SELECTING THE SLA GUARANTEE BY EVALUATING THE QOS AVAILABILITY

Ihab Sekhi
PhD student, University of Miskolc, Institute of Information Technology
3515 Miskolc, Miskolc-Egyetemváros, e-mail: ihab.razzaq@stu.edu.iq

Abstract

Consumers face confusion when selecting a Service level agreement from a Cloud Service Provider (CSP) due to the wide range of available options, such as Platform-as-a-Service, Infrastructure-as-a-Service, Software-as-a-Service and Network-as-a-service. CSPs provide their services to consumers based on the availability of computing and networking, which define the nature of the service and the corresponding costs. However, strict SLA adherence and achieving quality of service (QoS) can be challenging. In many cases, the availability of services falls short, making it difficult for consumers to choose the most suitable and guaranteed SLA among those offering similar functionality. Therefore, ensuring the availability of QoS becomes crucial for SLA selection and user satisfaction in cloud computing. In this paper, we introduce a fuzzy logic system-based model designed to classify SLA into 11 levels, ranging from 90% to 99.999%, contingent upon the QOS availability of computing (up and downtime) and QOS availability of networking (bandwidth, jitter, round-trip time, and packet loss) metrics. The research had two primary objectives: (i) To develop a versatile SLA model that employs nuanced techniques, diverging from typical CSP offerings, addressing both dominant forms of QoS availability. (ii) To improve the precision of SLA categorization, tailored to each user's specific requirements, enhancing task efficiency and cost-effectiveness. Our research was conducted using the MATLAB program.

Keywords: *Cloud Service Providers, uptime and downtime, round trip daley, Service Level Agreement, Fuzzy inference system*

1. Introduction

Cloud computing represents a cutting-edge paradigm in the world of networking. It enables seamless and instant access to computing resources, including applications, servers, storage space, services, and networks, without requiring any upfront investment. This technology offers high scalability, accommodating the varying needs of users. With cloud computing, individuals and businesses pay only for the resources they utilize. By harnessing the power of the cloud, data from all corners of the globe converge, making it readily accessible to users. The ultimate goal is to provide services to end users at any time and from any location.

Cloud Infrastructure provides three distinct service models for service delivery: software-as-a-service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS). Service providers offer these models in a reliable and cost-effective manner, earning the trust of their customers (Baliyan and Kumar, 2013).

As this technology becomes ubiquitous and utilized on desktops and mobile devices, particular challenges have surfaced for service providers and customers. The increasing number of cloud users and expanding storage capacity have heightened concerns regarding user privacy and security (Alhamad et al., 2011).

While cloud providers offer services and applications to their users, there are significant guarantee issues that need to be addressed. These include aspects like the availability, uptime, and downtime as stipulated in the SLA (Xiaoyong et al., 2015), as well as factors like throughput, delay or round trip time, variation/jitter, and packet loss related to network availability (Kihuya et al., ...). Consequently, customers may find it challenging to comprehend the SLA decision framework, which is essential for ensuring the timely and cost-effective fulfillment of their requested demands.

Users of cloud computing services need to be sure that their providers offer guarantees concerning quality of service (QOS) networking, such as Bandwidth, round-trip delay, jitter and packet loss, and quality of service (QOS) computing metrics, such as uptime and downtime. Before they can begin using cloud computing services, it is essential to conduct research on and engage in conversation with providers of cloud services to guarantee an SLA. Doing so must cultivate a favourable and trustworthy connection between the supplier and the customer. Furthermore in a cloud environment, defining a guarantee means figuring out what factors the provider has to consider. These characteristics include the level of performance and the speed and responsiveness of user task execution.

Cloud providers need to be transparent about their service offerings and performance metrics to demonstrate that they fulfil their guaranteed attributes. They can provide detailed documentation, service level agreements (SLAs), and performance reports to substantiate their claims.

The validation of the services provided by the cloud provider is a shared responsibility between the cloud provider and the customer. which is known as the shared responsibility model (Al Moteri, 2017).

As a security and compliance framework, the Shared Responsibility Model spells out what cloud service providers (CSPs) and customers need to do to protect all parts of the cloud environment, such as the hardware, infrastructure, endpoints, data, configurations, settings, operating system (OS), network controls, and access rights (Al Moteri, 2017). The model outlines where a cloud provider’s role and responsibility end and the customer’s begin. Regardless of whether to use IaaS, PaaS, or SaaS, the Shared Responsibility Model is part of the mix (Abery et al., 1998).

The manual selection of component services becomes more challenging to accomplish as a result of this variability. A method that offers complete transparency is required to solve this problem, particularly concerning the accessibility of computer and networking resources. The current methods for choosing an SLA can only handle users' formal requirements. It makes it very hard for them to consider non-quantifiable factors or unclear user opinions when choosing services. Many websites that use the graph user interface, for instance, can only offer service packages that the customers have selected, such as their own hardware and software resources, without naming those packages with guaranteed clarity. The difficulty lies in accurately expressing consumer preferences, which sometimes consist of nebulous beliefs, and incorporating these factors into the process of selecting services in order to ensure the selection of appropriate services (Qiqing et al., 2009).

The vagueness in user selection arises from human expression and unquantifiable features inherent in the services themselves. In light of this, we propose a service selection mechanism that allows users to define their "human opinions" for each factor in the service selection requirements, ensuring that the final service package aligns closely with their overall preferences. When considering Quality of Service (QoS) in SLA selection (Tran and Tsuji, 2008), it becomes imperative to incorporate all detailed user requirements. The final selection must surpass other alternatives, as QoS is a fundamental aspect of SLA selection, and the definition of quality is deeply tied to user preferences.

To achieve these objectives, any selection guarantee should perform three essential tasks: gather and represent information for user requirements, evaluate each available web service based on these requirements, and deliver an effective solution. Our proposed mechanism offers a practical and efficient approach to accomplishing these goals.

This paper is organized as follows:

- Section 2 presents the related work, focusing on QoS, parameters, and various trust models.
- Section 3 introduces the framework for the Cloud SLA Availability parameters.
- Section 4 elaborates on our proposed model and the calculation of SLA guarantees.
- Section 5 explains the process of fuzzification and defuzzification of the data presented.
- Finally, the paper concludes and summarizes the findings in the last section.

2. Related work

Patel et al. (Patel et al., 2009). Introduce an architecture for managing cloud Service Level Agreements (SLAs) using the Web Service Level Agreement (WSLA) specification. They employ WSLA to describe cloud SLAs, with some distinctions from earlier works on WSLA. The authors present three core WSLA services that facilitate cloud SLA automation. To enhance security measures, their approach involves incorporating trusted third parties to handle certain aspects of the process. Alhamad et al. (Alhamad et al., 2010). Outline the critical criteria that should be taken into account when formulating Service Level Agreements (SLAs) for Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). For IaaS, they include factors like boot time, scale up/down time, and response time.

Building on the work of Alhamad and Baset, Qiu et al. analyze 29 SLAs from various public cloud services, including 17 SLAs for IaaS (Qiu et al., 2013). They identify several attributes commonly mentioned in these SLAs and highlight some missing attributes that hold significant importance for the relationship between cloud providers and consumers. Notably, they observe that many SLAs lack specific provisions related to customer data, such as security, privacy, protection, and backup policies. On the other hand, every SLA looked at in their study consistently guaranteed availability as a feature.

