M\} \tag{3.6}$$

Each Sn has its own QoS attributes, so Sn represents an ordered pair in which qnm shows the mth QoS attribute for nth service. If there are total M qualitative attributes, then it will be represented by M.

TABLE 3.6: Main Variables in NDS-ROS

N	Number of Services	CS	Candidate services set
M	Number of Quality attributes	P_{jm}	jth user priority
U	Number of Users	R_j	QoS requirement of jth User
U_j	jth User	S_n	nth Service
qnm	mth QoS attribute of nth service	K	Number of candidate services

3.4.3 Assumptions on the QoS attributes

The QoS attribute models all the users and services with the quality attributes defined in previous chapter 2. The QoS methods, description, and units od QoS for measurement are mentioned in Table 3. The proposed approach was evaluated on SaaS prominent attributes of QoS shown in figure.

3.5 Mathematical Modeling of NDS-ROS

1. **Filtration Step-** The user Um requests for the service Sn and enters the requirements Rm for QoS attributes. The needs are taken in range and then matched with the list of available services in service set S. Thus, the candidate services filtered in this step will go in the candidate service set CS[n]. The candidate service is evaluated using equation (3.7)

$$CS_j[n] = Filtration(F)\{R_j, P_j, S\} \tag{3.7}$$

Where 1 $\leq$ n $\leq$ N, $CS_j[n]$ is the obtained candidate services for the jth user. The input data set includes R_j as the jth user's requirement, P_j is the priority taken from the jth user related to QoS attributes, and S is the service set. The services are filtered based on QoS attributes, where some parameters are maximized, and some are minimized. Suppose a parameter is maximized and b parameter to be minimized (a+b = M), as shown in equation (3.8) and (3.9).

$$F = \sum_{i=1}^{N}(\sum_{m=1}^{a}(R_j)) + N_{i=1}(\sum_{m=1}^{b}(R_j)) \tag{3.8}$$

$$R_j = U_jqm[min] \leq qnm \leq U_jqm[max] \tag{3.9}$$

2. **Sorting-** In the candidate service set CS[n], the merge sorting step is applied if $n \geq 2$. Therefore the candidate service set should have at least two services before the sorted function is applied on the candidate set. The sorted candidate service for each objective function evaluated using equation (3.10)

$$CO_m[Sn] = \sum_{m=1}^{M}(sortcandidateservice(CS[S_n])) \tag{3.10}$$

For all objectives (1 $\leq$ m $\leq$ M), we iteratively sort the candidate services by each objective m. The candidate objective set of the nth service sn, in this order, is composed of preceding solutions in the ordering $sn.CO_m$= $\{s_1$,s_2,. . . . . $s_{n-1}\}$.

3. **Ranking-** The sorted candidate service for each objective function is evaluated using the following equation (3.11)

$$RCO_m[Sn] = \sum_{m=1}^{M}(\sum_{n=1}^{N}(R[S_n]) \tag{3.11}$$

$$R[S_n] = r + 0.5 * (s - 1) \tag{3.12}$$

The rank r is initialized to 1 and will go to CS[n], s is also initialized to 1 and will remain one till the time duplicate services are seen in the objective set.

4. **Dominance Comparison-** The Pareto dominance relationship is discussed in this section. The idea of dominance comparison is to find the dominance relationship among the same ranked services lying in different objective functions. The dominance comparison will lead to optimal service stored in the optimal set. There are two possibilities in the dominance comparison.

The dominance comparison occurs between two different services having the same rank and indifferent objective functions.

$$OS = \sum_{m=1}^{M}(rank(si(qm))) \leq rank(sn(qm)) \tag{3.13}$$

| OS | = si

The dominance comparison between same ranked service and in the different objective function

$$OS = \sum_{m=1}^{M}(rank(sn(qm))) \leq rank(sn(qm)) \tag{3.14}$$

| OS | = sn

3.5.1 Dominance Comparison Rules

1. Compare services with the same rank in the different objective functions.
2. If the compared service is non-dominated to each other, then both the services are removed from the comparison set or in further comparison step.
3. If one service dominates another service in dominance comparison, then the dominating service is included in the optimal service set.

4. If optimal service has more than one dominating service, then the services again go for dominance comparison.
5. If two service has already compared in the previous round, then services are not compared again.
6. The execution of the algorithm will stop in two cases
   (a) If the optimal service set has only a single service after dominance comparison
   (b) If the optimal service set has all the non-dominating service, then the decision strategy(fitness function) is applied, and the optimal service is found. In the maximization objective function, the service has the highest value taken as an optimal value. In the case of minimization objective function, the service has the lowest value taken as optimal service.

3.5.2 Fitness Function

The fitness function is the candidate solution which helps in finding the ideal solution as per the objective function specified by the customer and keep the solution in the optimal solution set. The NDS-ROS algorithm helps in identifying the optimal service therefore it will store the optimal service in the optimal service only after all the dominance comparison among the ranked candidate service is completed. There are two conditions for optimal service set.

1. The optimal service set has only a single service after dominance comparison, and the service is dominating service compared to other services in the dominance comparison. The service Sn is optimal service, and it fits in user requirements.
$$\mid OS \mid = sn = 1 \tag{3.15}$$
2. The optimal service set has more than one service, and these services are non-dominating to each other. In this scenario, the fitness function for all the non-dominating services calculated. The fitness function is computed using each objective's priority and multiply with the QoS value for that objective; the value is added to the QoS value and the importance. The process continues for all the services stored in the objective function. The service which has the minimum value (in case of minimization problem) is taken as the optimal solution. Equation (3.16) shows the calculation of the fitness function.
$$\mid OS \mid = sn > 1 \tag{3.16}$$

Fitness function

$$FitnessFunction(ffqn) = \sum_{m=1}^{M} [sn(pj * qm) * (pj + 1 * qm + 1)] \tag{3.17}$$

Sn represents the n^{th} service with m^{th} quality of service attribute and prioritizes the user for attribute m.

3.6 NDS-ROS Algorithm

The NDS-ROS algorithm has four stages in the algorithm, including filtration, sorting, ranking, and dominance comparison. The NDS-ROS inputs include the set of services, users requesting services, requirements, and priorities, and the result consists of an optimal service.

The NDS-ROS architecture is shown in section 4 for the steps involved in NDS-ROS. Algorithm 1 shows the overall stages involved in service selection and finding optimal service, and algorithm 2, algorithm 3, algorithm 4, and algorithm 5 shows the respective steps of NDS-ROS.

Algorithm 1 NDS-ROS Algorithm: Non-dominated sorting and ranking services

```
Input: A service set S, user requirement set $R_j$, user priority set $P_j$
Output: Optimal service $S_n$
Start

Step 1. Filtration(CS[n]) /* select the candidate services from set S  */
Step 2. Sorting(COm[CS[n]) /* Sort the candidate services              */
Step 3. Ranking(R[CS[n]]) /* rank the candidate services               */
Step 4. Dominance comparison(OS[Sn]) /* compare two services           */
End
```

3.6.1 Filtration

The filtration step filters out relevant services as per user requirements and stores them in the candidate service set. The step works on important objectives: to understand users' requirements and minimize not required services. The filtration step involves the following sub-steps: firstly, the user requirements are taken in the range [min max]. Secondly, the cloud services Qos values matched with user requirements. Finally, the filtered services are stored in the candidate service set.

There are three scenarios involved in this step:

Case 1. if the conditions [range] match, then the service is selected for candidate service set else service is ignored.
Case 2. if there is only singleton service in the candidate set, then the service is considered an optimal service.
Case 3. if none of the user requirements is matched, then all the services are selected, or user requirements are again taken.

Algorithm 2 Filtration step

```
Input: Service set S, $R_j$, $P_j$
Output: Candidate service set $CS_j[n]$
Start

| CS | ← ∅ /* Candidate key is empty */
Filtration | CS | = select services from S such as $S_n(\min R_{jm} \leq$ qjn$\leq \max R_{jm})$
CS= $S_n$
if $CS= S_n$ then
    OS[n] = $S_n$ /* n=1, optimal service */
end
if $CS[n] = 0$ then
    CS[n] = | S | /* no service fits in range */
end
if $CS[n] = N$ then
    CS[n] = sorting(CS[n]) /* where n= 1 to N */
end
End
```

3.6.2 Sorting

The merge sort technique is applied to all the candidate services. Here, all the candidate services are sorted for each objective function. The ties are broken using lexicographical comparisons. The idea behind using merge sorting is its quickest response time, even for a large dataset, and having low complexity compared to other techniques. The NDS-ROS becomes efficient in the sorting process due to the merge sort. The sorting process involves sorting the candidate services from the first objective and then iteratively sorts the services for the rest of the objective. The algorithm 3 will take the candidate set CS[n] as an input. The sorting of candidate services is done in ascending order for each objective.

```
Algorithm 3 Sorting step
Input: Candidate Services N and Objective M
Output: Sorted set of services in COm
Start

COm ← ∅ /* Objective set is empty                                    */
for n= 1 to N do
    COm ← sorting(CS[n]) by mth objective /* Merge sort is used      */
    COm [Sn] ← CS[Sn]
end
Isranked(COm [Sn] ) ← false go to Algorithm 4
return COm [Sn] /* sorted services for each objective                */
End
```

3.6.3 Ranking

The third stage of the NDS-ROS algorithm includes ranking of the candidate services which are stored in the objective function set using the algorithm 3. The candidate services are already sorted here and therefore each of the candidate service is assigned a rank. The process of ranking of the candidate service is mentioned in algorithm 4 where the ranking function will assign ranks to the sorted candidate services (CS[Sn]). In this step, the sorted services are picked one by one from the objective set, and services are ranked for each objective function. The ranks are assigned in the same order in which the candidate services are sorted.

```
Algorithm 4 Ranking step
Input: Candidate objective set COm [Sn]
Output: Ranked candidate services R(CS[Sn])
Start

if ss ranked(COm [Sn] ) ← false then
    select Sn from COm /* n = 1 to N                                  */
    ${L_m}^{R(CS[n])} \leftarrow {L_m}^{R(CS[n])}$ U Sn /* include Sn to ${L_m}^{R(S)}$ [N*M]matrix */
end
if ranked(Sn) ← True then
    ${L_m}^{R(CS[n])} \leftarrow {L_m}^{R(CS[n])}$ U Sn /* s is already ranked */
end
return R(CS[Sn]) /* services ranked in each objective                */
End
```

3.6.4 Dominance Comparison

The algorithm 5 works on the finding of the Pareto-solution from the list of the ranked candidate services. The comparison among the candidate service will occur only through the dominance comparison rules mentioned in 3.5.1. which makes sure that duplicate comparisons or comparisons which can be avoided should be removed from the comparison set. The algorithm starts with picking up those services having the same rank and puts them in the dominance comparison set where the services are then compared pairwise to find their dominance relationships. Through the dominance comparison the rules will be followed which will eventually reduce the comparisons to identify the optimal service and remove duplicate comparisons from the list.

3.7 Summary

The NDS-ROS approach is summarized in algorithm 1. The candidate services are stored for each objective function separately and are later used to rank the candidate service for each objective function. The complete working of the proposed algorithm NDS-ROS algorithm works in four different steps. The user requests collected are converted in the form of objective functions, and then the services are then filtered from the massive list of accessible services. The filtered services should meet the QoS attribute value for each objective function mentioned by the user. The filtered services are stored in the candidate service set. In the second step, the filtered services sorted using a merge sort for each objective function in ascending order. When two services have the same parameter value for the same objective function, any service is considered on top of another service. It will not affect the complete ranking of the services or in the calculation of the optimal service. The sorted services for the objective functions are stored in the ascending order for minimization objective function. In the third step, the sorted services in the objective function set are ranked for the individual objective function set and are assigned Pareto fronts to them in the same rank order. The vital thing to take care of is the number of Pareto fronts is equal to candidate services. The fourth step involves comparing the services under different objective functions but has the same rank, and the dominated services are evaluated and stored in a set. If, in the end, there are two or more services in the dominated service set, then these services are further compared for dominance cases. If there is a single service in the dominance comparison set, then that service is considered as an optimal service required by the user.

Algorithm 5 Dominance Comparison

```
Input: Services $s_i$ , $s_n$
Output: optimal solution set OS[Sn]
Start

for $m.R(CS[s_n]) = k.\ R(CS[s_i])$ do
    different objectives /* same ranked services from */
    foreach COm (qm) do
        if $R(CS[Sn]) \Leftarrow R(CS[Si])$ then
            OS[Sn] = CS[Sn]
        end
        else
            ignore services (CS[Sn], CS[Sj] )
        end
        Repeat /* different services with same rank */
    end
end
if $|OS[Sn]| = 1$ then
    OS[n] = Sn /* optimal service */
end
else
    if optimal set $OS[n] \geq 2$ then
        /* dominating service */
        (OS[Sn,Sj])
    end
    if optimal set $OS[n] \geq 2$ then
        /* non-dominating service */
        FITNESSFUNCTION(ff) [Sn,Sj,Sk ]
    end
end
/* Calculate Fitness Function */
if ff [Sn] < ff [Sj] then
    |OS[n]| = Sn /* optimal service will be stored */
end
else
    if fitfun [Sn] < fitfun [Sj] then
        (OS[Sn,Sj]) /* Both Services are Optimal */
    end
end
End
```

Chapter 4

Experimental Environment

The chapter discusses the experimental environment used in the execution of the NDS-ROS algorithm. The efficiency of the NDS-ROS algorithm's along with three well-known algorithms: deductive sort, ENS-SS, and GBOS-SS. The experiments were conducted to compare the efficiency that a SaaS service user achieves when adopted a non-dominated sorting and ranking of services (NDS-ROS) algorithm. The experiments conducted to analyze the performance of all four algorithms based on the comparisons and execution time:

4.1 Cloud Dataset

The initial population is taken from the data set on cloud services collected from Cloud service data set from Kaggle. The data set contains the initial population with a diversity of services. The NDS-ROS algorithm's filtration step helps in filtering necessary services, and these services are called candidate services stored in the candidate service set. The above four algorithms are compared on two different performance parameters:

1. In this test, the varying number of candidate services [100-250] and the fixed number of objectives ranging from [2 to 5].
2. This test fixed the number of candidate services [100-250] and the varying objectives ranging from [2 to 5].

Execution Time required on Candidate Services K

1. In this test, the varying no candidate services [100-250] and the fixed number of objectives ranging from [2 to 5].

2. This test fixed the number of candidate services [100-250] at an increment of 50 and the varying objectives ranging from [2 to 5].

The simulator used in conducting experiments is JMetal Simulator, and the algorithm implemented in java. The results are evaluated for the proposed NDS-ROS algorithm with three existing algorithms. The performance parameter used in the study to calculate the NDS-ROS algorithm's efficiency includes the number of comparisons, total execution time is taken, and cyclomatic complexity. The NDS-ROS algorithm compares with all three algorithms discussed above, i.e., deductive sort, ENS-SS, and GBOS-SS. The computer and software versions used have the following features:

1. Windows 10 Standard
2. Intel i5 Processor with 4GHz.
3. 4 GB of RAM Memory
4. Java Version 1.8.0-121.64 bits
5. Simulator - J Metal

4.2 J Metal Simulator

The multi-objective optimization problem has increased attention as there is no single solution or point that augments all the objective functions simultaneously; thus, a MOP optimum consists of a set of points known as the Pareto optimal set. To obtain Pareto front of a MOP is the main objective of multi-objective optimization. The JMetal simulator helps solve multi-objective optimization problems and provides the users with an easy-to-use, flexible, and extensible tool. And also assist in the overall process of researching with metaheuristics algorithms. JMetal helps a range of applications, experimentation, and studying metaheuristics for resolving multi-objective optimization problems. jMetal contains both traditional and current optimizers and is efficient in managing benchmark problems and quality indicators to evaluate the algorithms' performance. The framework supports full experimental studies by providing complete jMetal's graphical interface. JMetal also generates statistical information automatically and execute faster experiments.

