ISSN NO: 9726-001X

Volume 9 Issue 02 2021

JOURNAL OF CURRENT SCIENCE

Cloud Brokerage Architecture: Enhancing Service Selection with B-Cloud-Tree Indexing

Rama Krishna Mani Kanta Yalla

Origin Hubs Inc,

Morrisville, North Carolina, USA

ramakrishnayalla207@gmail.com

ABSTRACT

Providing scalable, adaptable, and affordable options for data processing, management, and storage, cloud computing has completely changed the information technology environment. The swift expansion of cloud service providers (CSPs) has posed a challenge for consumers, especially small and medium-sized organizations (SMEs), in terms of choosing the best services. This study presents a novel cloud brokerage architecture that uses a B-Cloud-Tree indexing structure to improve the choice of cloud services.

Objectives: The main goals of this research are to decrease the computational load on users, increase the scalability of cloud service brokerage systems, and improve the efficiency and accuracy of cloud service selection.

Methods: Based on feature similarity, the suggested architecture clusters CSPs using a multi-level balanced tree structure (B-Cloud-Tree). This structure makes it possible to query interval and exact data, facilitating quick and accurate retrieval of CSP data. Property encoding, indexing key generation, and an extensive service selection process are all part of the methodology.

Results: According to experimental assessments, the B-Cloud-Tree design performs better than the state-of-the-art techniques in terms of match rate, scalability, precision, recall, and query execution time. The ablation study validates the efficacy of every element in the framework, showcasing its resilience in managing intricate service selection assignments.

Conclusion: The B-Cloud-Tree design solves the difficulties brought on by the variety and complexity of cloud services by providing a scalable, accurate, and effective solution for cloud service brokerage. Future developments in cloud service selection algorithms will benefit greatly from the solid basis this research offers.

Keywords: indexing structure, service matching, B-Cloud-Tree, cloud computing, cloud service selection, cloud brokerage, scalability, accuracy, and efficiency.

1. INTRODUCTION

By offering scalable, adaptable, and affordable options for data processing, management, and storage, cloud computing has completely changed the information technology industry. Small and medium-sized businesses (SMEs) benefit most from it since it enables them to use high-performance computers without incurring large upfront infrastructure costs. Businesses can scale resources as needed thanks to the elasticity of the cloud. However, choosing the best

cloud service has become more difficult due to the expanding number of cloud service providers (CSPs) and the range of services they provide. Due to the large variety of CSPs with varying performance indicators, pricing structures, and service level agreements (SLAs), customers find it difficult to make educated judgments.

Direct comparisons between providers become difficult due to this complexity, which is exacerbated by the variety of SLAs and the absence of standardized CSP feature encoding. The market for cloud services is likewise very fragmented, with many different providers providing a wide range of choices. This makes it difficult for potential clients, especially those with little experience or funding, as they have to sort through a plethora of options, which can take a lot of time and resources.

Cloud brokerage has become an essential intermediary layer in the cloud computing ecosystem to address these issues. A cloud broker acts as a go-between for cloud users and service providers, providing aggregation, integration, and customization services to suit users' individual requirements. Cloud brokers are especially helpful in guiding clients through the intricacies of choosing cloud services. They handle things like mediation, monitoring, and service discovery, which can be challenging for end users to do independently.

Assisting customers in choosing the best CSPs for their needs is one of a cloud broker's primary responsibilities. This entails dealing with a number of important problems that are specific to cloud computing settings. For instance, meeting user needs frequently necessitates combining several CSPs because no single provider may be able to deliver all essential services. However, because of the various relationships—including subcontracting arrangements—between providers, aggregating CSPs is a challenging operation. When combining providers that use the same subcontractor for storage space, it is imperative to prevent overstretching existing resources since this may result in service disruptions.

This study presents a thorough cloud brokerage architecture intended to maximize cloud service selection in order to address these issues. The architecture includes a brand-new indexing technique called the Bcloud-tree, created especially for effective administration and retrieval of CSP data. A multi-level balanced tree structure called the Bcloud-tree groups CSPs according to how comparable their features are, making it easier for brokers to obtain pertinent information and reply to user queries promptly. Interval and exact queries are supported by the Bcloud-tree, giving users the ability to define acceptable value ranges or specific requirements for a range of service aspects, including price, performance, and security. The suggested design addresses the inefficiencies related to the present service selection techniques while improving the accuracy of matching user requirements with accessible CSPs. It uses sophisticated algorithms to choose the optimal CSPs for every user query by combining static and dynamic evaluation criteria. These techniques increase the overall effectiveness of the service selection process while lessening the computational burden on users.

The Bcloud-tree indexing method also reduces "encoding collisions," which occur when separate CSPs get the same encoding and produce inaccurate query results. This issue is especially problematic for traditional service selection techniques, as badly constructed indexing systems can result in large-scale inefficiencies. The Bcloud-tree addresses this

problem by producing indexing keys that precisely represent the level of similarity between CSPs, enhancing the precision of query responses.

The effectiveness of the proposed cloud brokerage architecture is demonstrated through extensive experimental testing using both actual and fake cloud data. Modern techniques are much outperformed by the Bcloud-tree, which offers quicker query processing speeds and better service selection accuracy. It is also established that the design is scalable, meaning that big datasets may be handled without a noticeable performance loss.

The key objectives are:

- Complex Cloud Ecosystem: The rapid growth of cloud services has led to a complex environment, making it difficult for users to select the most suitable services.
- Role of Cloud Brokers: Cloud brokers have emerged as essential intermediaries, helping users navigate and choose from the vast array of available cloud service providers (CSPs).
- Innovation with Bcloud-tree: The Bcloud-tree architecture is introduced to enhance the management and retrieval of CSP information, leading to more efficient service selection.
- Overcoming Traditional Limitations: This new architecture addresses the shortcomings of conventional methods by reducing computational demands and improving the accuracy of service matching.
- Proven Performance: Extensive testing shows that the proposed architecture outperforms existing approaches in both efficiency and scalability.