However, Qiu et al. also note that their analysis lacks sufficient details regarding availability commitments and associated SLA penalties. This suggests that more explicit and relevant information is needed to enhance clarity and accountability in SLA agreements. With the evolving demands of network applications, the focus has shifted from prioritizing high throughput to encompassing other factors such as media quality, interactivity, and responsiveness. This evolution has led to a multidimensional definition of quality of Experience (QoE). In telecommunications networks, QoE is the degree of satisfaction or annoyance a user feels while using an application or service. Considering the user's personality and current state, the degree to which the application or service fulfills the user's expectations regarding utility and enjoyment (Brunnström et al., 2013), is a determining factor.

In his research, Baset (Baset, 2012), examines the Service Level Agreements (SLAs) of five Infrastructure as a Service (IaaS) and Platform as a Service (PaaS) providers, explicitly analyzing their compute and storage services. The study proposes a method to dissect cloud SLAs into various components, facilitating comparisons between cloud providers. This approach benefits cloud providers as it enables them to establish clear and well-defined SLAs for their services. In light of Baset's investigation framework, our focus centers on availability, and we conduct a detailed classification of the commitments made concerning availability.

3. Framework for the cloud Service Level Agreement availability guarantees

This section presents an availability guarantee framework designed to assess cloud Service Level Agreements (SLAs). This framework comprises two fundamental elements: the computation of availability for computing and networking services.

3.1. Availability computing calculation

The following criteria are commonly used to classify and rank availability (Baset, 2012). In practical scenarios, cloud availability calculation necessitates consideration of additional elements, such as:

$$Availability = \frac{ServiceTime - DownTime}{ServiceTime} \tag{1}$$

The Availability Commitment is the extent to which cloud providers guarantee the availability of their services. It is important to note that reliability is either similar to or a superset of service availability (Bauer and Adams, 2012). However, among all the surveyed SLAs, providers typically choose to express their commitment regarding the availability rate (Nabi et al., 2016). It is widely acknowledged that highly available systems, particularly those used in telecommunications, are expected to meet at least 99.999% availability, commonly called 5–9s availability requirements. Table 1. illustrates the maximum allowable downtime for a system, based on the different number of 9s availability required and various operating intervals. For instance, a 5–9s system permits only 5 minutes and 15 seconds of allowable downtime during continuous operation over one year (Toeroe and Tam, 2016).

Table 1. *Maximum allowable downtime for different availability levels*

Years of continuous operations	1	2	3
Availability	Maximum allowable downtime		
99.0000% (2–9s)	3 d 15 h 36 min 0 s	7 d 7 h 12 min 0 s	10 d 22 h 48 min 0 s
99.9000% (3–9s)	8 h 45 min 15 s	17 h 31 min 12 s	1 d 2 h 16 min 48 s
99.9900% (4–9s)	52 min 34 s	1 h 45 min 7 s	2 h 37 min 41 s
99.9990% (5–9s)	5 min 15 s	10 min 31 s	15 min 46 s
99.9999% (6–9s)	32 s 1 min 3 s	1 min 3 s	1 min 35 s

3.1.1. The measurement period

The Measurement Period refers to the timeframe in which cloud providers calculate their services' availability. There are two common forms: the billing month and the calendar month. The commitment level of cloud providers can vary depending on the length of the measurement period. Suppose the measurement period is set to one year. In that case, cloud providers can perform inconsistently for a few months while maintaining stability for the rest, still fulfilling the overall availability requirement. On the other hand, a measurement period of one month necessitates that providers consistently maintain stable and available services every month (Hauer et al., 2020).

3.1.2. Accuracy in service provision

Accuracy in service provision refers to the extent of failed services that cloud providers classify as unavailable. This involves assessing various levels of service components, such as virtual machines (VMs), hosts, or entire Availability Zones. For example, Amazon EC2 adopts a more stringent approach, considering a situation unavailable only when all running instances have no external connectivity in more than one Availability Zone within the same Region. Conversely, other providers like Aliyun Cloud take a broader perspective, considering any running instances that experience downtime as unavailable, irrespective of the component affected. One can view the availability of a system through the availability of its services. Service availability can be defined as:

$$Service\ Availability = \frac{Service\ Uptime}{Service\ Uptime + Service\ Outage} \tag{2}$$

Where $Service\ Outage(DownTime) = 100\% - Service\ Uptime$

and service uptime is the duration during which the system delivers the given service, which service outage (or also referred as downtime) is the period during which the service is not delivered (Nabi et al., 2016).

3.1.3. The accuracy in Time provision

The accuracy in Time provision, refers to the unit of downtime used in the measurement period. Currently, three types of unit downtime are prevalent: 1 minute, 5 minutes, and half an hour. The way downtime is handled varies among cloud providers. Sometimes, if the downtime does not align perfectly with the time granularity, certain clouds may exclude those periods from the total service downtime calculation. On the other hand, other providers would include such periods in the calculation. For example, consider a cloud service experiencing a downtime of 7 minutes with a time granularity of 5 minutes. In this scenario, the eventual downtime is either 5 minutes or 10 minutes, depending on the specific policies adopted by the cloud provider. This difference in handling time granularity becomes more pronounced when using more extended periods, such as half an hour, and can significantly impact the availability calculation (Nabi et al., 2016), define availability as

$$Availability = \frac{MTTF}{MTTF + MTTR} \tag{3}$$

where MTTF represents the mean-time-to-failure, and MTTR denotes the mean-time-to-recovery. This measure is based on the duration when the system is either up or down, which holds significance for users. Consequently, it is unsurprising that several cloud providers, such as Microsoft's Office 365 (Toeroe and Tam, 2016), employ this measure. Uptime corresponds to the time between failures, while downtime refers to the time taken to recover from a failure.

3.1.4. Exclusions

Exclusions refer to scenarios not considered when determining whether cloud services are available. Several events are not taken into account while calculating availability. In most cases, occurrences of natural disasters, regularly scheduled maintenance, network outages that occur beyond the demarcation point of the cloud provider, and internet attacks are excluded from coverage under this policy. Because these occurrences are deemed extraordinary and transient, they are not factored into the calculation of the availability of cloud services.is done because it is possible that they do not reflect the typical service performance of the provider.

3.2. Availability networking calculation

Our research introduces standard network definitions, including those we have chosen to focus on Bandwidth (BW), delay, jitter, and packet losses. We selected these key performance indicators (KPIs) because network administrators widely use them to assess the proper functioning of their networks. These indicators provide valuable insights into network performance and help determine if the network is operating as expected.

3.2.1. Bandwidth

The Bandwidth (BW) of a channel refers to the quantity of information that can be transmitted per unit of time, typically measured in bits per second. However, the concept of BW can be interpreted differently based on the specific underlying parameter (Strauss and Kaashoek). On the one hand,it may be used as asynonym for the capacity of apath.Given an end-to-end path consisting of a series of n ordered link i=1,...,n,we define the capacity of link i as the maximum transmission rate at the IP level, $C_{i.}$

Thus, the capacity of the path, C^*, is the total capacity of the path and is defined as the lowest capacity among the links' network bandwidths.

$$C^* = \min_{i=1,..,n} \{C_i\} \tag{4}$$

The links i_k such that $C_{i_k} = C^*$ are called the narrow links of the path. We note that more than one link may be the bottleneck.