4.3 Performance Parameter

The most crucial aspect of any algorithm is the performance, which tells the efficiency of that algorithm in carrying out a specific task. The performance parameter includes the algorithm's output generated in terms of computational resources, comparisons required, and execution time to find the optimal service. The NDS-ROS algorithm finds a set of solutions as close as possible to the optimal solutions. The three performance parameter used in the research to evaluate the performance of the NDS-ROS described below:

4.3.1 Number of Comparisons

The number of comparisons is the number of times the available candidate services get compared to give the result. To evaluate the NDS-ROS algorithm's efficiency with deductive sort, ENS-SS, and GBOS-SS, the number of comparisons recorded for the services ranging [100-250]. These comparisons were measured at objective sizes 2, 3, 4, and 5, respectively, to find the optimal service. The assessment criteria and the data set used in the experiment are the same for all the four algorithms. The comparison among the algorithm shows all the four algorithms' results if the objective functions increase and the services increase. The pairwise dominance comparison is made on the service having the same rank but is stored in the different objective sets. The total number of dominance comparisons taken by each of the algorithms is calculated. The algorithm with the least number of dominance comparisons is the efficient algorithm, and the NDS-ROS algorithm was efficient on the criteria.

4.3.2 Execution Time

The execution time also measures the computational efficiency of the proposed algorithm. The lesser the execution time, the faster the customer will get the optimal service, reducing his waiting time. The execution time is also directly proportional to the number of comparisons required until the optimal service is found. The lesser number of comparisons, the lesser execution time, and the algorithm will be more efficient. The four algorithms executed on the same set of parameters, and the execution time calculated for all the four algorithms. The result shows that the NDS-ROS algorithm yields faster results.

4.3.3 Computational Complexity

Computational complexity is the classification of the algorithm in terms of resource usage. A computation problem is solvable by applying some logic. An algorithm's complexity measures the amount of time and space taken by an algorithm to input a given size (n). The study calculates the computational complexity for all the four algorithms for time and space. The experimental results indicate that the NDS-ROS algorithm is found efficient in this parameter.

4.4 Computational Complexity

The standard method to judge an approach's efficiency is to compare an algorithm's overall performance and count the number of performance parameters on which an algorithm is an overall winner. To validate the results received from the algorithm is done through a sign test. The sign test is used as a statistical test that compares the NDS-ROS with other algorithms. The difference in performance scores of the two algorithms on a problem should be significant. The hypothesis testing intends to formally examine the two opposing conjectures (hypotheses), H0 and HA.

4.4.1 ANOVA

Analysis of variance (ANOVA) is a statistical tool that identifies the mean variances among the compared experimental groups. ANOVA conduct the empirical study with one dependent variable and measures the parametric outcomes along with numerous experimental groups within one or more independent variables. The independent variables in ANOVA are called factors, and levels are the different groups in each element. ANOVA computes the partitioning of variance, interactions, factors, a sum of squares, mean squares, F scores, post hoc tests, effect size, statistical power, etc. The one-way ANOVA matches the means among the groups and decides whether any of those means are statistically significantly dissimilar from each other. Specifically, it tests the null hypothesis and shown in equation (4.1)

$$H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4......\mu_k \tag{4.1}$$

where μ = group mean and k = number of groups.

Suppose the results of one-way ANOVA are statistically significant. In that case, the alternative hypothesis (Ha) is accepted, which means that group means that are statically significantly different from each There is also a two-way analysis of variance to identify the independent variables on one unceasing dependent variable.

The ANOVA also checks the impact on the independent variables on a dependent variable and the expected outcome.

4.5 Summary

The sample data taken from the population is used from the data set on cloud services from the Kaggle. The four algorithms deductive sort, ENS-SS, GBOS-SS and NDS-ROS will be compared in the next chapter on the basis of parameter including, the number of comparisons, total execution time, and computational complexity. The simulator which will be used for the experimental evaluation is "Jmetal" which is found to be an efficient simulator in solving multi-objective optimization problem. Analysis of Variance (ANOVA) a statistical tool will be used to find the variance in all the four algorithms.

Chapter 5

Experimental Results and Performance Analysis

The chapter discussed a complete analysis to show the NDS-ROS algorithm's performance by doing a comparative study. The performance analysis's primary goal is two-fold (1) to compare the NDS-ROS with deductive sort, ENS-SS, GBOS-SS approach to define the amount of enhancement in the outcomes (2) to raise the part of NDS-ROS towards mounting an agreement on the optimal service. The experiments conducted on the dataset with the candidate services range between [100-250] and objectives size [2,3,4 and 5]. All the four algorithms are executed in the Jmetal simulator on windows 10 PC with a 4 GHz Intel i5 processor and 4 GB of RAM.

5.1 Performance Analysis

The experimental study of the four algorithms [Deductive sort, ENS-SS, and GBOS-SS and NDS-ROS] compares them on three different performance parameters:

1. Computational complexity:- The computational complexity of GBOS-SS, Deductive sort, ENS-SS, and NDS-ROS algorithm is calculated for the worst case and best case time complexity.

2. Comparisons Required:- The comparisons required among the services to find the optimal service.

3. Execution Time:- The total time taken by the algorithm in giving the optimal service

5.2 Computational Complexity

The complexity of any algorithm depends on the methodology involved in finding the solution. This section briefly explains the deductive sort, ENS-SS, and GBOS-SS algorithm and their time complexity taken from the existing literature. The time complexity of the NDS-ROS converses in detail. Best and worst time case complexity is evaluated for all the four steps, i.e., filtration, sorting, ranking, and dominance comparison.

5.2.1 Deductive Sort Algorithm

The deductive approach begins with matching solution P1 with other solutions one by one. The number of comparisons by P1 will continue till the time any other solution does not dominate P1. As soon as P1 dominates by some other solution in the list or is dominated by other solutions, then the dominating solution is assigned to the front F1, and P1 will no longer be involved in further comparing the front F1. Similarly, solution P2, P3, etc. are compared with each solution in the list. The dominance comparison will last until all the solutions are allocated fronts. In deductive sort, there are many unnecessary comparisons, and many of them are also duplicate comparisons.

5.2.2 ENS-SS Algorithm

ENS-SS approach is different from other approaches in the way comparisons between the solutions is done. Usually, the other approaches compare the solutions before assigning it to the front, while the ENS approach matches the solution with only those assigned to the front. In ENS, the population is sorted first based on the first objective before the ENS approach is applied. Thus, the solutions included to the front will not dominate any solution added before, and as a result, ENS can escape only some of the duplicate comparisons.

5.2.3 GBOS-SS

Generalized Binary Order sequential search (GBOS-SS) is an efficient algorithm and is an extension of ENS. The approach uses a novel binary order search to speed up sorting and rapidly explores through the front to check if the solution is dominated. GBOS-SS can reduce the number of duplicate comparisons only but can handle large population size and objectives.

5.2.4 NDS-ROS Algorithm

The NDS-ROS algorithm's computational complexity to find optimal service depends on four steps: filtration, sorting ranking, and dominance comparison. The best and worst time case complexity is calculated for all four steps separately, and the full time complexity of the algorithm is calculated. T represents the time complexity, N is the total service list, and M represents the size of the total objective.

1. **Filtration Step:** In the filtration step, the QoS value of the available services gets compared with user requirements. As soon as the services are matched successfully with needs, they are filtered to the candidate set. The algorithm 2 shows the total number of available services "N," and all these services must be searched at least once. Therefore, the worst-time complexity infiltration algorithm T(n) is O (n). Even if first service gets matched with user requirements for best-case, the algorithm still checks all the services and looks for its match. Therefore, the best case time complexity is T(n) is O(n). There are total M objectives; thus, time complexity becomes O(NM). The worst and best case time complexity is the same in the filtration step shown in equation (5.1), (5.2) & (5.3). Since there are total N services n1,n2,. . . . . . .,N, therefore the total number of matching required are N. Once the service is matched, it will be removed from the list; therefore, next matching will be from (N-1) comparisons and so on, as shown in equation (5.2). The number of objectives is m1, m2,. . . . . , M.

$$Num\_matched = \sum_{m=1}^{M} N \tag{5.1}$$

Time Complexity [Worst Case/ Best Case]

$$Timecomplexity_{worst/best} = M.Num\_comp_{worst/best} \tag{5.2}$$

$$Timecomplexity_{worst/best} = O(MN) \tag{5.3}$$

2. **Time Complexity: Sorting Step:** The merge sort technique is used in the NDS-ROS algorithm because it is considered more efficient and works faster than other sorting techniques, even on large datasets. The worst-case time complexity for sorting maximum N candidate services for each objective function is O(NlogN). The best-case time complexity for sorting full N candidate service is also O(NlogN). Therefore the time complexity for merge sort comes out to be the same in both the worst and the best case. Thus equation (5.4) & (5.5) shows the time complexity of sorting steps with M objectives as :

Worst Case/ Best Case Time Complexity

$$Timecomplexity_{worst/best} = O(MNLogN)+O(M-1)(NLogN)+O(M-2)(NLogN) \tag{5.4}$$

$$Timecomplexity_{worst/best} = O(MNLogN) \tag{5.5}$$

3. **Time Complexity: Ranking Step**
The Ranks are assigned to the sorted candidate service stored in each objective function, and the ranks given to the services in the same order as these services are sorted. Suppose there are total N candidate services n1,n2,. . . . . . .N for total M objective function m1,m2. . . ..M. Therefore for all the candidate services at least once be assigned the ranks. In case N services are in the worst-case time complexity, then the rank is assigned starting from service [1 to N]; therefore, the time complexity will be O(N). In case there is only a single service, then the best case complexity will be O(1). Therefore equations (5.6) and(5.7) shows the time complexity of ranking step.

Worst Time Complexity
$Timecomplexity_{worst}$ = M. Num_compworst = M. N

$$Timecomplexity_{worst} = O(MN) \tag{5.6}$$

Best Time Complexity
$Timecomplexity_{best}$ = M. Num_compbest = M. (1)

$$Timecomplexity_{best} = O(M) \tag{5.7}$$

4. **Time Complexity: Dominance Comparison:** The NDS-ROS algorithm computes pairwise dominance comparison for the same ranked services but from a different objective set at least once. In Dominance comparison, there will be two cases :

Case1: If the two different services have the same rank in the different objective functions. Therefore if there are total N services in the first objective set and total N services in the second objective set, then in worst-case time complexity, will have O(N2).

Case 2: If the two same services have the same rank in the different objective functions. Therefore if there are total N services in the first objective set and full N services in the second objective set, but as the service is the same, therefore best time complexity, the order will O(N).

If there are M objectives, the overall time complexity includes the worst, and best case is shown in equation (5.8) and (5.9)

Worst Time Complexity

$$Timecomplexity_{worst} = M.Num_{c}omp_{worst} = O(MN^2) \tag{5.8}$$

Best Time Complexity

$$Timecomplexity_{best} = M.Num_{c}omp_{best} = M.(N) = O(MN) \tag{5.9}$$

The complexity of an approach is calculated when all the steps of an algorithm are covered; therefore, the total worst-case time complexity is shown in (5.10) and best case time complexity (5.11) calculated from all the four steps: filtration, sorting, ranking, and dominance comparison.

The overall worst time complexity

$T_{worst} = T_{filtration} + T_{sort} + T_{ranked} + T_{dominancecheck}$

= O(MN) + O(MNLogN) + O(MN) + $O(MN^2)$

$$T_{worst} = O(MN^2) \tag{5.10}$$

The overall best time complexity

$T_{best} = T_{filtration} + T_{sort} + T_{ranked} + T_{dominancecheck}$

=O(MN) + O(MNLogN) + O(1) + O(MN)

$$T_{best} = O(MNLOGN) \tag{5.11}$$

TABLE 5.1 displays the computational complexity of different algorithms, the best and worst time complexity is calculated, and also, the space complexity is evaluated for all the four algorithms.

Table 5.1: Time and Space complexity Comparisons of all four algorithms

Algorithm	**Best-time complexity**	**Worst-time complexity**	**Space complexity**
NDS-ROS	O(MNlogN)	$O(MN^2)$	O(MN)
GBOS-SS	O(MNlogN)	$O(MN^2)$	O(MN)
ENS-SS	O(MN$\sqrt{N}$)	$O(MN^2)$	O(1)
Deductive sort	O(MN$\sqrt{N}$)	$O(MN^2)$)	O(N)

5.3 Experimental Outcomes

5.3.1 Number of Comparisons

To evaluate the comparisons required by all the four algorithms, NDS-ROS, Deductive Sort, ENS-SS, and GBOS-SS. The NDS-ROS's performance is analyzed from the experimental results, calculated using windows 10 with intel i5 processor, and implemented on Jmetal 4.0. The performance of all four algorithms compared to finding each approach's efficiency in the first performance parameter is the number of comparisons. The two experiments were carried out.

1. Fixed objective size from [2, 3,4, and 5] and with varying candidate service (k) [100 -250] with an increment of 10.

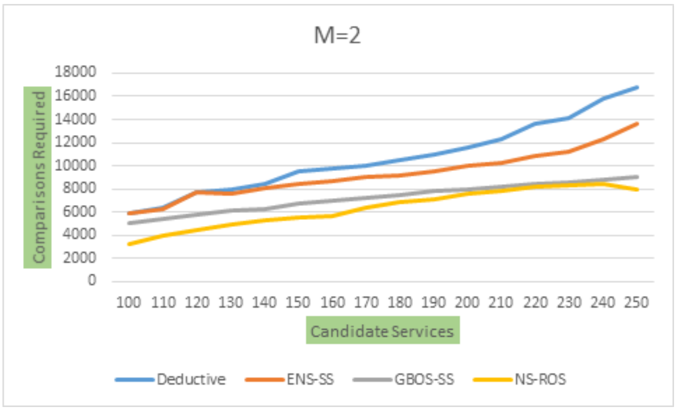


FIGURE 5.1: Dominance Comparisons at M=2

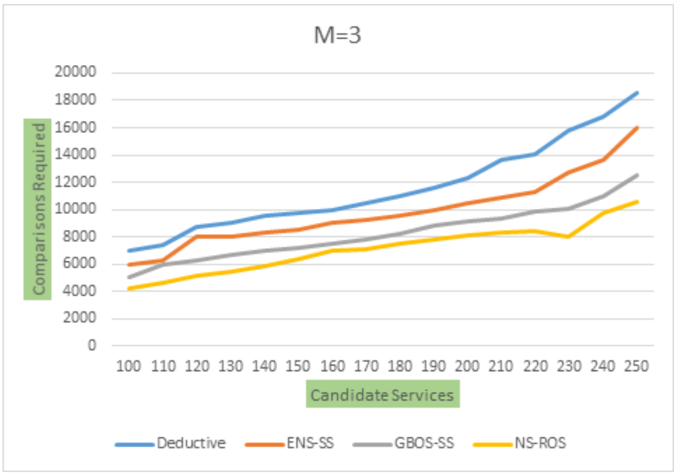


FIGURE 5.2: Dominance Comparisons at M=3

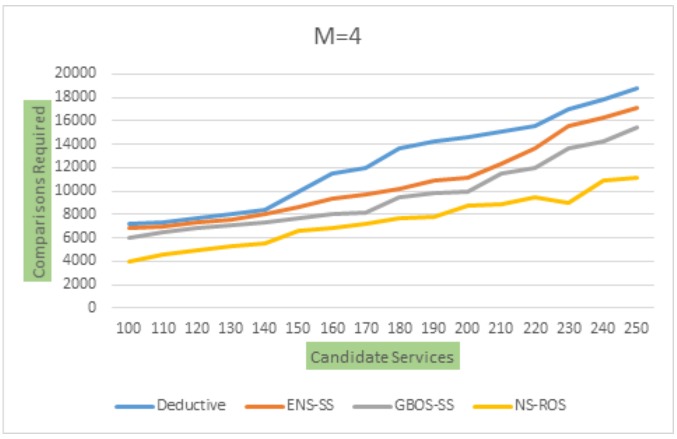


FIGURE 5.3: Dominance Comparisons at M=4

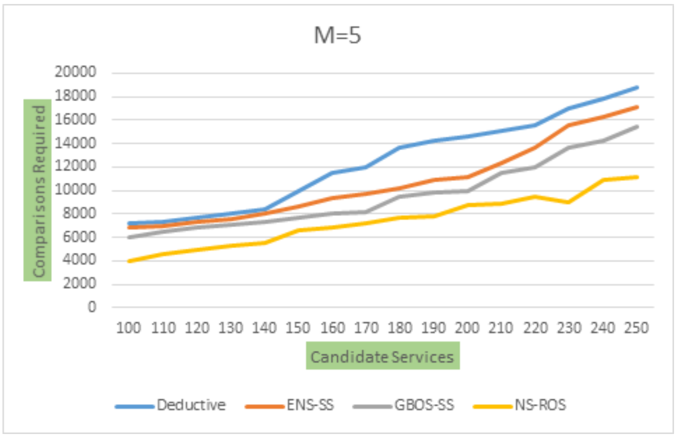


FIGURE 5.4: Dominance Comparisons at M=5

2. Fixed candidate service (K) between [100 -250] with an increment of 50 and a varying objective function [2, 3,4, and 5].