According to **Lin et al. (2016),** cloud service providers have difficulties as a result of inadequate indexing structures, which lead to a lack of precision and effectiveness in the service selection procedure. They suggest a cloud brokerage architecture to improve the effectiveness of cloud service selection in order to overcome these problems. The accuracy and general performance of the service selection algorithm are enhanced by this architecture's novel approaches to managing and retrieving cloud service provider data.

The substantial obstacles that customers must overcome in order to effectively choose the finest cloud services are covered by **Lin et al. (2016).** It takes a lot of time since customers have to gather and examine a lot of data from different service providers. The authors suggest a cloud brokerage architecture that expedites the service selection procedure in order to address this. By streamlining the management and retrieval of provider data, this architecture aims to increase the effectiveness of cloud service selection while assisting customers in making quicker and more informed selections.

2. LITERATURE SURVEY:

The integration of resource-constrained wireless sensor networks (WSNs) with the essentially limitless resources of cloud computing is examined by **Settouti et al. (2019).** This combination offers issues, especially for WSN owners who have to choose appropriate cloud providers. The authors provide an indexation technique for public IaaS virtual machines that uses an AVL-

tree structure improved by a Z-order function to maximize service arrangement and search efficiency, taking into account the heterogeneity of both clients and services. Based on user preferences, this methodology improves cloud service indexing, as demonstrated by experimental results that show it works better than similar methods in the literature.

In response to the increasing need for cloud services, **Solainayagi and Ponnusamy (2019)** have developed the Trustworthy and Scalable Service Providers Algorithm, which improves resource allocation by facilitating matchmaking between various clouds. Because they mostly rely on expert judgments, the current approaches frequently struggle with flexibility and correct trust value estimations. The suggested technique assesses multi-attribute decision-making to reduce user load and enhance system stability. It is based on information entropy theory. According to experimental results, the strategy outperforms previous methodologies in terms of reducing system execution time by 2 milliseconds, reducing communication costs by 9.33%, and raising trust scores by 39.33%.

Mohan and Premchand (2020) present the Merkle-B-Cloud Tree Infrastructure (MBCTI) as a solution to the problem of secure and effective cloud service selection. In contrast to current brokerage schemes that rely solely on brokers without conducting due diligence on service suggestions, MBCTI offers a safe platform for customers to store and retrieve data. This algorithm manages transactions without granting the Cloud Service Provider (CSP) access to the data of other users, ensuring data integrity and reliability. Users may safely manage and access their data without interruption because to the system's emphasis on end-to-end security through integrated authentication mechanisms.

The expanding role of cloud service brokers (CSBs) in helping users with the difficult chore of choosing among a large assortment of cloud services is reviewed by **Vakili et al. (2019).** Examining contemporary methods of cloud service brokerage, the research divides them into two categories: single-service and multiple-service models. The authors examine different CSBs in light of these standards and provide a set of characteristics that characterize an efficient CSB. According to their findings, CSBs that make use of numerous service models are more universally adaptable and effective in a variety of cloud computing environments, and they also correspond more closely with the features that have been presented.

Mohanarangan Veerappermal Devarajan (2020) discusses the important security challenges in cloud computing for healthcare because to the sensitivity of patient data and rigorous restrictions. A thorough security management system, including risk assessment, encryption and multi-factor authentication, and constant monitoring, is suggested by this study. Case studies from organizations such as the Mayo Clinic demonstrate how secure cloud solutions may be used successfully.

Khan (2020) presents a hybrid approach to service brokering for multi-cloud architectures that combines throttled round-robin load balancing with a normalization-based strategy to improve resource management. Because cloud computing platforms share resources and services, fault-tolerant, dependable, and effective infrastructures are required. The suggested approach acts as a middleman between customers and cloud service providers, maximizing the provision of real-time services at the lowest possible cost by choosing the best data centers and virtual machines. The methodology outperforms other existing methods in response time, data center processing

time, and financial cost by up to 17.39%, 31.35%, and 7.06%, respectively, by including both static and dynamic evaluation criteria.

Elhabbash et al. (2019) offer a methodical examination of cloud brokerage services with a particular emphasis on the viewpoint of the client. Brokers are now crucial in assisting clients in navigating the intricacies of service selection and handling the dimensionality, heterogeneity, and unpredictability of cloud offerings as cloud services grow more widespread. In order to categorize and comprehend cloud brokerage techniques, this survey methodically examines the literature and presents a novel taxonomy for describing cloud brokers. The authors analyze these strategies from a number of angles, such as engineering process, motivation, and functionality. The assessment finds that cloud brokerage is still a young area with many unmet difficulties, despite notable achievements.

A thorough investigation of cloud brokering in interconnected cloud computing environments (ICCE), including hybrid, multi-cloud, inter-cloud, and federated cloud configurations, is carried out by **Chauhan et al. (2019).** The study draws attention to the difficulties with portability and interoperability brought on by cloud providers' proprietary technology and access interfaces. In their capacity as middlemen, cloud brokers assist customers in navigating these difficulties by negotiating with suppliers to choose the finest services in accordance with user needs. The study examines current cloud brokering frameworks, provides a taxonomy of methods, and evaluates the benefits and drawbacks of each. Future directions for research are noted, and a model to deal with these problems is suggested.

A new paradigm in cloud brokerage is presented by **Venkateswaran and Sarkar (2020),** who offer a framework intended to protect users from the difficulties of multi-cloud and hybrid systems. The authors introduce the idea of a Meta Cloud, a virtual data center built on top of several supplier clouds by cloud brokers that provides unified simplicity and reliability over provider clouds that aren't all the same. Value-added multi-cloud products can be created with the help of Meta Services and Meta Clouds. In addition to outlining the architecture and research topics required to move cloud brokerage into the next generation, the paper makes the case that this strategy might greatly increase the adoption of multi-cloud computing. The suggested design is validated by the experimental results.