On the other hand, the BW of a path may refer to the available BW of a path, which is the unused capacity of a channel in a specific moment in time. This metric is complementary to the current used BW given a utilization factor:

$u_i^t \in [0,1]$

the available BW at time t of link i is:

$$A_t^* = \min_{i=1,..,n} C_i(1 - u_i^t\ ) \tag{5}$$

Thus the available BW of a path depends on C^* , the amount of traffic passing through it, and the number of competing clients—which is particularly notorious in wireless scenarios. The link ik such that $A_{i_k} = A^*$ is called the tight link. This instantaneous measure is usually reported as averaged over a time interval [t, t + τ]:

$$\overline{A^*}(t, t+\tau) = \min_{i=1,..,n} C_i\big(1 - \overline{u_i}(t, t+\tau)\big) \tag{6}$$

The bulk transfer capacity (BTC) refers to the upper limit of data transmission per unit of time achievable by a congestion management method, such as TCP, when implemented within a protocol. The statistic in question is influenced by various elements (Ramos et al., 2011), including the quantity of concurrent TCP sessions and conflicting traffic from the User Datagram Protocol (UDP), among other variables. In order to conduct measurements of body weight (BW), two approaches can be employed: an active method or a passive approach. The efficacy of active techniques is influenced by the choice of transport protocol, resulting in potential variations in the reported parameters of measurements. For instance, the utilization of the packet train technique (Ramos et al., 2011), which employs UDP, enables precise determination of the path's capacity C*. Conversely, estimations of the BTC can be obtained by measurements conducted with TCP traffic. Passive techniques are dependent on the monitoring of bandwidth utilization by applications or hosts, thereby accounting for the number of transmitted bytes within a specific time frame. Absolute thresholds are not that helpful, but when the client detects bandwidth is low (< 100 Kbps) audio quality can easily be impacted by other applications or network congestion. To gain a deeper insight into bandwidth and its associated availability, please consult the extended information presented in Table 2.

Table 2. *Bandwidth availability*

Bandwidth (BW)	QOS network availability
BW <500 Mbps	[90% ,92%]
500 Mbps <= BW <1Gbps	[93%, 95%]
1Gbps Mbps <= BW =<2.5Gbps.	[96%, 98%]
BW >2.5Gbps	[99.999]

3.2.2. Round trip time (Delay)

Delay is the amount of time that it takes a byte to travel the distance from the moment it left a host until it reaches its destination. Delay may refer either to one-way delay (OWD) (Almes et al., 2016), or to round-trip time (RTT)—the sum of both OWDs between a pair of hosts.

OWD estimation is challenging, as it requires very accurate and synchronized clocks at both ends of the connection. Therefore, RTT measurements are usually preferred, as they bypass this problem—we note that in this case, the involved times can be gathered from the same clock. Assuming that both latencies are of the same order (symmetric paths), the OWD can be estimated as half the RTT between a request and a reply, as suggested in protocols such as Q4S (Aranda et al., 2020).

A common way of estimating the RTT of a path is by actively measuring it using mechanisms such as ping (Almes et al., 1999). Measuring the RTT passively is harder, as it requires knowledge of the upper-layer protocols on top of TCP to correctly pair requests with replies. However, Nagle's algorithm or the delayed acknowledgment (ACK) implemented in TCP make this task difficult, as replies may have additional delays, which leads to overestimations. Despite these limitations, we can still measure the RTT by looking at the timestamps of the packets sent during the three-way handshake of a TCP session. As depicted in Figure 1, we can estimate the RTT by subtracting the times between the segments with the TCP synchronize (SYN) and ACK control flags set:

$$RTT = t_{ACK} \ - \ t_{SYN} \tag{7}$$

or between the SYN, ACK and the client SYN:

$$RTT = t_{SYN,ACK} \ - \ t_{SYN} \tag{8}$$

Depending on the point in the path our network probe is measuring, we can chose between one or the other. If we are closer to the server, we should use Equation (7). If we are closer to the client, we should use Equation (8) instead.

If we are closer to the client, we should use Equation (8) instead.

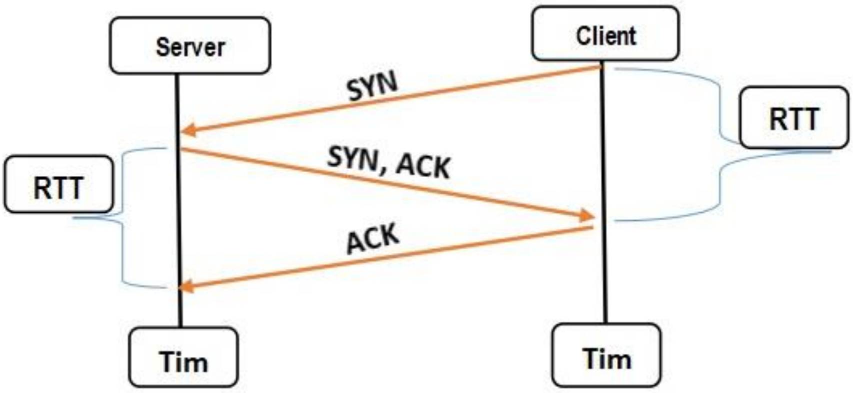


Figure 1. *Measuring round-trip time (RTT) in a three-way handshake of the Transmission Control Protocol (TCP).*

To gain a deeper insight into Avg. Round Trip Time packets and their associated availability, please consult the extended information presented in Table 3.

Table 3. *Round Trip Time availability*

Round Trip Time (RTT)	QOS network availability
RTT > 500 ms	[90% ,91%]
200 < RTT <=500 ms	[92% ,93%]
150 < RTT<=200 ms	[94% ,95%]
100 < RTT<=150 ms	[96% ,97%]
RTT<=100 ms	[98%, 99.999]

3.2.3. Variability of latency or Jitter

Jitter is a critical performance indicator because of the impact it exerts on the quality of multimedia applications such as Voice over IP (VoIP). As this KPI provides an estimation of how network latency varies, there are several definitions depending on how this variability is measured. The definition we have used in our experiments follows that in (Karmakar et al., 2017) and equivalently in (Toeroe and Tam, 2016). Given n + 1 OWD measurements,$\{l_i\}_{i=0}^{n}$, we compute the n pairwise differences $\{\Delta_j\}_{j=1}^{n}$:

$$\Delta_j = /lj - lj - 1/ \qquad (9)$$

and define jitter using a statistic of the centrality of the $\{\Delta_j\}$ such as the mean or the median.

Other definitions of jitter are given as the result of an exponential filter (Karmakar et al., 2017), of the $\{\Delta_j\}$ with parameter 1/16, or by computing the standard deviation of the $\{l_j\}$.

For a comprehensive exploration of the average jitter for packets and the availability associated with them, please refer to the additional details provided in Table 4.

Table 4. *Jitter availability*

Jitter	QOS network avialability
1<= Jitter <=15	[99.999]
15< Jitter <=20	[97% ,98%]
20< Jitter <=25	[95% ,96%]
25< Jitter <=30	[94% ,93%]
30< Jitter <=40	[92%,91%]
40< Jitter <=45	[90%]

3.2.4. Packet loss

Packet loss indicates network saturation, occurring when both routers and hosts receive packet rates beyond their processing capacities, leading to dropped packets. Additionally, bit errors can result in packet loss from hardware errors or random noise, which is particularly common in wireless communications.