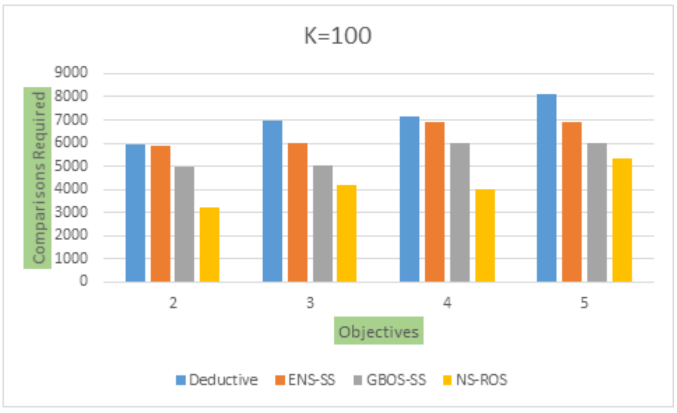


FIGURE 5.5: Dominance Comparisons at K=100

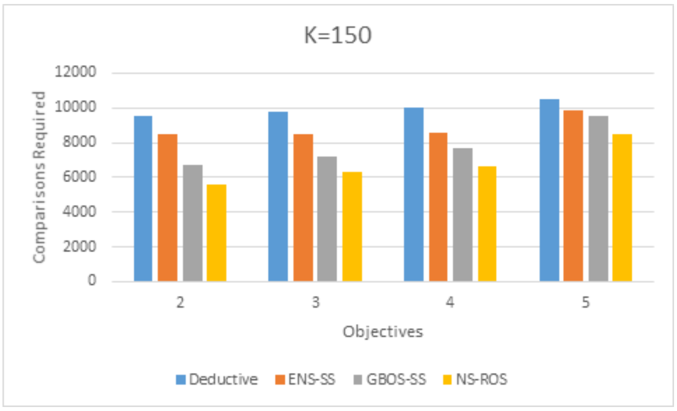


FIGURE 5.6: Dominance Comparisons at K=150

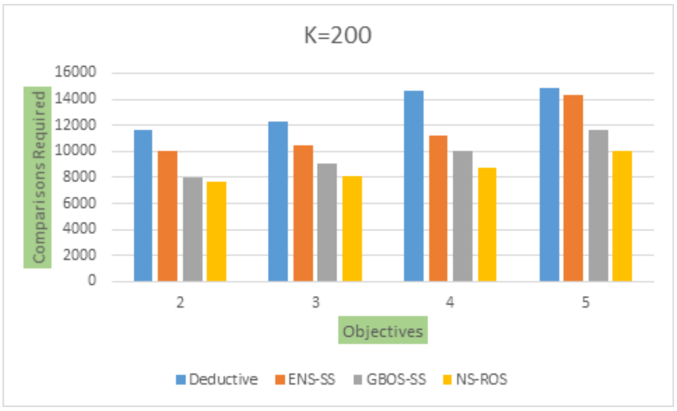


FIGURE 5.7: Dominance Comparisons at K=200

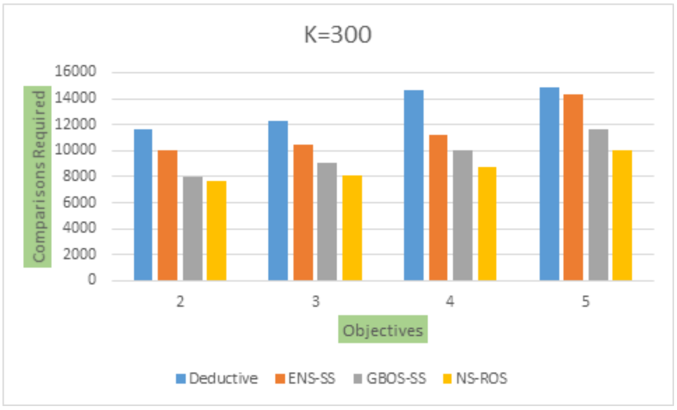


FIGURE 5.8: Dominance Comparisons at K=250

5.3.2 Execution Time

The NDS-ROS algorithm compared with three algorithms deductive Sort, ENS-SS, and GBOS-SS on the execution time. The NDS-ROS's performance is analyzed from the experimental results, calculated using windows 10 with intel i5 processor, and implemented on Jmetal 4.0. To compare, the algorithms, experiments are carried out.

1. Fixed objective size from [2, 3,4, and 5] and with varying candidate service (k) [100 -250] with an increment of 10

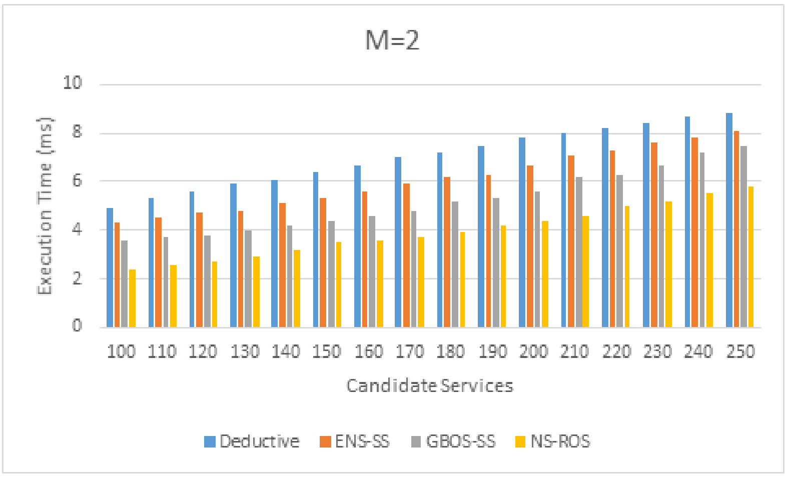


FIGURE 5.9: Execution Time at M=2

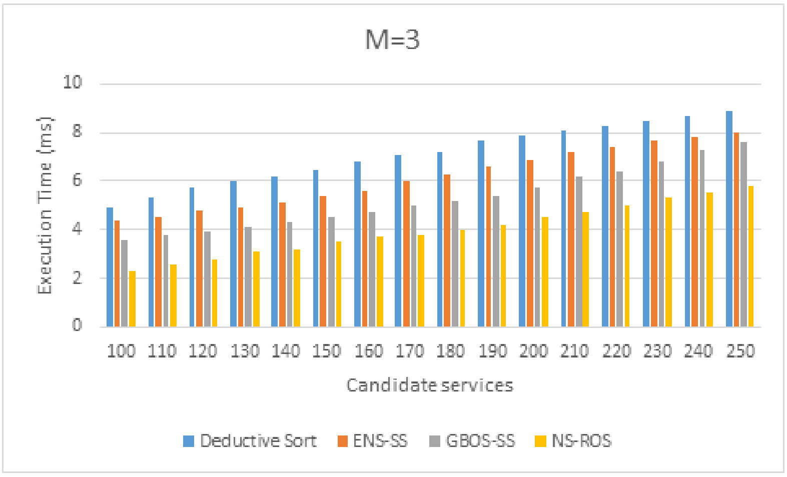


FIGURE 5.10: Execution Time at M=3

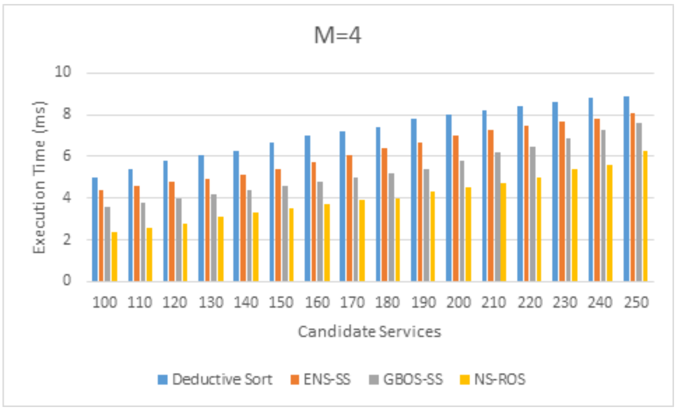


FIGURE 5.11: Execution Time at at M=4

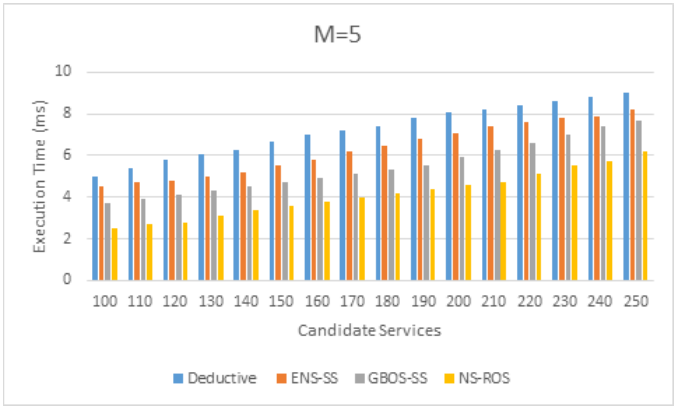


FIGURE 5.12: Execution Time at at M=5

2. Fixed candidate service (K) between [100 -250] with an increment of 50 and a varying objective function [2, 3,4, and 5].

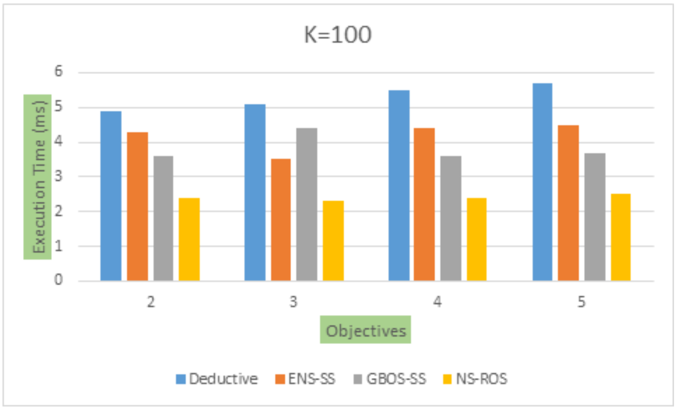


FIGURE 5.13: Execution Time at K=100

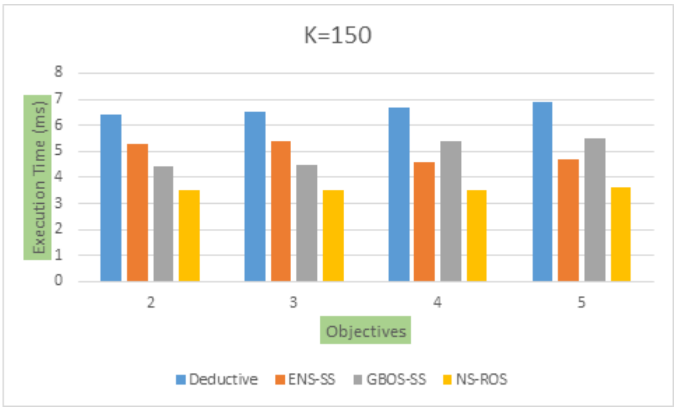


FIGURE 5.14: Execution Time at K=150

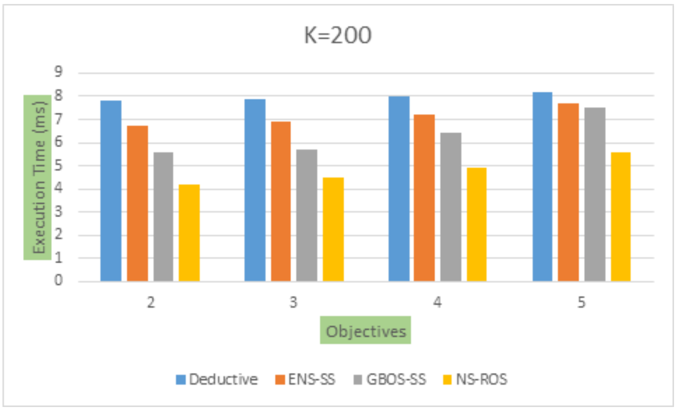


FIGURE 5.15: Execution Time at K=200

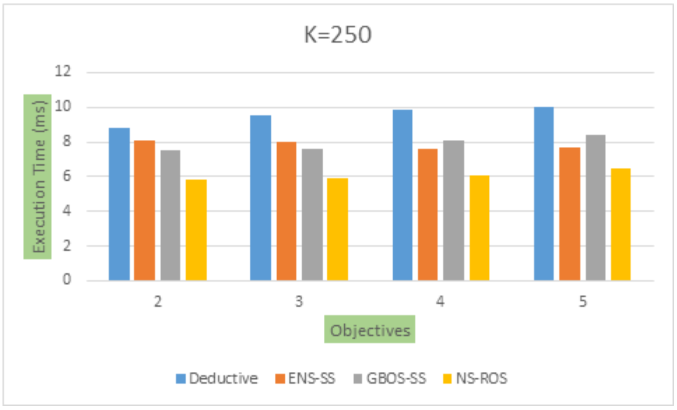


FIGURE 5.16: Execution Time at K=250

5.4 Validation of NDS-ROS

The statistical method will help decide if there is a significant difference among the average number of comparisons in Deductive Sort, ENS-SS, GBOS-SS, and NDS-ROS algorithms. The one-way ANOVA is adopted because the algorithms' mean and variance are unknown. ANOVA test determines if there is any significant difference among means of two or more independent groups. An ANOVA test is conducted on the number of comparisons is to find a substantial difference.

5.4.1 One-way analysis of variance (ANOVA)

The one-way ANOVA matches the means among the groups, defines the null hypothesis, and finds the means difference significantly differs. There are two types of views for a one-way ANOVA, null hypothesis and alternative hypothesis.

$$H_o : \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5.......\mu_k \tag{5.12}$$

where μ represents the mean of the group, and k represents the group size. The alternative hypothesis is accepted if the one-way ANOVA results are statistically significant; else, the alternative hypothesis (Ha) is accepted. The one-way ANOVA is not efficient in telling about the specific groups which are statistically expressively different from each other, and consequently, a post hoc test is required. The Deductive Sort (DS) is represented by "1", ENS-SS is represented by "2", GBOS-SS is represented by "3," and "4" algorithms represent NDS-ROS are compared based on one variable. One way ANOVA was conducted with the Dependent Variable (number of comparisons) and Independent variable (Deductive Sort, ENS-SS, GBOS-SS, NDS-ROS Algorithms). Homogeneity of variance was checked using Levene's test.