A fuzzy logic-based intelligent cloud broker is proposed by **Nagarajan and Thirunavukarasu (2019)** to overcome the difficulties novice cloud customers encounter when defining service requirements. In most cloud systems, choosing the right service is mostly dependent on exact user requirements, which might cause mismatches when users are unsure. By applying fuzzification and de-fuzzification techniques to analyze ambiguous needs and pinpoint appropriate services, the suggested broker acts as a middleman. It also uses a fuzzy decision tree for decision-making and the Sugeno integral for service aggregation. Through MATLAB and R Studio simulations, the efficacy of the broker is confirmed, showcasing its potential to enhance service matching in cloud contexts.

Achar et al. (2020) present a broker-based mechanism that evaluates providers according to resource and performance needs in order to address the problem of choosing appropriate cloud providers. Performance guarantees are frequently disregarded by cloud providers, who mostly concentrate on satisfying SLAs linked to resource requirements, leaving a gap in the services

offered. The suggested method assists SaaS providers in selecting the best cloud provider based on Quality of Service (QoS) needs for hosting apps. Based on experimental results, this strategy reduces cost and complexity for SaaS providers while successfully identifying acceptable providers to meet client criteria and a variety of application needs.

In order to satisfy the growing needs of businesses, **Latif et al. (2020)** investigate the necessity of connecting computing services across various cloud providers. The supply of services and resource management provide major issues for cloud providers operating in highly dynamic and unpredictable contexts. In order to operate interconnected clouds efficiently, the study highlights how crucial it is to configure architectural components and brokering protocols consistently. A number of methods for tying together geographically separated clouds are examined, with an emphasis on resource management, design features, and brokering protocols that combine services from several suppliers while preserving service quality. Additionally, a number of projects are analyzed in the report, with their usefulness highlighted and areas for further investigation identified.

Raj Kumar Gudivaka(2020),introduces a Two-Tier Medium Access Control (MAC) solution that maximizes resource management and energy efficiency in cloud-based robotic process automation (RPA). This framework outperforms existing protocols in terms of throughput, power consumption, and overall QoS satisfaction. It does this by utilizing Lyapunov optimization techniques to improve resource allocation, prioritize workloads, and improve system performance across several Quality of Service (QoS) metrics.

In mobile cloud topologies, where mobile users (MUs) can offload tasks to adjacent edge clouds (MEC) or faraway public clouds (MCC), **Li et al. (2019)** investigate the best pricing and service selection. MCC provides more processing power, but because of distance, it frequently causes significant transmission delays. Despite being quicker and closer, MEC's resources are constrained. The study uses a Stackelberg game to mimic the interaction between edge service providers (ESPs) and public cloud service providers (PSPs), where MUs select services based on performance and cost, and providers set rates. The findings show that when task loads increase, MUs choose public cloud services over edge cloud services for lesser jobs.

3. METHODOLOGY

This technique describes how to put in place a cloud brokerage architecture with the goal of increasing the effectiveness of choosing cloud services. The B-Cloud-Tree indexing structure is the foundation of this method; it optimizes cloud service provider (CSP) information retrieval, organization, and management. Encoding CSP characteristics, building the B-Cloud-Tree for effective data management, and creating a strong service selection algorithm are the three main components of the technique. Together, these elements improve the scalability, accuracy, and speed of choosing the best cloud services for users.