To estimate packet loss with UDP traffic, measurement protocols like Q4S or IPPM (Klir and Yuan, 1996), utilize sequence numbers, similar to the approach used by TCP for its reliable transfer capability. Packet loss is the ratio of non-received packets to the total expected number. Packet loss occurs when the number of packets not received compares to the total anticipated amount. To gain an in-depth understanding of the average packet loss rate for containers and their corresponding availability, please consult Table 5, which contains further information on this topic.

Table 5. *Packet loss availability*

Packet loss	QOS network avialability
Packet loss <=1	[99.999]
1 < Packet loss <=2	[98%]
2 < Packet loss <=3	[97%]
3 < Packet loss <=4	[96%]
4 < Packet loss <=5	[95%]
5 < Packet loss <=7	[93%, 94%]
7< Packet loss <=10	[91%,92%]
10< Packet loss <30	[90%]

4. Proposed model

In the context provided, cloud services have become popular in distributed technology because they allow users to rent computing, network, and storage resources without heavy investments in integrating and managing IT infrastructure. Users only pay for the services they utilize, which eliminates the need for extensive upfront costs.

Establishing trust between cloud providers and users is crucial before any interaction occurs. Service Level Agreements (SLAs) are significant in this trust-building process. SLAs encompass various dimensions, including computing and networking availability elements, as well as linguistic terms that characterize each aspect, for example, the up/downtime, which represents the availability of QOS computing or network latency, and the up/download packets, which represent the availability of QOS in networking.

To address SLA guarantee in this context, an intelligent fuzzy theory-based SLA guarantee model is discussed. This model calculates the SLA guarantee value for each cloud service provider by considering specific computing parameters, such as uptime and downtime, and networking parameters, like bandwidth, round-trip delay, latency, and packet loss.

Fuzzy logic is applied to these parameters within the model. By giving membership values to different linguistic terms or fuzzy sets, fuzzy logic makes it possible to represent and change data that is not exact or sure. By applying vague logic criteria to the computing and networking parameters, the model calculates fuzzy values and then, by fuzzy inference, produces an ambiguous result. By the last process of FIS defuzzification, the impact of getting a crisp output contributes to the overall SLA guarantee value.

Figure 2 illustrates the system components involved in this model. The SLA manager acts as the coordinator for each computing and networking service, overseeing their performance. The Quality of Service (QOS) for computing and networking is an input parameter for the fuzzy logic system, influencing the SLA guarantee value calculation.

These parameters are proposed to be consistent with the output of our system. as illustrated in table number 6.

Table 6. *The contents of the guarantee*

the guarantee	The criteria			
	Daily	Weekly	Monthly	Yearly
90	2 hours, 24 minutes	16 hours, 48 minutes	3 days, 26 minutes, 55 seconds.	36 days, 5 hours, 22 minutes, 55 seconds
90.9999	2 hours, 9 minutes, 36 seconds	15 hours, 7minutes, 13 seconds	2 days, 17 hours, 12 minutes, 16 seconds.	32 days, 14 hours, 27 minutes, 9 seconds
91.9998	1hours, 55 minutes, 12 seconds	13 hours, 26 minutes, 25 seconds	2 days, 9 hours, 57 minutes, 37 seconds.	28 days, 23 hours, 31 minutes, 23 seconds
92.9997	1hours, 40 minutes, 48 seconds	11 hours, 45 minutes, 38 seconds	2 days, 2 hours, 42 minutes, 58 seconds.	25 days, 8 hours, 35 minutes, 37 seconds
93.9996	1hours, 26 minutes, 24 seconds	10 hours, 4 minutes, 50 seconds	1 days, 19 hours, 28 minutes, 19 seconds.	21 days, 17 hours, 39 minutes, 50 seconds
94.9995	1hours, 12 minutes, 0.43 seconds	8 hours, 24 minutes, 3 seconds	1 days, 12 hours, 13 minutes,	18 days, 2 hours, 44 minutes,

			40 seconds.	4.1 seconds
95.9994	57 minutes, 37 seconds	6 hours, 43 minutes, 16 seconds	1 day, 5 hours, 13 minutes, 26 seconds.	14 days, 14 hours, 41 minutes, 5 seconds
96.9993	43 minutes, 13 seconds	5 hours, 2minutes, 28 seconds	21 hours, 55 minutes, 11 seconds.	10 days, 23 hours, 2 minutes, 8 seconds
97.9992	28 minutes, 49 seconds	3 hours, 21 minutes, 41 seconds	14 hours, 36 minutes, 56 seconds.	7 days, 7 hours, 23 minutes, 10 seconds
98.9991	14 minutes, 25 seconds	1 hours, 40 minutes, 53 seconds	7 hours, 18 minutes, 41 seconds.	3 days, 15 hours, 44 minutes, 13 seconds
99.999	0.86 seconds	6 seconds	26 seconds.	5 minutes, 13 seconds

Table 7 present the universe of discourse for the inputs, which defines the range or domain of values that the QOS parameters can take. The table lists the possible values or ranges for the QOS (computing and networking) parameters. This lets the fuzzy logic system look at them and figure out what the fuzzy values should be.

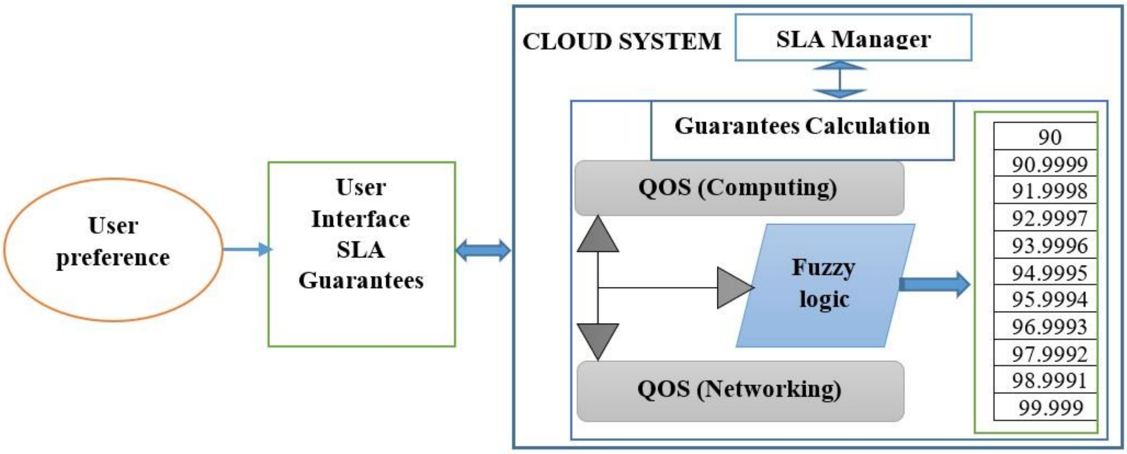


Figure 2. *Proposed SLA guarantee model.*

Table 7. *The universe of discourse for both inputs*

The universe of discourse for both (Computing and networking) inputs		
Low [89.99 90 95], Medium [90 95 99.999] and High [95 99.99 100]		
89.99	93.39966	96.89931
90	93.49965	96.9993
90.09999	93.59964	97.09929
90.19998	93.69963	97.19928
90.29997	93.79962	97.29927
90.39996	93.89961	97.39926
90.49995	93.9996	97.49925
90.59994	94.09959	97.59924
90.69993	94.19958	97.69923
90.79992	94.29957	97.79922
90.89991	94.39956	97.89921
90.9999	94.49955	97.9992
91.09989	94.59954	98.09919
91.19988	94.69953	98.19918
91.29987	94.79952	98.29917
91.39986	94.89951	98.39916
91.49985	94.9995	98.49915
91.59984	95.09949	98.59914
91.69983	95.19948	98.69913
91.79982	95.29947	98.79912
91.89981	95.39946	98.89911
91.9998	95.49945	98.9991
92.09979	95.59944	99.09909
92.19978	95.69943	99.19908
92.29977	95.79942	99.29907
92.39976	95.89941	99.39906
92.49975	95.9994	99.49905
92.59974	96.09939	99.59904
92.69973	96.19938	99.69903