Algorithm	N	Mean	Stand. Deviation	Stand. Error	Lower Bound	Upper Bound	Minimum	Maximum
1[DS]	16	7058.750	1250.5619	312.6405	6392.373	7725.127	4900.0	8800.0
2 (ENS-SS)	16	7058.750	1250.5619	312.6405	6392.373	7725.127	4900.0	8800.0
3(GBOS-SS)	16	5215.313	1296.4612	324.1153	4524.477	5906.148	3590.0	7450.0
4(NDS-ROS)	16	3997.500	1062.5912	265.6478	3431.285	4563.715	2400.0	5800.0
Total	64	5832.578	1769.0200	221.1275	5390.690	6274.466	2400.0	8800.0

Table 5.2: ANOVA Test at M=2

Levene Statistic	df1	df2	Sig.
0.384	3	60	0.765

TABLE 5.3: Test of Homogeneity of Variances [M=2]

	Sum of squares	df	Mean Square	f	Sig.
Between Groups	108088376.172	3	36029458.724	24.272	.000
Within Groups	89065823.438	60	1484430.391		
Total	197154199.609	63			

TABLE 5.4: ANOVA Analysis at M=2

	Sum of squares	df	Mean Square	f	Sig.
Between Groups	108088376.172	3	36029458.724	24.272	.000
Within Groups	89065823.438	60	1484430.391		
Total	197154199.609	63			

TABLE 5.5: ANOVA Analysis at M=2

	(I)	(J)	Mean	fStd.Error	Sig.	Lower B	Uper B
TukeyHSD	1	2	.0000	430.7596	1.000	-1138.291	1138.291
		3	1843.4375*	430.7596	.000	705.146	2981.729
		4	3061.2500*	430.7596	.000	1922.959	4199.541
TukeyHSD	2	1	.0000	430.7596	1.000	-1138.291	1138.291
		3	1843.4375*	430.7596	.000	705.146	2981.729
		4	3061.2500*	430.7596	.000	1922.959	4199.541
TukeyHSD	3	1	-1843.4375*	430.7596	.000	-2981.729	-705.146
		2	-1843.4375*	430.7596	.000	-2981.729	-705.146
		4	1217.8125*	430.7596	.032	79.521	2356.104
TukeyHSD	4	1	-3061.2500*	430.7596	.000	-4199.541	-1922.959
		2	-3061.2500*	430.7596	.000	-4199.541	-1922.959
		3	-1217.8125*	430.7596	.032	-2356.104	-79.521
Bonferroni	1	2	.0000	430.7596	1.00	-1175.350	1175.350
		3	1843.4375*	430.7596	.000	668.088	3018.787
		4	3061.2500*	430.7596	.000	1885.900	4236.600
Bonferroni	2	1	.0000	430.7596	1.00	-1175.350	1175.350
		3	1843.4375*	430.7596	.000	668.088	3018.787

		4	3061.2500*	430.7596	.000	1885.900	4236.600
Bonferroni	3	1	-1843.4375*	430.7596	.000	-3018.787	-668.088
		2	-1843.4375*	430.7596	.000	-3018.787	-668.088
		4	1217.8125*	430.7596	.038	42.463	2393.162
Bonferroni	4	1	-3061.2500*	430.7596	.000	-4236.600	-1885.900
		2	-3061.2500*	430.7596	.000	-4236.600	-1885.900
		3	-1217.8125*	430.7596	.038	-2393.162	-42.463
Bonferroni	3	1	-1843.4375*	450.3274	.002	-3068.018	-618.857
		2	-1843.4375*	450.3274	.002	-3068.018	-618.857
		4	1217.8125*	419.0698	.033	75.786	2359.839
Bonferroni	4	1	-3061.2500*	410.2594	.000	-4178.464	-1944.036
		2	-3061.2500*	410.2594	.000	-4178.464	-1944.036
		3	-1217.8125*	419.0698	.033	-2359.839	-75.786

TABLE 5.6: Post Hoc Tests [Multiple Comparisons] at M=2

The output in the abow tables shows the result of Turkey's test. These results display subsets of the group with statistically similar means. The first subset contains the NDS-ROS algorithm, the second subset contains ENS-SS and GBOS-SS algorithm, and the third subset contains GBOS-SS and Deductive sort algorithm. The results display that ENS and GBOS-SS algorithm has similar means. Also, the deductive sort algorithm and the GBOS-SS algorithm have identical means. The only algorithm that has significantly different standards is the NDS-ROS algorithm. The test provides significant value for each subset, which shows that among the deductive sort, ENS-SS, GBOS-SS, and NDS-ROS algorithm that only the NDS-ROS algorithm is found to be significantly different (as indicated by the values of sig. that is less than 0.05)

5.5 Summary

Analysis of the result for Deductive Sort, ENS-SS, GBOS-SS, and NDS-ROS:

The one-way variance (ANOVA) for all the four algorithms, the Dependent Variable (number of comparisons) were found to be significantly affected by the Independent variable (Deductive Sort, ENS-SS, GBOS-SS, NDS-ROS Algorithms). The homogeneity of variance for all the four objective function [m=2 to 5] shows the values as .765, .741, .752,.791. There is no significant result showing that the requirement of homogeneity for variance is met, and the ANOVA test can be considered robust. Then ,F (3,60)= 24.27, p= .000 for M=2. Pairwise comparisons Using Tukey and Boferron with adjusted p-values showed there are significant differences in DV between algorithms. The F (3,60)= 18.822, p= .000 for M=3. Pairwise comparisons Using Tukey and Boferron with adjusted p-values showed there are significant differences in DV between algorithms. The F (3,60)= 19.842, p= .000 for M=4. Pairwise comparisons Using Tukey and Boferron with adjusted p-values showed there are significant differences in DV between algorithms. The F (3,60)= 19.318, p= .000 for M=5. Pairwise comparisons Using Tukey and Bonferroni with adjusted p-values showed that there were significant differences in DV between algorithms.

Chapter 6

Conclusion and Future Work

The chapter converses with the varying demands of users due to the expansion of technologies. One such technology is cloud computing, which offers a developed business model on a subscription basis for enabling the service providers to propose different services having quality attributes. Due to the huge list of services, discovery an appropriate service according to users' changing requirements has become a significant task. The service selection and ranking systems support the users in selecting applicable service considering customized requirements. The existing literature models the SaaS cloud selection as a multi-objective problem. The efficient QoS-based SaaS cloud selection approach is the need of an hour. The proposed a non-dominated sorting and ranking of service approach called NDS-ROS is efficient in (1) removing the research issue in the SaaS selection. (2) achieve services selection and ranking using QoS perspective; (3) removed the limitations in existing algorithms used for the multi-objective optimization problem, and (4) identify the optimal service. The algorithm considers QoS attributes to find the optimal service and motivates the cloud users and the decision-maker to opt for more cloud services as their waiting time is reduced. The NDS-ROS algorithm uses a filtration step to identify the candidate services that fulfil the consumer's initial necessities. The algorithm also sorts and then ranks the cloud services for each objective function. Lastly, all the same, ranked candidate services undergo pairwise dominance comparison. The fitness function is applied if an optimal set has more than two non-dominating services.

The non- dominated sorting algorithm NDS-ROS is established and applied to find optimal service in the SaaS cloud model by considering user's requirements. The proposed NDS-ROS algorithm's basic idea is to use a quick sorting algorithm in a non-dominated ranking. The comparative result of the deductive sort, ENS-SS, GBOS-SS, and the NDS-ROS algorithm is evaluated on the parameters, including computational complexity, number of comparisons, and execution time. The NDS-ROS algorithm outperforms the compared algorithms in all the parameters.

The experimental study conducted with a varying number of services and objectives shows that the NDS-ROS algorithm is efficient in execution time and comparisons. The complexity calculated for the algorithm for four steps – filtration, sorting, ranking, and dominance comparison has the worst time complexity as T worst= O(MN2), whereas the best time complexity as (MNlogN). The NDS-ROS algorithm's efficiency is improved due to its filtration, which reduces unnecessary services to a great extent by filtering them in the initial stage, and the dominance comparison removes all the duplicate comparisons. The execution time is directly proportional to the total comparisons required. If comparisons increases, then execution time increases, and vice versa. The results collected from the experimental study show that the NDS-ROS algorithm has the least number of comparisons. Therefore the execution time is also less than the compared algorithms.

A working example illustrates the service selection process using the NDS-ROS algorithm, which has four stages: filtering, sorting, ranking, and dominance comparison. The filtration step lowers the search space in the initial step that significantly saves DMs' time and effort. Most existing studies lack the filtration step, maximizing the threat of selecting complete services. In the sorting, the candidate services are sorted based on QoS attributes value for each objective function. The sorted services are ranked so that the best services in that objective lies among the top numbers. The dominance comparison compares only those services having the same rank but stored in different objective functions. The dominance rules applied at the time of dominance comparison reducing duplicate comparison between services and yielding fast results. The filtration and dominance comparisons efficiently reduce the search space and remove identical and un-necessary comparison, thereby increasing the overall searching process efficiency. The number of objectives taken was 2,3,4,5, and QoS values are taken for the study to evaluate the cloud services. The research combined four varied, SaaS QoS norms. The filtration stage takes the user requirements for QoS attributes, and the priority is taken from the user to evaluate the fitness function. Different steps are applied and comprehensive analysis to appreciate the NDS-ROS approach's role in developing an optimal service agreement. In brief, NDS-ROS offers a complete step-by-step mechanism to evaluate cloud services' performance from QoS viewpoints.
The NDS-ROS algorithm effectively finds a consensual result in comparison to other non-dominating approaches.

The NS- ROS algorithm has some advantage over other algorithms, but there is one drawback: the sorting time is increased with objective size and increases execution time. Also, qualitative parameters can be considered in future research to improve trust.

Future Work

In the future, the NDS-ROS approach can be extended to include subjective parameters for SaaS cloud selection. The data science and machine learning can be applied to improve searching and ranking process in the cloud.

Chapter 7

List of Publications

Patents

1. Sirohi, P., Agarwal, A., Maheshwari, P.,Dewangan, B.K., & Choudhury, T. (2020a). An efficient approach for saas cloud service selection and ranking [Govt. of India, File No. 202011045036 A]. Indian Patents.

Scopus Indexed Journals

1. Sirohi, Preeti, Amit Agarwal, and Piyush Maheshwari "A framework for ranking of cloud services using non-dominated sorting" International Journal of Engineering and Advanced Technology(IJEAT)" ISSN: 2249 – 8958Volume-8 Issue-5, 2018.

2. Sirohi, Preeti, Amit Agarwal, and Piyush Maheshwari "Systematic Literature Survey using Quality Parameter for Cloud Computing" International Journal of Recent Technology and Engineering (IJRTE)" ISSN- 2277-3878 (Online) Volume 7 issue 6c, 2019.

3. Sirohi, Preeti, Amit Agarwal, and Piyush Maheshwari "A Comparative study of Cloud Computing Service Selection" International Journal of Engineering and Advanced Technology(IJEAT)" ISSN: 2249-8958 Volume-8 Issue-5, 2019.

4. Sirohi, P., Agarwal, A., Maheshwari, P.,Dewangan, B. K., & Choudhury, T. (2020b). Saas-cloudselection by optimizing energy utilization and response time.Journal of Green Engineering. Volume 10,Issue 9, Pages 6966–6989.

Conferences

1. Sirohi, Preeti, Amit Agarwal, and Piyush Maheshwari. " Systematic Literature review for Cloud Computing." International Conference on Future Computing and Communication Technology ICFCCT, 2018 (MIET , Meerut)
2. Sirohi, Preeti, Amit Agarwal, and Piyush Maheshwari. "Framework for Cloud Service Selection and Ranking." International Conference on Advances in Engineering Science Management & Technology (ICAESMT)-2019, Uttaranchal University, Dehradun, India. 2019.

References

Aazam, M., & Huh, E.-N. (2017). Cloud broker service-oriented resource management model. *Transactions on Emerging Telecommunications Technologies*, *28*(2), e2937.

Abbas, A., Pimenov, D. Y., Erdakov, I., Mikolajczyk, T., El Danaf, E., & Taha, M. (2017). Minimization of turning time for high-strength steel with a given surface roughness using the edgeworth–pareto optimization method. *The International Journal of Advanced Manufacturing Technology*, *93*(5-8), 2375–2392.

Abdel-Basset, M., Manogaran, G., Gamal, A., & Chang, V. (2019). A novel intelligent medical decision support model based on soft computing and iot. *IEEE Internet of Things Journal*, *7*(5), 4160–4170.

Achar, R., & Thilagam, P. S. (2014). A broker based approach for cloud provider selection. In *2014 international conference on advances in computing, communications and informatics (icacci)* (pp. 1252–1257).

Ali, A. S., Ludwig, S. A., & Rana, O. F. (2005). A cognitive trust-based approach for web service discovery and selection. In *Third european conference on web services (ecows'05)* (pp. 12–pp).

Annette, R., & Banu, A. (2015). A service broker model for cloud based render farm selection. *arXiv preprint arXiv:1505.06542*.

Arab, B. S. (2010). *Custom windows performance counters monitoring mechanism for measuring quality of service attributes and stability coefficient service-oriented architecture* (Unpublished doctoral dissertation). Universiti Putra Malaysia.

Ardagna, D., & Pernici, B. (2005). Global and local qos constraints guarantee in web service selection. In *Ieee international conference on web services (icws'05).*

Beimborn, D., Miletzki, T., & Wenzel, S. (2011). Platform as a service (paas). *Business & Information Systems Engineering*, *3*(6), 381–384.

Benlian, A., Hess, T., & Buxmann, P. (2009). Drivers of saas-adoption–an empirical study of different application types. *Business & Information Systems Engineering*, *1*(5), 357.

Beume, N., Naujoks, B., & Emmerich, M. (2007). Sms-emoa: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, *181*(3), 1653–1669.

Bhardwaj, S., Jain, L., & Jain, S. (2010). Cloud computing: A study of infrastructure as a service (iaas). *International Journal of engineering and information Technology*, *2*(1), 60–63.

Binnig, C., Kossmann, D., Kraska, T., & Loesing, S. (2009). How is the weather tomorrow? towards a benchmark for the cloud. In *Proceedings of the second international workshop on testing database systems* (pp. 1–6).

Bohn, R. B., Messina, J., Liu, F., Tong, J., & Mao, J. (2011). Nist cloud computing reference architecture. In *2011 ieee world congress on services* (pp. 594–596).

Boutkhoum, O., Hanine, M., Agouti, T., & Tikniouine, A. (2016). Selection problem of cloud solution for big data accessing: fuzzy ahp-promethee as a proposed methodology. *Journal of Digital Information Management*, *14*(6).

Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009). Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, *25*(6), 599–616.

Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., & Buyya, R. (2011). Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience*, *41*(1), 23–50.

Cao, B.-Q., Li, B., & Xia, Q.-M. (2009). A service-oriented qos-assured and multi-agent cloud computing architecture. In *Ieee international conference on cloud computing* (pp. 644–649).

Chan, S.-H. G., & Tobagi, F. (2001). Distributed servers architecture for networked video services. *IEEE/ACM transactions on networking*, *9*(2), 125–136.

Chen, C.-T., & Lin, K.-H. (2010). A decision-making method based on interval-valued fuzzy sets for cloud service evaluation. In *4th international conference on new trends in information science and service science* (pp. 559–564).

Chen, G., Bai, X., Huang, X., Li, M., & Zhou, L. (2011). Evaluating services on the cloud using ontology qos model. In *Proceedings of 2011 ieee 6th international symposium on service oriented system (sose)* (pp. 312–317).

Chen, L., Feng, Y., Wu, J., & Zheng, Z. (2011). An enhanced qos prediction approach for service selection. In *2011 ieee international conference on services computing* (pp. 727–728).

Chou, S.-Y., Chang, Y.-H., & Shen, C.-Y. (2008). A fuzzy simple additive weighting system under group decision-making for facility location selection with objective/subjective attributes. *European Journal of Operational Research*, *189*(1), 132–145.

Coello, C. A. C., Lamont, G. B., Van Veldhuizen, D. A., et al. (2007). *Evolutionary algorithms for solving multi-objective problems* (Vol. 5). Springer.

Cusumano, M. (2010). Cloud computing and saas as new computing platforms. *Communications of the ACM*, *53*(4), 27–29.

Deb, K. (1999). Solving goal programming problems using multi-objective genetic algorithms. In *Proceedings of the 1999 congress on evolutionary computation-cec99 (cat. no. 99th8406)* (Vol. 1, pp. 77–84).