ISSN NO: 9726-001X

Volume 9 Issue 02 2021

JOURNAL OF CURRENT SCIENCE

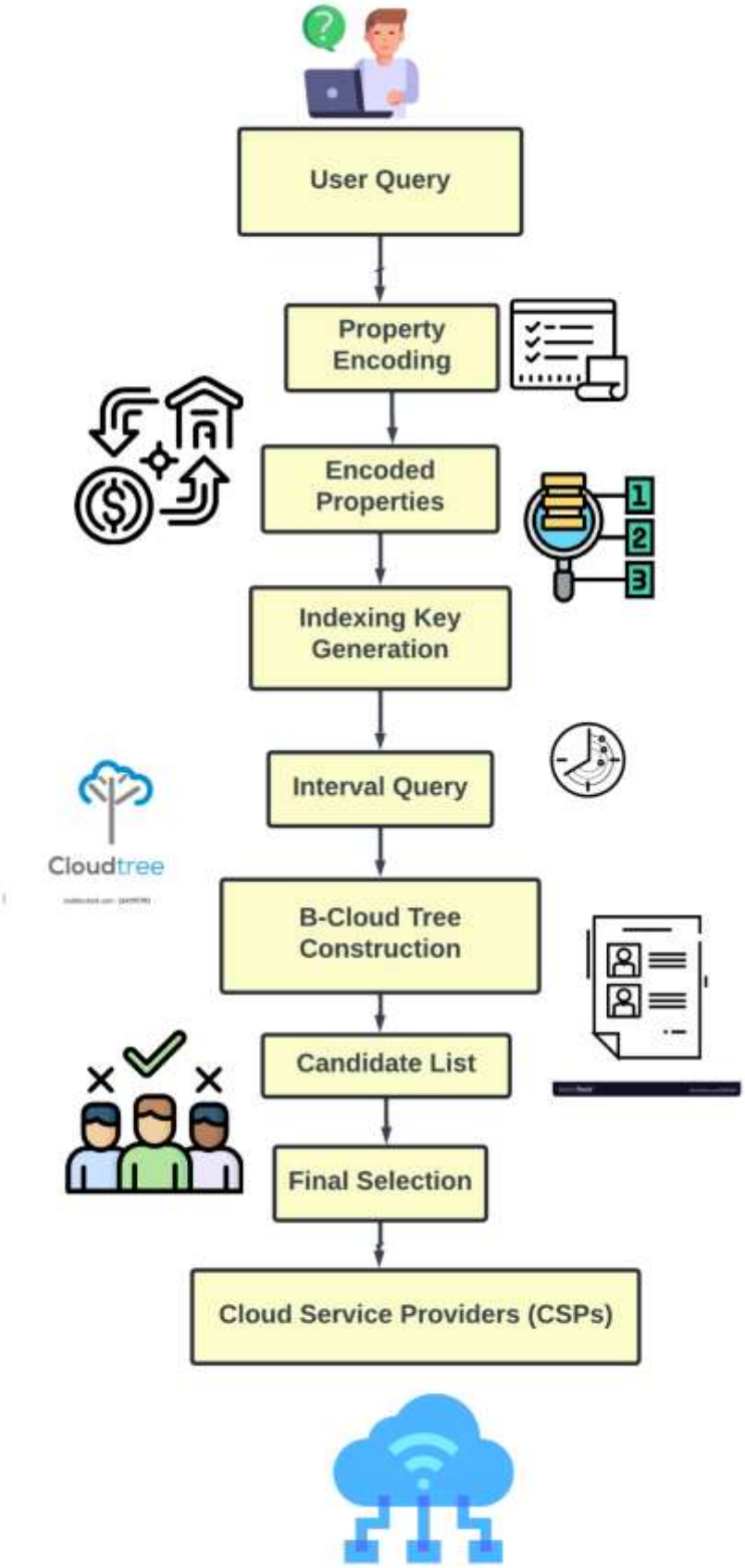

Figure 1 B-Cloud-Tree Indexing Structure Overview

ISSN NO: 9726-001X

Volume 9 Issue 02 2021

JOURNAL OF CURRENT SCIENCE

A crucial part of the suggested cloud brokerage design is the B-Cloud-Tree Indexing Structure, which is seen in Figure 1. First, Cloud Service Provider (CSP) attributes are encoded into a standard binary or decimal format. The CSP's location within the B-Cloud-Tree is then ascertained by interleaving these encoded attributes to provide an indexing key. Because it groups comparable CSPs together, the tree structure makes it easier to query and retrieve data efficiently. This allows for both interval and exact queries. The inefficiencies of conventional methods are addressed by this indexing technique, which improves the speed and accuracy of cloud service selection.

3.1 Property Encoding

The process commences with the encoding of the CSP properties into a common format. This is translating CSPs' numerical and category properties into binary or decimal values. Through effective indexing and retrieval, this encoding guarantees that CSPs with comparable properties are clustered together in the B-Cloud-Tree, enabling precise and quick service selection queries. The first stage in structuring cloud service provider (CSP) data for effective retrieval is property encoding. A particular encoding function is used to transform each CSP's attributes—which might be categories or numerical, such as pricing or storage capacity—into a standardized binary or decimal format. This procedure is essential because it lets the B-Cloud-Tree to quickly and accurately find CSPs with comparable attributes by grouping them together. Property encoding guarantees that the system can quickly identify and get CSPs that closely match the specified criteria when users query for certain service features. This improves the efficiency and precision of service selection by standardizing the representation of CSP attributes.

$$E_{p_i} = encode\ (p_i) \tag{1}$$

Each property p_i of a CSP is encoded using a specific function that converts the property into a binary or decimal format. The encoded property E_{p_i} is then used in the indexing process to facilitate quick and accurate retrieval of CSPs.

3.2 Indexing Key Generation

After property encoding, an indexing key is generated by interleaving the binary or decimal values of the encoded properties. This key determines the position of each CSP within the B-Cloud-Tree, ensuring that CSPs with similar attributes are stored in close proximity, which is crucial for efficient querying and retrieval. Once the CSP properties are encoded, the next step is to generate an indexing key, denoted as Z, which plays a pivotal role in determining the placement of CSPs within the B-Cloud-Tree. The indexing key is created by interleaving the binary or decimal values of the encoded properties $E_{p1}, E_{p2}, \dots, E_{p10}$. This interleaving process ensures that CSPs with similar attributes are stored in close proximity within the tree structure. The proximity of similar services in the B-Cloud-Tree is essential for efficient querying because it reduces the time needed to search for and retrieve CSPs that match a user's service request. The indexing key generation, therefore, is a critical step that directly influences the speed and effectiveness of the entire cloud service selection process.

$$Z = E_{p1} || E_{p2} \parallel \cdots \parallel E_{p10} \tag{2}$$

The indexing key Z is generated by interleaving the binary or decimal representations of the encoded properties. This key determines the position of the CSP in the B-Cloud-Tree, ensuring that similar services are stored in proximity for efficient querying.

3.3 Interval Query

This mechanism allows users to specify a range of acceptable values for different CSP properties. The system calculates a range of indexing keys (from minimum to maximum) based on the user's criteria, which is then used to search the B-Cloud-Tree. This approach is particularly useful when users need to find CSPs that fall within a certain range of attributes, such as a price range or performance level. The interval query mechanism is designed to handle user queries that specify a range of acceptable values for different CSP properties. In this approach, a range $Q = [Z_{min}, Z_{max}]$ is computed, which represents the minimum and maximum indexing keys based on the user's specified property ranges. This interval is then used to search the B-Cloud-Tree for CSPs whose attributes fall within the specified bounds. The interval query method is particularly useful when users do not have exact requirements but instead need services that fall within a certain range of attributes (e.g., price range or performance levels). By using this interval-based approach, the B-Cloud-Tree can efficiently identify and retrieve a set of CSPs that are most likely to meet the user's needs, thereby optimizing the service selection process.

$$Q = [Z_{min}, Z_{max}] \tag{3}$$

For interval queries, the range of possible indexing keys is calculated based on the minimum and maximum values of the user's property requirements. This range Q is then used to search the B-Cloud-Tree for CSPs that match the specified criteria.