92.79972	96.29937	99.79902
92.89971	96.39936	99.89901
92.9997	96.49935	99.999
93.09969	96.59934	
93.19968	96.69933	
93.29967	96.79932	100

The guaranteed value is calculated using fuzzy logic theory. Guarantee can to some extent belong to a fuzzy set and set membership function is used to represent it. Let X = $\{x_0, x_1, x_2\ldots, x_n\}$, the domain set where x is the elements of the set and i =0,1,2, 3,…, n. ∀x ∈ X and X can be represented by

$$X \rightarrow \; [0,1], \mu(x) \; \in [0,1] \tag{10}$$

In a cloud computing environment, the SLA guarantee can be described as the degree or membership of fuzzy sets in X, representing different guarantee levels. Three fuzzy logic sets are utilized to express this guarantee, each representing specific degrees of assurance. The values of these parameters are determined in Section 3, and their respective results are estimated for each. After obtaining these values, fuzzy logic is applied to them. By analysing the value of G, one can easily interpret the reputation and quality of the resources offered by a specific cloud provider. These factors benefit both the provider and the cloud users.

How the user selects the SLA guarantee through the cloud system?

The selection of the SLA guarantee is based on the criteria outlined in Section 3 of this paper. Suppose the cloud system comprises K cloud resources or service level agreements (SLAs) named SLA1, SLA2, SLA3,..., SLAk, all of which meet the specified requirements. Cloud users proceed by selecting the desired SLA guarantee and essential Quality of Service (QOS) criteria through the SLA manager. This selection is then passed on to the Fuzzy logic system for further processing. The Fuzzy logic system classifies and sorts the selection based on the availability computing and networking values parameters offered by various availability resources provided by the Cloud Service Provider (CSP).

For instance, let us consider two SLA guarantees: SLAx and SLAy. SLAx may have better availability than SLAy, but it does not meet the user's specific desires. In this scenario, user A chooses SLAy as their preferred SLA guarantee; Subsequently, the SLA manager facilitates the preparation of an agreement between user A and the cloud provider, involving some negotiation and compromises. Once the agreement is finalized, the cloud provider executes the job assigned by user A based on the selected SLA guarantee.

5. Fuzzification and defuzzification

5.1. Fuzzification

Fuzzy logic involves mapping a dataset to scalar data as output. This system comprises four main components: fuzzification, inference rules, decision components, and defuzzification. Figure 3 illustrates the components of the Fuzzy logic system.

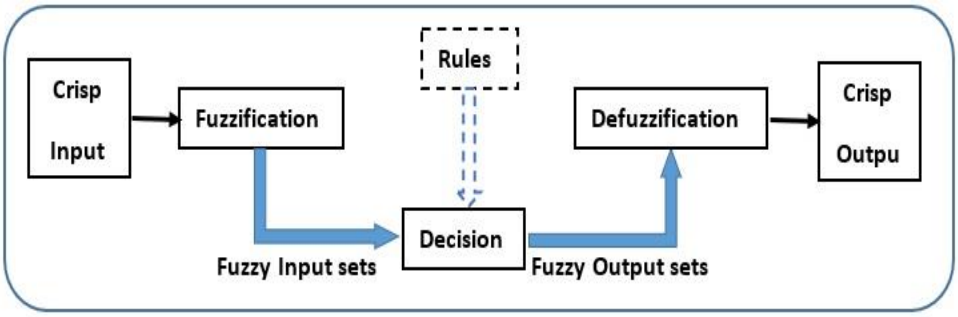


Figure 3. *Fuzzification Process*

The fuzzification system takes crisp values as input and transforms them into fuzzy logic sets using linguistic set variables, terms, and fuzzy membership functions. This process is known as fuzzification. Subsequently, fuzzy inference rules are applied to obtain the fuzzy outcome value. The final step is defuzzification, which returns the fuzzy outcome to a crisp output value (Klir and Yuan, 1996).

5.2. Fuzzy inputs

The model presented utilizes a triangular membership function, as described in equation (26).l,m, and n represent the d coordinates of the three vertices of $\mu A(d)$ to represent fuzzy sets. The membership function is defined by three vertices, denoted as l, m, and n, which correspond to the lower boundary, centre, and upper boundary of the fuzzy set A. The membership degree is zero at the lower and upper boundaries (l and n) and one at the centre (m).

Additionally, the model converts crisp input values into fuzzy sets. By applying a fuzzy logic system, the availability values for computing and networking are calculated. These fuzzy sets are then used to derive the final result, representing the SLA guarantee.

The range for all three levels (Low, Medium, and High) is defined as follows:

- For Low, l=89.99, m=90, n=95.
- For Medium, l=90, m=95, n=99.99.
- For High, l=95, m=99.99, n=100.

$$Triangled(d: l, m, n) = \begin{cases} 0, & d < l \\ d - l/m - l, & l \leq d \leq m \\ n - d/n - m, & m \leq d \leq n \\ 0, & n \leq d \end{cases} \tag{11}$$

The (x-axis or d) represents the availability of Computing and Networking as an input parameter to the fuzzy logic system. The (y-axis or μ_A (d)) represents the degree of membership function. Furthermore, the value is calculated using the triangular membership function in equation (11). For example, if the availability of Computing value is (90.09999) and Networking availability value is (90.09999). and falls in the Low range, the fuzzy output value is calculated using equation (14) (Kim and Cho, 1998), resulting is a value of (90.6).

$$Z^{*} = \frac{\int mB(z)zdz}{\int mB(z)zdz} \quad (12)$$

Where $mB(z)$is the centroid fuzzy membership function. Table 8 presents a selection of generated values within the specified range. By utilizing all that generated values, the membership function of the Computing and Networking graph is plotted in Figure 4 and Figure 5. Furthermore, Figure 6 presents the results corresponding to the availability of computing and networking. We observed that the test outcomes were displayed with a precision level to one decimal place. This occurred due to our employment of a fuzzy logic designer for constructing the model, as opposed to using direct commands. Additionally, this approach tends to yield results that are numerically closer to the next higher integer as the decimal value decreases.