De Boer, L., van der Wegen, L., & Telgen, J. (1998). Outranking methods in support of supplier selection. *European Journal of Purchasing & Supply Management*, *4*(2-3), 109–118.

Dhiman, G., & Kumar, V. (2018). Multi-objective spotted hyena optimizer: a multi-objective optimization algorithm for engineering problems. *Knowledge-Based Systems*, *150*, 175–197.

Dikaiakos, M. D., Katsaros, D., Mehra, P., Pallis, G., & Vakali, A. (2009). Cloud computing: Distributed internet computing for it and scientific research. *IEEE Internet computing*, *13*(5), 10–13.

Ding, S., Xia, C., Wang, C., Wu, D., & Zhang, Y. (2017). Multi-objective optimization based ranking prediction for cloud service recommendation. *Decision Support Systems*, *101*, 106–114.

Ding, S., Yang, S., Zhang, Y., Liang, C., & Xia, C. (2014). Combining qos prediction and customer satisfaction estimation to solve cloud service trustworthiness evaluation problems. *Knowledge-Based Systems*, *56*, 216–225.

Dong, W. E., Nan, W., & Xu, L. (2013). Qos-oriented monitoring model of cloud computing resources availability. In *2013 international conference on computational and information sciences* (pp. 1537–1540).

Drozdik, M., Akimoto, Y., Aguirre, H., & Tanaka, K. (2014). Computational cost reduction of nondominated sorting using the m-front. *IEEE Transactions on Evolutionary Computation*, *19*(5), 659–678.

Dubey, A., & Wagle, D. (2007). Delivering software as a service. *The McKinsey Quarterly*, *6*(2007), 2007.

Erickson, M., Mayer, A., & Horn, J. (2002). Multi-objective optimal design of groundwater remediation systems: application of the niched pareto genetic algorithm (npga). *Advances in Water Resources*, *25*(1), 51–65.

Erl, T. (1900). *Service-oriented architecture: concepts, technology, and design.* Pearson Education India.

Ferris, J. M., & Riveros, G. E. (2018, May 15). *Verifying software license compliance in cloud computing environments.* Google Patents. (US Patent 9,971,880)

Figueira, J., Mousseau, V., & Roy, B. (2005). Electre methods. In *Multiple criteria decision analysis: State of the art surveys* (pp. 133–153). Springer.

Fletcher, K. K., & Liu, X. F. (2015). A collaborative filtering method for personalized preference-based service recommendation. In *2015 ieee international conference on web services* (pp. 400–407).

Fonseca, C. M., Fleming, P. J., et al. (1993). Genetic algorithms for multiobjective optimization: Formulationdiscussion and generalization. In *Icga* (Vol. 93, pp. 416–423).

Gabbani, D., & Magazine, M. (1986). An interactive heuristic approach for multi-objective integer-programming problems. *Journal of the Operational Research Society*, *37*(3), 285–291.

Gao, J., Pattabhiraman, P., Bai, X., & Tsai, W.-T. (2011). Saas performance and scalability evaluation in clouds. In *Proceedings of 2011 ieee 6th international symposium on service oriented system (sose)* (pp. 61–71).

Garg, S. K., Versteeg, S., & Buyya, R. (2011). Smicloud: A framework for comparing and ranking cloud services. In *2011 fourth ieee international conference on utility and cloud computing* (pp. 210–218).

Garg, S. K., Versteeg, S., & Buyya, R. (2013). A framework for ranking of cloud computing services. *Future Generation Computer Systems*, *29*(4), 1012–1023.

Gates III, W. H., Flake, G. W., Bolosky, W. J., Dani, N. V., Glasser, D. S., Gounares, A. G., . . . Meijer, H. J. M. (2011, September 6). *Hardware architecture for cloud services.* Google Patents. (US Patent 8,014,308)

Ghosh, N., Ghosh, S. K., & Das, S. K. (2014). Selcsp: A framework to facilitate selection of cloud service providers. *IEEE transactions on cloud computing*, *3*(1), 66–79.

Gibson, J., Rondeau, R., Eveleigh, D., & Tan, Q. (2012). Benefits and challenges of three cloud computing service models. In *2012 fourth international conference on computational aspects of social networks (cason)* (pp. 198–205).

Godse, M., & Mulik, S. (2009). An approach for selecting software-as-a-service (saas) product. In *2009 ieee international conference on cloud computing* (pp. 155–158).

Golden, B. L., Wasil, E. A., & Harker, P. T. (1989). The analytic hierarchy process. *Applications and Studies, Berlin, Heidelberg.*

Gonçalves Junior, R., Rolim, T., Sampaio, A., & Mendonça, N. C. (2015). A multi-criteria approach for assessing cloud deployment options based on non-functional requirements. In *Proceedings of the 30th annual acm symposium on applied computing* (pp. 1383–1389).

Gong, C., Liu, J., Zhang, Q., Chen, H., & Gong, Z. (2010). The characteristics of cloud computing. In *2010 39th international conference on parallel processing workshops* (pp. 275–279).

Görener, A. (2012). Comparing ahp and anp: an application of strategic decisions making in a manufacturing company. *International Journal of Business and Social Science*, *3*(11).

Goyal, N., Pandey, A. K., Gupta, S. K., & Pandey, R. (2019). Suppleness of multi-tenancy in cloud computing: Advantages, privacy issues and risk factors. In *Proceedings of international conference on sustainable computing in science, technology and management (suscom), amity university rajasthan, jaipur-india.*

Guzek, M., Gniewek, A., Bouvry, P., Musial, J., & Blazewicz, J. (2015). Cloud brokering: Current practices and upcoming challenges. *IEEE Cloud Computing*, *2*(2), 40–47.

Hao, Y., Zhang, Y., & Cao, J. (2010). Web services discovery and rank: An information retrieval approach. *Future generation computer systems*, *26*(8), 1053–1062.

He, Q., Han, J., Yang, Y., Grundy, J., & Jin, H. (2012). Qos-driven service selection for multi-tenant saas. In *2012 ieee fifth international conference on cloud computing* (pp. 566–573).

Hernández Gómez, R., & Coello Coello, C. A. (2015). Improved metaheuristic based on the r2 indicator for many-objective optimization. In *Proceedings of the 2015 annual conference on genetic and evolutionary computation* (pp. 679–686).

Hey, T., & Trefethen, A. (2003). Grid computing: Making the global infrastructure a reality. In *The data deluge: An e-science perspective.* Wiley.

Hiessl, H., Duckstein, I., & Plate, E. (1985). Multiobjective q-analysis with concordance and discordance concepts. *Applied Mathematics and Computation*, *17*(2), 107–122.

Hong, D. H., & Choi, C.-H. (2000). Multicriteria fuzzy decision-making problems based on vague set theory. *Fuzzy sets and systems*, *114*(1), 103–113.

Huang, A. F., Lan, C.-W., & Yang, S. J. (2009). An optimal qos-based web service selection scheme. *Information Sciences*, *179*(19), 3309–3322.

Hussain, A., Chun, J., & Khan, M. (2020). A novel framework towards viable cloud service selection as a service (cssaas) under a fuzzy environment. *Future Generation Computer Systems*, *104*, 74–91.

Hwang, C.-L., & Yoon, K. (1981). Methods for multiple attribute decision making. In *Multiple attribute decision making* (pp. 58–191). Springer.

Ibrahim, S., He, B., & Jin, H. (2011). Towards pay-as-you-consume cloud computing. In *2011 ieee international conference on services computing* (pp. 370–377).

Jahani, A., Derakhshan, F., & Khanli, L. M. (2017). Arank: A multi-agent based approach for ranking of cloud computing services. *Scalable Computing: Practice and Experience*, *18*(2), 105–116.

Jahani, A., Khanli, L. M., & Razavi, S. N. (2014). W_sr: A qos based ranking approach for cloud computing service. *Computer Engineering and Applications Journal*, *3*(2), 55–62.

Jannat, S., Khaled, A., & Paul, S. K. (2010). Optimal solution for multi-objective facility layout problem using genetic algorithm. In *Proc. 2010 intl. conference on industrial engineering and operations management.*

Jensen, M. T. (2003). Reducing the run-time complexity of multiobjective eas: The nsga-ii and other algorithms. *IEEE Transactions on Evolutionary Computation*, *7*(5), 503–515.

Jøsang, A., Ismail, R., & Boyd, C. (2007). A survey of trust and reputation systems for online service provision. *Decision support systems*, *43*(2), 618–644.

Kanagasabai, R., et al. (2012). Owl-s based semantic cloud service broker. In *2012 ieee 19th international conference on web services* (pp. 560–567).

Karim, R., Ding, C., & Miri, A. (2013). An end-to-end qos mapping approach for cloud service selection. In *2013 ieee ninth world congress on services* (pp. 341–348).

Kavis, M. J. (2014). *Architecting the cloud: design decisions for cloud computing service models (saas, paas, and iaas).* John Wiley & Sons.

Keller, A., & Ludwig, H. (2003). The wsla framework: Specifying and monitoring service level agreements for web services. *Journal of Network and Systems Management*, *11*(1), 57–81.

Khezrian, M., Kadir, W. M. W., Ibrahim, S., Kalantari, A., et al. (2012). A hybrid approach for web service selection. *International Journal of Computational Engineering Research*, *2*(1), 190–198.

Khowfa, W., & Silasai, O. (2017). The integration of association rules and ahp in cloud service selection. *International Journal of Applied Engineering Research*, *12*(24), 15814–15820.

Knowles, J. D., Watson, R. A., & Corne, D. W. (2001). Reducing local optima in single-objective problems by multi-objectivization. In *International conference on evolutionary multi-criterion optimization* (pp. 269–283).

Kukkonen, S., & Deb, K. (2006). A fast and effective method for pruning of non-dominated solutions in many-objective problems. In *Parallel problem solving from nature-ppsn ix* (pp. 553–562). Springer.

Kumar, N., & Agarwal, S. (2014). Qos based cloud service provider selection framework. *Research Journal of Recent Sciences*, *2277*, 2502.

Kumar, R. D., & Zayaraz, G. (2011). A qos aware quantitative web service selection model. *International Journal on Computer Science and Engineering*, *3*(4), 1534–1538.

Kumar, R. R., Mishra, S., & Kumar, C. (2017). Prioritizing the solution of cloud service selection using integrated mcdm methods under fuzzy environment. *The Journal of Supercomputing*, *73*(11), 4652–4682.

Lai, Y.-J., Liu, T.-Y., & Hwang, C.-L. (1994). Topsis for modm. *European journal of operational research*, *76*(3), 486–500.

Latif, R., Abbas, H., Assar, S., & Ali, Q. (2014). Cloud computing risk assessment: a systematic literature review. In *Future information technology* (pp. 285–295). Springer.

Lewis, G. (2010). Basics about cloud computing. *Software engineering institute carniege mellon university, Pittsburgh*.

Li, K., Zhang, Q., Kwong, S., Li, M., & Wang, R. (2013). Stable matching-based selection in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, *18*(6), 909–923.

Li, X., Liang, H., & Zhang, X. (2016). Trust based service selection in cloud computing environment. *International Journal of Smart Home*, *10*(11), 39–50.

Lin, H. Y., Doong, J.-G., & Hsieh, M.-Y. (2015). Building a prototype of middleware using windows communication foundations. In *Environmental science and information application technology: Proceedings of the 2014 5th international conference on environmental science and information application technology (esiat 2014), hong kong, november 7-8, 2014* (p. 157).

Lin, H.-Y., Hsu, P.-Y., & Sheen, G.-J. (2007). A fuzzy-based decision-making procedure for data warehouse system selection. *Expert systems with applications*, *32*(3), 939–953.

Liu, L., & Zhang, M. (2015). Multi-objective optimization model with ahp decision-making for cloud service composition. *KSII Transactions on Internet & Information Systems*, *9*(9).

Liu, M., Ding, W., Park, D. H., Fang, Y., Yan, R., & Hu, X. (2017). Which used product is more sellable? a time-aware approach. *Information Retrieval Journal*, *20*(2), 81–108.

Lo, C.-C., Chen, D.-Y., Tsai, C.-F., & Chao, K.-M. (2010). Service selection based on fuzzy topsis method. In *2010 ieee 24th international conference on advanced information networking and applications workshops* (pp. 367–372).

Lombardi, F., & Di Pietro, R. (2011). Secure virtualization for cloud computing. *Journal of network and computer applications*, *34*(4), 1113–1122.

Lucas-Simarro, J. L., Aniceto, I. S., Moreno-Vozmediano, R., Montero, R. S., & Llorente, I. M. (2013). A cloud broker architecture for multicloud environments. *Large Scale Network-Centric Distributed Systems*, 359–376.

Luo, J.-Z., Jin, J.-H., Song, A.-b., & Dong, F. (2011). Cloud computing: architecture and key technologies. *Journal of China Institute of Communications*, *32*(7), 3–21.

Mabrouk, N. B., Beauche, S., Kuznetsova, E., Georgantas, N., & Issarny, V. (2009). Qos-aware service composition in dynamic service oriented environments. In *Acm/ifip/usenix international conference on distributed systems platforms and open distributed processing* (pp. 123–142).

Mahmoodzadeh, S., Shahrabi, J., Pariazar, M., & Zaeri, M. (2007). Project selection by using fuzzy ahp and topsis technique. *World Academy of Science, Engineering and Technology*, *30*, 333–338.

Malathi, M. (2011). Cloud computing concepts. In *2011 3rd international conference on electronics computer technology* (Vol. 6, pp. 236–239).

Marler, R. T., & Arora, J. S. (2010). The weighted sum method for multi-objective optimization: new insights. *Structural and multidisciplinary optimization*, *41*(6), 853–862.

Mell, P., Grance, T., et al. (2011). The nist definition of cloud computing.

Menchaca-Mendez, A., & Coello, C. A. C. (2015). Gde-moea: a new moea based on the generational distance indicator and ε-dominance. In *2015 ieee congress on evolutionary computation (cec)* (pp. 947–955).

Menzel, M., & Ranjan, R. (2012). Cloudgenius: decision support for web server cloud migration. In *Proceedings of the 21st international conference on world wide web* (pp. 979–988).

Menzel, M., Schönherr, M., & Tai, S. (2013). (mc2) 2: criteria, requirements and a software prototype for cloud infrastructure decisions. *Software: Practice and experience*, *43*(11), 1283–1297.

Milan, A., Rezatofighi, S. H., Garg, R., Dick, A., & Reid, I. (2017). Data-driven approximations to np-hard problems. In *Thirty-first aaai conference on artificial intelligence.*

Mishra, S., Mondal, S., Saha, S., & Coello, C. A. C. (2018). Gbos: Generalized best order sort algorithm for non-dominated sorting. *Swarm and Evolutionary Computation*, *43*, 244–264.

Mohammadshahi, Y. (2013). A state-of-art survey on tqm applications using mcdm techniques. *Decision Science Letters*, *2*(3), 125–134.

Moscato, F., Aversa, R., Di Martino, B., Fortiş, T.-F., & Munteanu, V. (2011). An analysis of mosaic ontology for cloud resources annotation. In *2011 federated conference on computer science and information systems (fedcsis)* (pp. 973–980).

Mousavi-Nasab, S. H., & Sotoudeh-Anvari, A. (2017). A comprehensive mcdm-based approach using topsis, copras and dea as an auxiliary tool for material selection problems. *Materials & Design*, *121*, 237–253.

Mousseau, V., & Slowinski, R. (1998). Inferring an electre tri model from assignment examples. *Journal of global optimization*, *12*(2), 157–174.

Nickolov, P., Armijo, B., & Miloushev, V. (2013, April 23). *Globally distributed utility computing cloud.* Google Patents. (US Patent 8,429,630)

Ostermann, S., Iosup, A., Yigitbasi, N., Prodan, R., Fahringer, T., & Epema, D. (2009). A performance analysis of ec2 cloud computing services for scientific computing. In *International conference on cloud computing* (pp. 115–131).