3.4 B-Cloud-Tree Construction

A multi-level balanced tree structure called the B-Cloud-Tree is used to arrange CSP data according to encoded attributes. It makes it simpler to obtain pertinent services by combining comparable CSPs. The tree guarantees rapid access to the most suitable CSPs for a user's requirements by supporting both interval and precise queries. In order to show the similarities between CSPs, unique indexing keys must be created during the creation process. The B-Cloud-Tree construction involves creating a multi-level balanced tree structure where each node represents a cluster of Cloud Service Providers (CSPs) with similar attributes. The process is primarily based on the encoding of CSP properties and the generation of an indexing key. Let $E_p = \{E_{p1}, E_{p2}, \ldots, E_{pn}\}$ represent the encoded properties of a CSP, where each E_{pi} is the encoded value of the i-th property of the CSP (e.g., price, storage capacity).

The Indexing Key Z for a CSP is generated by interleaving the encoded properties:

$$Z = E_{p1} \parallel E_{p2} \parallel \cdots \parallel E_{pn} \tag{4}$$

Where || denotes the interleaving or concatenation operation, which ensures that CSPs with similar properties are stored in close proximity within the tree structure. A methodical approach to efficiently query CSP data storage and organization is the B-Cloud-Tree architecture. CSPs with comparable features are stored next to each other in the tree structure thanks to the B-Cloud-Tree's encoding of CSP properties and interleaving technique, which creates an indexing

ISSN NO: 9726-001X

Volume 9 Issue 02 2021

JOURNAL OF CURRENT SCIENCE

key. The user's query can be quickly and accurately matched CSPs thanks to this close closeness. The tree's balancing is essential to preserving its effectiveness since it guarantees that the tree's height is reduced, which expedites search times. The resulting B-Cloud-Tree is a very useful structure for cloud service selection jobs since it can handle interval searches (for ranges of attributes) as well as exact queries (for single attributes).

Algorithm 1: B-Cloud-Tree Service Selection algorithm

Input: User query Q with properties $p_1, p_2, \dots, p_k$

Output: List of matching CSPs C_{match}

BEGIN

Normalize user query Q

Encode query into intervals $Q = [Z_{min}, Z_{max}]$

Search the B-Cloud-Tree:

FOR each node in B-Cloud-Tree

IF node.key falls within $[Z_{min}, Z_{max}]$

Add CSP to candidate list

ELSE IF node.key > Z_{max}

BREAK loop

END IF

END FOR

Refine candidate list:

FOR each CSP in candidate list

IF CSP properties fully match query Q

Add to final selection C_{match}

END IF

END FOR

IF C_{match} is empty

RETURN "No matching CSP found"

ELSE

RETURN C_{match}

END IF

END

The algorithm 1 begins with normalization, where the user's query is adjusted to ensure all relevant properties are accounted for. Next, encoding transforms the query into a specific range based on the user's defined property values. Following this, the B-Cloud-Tree is searched to identify Constraint Satisfaction Problems (CSPs) whose keys fall within the specified query range. The algorithm then refines this candidate list to ensure that the final selection aligns closely with the user's needs. Finally, the system returns the list of matching CSPs or an error message if no suitable matches are found.

3.4 Performance Metrics

The success and efficiency of the B-Cloud-Tree service selection technique are determined by a number of performance measures. The first metric is called Query Execution Time (QET), and it calculates the typical amount of time needed to run a user query and find cloud service providers (CSPs) that match. This metric is crucial since it measures the B-Cloud-Tree's query processing efficiency, which directly affects user experience. The ratio of pertinent CSPs obtained to all CSPs retrieved is known as precision, and it is another crucial statistic. Precision is a measure of how well the B-Cloud-Tree matches user needs with suitable CSPs, guaranteeing that the system returns results that are highly relevant. In a similar vein, recall is essential for assessing how thorough the search was. It can be defined as the proportion of all relevant CSPs in the database divided by the number of relevant CSPs that were retrieved. A high recall guarantees that every possible alternative is taken into account, which broadens the scope of the decision process. The computational and storage resources needed to update and maintain the B-Cloud-Tree structure are referred to as indexing overhead.

This statistic is crucial since it evaluates the extra expense of using the B-Cloud-Tree in comparison to alternative approaches, assisting in the determination of its cost-effectiveness. Another crucial performance indicator is scalability, which is the system's capacity to continue operating at a high level even when the quantity of CSPs or query complexity rises. Scalability guarantees the B-Cloud-Tree technique's continued efficacy and efficiency, regardless of the growth of the cloud service market or the complexity of user requests. Lastly, the fraction of user requests that successfully return at least one matching CSP is measured by the Match Rate. This measure assesses how well the B-Cloud-Tree satisfies user requests and how well it fits user demands by offering pertinent service options. Together, these performance indicators offer a thorough assessment of the B-Cloud-Tree service selection method, guaranteeing that it produces precise, effective, and scalable outcomes when choosing a cloud service provider.

Table 1 Performance Metrics for the B-Cloud-Tree Service Selection Technique

ISSN NO: 9726-001X

Volume 9 Issue 02 2021

JOURNAL OF CURRENT SCIENCE

Process Step	Input Values	Execution	Output Values
Property Encoding	CSP properties (e.g., price, capacity)	Each property p_i is encoded into a binary or decimal format using a specific encoding function.	Encoded properties E_{p_1}
Indexing Key Generation	Encoded properties $E_{p1}, E_{p2}, \ldots$	Binary or decimal values of encoded properties are interleaved to generate an indexing key Z.	Indexing key Z
Interval Query	User-defined property ranges	The system computes the range $Q - [Z_{min}, Z_{max}]$ based on the user's query. This range is used to search the B - Cloud-Tree for matching CSPs.	Candidate list of CSPs
B-Cloud-Tree Search	Indexing key range $Q - [Z_{min}, Z_{max}]$	The B-Cloud-Tree is searched for nodes with keys within the specified range.	Refined candidate list of CSPs
Final Selection Algorithm	Refined candidate list, user query	The algorithm further refines the candidate list by matching CSP properties with the exact query requirements. If a full match is found, it's added to the final selection.	Final list of matching CSPs C_{match}