Table 8. *Computing and networking availability inputs and SLA Guarantee outputs*

INPUT PARAMETERS		OUTPUT PARAMETERS
COMPUTING	NETWORKING	SLA GUARANTEES
90	90	90.3
90.09999	90.09999	90.6
90.19998	90.19998	90.9
90.29997	90.29997	91.1
95.59944	95.59944	95.3
95.69943	95.69943	95.4
95.79942	95.79942	95.5
95.89941	95.89941	95.7
97.89921	97.89921	96.8
97.9992	97.9992	96.9
98.09919	98.09919	96.9
98.19918	98.19918	97
99.79902	99.79902	98.6
99.89901	99.89901	98.8
99.999	99.999	99

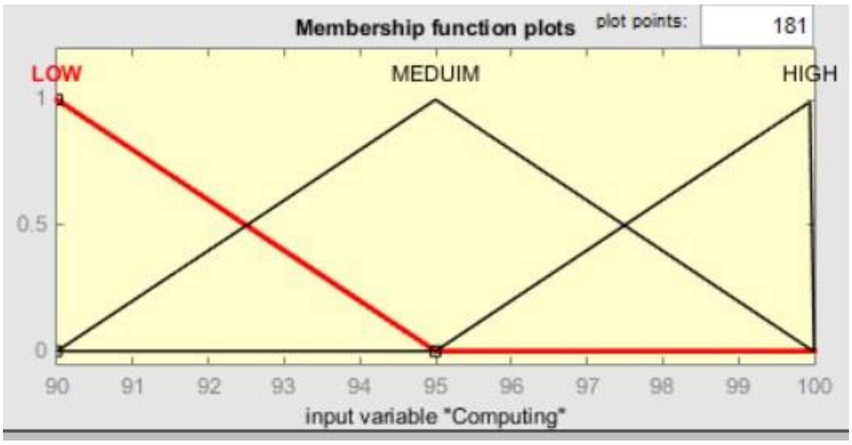


Figure 4. *Membership function for the Computing.*

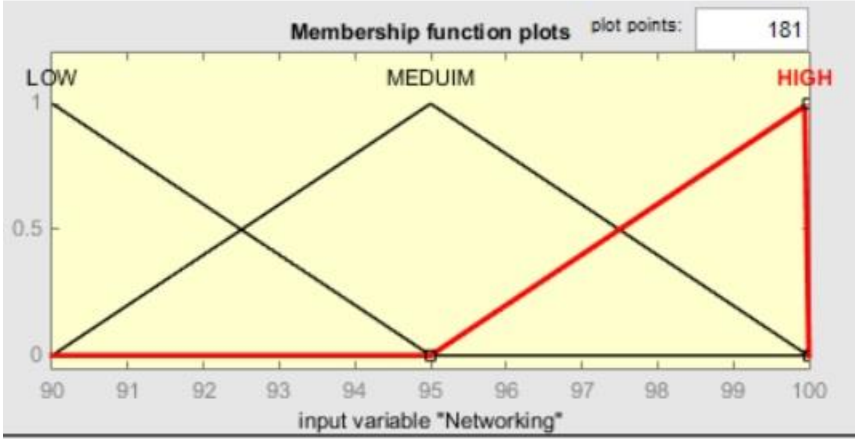
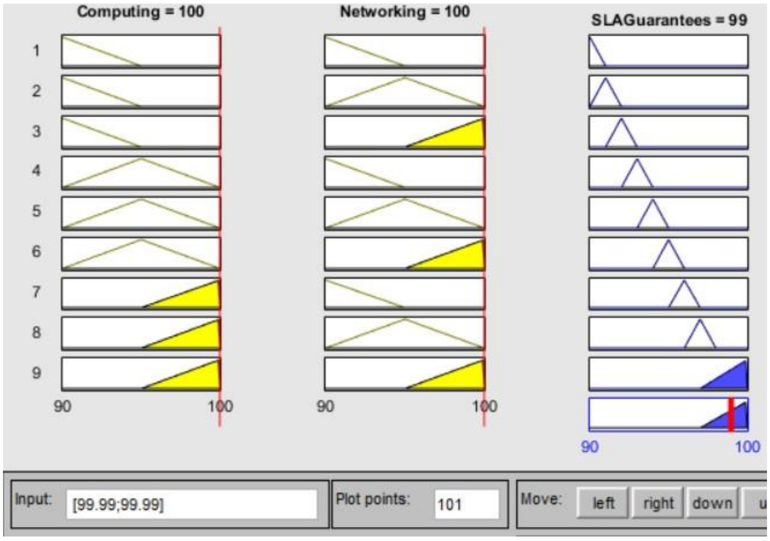


Figure 5. *Membership function for the Networking.*



Input: [99.99;99.99]

Plot points: 101

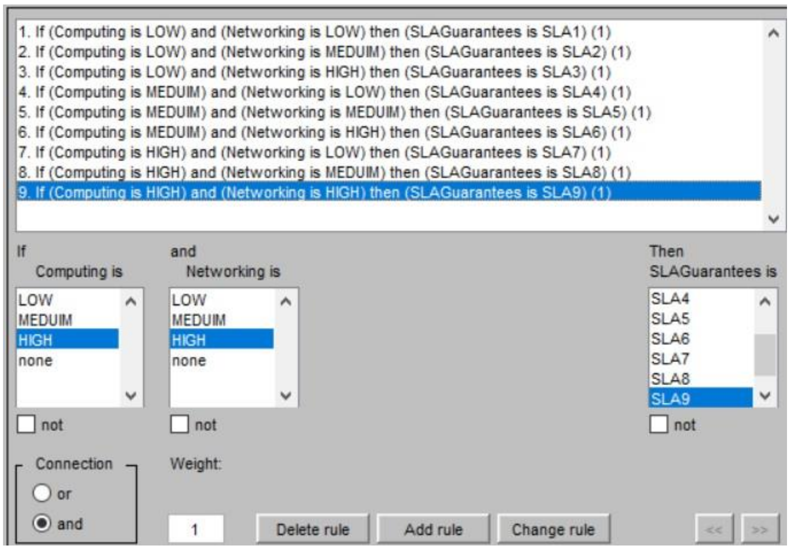
Figure 6. *The equivalent guarantee percentage for both availability computing and networking.*

5.3. Fuzzy inference rules

Inference rules serve as the procedural steps employed to transform a given input value into a fuzzified output value. This mapping strategy finds application in decision-making processes and dealing with fuzzy patterns. There are two main concepts in this context: Linguistic Fuzzy rules and If-Then-Else rules. The Linguistic Fuzzy rules utilize English words and sentences to express the values. On the other hand, If-Then-Else rules consist of two parts: the antecedents and the consequent. These parts are comprised of linguistic variable propositions.

$$R(q)\text{: IF } x \in P1 \text{ and ….and } xn \in Pn \text{ THEN } G \in qj$$

where q ranges from 1 to w representing the total number of rules. P and G denote the fuzzy values for parameters and Guarantee, respectively. Based on the rules mentioned above, the proposed model is established. Our extension model comprises 9 fuzzy rules, which can be illustrated as a 3x3 matrix.For instance, one of the inference rules is: "If Computing is low, and Networking is low, Availability is low, then the final SLA guarantee outcome is low." By applying these rules, we can generate fuzzy values for SLA Guarantees.