Özlen, M., & Azizoğlu, M. (2009). Multi-objective integer programming: A general approach for generating all non-dominated solutions. *European Journal of Operational Research*, *199*(1), 25–35.

Pan, Y., Ding, S., Fan, W., Li, J., & Yang, S. (2015). Trust-enhanced cloud service selection model based on qos analysis. *PloS one*, *10*(11), e0143448.

Pandey, S., & Daniel, A. (2017). Qocs and cost based cloud service selection framework. *Int. J. Eng. Trends Technol.(IJETT)*, *48*(3), 167–172.

Pawar, P. S., Rajarajan, M., Nair, S. K., & Zisman, A. (2012). Trust model for optimized cloud services. In *Ifip international conference on trust management* (pp. 97–112).

Peng, C., Sun, H., & Guo, J. (2010). Multi-objective optimal pmu placement using a non-dominated sorting differential evolution algorithm. *International Journal of Electrical Power & Energy Systems*, *32*(8), 886–892.

Phelps, S., & Köksalan, M. (2003). An interactive evolutionary metaheuristic for multiobjective combinatorial optimization. *Management Science*, *49*(12), 1726–1738.

Pilat, M., & Neruda, R. (2015). Incorporating user preferences in moea/d through the coevolution of weights. In *Proceedings of the 2015 annual conference on genetic and evolutionary computation* (pp. 727–734).

Prasad, G. V., Prasad, A. S., & Rao, S. (2016). A combinatorial auction mechanism for multiple resource procurement in cloud computing. *IEEE Transactions on Cloud Computing*, *6*(4), 904–914.

Qi, Y., Ma, X., Liu, F., Jiao, L., Sun, J., & Wu, J. (2014). Moea/d with adaptive weight adjustment. *Evolutionary computation*, *22*(2), 231–264.

Rai, D., & Kumar, P. (2016). Instance based multi criteria decision model for cloud service selection using topsis and vikor. *International Journal of Computer Engineering and Technology*, *7*(1), 78–87.

Raj, J. R., & Sasipraba, T. (2012). web service discovery based on computation of semantic similarity distance and qos normalization. *Indian journal of computer science and engineering*, *3*(2), 235–239.

Rani, D., & Ranjan, R. K. (2014). A comparative study of saas, paas and iaas in cloud computing. *International Journal of Advanced Research in Computer Science and Software Engineering*, *4*(6).

Roy, P. C., Islam, M. M., & Deb, K. (2016). Best order sort: a new algorithm to non-dominated sorting for evolutionary multi-objective optimization. In *Proceedings of the 2016 on genetic and evolutionary computation conference companion* (pp. 1113–1120).

Saaty, T. L. (1988). What is the analytic hierarchy process? In *Mathematical models for decision support* (pp. 109–121). Springer.

Saaty, T. L., & Vargas, L. G. (2013). The analytic network process. In *Decision making with the analytic network process* (pp. 1–40). Springer.

Sakawa, M., Kato, K., & Nishizaki, I. (2003). An interactive fuzzy satisficing method for multiobjective stochastic linear programming problems through an expectation model. *European journal of operational research*, *145*(3), 665–672.

Salama, M., Shawish, A., & Zeid, A. (2013). A generic framework for modeling and simulation of cloud computing services. *International Journal of Computer Applications*, *77*(17).

Sardinas, R. Q., Santana, M. R., & Brindis, E. A. (2006). Genetic algorithm-based multi-objective optimization of cutting parameters in turning processes. *Engineering Applications of Artificial Intelligence*, *19*(2), 127–133.

Saripalli, P., & Pingali, G. (2011). Madmac: Multiple attribute decision methodology for adoption of clouds. In *2011 ieee 4th international conference on cloud computing* (pp. 316–323).

Sasikaladevi, N. (2016). Trust based cloud service composition framework. *International Journal of Grid and Distributed Computing*, *9*(1), 99–104.

Savu, L. (2011). Cloud computing: Deployment models, delivery models, risks and research challenges. In *2011 international conference on computer and management (caman)* (pp. 1–4).

Seethamraju, R. (2015). Adoption of software as a service (saas) enterprise resource planning (erp) systems in small and medium sized enterprises (smes). *Information systems frontiers*, *17*(3), 475–492.

Sevkli, M. (2010). An application of the fuzzy electre method for supplier selection. *International Journal of Production Research*, *48*(12), 3393–3405.

Shawky, D. M., & Ali, A. F. (2012). Defining a measure of cloud computing elasticity. In *2012 1st international conference on systems and computer science (icscs)* (pp. 1–5).

Siegel, J., & Perdue, J. (2012). Cloud services measures for global use: the service measurement index (smi). In *2012 annual srii global conference* (pp. 411–415).

Silas, S., Rajsingh, E. B., & Ezra, K. (2012). Efficient service selection middleware using electre methodology for cloud environments. *Information Technology Journal*, *11*(7), 868.

Smith, J., & Nair, R. (2005). *Virtual machines: versatile platforms for systems and processes.* Elsevier.

Srivastava, A., & Sorenson, P. G. (2010). Service selection based on customer rating of quality of service attributes. In *2010 ieee international conference on web services* (pp. 1–8).

Stienhans, F., & Klimentiev, M. (2011, August 9). *Systems and methods for dynamically provisioning cloud computing resources.* Google Patents. (US Patent 7,996,525)

Stojanovic, M. D., Rakas, S. V. B., & Acimovic-Raspopovic, V. S. (2010). End-to-end quality of service specification and mapping: The third party approach. *Computer Communications*, *33*(11), 1354–1368.

Sun, L., Ma, J., Zhang, Y., Dong, H., & Hussain, F. K. (2016). Cloud-fuser: Fuzzy ontology and mcdm based cloud service selection. *Future Generation Computer Systems*, *57*, 42–55.

Sundareswaran, S., Squicciarini, A., & Lin, D. (2012). A brokerage-based approach for cloud service selection. In *2012 ieee fifth international conference on cloud computing* (pp. 558–565).

Supriya, M., Sangeeta, K., & Patra, G. (2016). Trustworthy cloud service provider selection using multi criteria decision making methods. *Engineering Letters*, *24*(1).

Tajvidi, M., Ranjan, R., Kolodziej, J., & Wang, L. (2014). Fuzzy cloud service selection framework. In *2014 ieee 3rd international conference on cloud networking (cloudnet)* (pp. 443–448).

Talja, S. (1999). Analyzing qualitative interview data: The discourse analytic method. *Library & information science research*, *21*(4), 459–477.

Tho, Q. T., Hui, S. C., Fong, A. C. M., & Cao, T. H. (2006). Automatic fuzzy ontology generation for semantic web. *IEEE transactions on knowledge and data engineering*, *18*(6), 842–856.

Tsai, W.-T., & Sun, X. (2013). Saas multi-tenant application customization. In *2013 ieee seventh international symposium on service-oriented system engineering* (pp. 1–12).

Upadhyay, N. (2017). Managing cloud service evaluation and selection. *Procedia computer science*, *122*, 1061–1068.

Urena, R., Kou, G., Dong, Y., Chiclana, F., & Herrera-Viedma, E. (2019). A review on trust propagation and opinion dynamics in social networks and group decision making frameworks. *Information Sciences*, *478*, 461–475.

Van Veldhuizen, D. A., & Lamont, G. B. (2000). On measuring multiobjective evolutionary algorithm performance. In *Proceedings of the 2000 congress on evolutionary computation. cec00 (cat. no. 00th8512)* (Vol. 1, pp. 204–211).

Varia, J. (2008). Cloud architectures. *White Paper of Amazon, jineshvaria. s3. amazonaws. com/public/cloudarchitectures-varia. pdf*, *16*.

Velte, T., Velte, A., & Elsenpeter, R. (2009). *Cloud computing, a practical approach.* McGraw-Hill, Inc.

Wanchun, D., Chao, L., Xuyun, Z., & Chen, J. (2011). A qos-aware service evaluation method for co-selecting a shared service. In *2011 ieee international conference on web services* (pp. 145–152).

Wang, L., Zhan, J., Shi, W., & Liang, Y. (2011). In cloud, can scientific communities benefit from the economies of scale? *IEEE Transactions on Parallel and Distributed Systems*, *23*(2), 296–303.

Wei-Wen, W. (2011). Developing an explorative model for saas adoption. *Expert systems with applications*, *38*(12), 15057–15064.

Wischik, D., Handley, M., & Braun, M. B. (2008). The resource pooling principle. *ACM SIGCOMM Computer Communication Review*, *38*(5), 47–52.

Wu, Q., Zhu, Q., Jian, X., & Ishikawa, F. (2014). Broker-based sla-aware composite service provisioning. *Journal of Systems and Software*, *96*, 194–201.

Xing, Y., & Zhan, Y. (2012). Virtualization and cloud computing. In *Future wireless networks and information systems* (pp. 305–312). Springer.

Yadav, N., Singh, V., & Kumari, M. (2014). Generalized reliability model for cloud computing. *International Journal of Computer Applications*, *88*(14).

Yang, W., Zhang, C., Shao, Y., Shi, Y., Li, H., Khan, M., ... others (2014). A hybrid particle swarm optimization algorithm for service selection problem in the cloud. *Int J Grid Distrib Comput*, *7*(4), 1–10.

Yazdani-Chamzini, A., Yakhchali, S. H., & Mahmoodian, M. (2013). Risk ranking of tunnel construction projects by using the electre technique under a fuzzy environment. *International Journal of Management Science and Engineering Management*, *8*(1), 1–14.

Ye, Z., Zhou, X., & Bouguettaya, A. (2011). Genetic algorithm based qos-aware service compositions in cloud computing. In *International conference on database systems for advanced applications* (pp. 321–334).

Zahariev, A. (2009). Google app engine. *Helsinki University of Technology*, 1–5.

Zavadskas, E. K., Zakarevicius, A., & Antucheviciene, J. (2006). Evaluation of ranking accuracy in multi-criteria decisions. *Informatica*, *17*(4), 601–618.

Zeng, L., Benatallah, B., Ngu, A. H., Dumas, M., Kalagnanam, J., & Chang, H. (2004). Qos-aware middleware for web services composition. *IEEE Transactions on software engineering*, *30*(5), 311–327.

Zhang, M., Ranjan, R., Nepal, S., Menzel, M., & Haller, A. (2012). A declarative recommender system for cloud infrastructure services selection. In *International conference on grid economics and business models* (pp. 102–113).

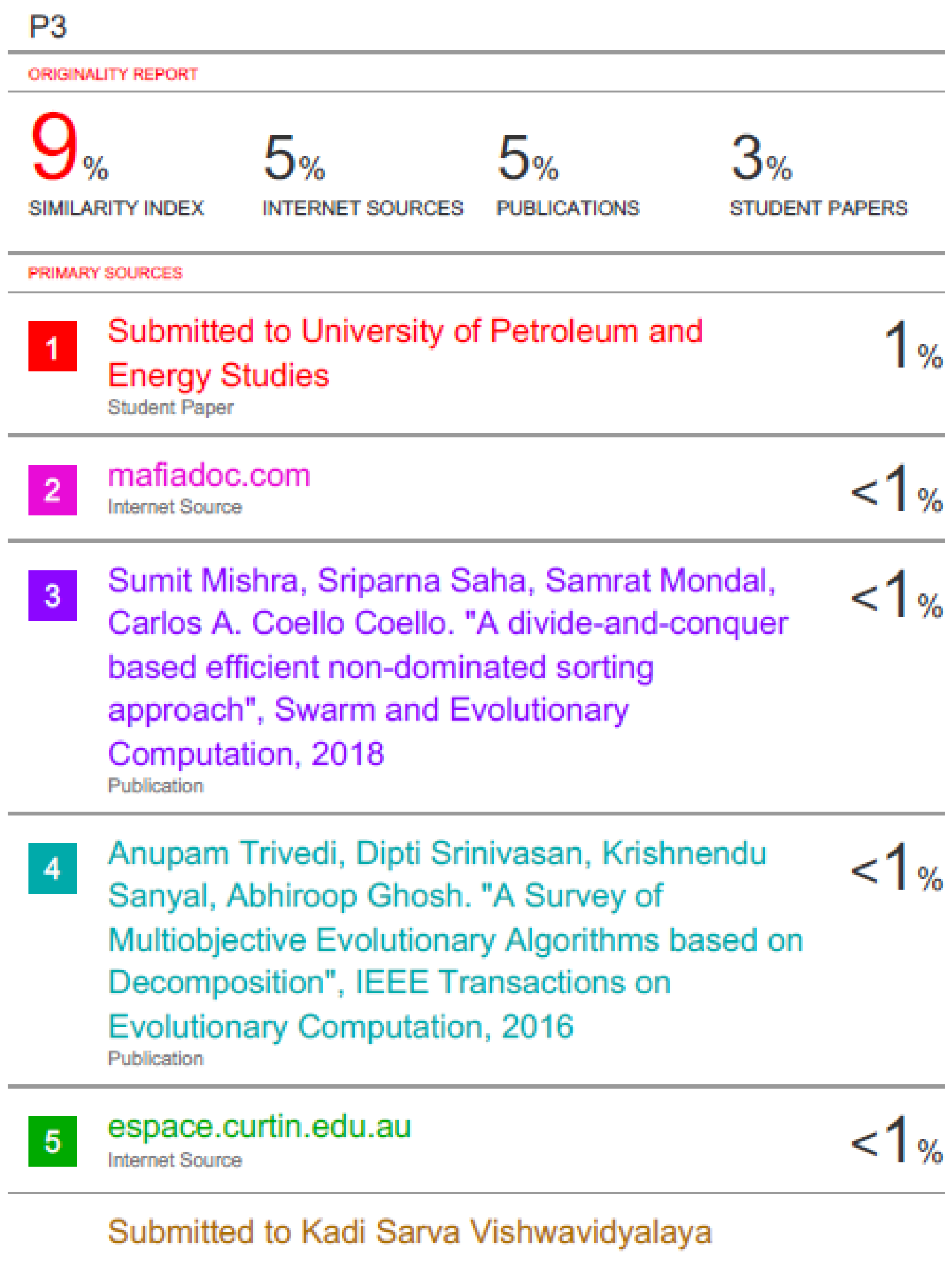
Zhang, Q., & Li, H. (2007). Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on evolutionary computation*, *11*(6), 712–731.

Zhang, X., Tian, Y., Cheng, R., & Jin, Y. (2016). Empirical analysis of a tree-based efficient non-dominated sorting approach for many-objective optimization. In *2016 ieee symposium series on computational intelligence (ssci)* (pp. 1–8).

Zheng, Z., Wu, X., Zhang, Y., Lyu, M. R., & Wang, J. (2012). Qos ranking prediction for cloud services. *IEEE transactions on parallel and distributed systems*, *24*(6), 1213–1222.

Zitzler, E., & Künzli, S. (2004). Indicator-based selection in multiobjective search. In *International conference on parallel problem solving from nature* (pp. 832–842).

Zydallis, J. B., Van Veldhuizen, D. A., & Lamont, G. B. (2001). A statistical comparison of multiobjective evolutionary algorithms including the momga-ii. In *International conference on evolutionary multi-criterion optimization* (pp. 226–240).