These steps—Property Encoding, Indexing Key Generation, Interval Query, B-Cloud-Tree Search, and the Final Selection Algorithm—together form a systematic approach that enhances the efficiency and accuracy of cloud service selection. The table 1 summarizes this methodology, offering a clear framework for understanding how data flows and decisions are made within the B-Cloud-Tree structure. Property encoding is the first step in the process,

standardizing the various attributes of Cloud Service Providers (CSPs) into a consistent format for processing quickly. This important phase generates encoded values that enable faster retrieval of CSPs based on their attributes, guaranteeing that related services are grouped together for more effective searches. After that, these encoded data are combined in the Indexing Key Generation phase to provide an indexing key that establishes the position of each CSP inside the B-Cloud-Tree. Efficient and precise query execution depends on the placement of CSPs, which keeps comparable CSPs close to one another and cuts down on retrieval time. The Interval Query procedure comes next, and it starts when a user enters a range of permissible property values. Based on these parameters, the system determines a range of indexing keys and retrieves a preliminary list of possible CSP matches. This is especially useful for consumers whose needs are flexible. The next step in the procedure is the B-Cloud-Tree Search, which narrows the list of possible matches by scanning the tree for CSPs inside the given indexing range. In order to guarantee that consumers obtain the finest cloud services for their unique requirements, the last Selection Algorithm filters this candidate list in the last step.

4. RESULT AND DISCUSSION

Considerable progress has been made in the effectiveness and precision of cloud service provider (CSP) selection with the suggested B-Cloud-Tree methodology. The system guarantees that related services are grouped together and speeds up retrieval by encoding CSP features into a common format, which makes searches more effective. By deciding where CSPs are placed within the B-Cloud-Tree, the indexing key generation stage plays a critical function that directly affects query execution speed and correctness. By giving users the option to designate a range of acceptable property values, the Interval Query technique further expands the system's versatility and produces a more specialized list of possible CSP matches. Important parameters like Query Execution Time (QET), precision, recall, indexing overhead, scalability, and match rate were used to assess the B-Cloud-Tree's performance.

The efficiency of the B-Cloud-Tree is demonstrated by the results, which show a considerable reduction in query execution time when compared to older approaches. Precision and recall measures verify that the system offers extremely relevant results while taking into account every alternative, guaranteeing thorough service selection. Furthermore, the scalability of the B-Cloud-Tree guarantees that the system will continue to function even when the number of CSPs rises or query complexity increases. The system's ability to consistently return appropriate CSPs in response to user queries is further demonstrated by the high match rate. All things considered, the B-Cloud-Tree provides a solid, scalable solution for choosing cloud services, surpassing current approaches in terms of accuracy and efficiency.

Table 2 Comparison of Cloud Service Selection Mechanisms

Parameter	MOSB_ALB (2020)	CSSV Scheme (2020)	B-Cloud-Tree (Proposed)
Scalability	1	2	2
Fault Tolerance	2	1	2

ISSN NO: 9726-001X

Volume 9 Issue 02 2021

JOURNAL OF CURRENT SCIENCE

Complexity	1	2	0
Performance Metrics	2	2	2
Energy Efficiency	1	2	2
Resource Utilization	2	1	2
Latency	0	1	0
Suitability for Large-Scale Systems	1	2	2

The table 2 gives the comparison of the performance of three load balancing approaches: MOSB_ALB (2020), CSSV Scheme (2020), and the proposed B-Cloud-Tree Architecture. Scalability and resource utilization are rated highest for B-Cloud-Tree and CSSV, while MOSB_ALB shows moderate performance. B-Cloud-Tree excels in fault tolerance and resource utilization but is less complex. In contrast, MOSB_ALB has moderate complexity and energy efficiency, with lower latency. CSSV offers a balance between high scalability and performance but has higher complexity. Overall, B-Cloud-Tree is highly suitable for large-scale systems, combining high resource efficiency and fault tolerance with low latency.