1. If (Computing is LOW) and (Networking is LOW) then (SLAGuarantees is SLA1) (1)
2. If (Computing is LOW) and (Networking is MEDUIM) then (SLAGuarantees is SLA2) (1)
3. If (Computing is LOW) and (Networking is HIGH) then (SLAGuarantees is SLA3) (1)
4. If (Computing is MEDUIM) and (Networking is LOW) then (SLAGuarantees is SLA4) (1)
5. If (Computing is MEDUIM) and (Networking is MEDUIM) then (SLAGuarantees is SLA5) (1)
6. If (Computing is MEDUIM) and (Networking is HIGH) then (SLAGuarantees is SLA6) (1)
7. If (Computing is HIGH) and (Networking is LOW) then (SLAGuarantees is SLA7) (1)
8. If (Computing is HIGH) and (Networking is MEDUIM) then (SLAGuarantees is SLA8) (1)
9. If (Computing is HIGH) and (Networking is HIGH) then (SLAGuarantees is SLA9) (1)

If Computing is: HIGH
and Networking is: HIGH
Then SLAGuarantees is: SLA9
Connection: ○ or ● and
Weight: 1

Figure 7. *Correlations between inputs and outputs.*

5.4. Defuzzification

Once the Fuzzification process is completed, the subsequent step involves Defuzzification to obtain precise (crisp) values using a mathematical method. In the proposed model, the widely used and popular centroid method of Defuzzification, represented by equation (12), is employed for this purpose. To visualize the SLA guarantees, the Triangular Membership Function, as depicted in Figure 8, is used for plotting. The graph is divided into nine categories, with the guarantee values represented on the X-axis. At the same time, Figure 9 depicts the illuminated surface utilized for evaluating the plot points of both input and output. The SLA categories and their respective value ranges are as follows:

- SLA1: [89.99, 90, 91]
- SLA2: [90, 91, 92]
- SLA3: [91, 92, 93]
- SLA4: [92, 93, 94]
- SLA5: [93, 94, 95]
- SLA6: [94, 95, 96]
- SLA7: [95, 96, 97]
- SLA8: [96, 97, 98]
- SLA9: [97, 99.99, 100]

For insight into the centre of gravity (COG) defuzzification process, review the steps detailed in the defuzzification sequence for ambiguous values, as applied in the optimal guarantee scenario of five nines (99.999%) SLA availability, along with other contract specifications presented in Table 9.

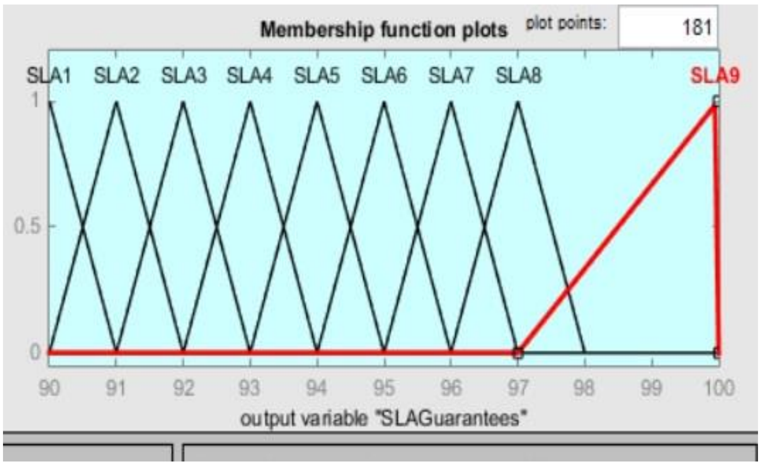

plot points: 181

Figure 8. *SLA Guarantees.*

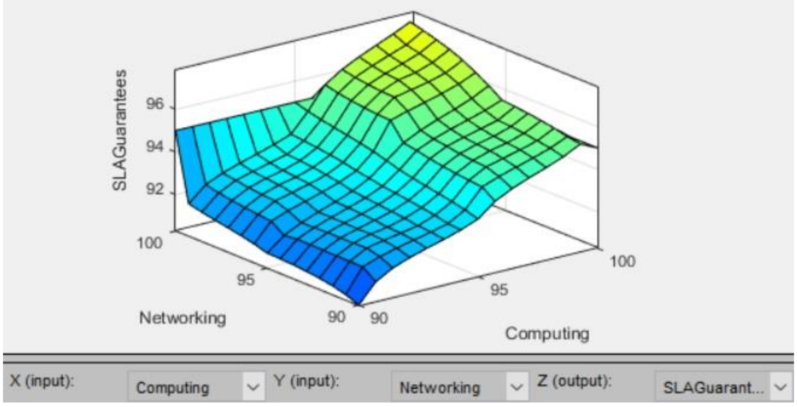

X (input): Computing
Y (input): Networking
Z (output): SLAGuarant...

Figure 9. *Surface Evaluation of the input-output working.*

5.4.1. Example of COG (SLA Five nines availability (99.999%))

- Determine the membership function's degree corresponding to the SLA's five nines availability.
- Implement the center of gravity (COG) defuzzification technique.

$$COG = \frac{\int((99.999) * 0.999950495) + ((99.999) * 0.999950495)}{\int(0.999950495 + 0.999950495)}$$

COG=99.999.

5.4.2. Fuzzy logic outputs: Assessing SLA availability guarantees

Table 9 showcases the output of our model, which dynamically segments SLA into 11 categories, from the classic or light availability SLA with a 90% guarantee to the premium, optimal availability SLA at a 99.999% guarantee. Figure 10 shows the application of the centre of gravity method to the entire spectrum of Quality of Service (QoS) for computing and network SLA availabilities.

Table 9. *Centre of gravity (COG) defuzzification process*

Availability		Degree of Membership (MF)Functions		Calculus		COG
Computing	Networking	Computing	Networking	Computing And Networking	Membership Functions	SLA guarantees
90	90	1	1	90	2	90
90.9999	90.9999	0.80002	0.80002	72.80174	1.60004	90.9999
91.9998	91.9998	0.60004	0.60004	55.20355999	1.20008	91.9998
92.9997	92.9997	0.40006	0.40006	37.20545998	0.80012	92.9997
93.9996	93.9996	0.666533333	0.666533333	62.65386669	1.333066666	93.9996
94.9995	94.9995	0.9998	0.9998	94.9805001	1.9996	94.9995
95.9994	95.9994	0.666866667	0.666866667	64.01879991	1.333733334	95.9994
96.9993	96.9993	0.399939988	0.399939988	38.79389888	0.799879976	96.9993
97.9992	97.9992	0.599979996	0.599979996	58.79755962	1.199959992	97.9992
98.9991	98.9991	0.799979996	0.799979996	79.19729962	1.599959992	98.9991
99.999	99.999	0.999950495	0.999950495	99.99404955	1.99990099	99.999

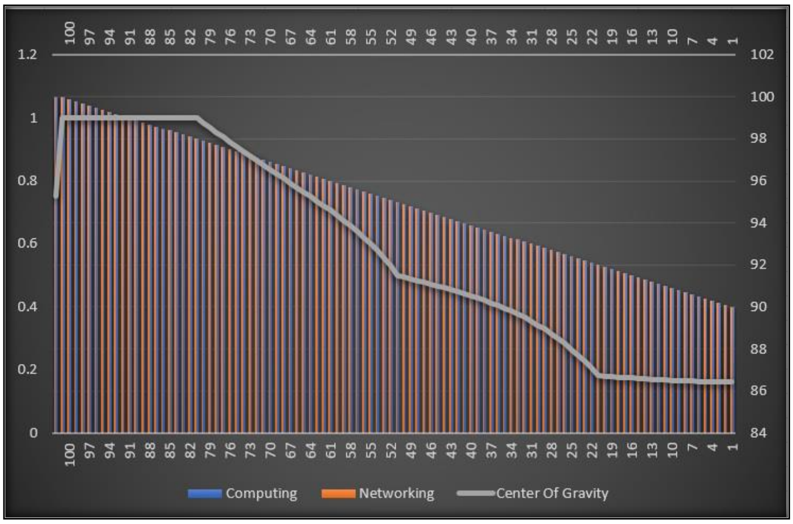


Figure 10. *Centre of gravity (COG) defuzzification process*

6. Conclusion

This study introduces an advanced method for speculative execution in modelling SLA evaluations for cloud services using fuzzy logic. It dissects SLAs into 11 distinct categories based on the level of service guarantee, ranging from a light 90% to an optimal 99.999% availability, encompassing both network (bandwidth, round trip time, jitter, and packet loss) and computing (uptime and downtime) QoS metrics.