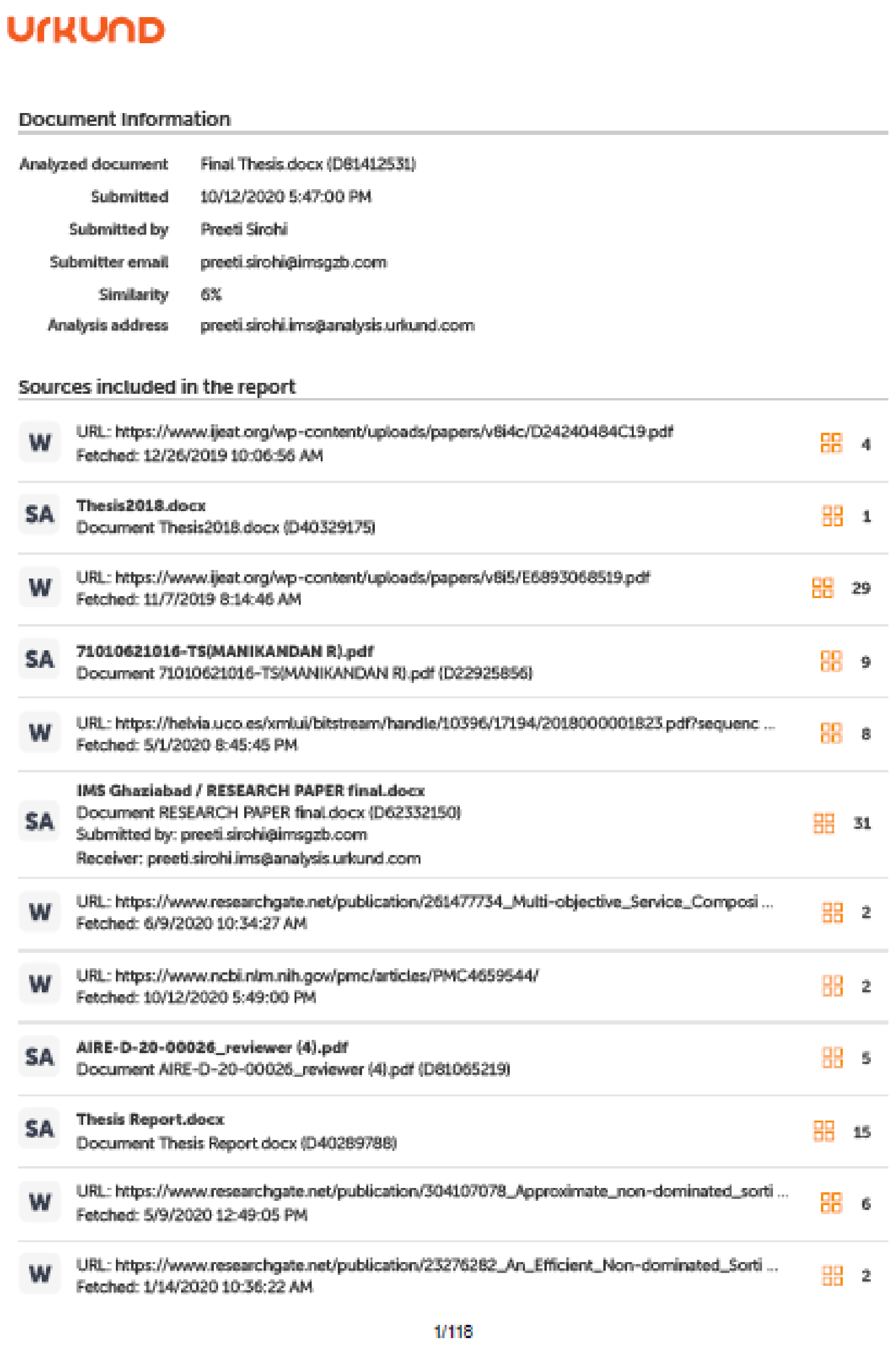
P3

ORIGINALITY REPORT

9%	5%	5%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to University of Petroleum and Energy Studies Student Paper	1%
2	mafiadoc.com Internet Source	<1%
3	Sumit Mishra, Sriparna Saha, Samrat Mondal, Carlos A. Coello Coello. "A divide-and-conquer based efficient non-dominated sorting approach", Swarm and Evolutionary Computation, 2018 Publication	<1%
4	Anupam Trivedi, Dipti Srinivasan, Krishnendu Sanyal, Abhiroop Ghosh. "A Survey of Multiobjective Evolutionary Algorithms based on Decomposition", IEEE Transactions on Evolutionary Computation, 2016 Publication	<1%
5	espace.curtin.edu.au Internet Source	<1%
	Submitted to Kadi Sarva Vishwavidyalaya	



URKUND

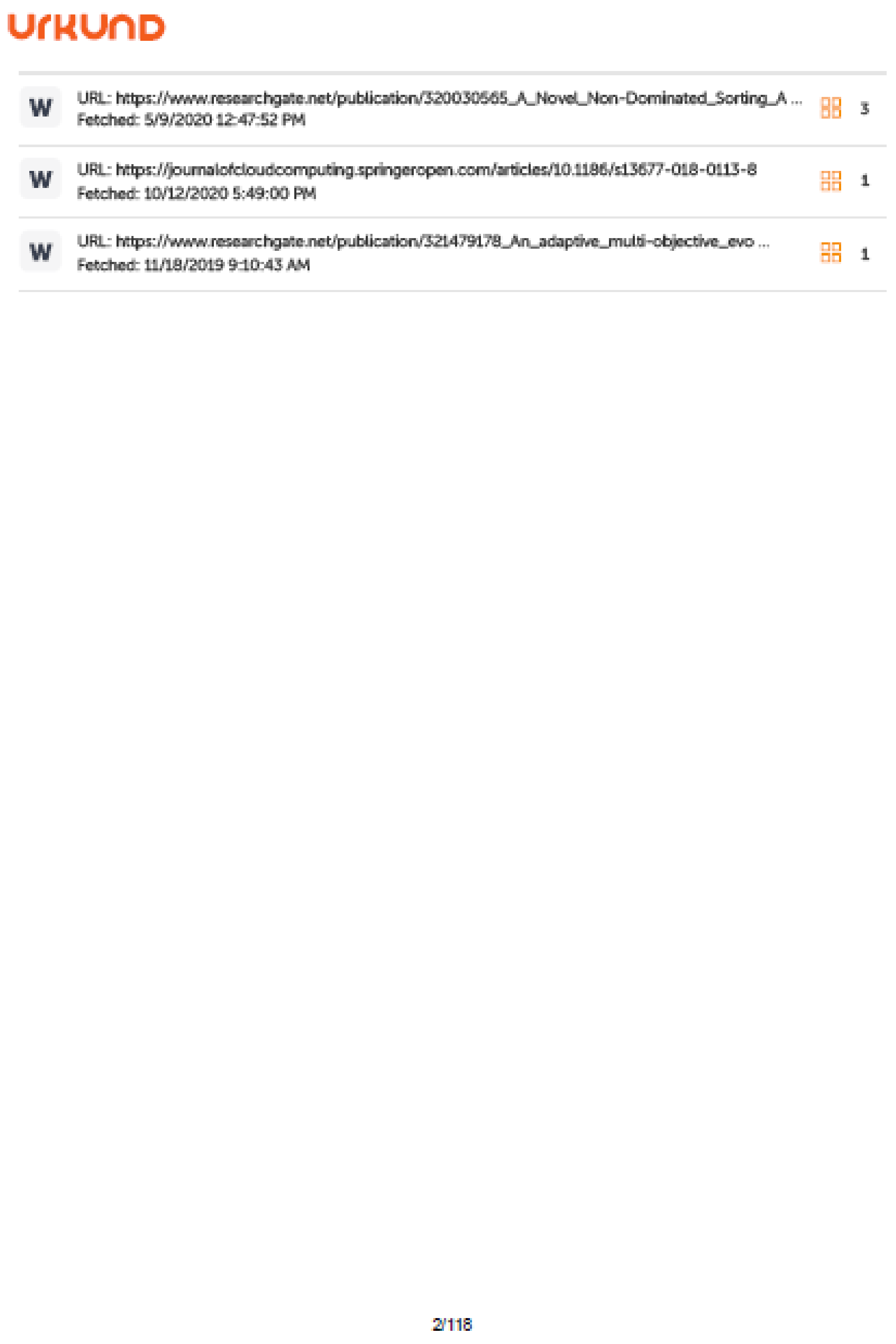
Document Information

Analyzed document	Final Thesis.docx (D81412531)
Submitted	10/12/2020 5:47:00 PM
Submitted by	Preeti Sirohi
Submitter email	preeti.sirohi@imsgzb.com
Similarity	6%
Analysis address	preeti.sirohi.ims@analysis.urkund.com

Sources included in the report

W	URL: https://www.ijeat.org/wp-content/uploads/papers/v8i4c/D24240484C19.pdf Fetched: 12/26/2019 10:06:56 AM	4
SA	**Thesis2018.docx** Document Thesis2018.docx (D40329175)	1
W	URL: https://www.ijeat.org/wp-content/uploads/papers/v8i5/E6893068519.pdf Fetched: 11/7/2019 8:14:46 AM	29
SA	**71010621016-TS(MANIKANDAN R).pdf** Document 71010621016-TS(MANIKANDAN R).pdf (D22925856)	9
W	URL: https://helvia.uco.es/xmlui/bitstream/handle/10396/17194/2018000001823.pdf?sequenc ... Fetched: 5/1/2020 8:45:45 PM	8
SA	**IMS Ghaziabad / RESEARCH PAPER final.docx** Document RESEARCH PAPER final.docx (D62332150) Submitted by: preeti.sirohi@imsgzb.com Receiver: preeti.sirohi.ims@analysis.urkund.com	31
W	URL: https://www.researchgate.net/publication/261477734_Multi-objective_Service_Composi ... Fetched: 6/9/2020 10:34:27 AM	2
W	URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4659544/ Fetched: 10/12/2020 5:49:00 PM	2
SA	**AIRE-D-20-00026_reviewer (4).pdf** Document AIRE-D-20-00026_reviewer (4).pdf (D81065219)	5
SA	**Thesis Report.docx** Document Thesis Report.docx (D40289788)	15
W	URL: https://www.researchgate.net/publication/304107078_Approximate_non-dominated_sorti ... Fetched: 5/9/2020 12:49:05 PM	6
W	URL: https://www.researchgate.net/publication/23276282_An_Efficient_Non-dominated_Sorti ... Fetched: 1/14/2020 10:36:22 AM	2

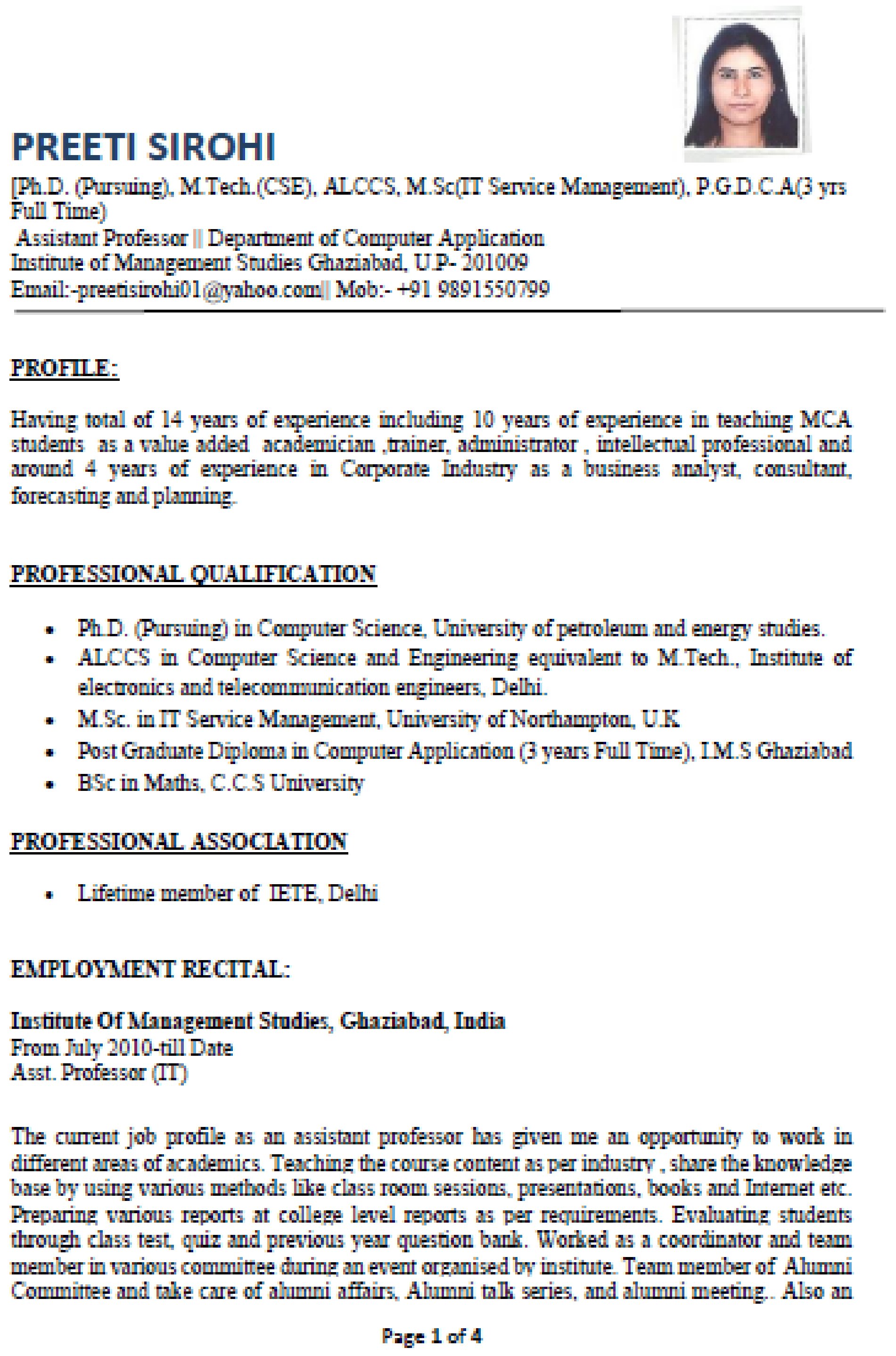
1/118



URKUND

W	URL: https://www.researchgate.net/publication/320030565_A_Novel_Non-Dominated_Sorting_A ... Fetched: 5/9/2020 12:47:52 PM	🟧	3
W	URL: https://journalofcloudcomputing.springeropen.com/articles/10.1186/s13677-018-0113-8 Fetched: 10/12/2020 5:49:00 PM	🟧	1
W	URL: https://www.researchgate.net/publication/321479178_An_adaptive_multi-objective_evo ... Fetched: 11/18/2019 9:10:43 AM	🟧	1

2/118



PREETI SIROHI

[Ph.D. (Pursuing), M.Tech.(CSE), ALCCS, M.Sc(IT Service Management), P.G.D.C.A(3 yrs Full Time)
Assistant Professor || Department of Computer Application
Institute of Management Studies Ghaziabad, U.P- 201009
Email:-preetisirohi01@yahoo.com|| Mob:- +91 9891550799

PROFILE:

Having total of 14 years of experience including 10 years of experience in teaching MCA students as a value added academician ,trainer, administrator , intellectual professional and around 4 years of experience in Corporate Industry as a business analyst, consultant, forecasting and planning.