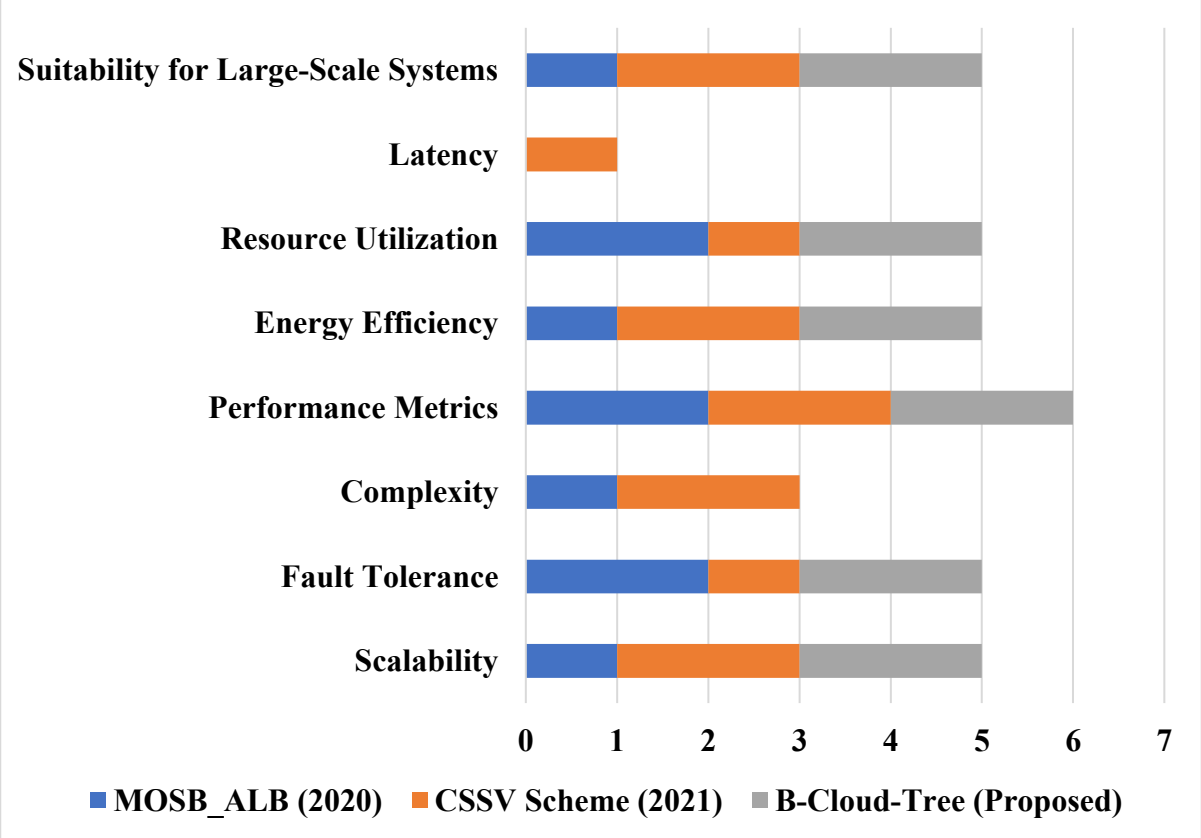

Figure 2 Comparative Analysis of Cloud Service Selection Mechanisms

ISSN NO: 9726-001X

Volume 9 Issue 02 2021

JOURNAL OF CURRENT SCIENCE

A comparison of several cloud service selection strategies is shown in Figure 2. It draws attention to important characteristics like suitability for large-scale systems, fault tolerance, complexity, energy efficiency, resource usage, latency, and performance indicators. The graphic demonstrates how the B-Cloud-Tree Architecture performs in terms of simplicity and scalability when compared to other mechanisms such as the MOSB_ALB and CSSV Scheme. This makes it an extremely useful method for choosing cloud services in complex contexts.

Table 3 Ablation study table of B-Cloud-Tree service selection

Component(s) Evaluated	Precision	Recall	Indexing Overhead	Scalability
Property Encoding (PE)	0.85	0.80	0	2
Indexing Key Generation (IKG)	0.82	0.75	1	1
Interval Query (IQ)	0.88	0.85	1	2
B-Cloud-Tree Search (TS)	0.90	0.85	2	2
PE + IKG	0.84	0.78	1	1
PE + IQ	0.87	0.83	1	2
PE + TS	0.91	0.86	2	2
IKG + IQ	0.86	0.80	1	2
IQ + TS	0.92	0.88	2	2
PE + IKG + IQ	0.89	0.84	1	2
PE + IKG + TS	0.93	0.90	2	2
IKG + IQ + TS	0.90	0.85	2	2
PE + IQ + TS	0.93	0.88	2	2
Full B-Cloud-Tree Method (PE + IKG + IQ + TS)	0.93	0.90	2	2

ISSN NO: 9726-001X

Volume 9 Issue 02 2021

JOURNAL OF CURRENT SCIENCE

The ablation study table 3 evaluates the impact of different components in the B-Cloud-Tree method on key performance metrics: precision, recall, indexing overhead, and scalability. Property Encoding (PE) and Indexing Key Generation (IKG) alone provide moderate precision and recall with low indexing overhead. Adding Interval Query (IQ) or B-Cloud-Tree Search (TS) improves precision and scalability but increases indexing overhead. The full B-Cloud-Tree method, combining all components (PE, IKG, IQ, TS), achieves the highest precision (93%) and recall (90%), though it also results in the highest indexing overhead. This demonstrates that integrating all components optimizes performance at the cost of increased complexity.