A Mamdani fuzzy inference system is used in our new method to group different QoS availability metrics into a single SLA category, like SLA(90%). This is based on networking QoS values (like BW<500 Mbps, RTT>500 ms, jitter between 40 and 45 ms, and packet loss between 10 and 30 ms) and computing downtime (up to 36 days, 5 hours, 22 minutes, and 55 seconds). This same system is adept at discerning the remaining ten SLA levels, each with its tailored criteria.

Our empirical analysis expanded on the conventional four SLA availability values offered by cloud service providers to a more comprehensive set of 11 values. The research aimed to achieve two primary goals: (i) Design a versatile availability SLA model that applies non-standard, more nuanced techniques compared to typical CSP offerings, addressing both dominant forms of QoS availability. (ii) Enhance the accuracy of categorizing SLA guarantees to align with each user's specific needs for efficient and cost-effective task execution.

The findings demonstrate that our model significantly elevates performance over existing cloud provider models by utilizing detailed fuzzy logic to yield descriptive SLA results. It empowers users to make informed decisions in choosing the most precise and fitting SLA guarantees from cloud resource providers.

References

[1] Baliyan, N., and Kumar, S.: *Quality assessment of software as a service on cloud using fuzzy logic*, 2013 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), pp. 1-6. IEEE. **https://doi.org/10.1109/CCEM.2013.6684439**

[2] Alhamad, M., Dillon, T., and Chang, E. (2011). A trust-evaluation metric for cloud applications. *International Journal of Machine Learning and Computing*, 1(4), 416. **https://doi.org/10.7763/IJMLC.2011.V1.62**

[3] Xiaoyong, Y., Ying, L., Tong, J., Tiancheng, L., and Zhonghai, W.: *An analysis on availability commitment and penalty in cloud SLA*, 2015 IEEE 39th Annual Computer Software and Applications Conference, Vol. 2, pp. 914-919. IEEE. **https://doi.org/10.1109/COMPSAC.2015.39**

[4] Kihuya, W. B., Otieno, C., and Rimiru, R. Analysis of computer network quality of experience using Fuzzy logic model: A Survey. **https://doi.org/10.9790/1813-0804028596**

[5] Al Moteri, M. A. (2017). Decision support for shared responsibility of cloud security metrics.

[6] Abery, B., Bonner, M., Fossum, P., Koch, T., Montie, J., Nordness, K., ... and Vandercook, T. (1998). The shared responsibility framework of social interaction for collective investment: Introducing a model to enhance school improvement.

[7] Qiqing, F., Xiaoming, P., Qinghua, L., and Yahui, H.: *A global qos optimizing web services selection algorithm based on moaco for dynamic web service composition*, 2009 International forum on information technology and applications, Vol. 1, pp. 37-42. IEEE. **https://doi.org/10.1109/IFITA.2009.91**

[8] Tran, V. X., and Tsuji, H.: *QoS based ranking for web services: Fuzzy approaches*, 2008 4th International Conference on Next Generation Web Services Practices, pp. 77-82. IEEE. **https://doi.org/10.1109/NWeSP.2008.41**

[9] Patel, P., Ranabahu, A. H., and Sheth, A. P. (2009). Service level agreement in cloud computing.

[10] Alhamad, M., Dillon, T., and Chang, E.: *Conceptual SLA framework for cloud computing*, 2010 4th IEEE International Conference on Digital Ecosystems and Technologies. pp. 606-610. IEEE. **https://doi.org/10.1109/DEST.2010.5610586**

[11] Qiu, M. M., Zhou, Y., & Wang, C.: *Systematic analysis of public cloud service level agreements and related business values*, 2013 IEEE International Conference on Services Computing, pp. 729-736. IEEE. **https://doi.org/10.1109/SCC.2013.24**

[12] Brunnström, K., Beker, S. A., De Moor, K., Dooms, A., Egger, S., Garcia, M. N., ... and Zgank, A. (2013). Qualinet white paper on definitions of quality of experience.

[13] Baset, S. A. (2012). Cloud SLAs: present and future. *ACM SIGOPS Operating Systems Review*, 46(2), 57–66. **https://doi.org/10.1145/2331576.2331586**

[14] Bauer, E., and Adams, R. (2012). *Reliability and availability of cloud computing*. John Wiley & Sons. **https://doi.org/10.1002/9781118393994**

[15] Nabi, M., Toeroe, M., and Khendek, F. (2016). Availability in the cloud: State of the art. *Journal of Network and Computer Applications*, 60, 54–67. **https://doi.org/10.1016/j.jnca.2015.11.014**

[16] Toeroe, M., and Tam, F. (Eds.). (2012). *Service availability: principles and practice*. John Wiley & Sons. **https://doi.org/10.1002/9781119941378**

[17] Hauer, T., Hoffmann, P., Lunney, J., Ardelean, D., and Diwan, A.: *Meaningful availability*, 2020 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20), pp. 545-557.

[18] Murray, K. (2011). *Microsoft Office 365: Connect and collaborate virtually anywhere, anytime*. Microsoft Press.

[19] Strauss, J., and Kaashoek, M. F.: Estimating bulk transfer capacity.

[20] Ramos, J., del Río, P. S., Aracil, J., and de Vergara, J. L. (2011). On the effect of concurrent applications in bandwidth measurement speedometers. *Computer Networks*, 55(6), 1435–1453. **https://doi.org/10.1016/j.comnet.2010.10.022**

[21] Almes, G., Kalidindi, S., Zekauskas, M., and Morton, A. (2016). A one-way delay metric for IP performance metrics (IPPM), No. rfc7679. **https://doi.org/10.1016/j.comnet.2010.10.022**

[22] Aranda, J. J., Cortés, M., Salvachúa, J., Narganes, M., and Martínez-Sarriegui, I. (2020). RFC 8802 The Quality for service (Q4S) protocol. **https://doi.org/10.17487/RFC8802**

[23] Almes, G., Zekauskas, M. J., Kalidindi, S. (1999). A Round-trip Delay Metric for IPPM. 1–20. **https://doi.org/10.17487/rfc2681**

[24] Karmakar, R., Chattopadhyay, S., & Chakraborty, S. (2017). Impact of IEEE 802.11 n/ac PHY/MAC high throughput enhancements on transport and application protocols. - A survey. IEEE. *Communications Surveys & Tutorials*, 19(4), 2050–2091. **https://doi.org/10.1109/COMST.2017.2745052**

[25] Klir, G. J., and Yuan, B. (1996). Fuzzy sets and fuzzy logic: theory and applications. Possibility Theory versus Probab. *Theory*, 32(2), 207–208. **https://doi.org/10.5860/CHOICE.33-2786**

[26] Pedrycz, W. (1994). Why triangular membership functions? *Fuzzy sets and Systems*, 64(1), 21–30. **https://doi.org/10.1016/0165-0114(94)90003-5**

[27] Kim, D., and Cho, I. H. (1998). An optimal COG defuzzification method for a Fuzzy Logic Controller. *Soft Computing in Engineering Design and Manufacturing*, 401–409. Springer London. **https://doi.org/10.1007/978-1-4471-0427-8_44**