PROFESSIONAL QUALIFICATION

- Ph.D. (Pursuing) in Computer Science, University of petroleum and energy studies.
- ALCCS in Computer Science and Engineering equivalent to M.Tech., Institute of electronics and telecommunication engineers, Delhi.
- M.Sc. in IT Service Management, University of Northampton, U.K
- Post Graduate Diploma in Computer Application (3 years Full Time), I.M.S Ghaziabad
- BSc in Maths, C.C.S University

PROFESSIONAL ASSOCIATION

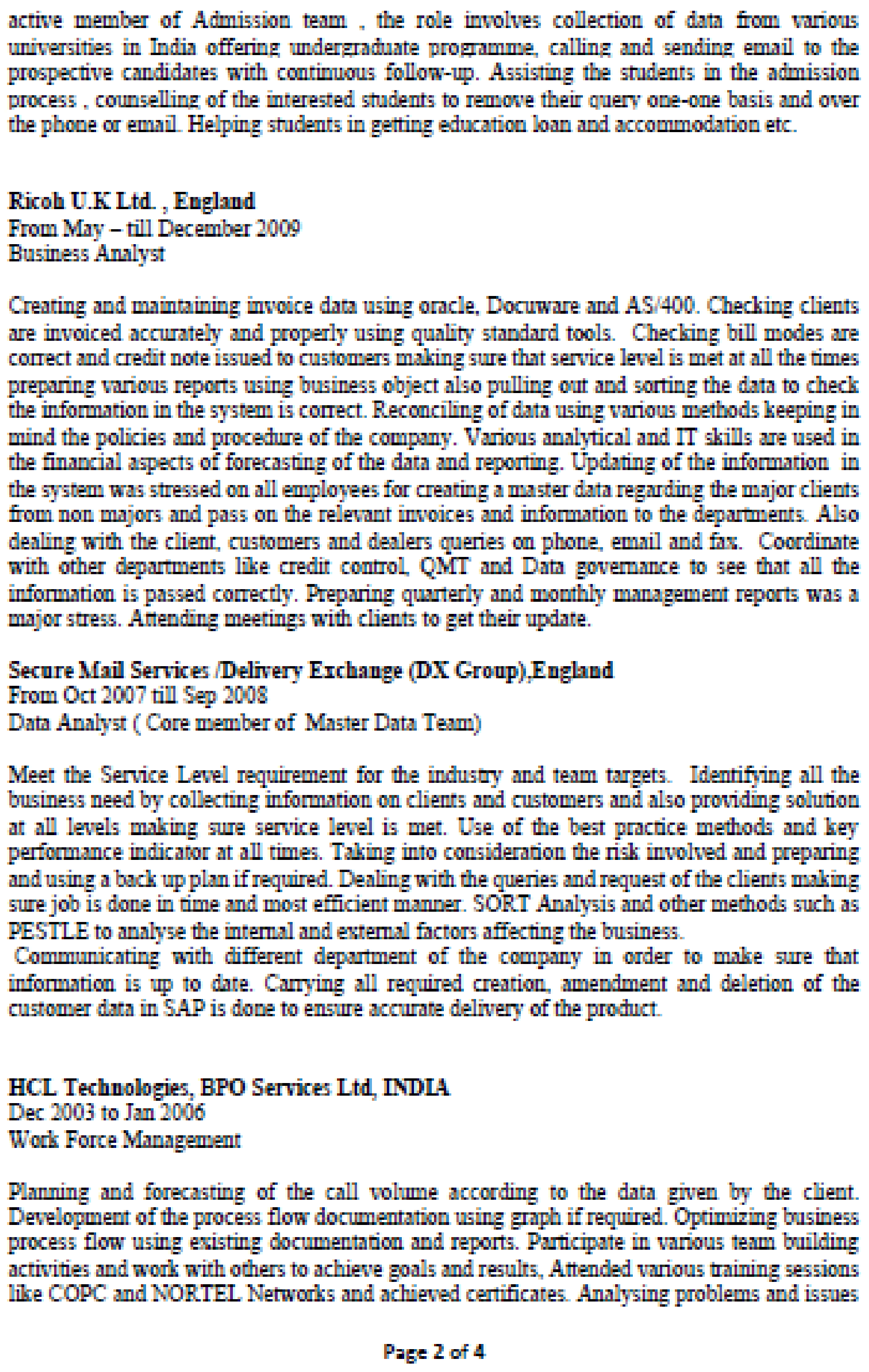
- Lifetime member of IETE, Delhi

EMPLOYMENT RECITAL:

Institute Of Management Studies, Ghaziabad, India
From July 2010-till Date
Asst. Professor (IT)

The current job profile as an assistant professor has given me an opportunity to work in different areas of academics. Teaching the course content as per industry , share the knowledge base by using various methods like class room sessions, presentations, books and Internet etc. Preparing various reports at college level reports as per requirements. Evaluating students through class test, quiz and previous year question bank. Worked as a coordinator and team member in various committee during an event organised by institute. Team member of Alumni Committee and take care of alumni affairs, Alumni talk series, and alumni meeting.. Also an

Page 1 of 4



active member of Admission team , the role involves collection of data from various universities in India offering undergraduate programme, calling and sending email to the prospective candidates with continuous follow-up. Assisting the students in the admission process , counselling of the interested students to remove their query one-one basis and over the phone or email. Helping students in getting education loan and accommodation etc.

Ricoh U.K Ltd. , England
From May – till December 2009
Business Analyst

Creating and maintaining invoice data using oracle, Docuware and AS/400. Checking clients are invoiced accurately and properly using quality standard tools. Checking bill modes are correct and credit note issued to customers making sure that service level is met at all the times preparing various reports using business object also pulling out and sorting the data to check the information in the system is correct. Reconciling of data using various methods keeping in mind the policies and procedure of the company. Various analytical and IT skills are used in the financial aspects of forecasting of the data and reporting. Updating of the information in the system was stressed on all employees for creating a master data regarding the major clients from non majors and pass on the relevant invoices and information to the departments. Also dealing with the client, customers and dealers queries on phone, email and fax. Coordinate with other departments like credit control, QMT and Data governance to see that all the information is passed correctly. Preparing quarterly and monthly management reports was a major stress. Attending meetings with clients to get their update.

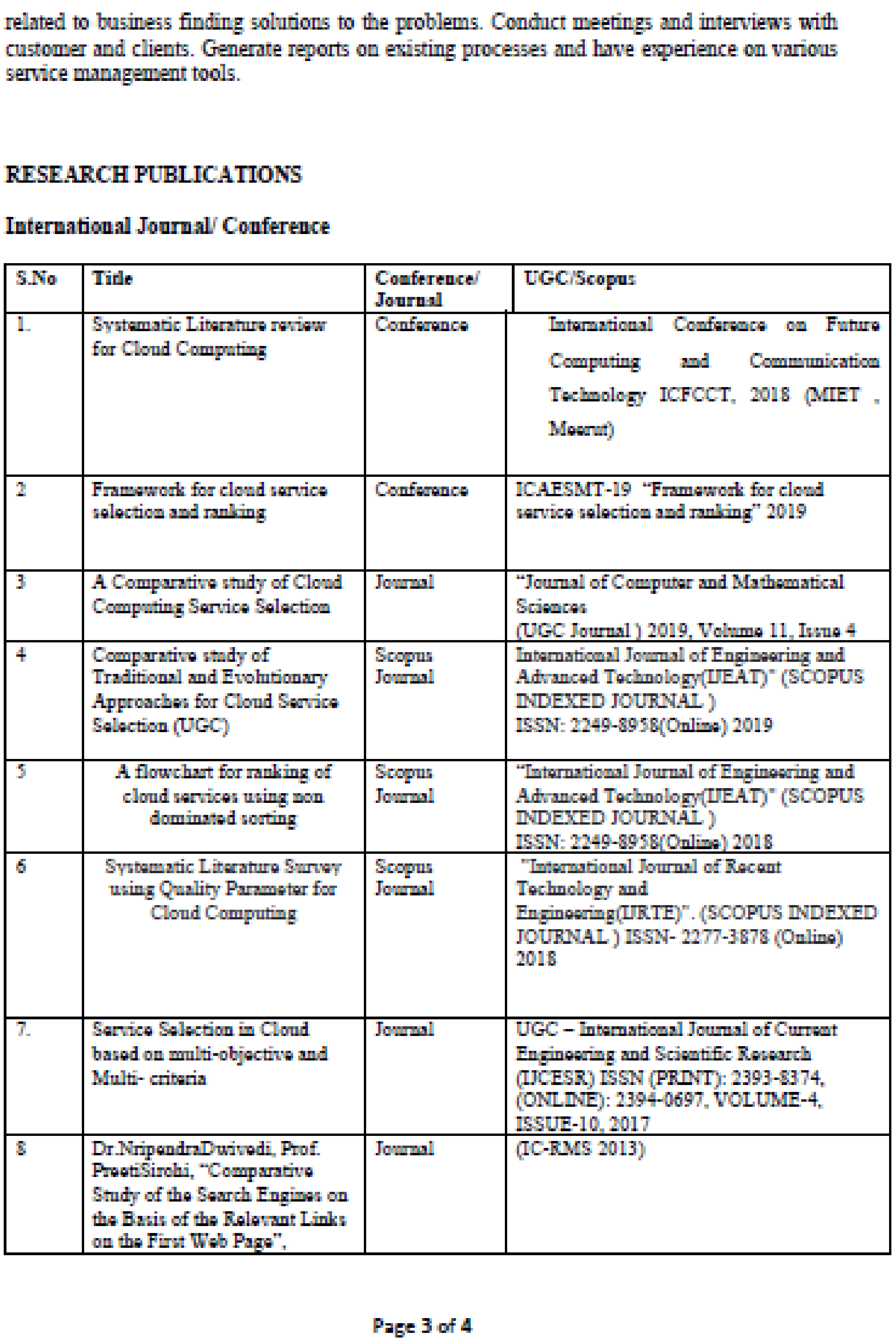
Secure Mail Services /Delivery Exchange (DX Group),England
From Oct 2007 till Sep 2008
Data Analyst (Core member of Master Data Team)

Meet the Service Level requirement for the industry and team targets. Identifying all the business need by collecting information on clients and customers and also providing solution at all levels making sure service level is met. Use of the best practice methods and key performance indicator at all times. Taking into consideration the risk involved and preparing and using a back up plan if required. Dealing with the queries and request of the clients making sure job is done in time and most efficient manner. SORT Analysis and other methods such as PESTLE to analyse the internal and external factors affecting the business.
Communicating with different department of the company in order to make sure that information is up to date. Carrying all required creation, amendment and deletion of the customer data in SAP is done to ensure accurate delivery of the product.

HCL Technologies, BPO Services Ltd, INDIA
Dec 2003 to Jan 2006
Work Force Management

Planning and forecasting of the call volume according to the data given by the client. Development of the process flow documentation using graph if required. Optimizing business process flow using existing documentation and reports. Participate in various team building activities and work with others to achieve goals and results, Attended various training sessions like COPC and NORTEL Networks and achieved certificates. Analysing problems and issues

Page 2 of 4



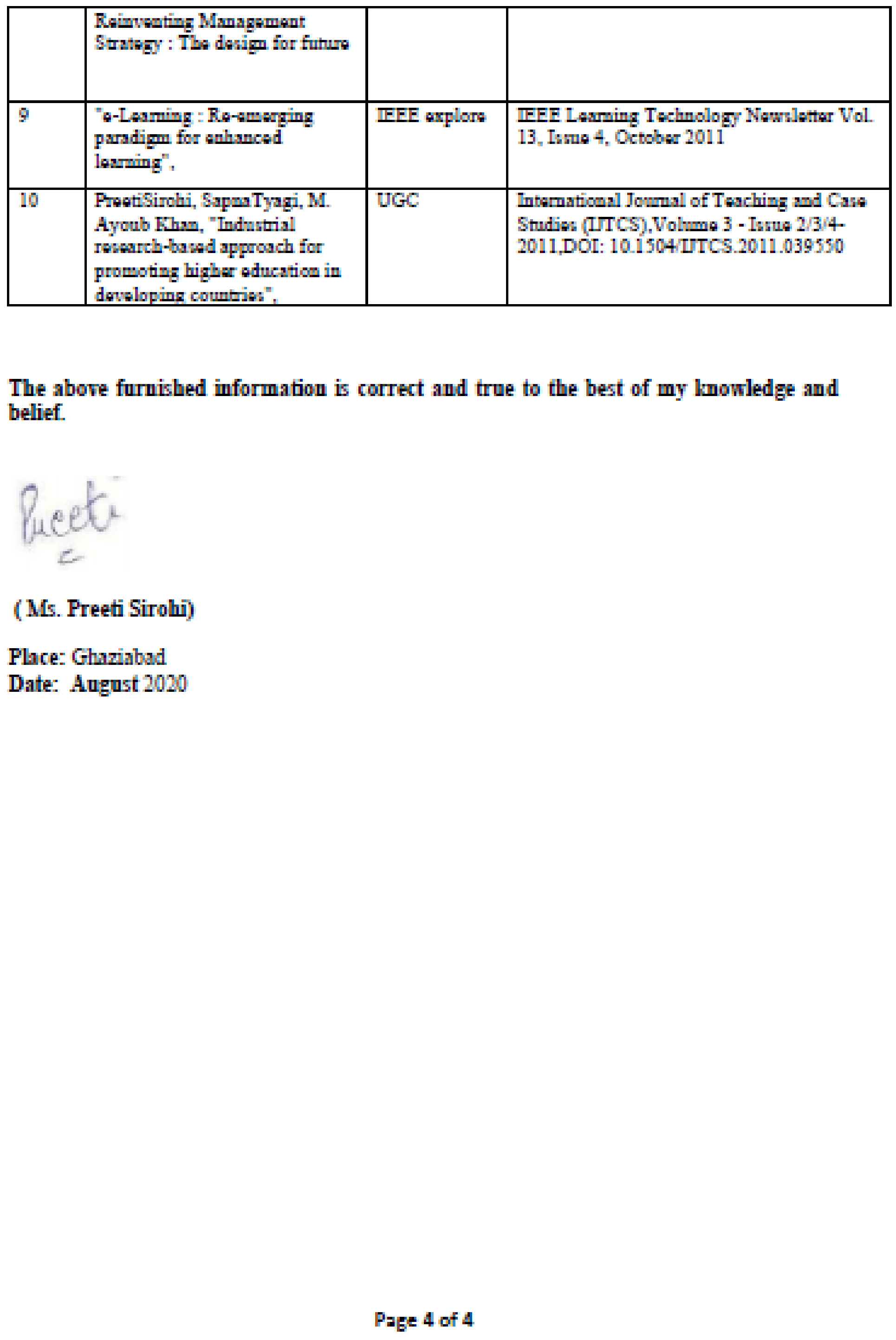
related to business finding solutions to the problems. Conduct meetings and interviews with customer and clients. Generate reports on existing processes and have experience on various service management tools.

RESEARCH PUBLICATIONS

International Journal/ Conference

S.No	Title	Conference/ Journal	UGC/Scopus
1.	Systematic Literature review for Cloud Computing	Conference	International Conference on Future Computing and Communication Technology ICFCCT, 2018 (MIET , Meerut)
2	Framework for cloud service selection and ranking	Conference	ICAESMT-19 "Framework for cloud service selection and ranking" 2019
3	A Comparative study of Cloud Computing Service Selection	Journal	"Journal of Computer and Mathematical Sciences (UGC Journal) 2019, Volume 11, Issue 4
4	Comparative study of Traditional and Evolutionary Approaches for Cloud Service Selection (UGC)	Scopus Journal	International Journal of Engineering and Advanced Technology(IJEAT)" (SCOPUS INDEXED JOURNAL) ISSN: 2249-8958(Online) 2019
5	A flowchart for ranking of cloud services using non dominated sorting	Scopus Journal	"International Journal of Engineering and Advanced Technology(IJEAT)" (SCOPUS INDEXED JOURNAL) ISSN: 2249-8958(Online) 2018
6	Systematic Literature Survey using Quality Parameter for Cloud Computing	Scopus Journal	"International Journal of Recent Technology and Engineering(IJRTE)". (SCOPUS INDEXED JOURNAL) ISSN- 2277-3878 (Online) 2018
7.	Service Selection in Cloud based on multi-objective and Multi- criteria	Journal	UGC – International Journal of Current Engineering and Scientific Research (IJCESR) ISSN (PRINT): 2393-8374, (ONLINE): 2394-0697, VOLUME-4, ISSUE-10, 2017
8	Dr.NripendraDwivedi, Prof. PreetiSirohi, "Comparative Study of the Search Engines on the Basis of the Relevant Links on the First Web Page",	Journal	(IC-RMS 2013)

Page 3 of 4



	Reinventing Management Strategy : The design for future		
9	"e-Learning : Re-emerging paradigm for enhanced learning",	IEEE explore	IEEE Learning Technology Newsletter Vol. 13, Issue 4, October 2011
10	PreetiSirohi, SapnaTyagi, M. Ayoub Khan, "Industrial research-based approach for promoting higher education in developing countries",	UGC	International Journal of Teaching and Case Studies (IJTCS),Volume 3 - Issue 2/3/4-2011,DOI: 10.1504/IJTCS.2011.039550

The above furnished information is correct and true to the best of my knowledge and belief.

[illegible]

(Ms. Preeti Sirohi)

Place: Ghaziabad
Date: **August** 2020

Page 4 of 4