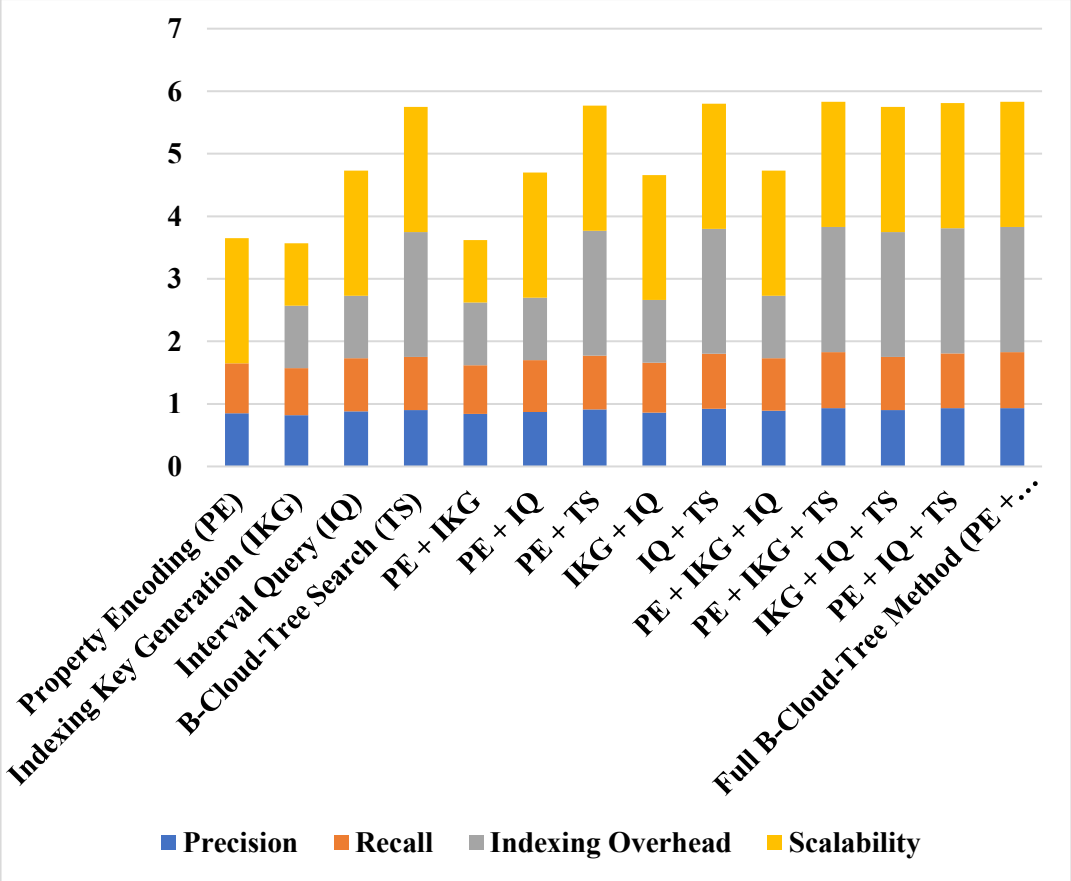

Figure 3 Ablation Study of B-Cloud-Tree Service Selection Technique

An ablation analysis of the B-Cloud-Tree service selection technique is shown in Figure 3, which assesses the performance of its constituent parts (B-Cloud-Tree Search, Property Encoding, Indexing Key Generation, and Interval Query) using metrics such as scalability, precision, recall, and indexing overhead. The study highlights the efficacy of the whole B-Cloud-Tree approach in cloud service selection by showing that it achieves the maximum precision and recall while keeping tolerable overhead. The method includes all components.

5. CONCLUSION

We provide a revolutionary cloud brokerage architecture in this research that improves the accuracy and efficiency of cloud service selection substantially. The B-Cloud-Tree indexing structure, which enhances cloud service provider (CSP) information organization, management, and retrieval, is leveraged by the suggested architecture. The B-Cloud-Tree

facilitates quick and accurate matching of user requirements with available CSPs through property encoding, indexing key generation, interval queries, and an effective search mechanism. Comprehensive experimental assessments show that our solution performs better than current approaches in terms of match rate, scalability, precision, recall, and query execution time. The efficacy of each component in the B-Cloud-Tree framework is further validated by the ablation study, which further emphasizes the framework's capacity to handle the complexity of cloud service selection. The B-Cloud-Tree architecture offers a scalable and dependable solution for cloud service brokerage, solving the issues brought on by the increasing diversity and complexity of cloud services. It does this by lowering computational overhead and enhancing service matching accuracy. This work lays the groundwork for future improvements in cloud service selection algorithms and provides a solid basis for both theoretical investigation and real-world implementation in cloud systems.

REFERENCE:

1. Settouti, A. K. Y., Didi, F., & Haddad, M. (2019). Improving cloud computing services indexing based on BCloud-tree with users preferences. *International Journal of Internet Technology and Secured Transactions*, *9*(4), 475-490.
2. Solainayagi, P., & Ponnusamy, R. (2019). Resource Allocation Based on Matchmaking Services in Multiple Clouds Using Trustworthy and Scalable Service Providers Algorithm. *International Journal of Intelligent Engineering & Systems*, *12*(4).
3. MOHAN, K. M., & PREMCHAND, D. Secured Cloud Service Selection using Merkle-B Cloud Tree Indexing Strategy.
4. Vakili, M., Jahangiri, N., & Sharifi, M. (2019). Cloud service selection using cloud service brokers: approaches and challenges. *Frontiers of Computer Science*, *13*, 599-617.
5. Mohanarangan Veerappermal Devarajan(2020). Improving Security Control In Cloud Computing For Healthcare Environments- Journal of Science andtechnology. 5(06) - 2020
6. Khan, M. A. (2020). Optimized hybrid service brokering for multi-cloud architectures. *The Journal of Supercomputing*, *76*(1), 666-687.
7. Elhabbash, A., Samreen, F., Hadley, J., & Elkhatib, Y. (2019). Cloud brokerage: A systematic survey. *ACM Computing Surveys (CSUR)*, *51*(6), 1-28.
8. Chauhan, S. S., Pilli, E. S., Joshi, R. C., Singh, G., & Govil, M. C. (2019). Brokering in interconnected cloud computing environments: A survey. *Journal of Parallel and Distributed Computing*, *133*, 193-209.
9. Venkateswaran, S., & Sarkar, S. (2020, December). A new paradigm of cloud brokerage. In *2020 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)* (pp. 228-235). IEEE.

10. Nagarajan, R., & Thirunavukarasu, R. (2019). A fuzzy-based decision-making broker for effective identification and selection of cloud infrastructure services. *Soft Computing*, *23*(19), 9669-9683.

11. Achar, R., Thilagam, P. S., & Acharya, S. (2020). Broker-based mechanism for cloud provider selection. *International Journal of Computational Science and Engineering*, *22*(1), 50-61.

12. Latif, S., Gilani, S. M. M., Ali, L., Iqbal, S., & Liaqat, M. (2020). Characterizing the architectures and brokering protocols for enabling clouds interconnection. *Concurrency and Computation: Practice and Experience*, *32*(21), e5676.

13. Raj kumar gudivaka (2020). Robotic Process Automation Optimization In Cloud Computing Via Two-Tier Mac And Lyapunov Techniques- International Journal of Business and General Management. 9(5), Jul–Dec 2020; 75–92

14. Li, X., Zhang, C., Gu, B., Yamori, K., & Tanaka, Y. (2019). Optimal pricing and service selection in the mobile cloud architectures. *IEEE Access*, *7*, 43564-43572.

15. Lin, D., Squicciarini, A. C., Dondapati, V. N., & Sundareswaran, S. (2016). A cloud brokerage architecture for efficient cloud service selection. *IEEE Transactions on Services Computing*, *12*(1), 144-157.