

RESEARCH ARTICLE

Uncertain Big QoS Data-Driven Efficient SaaS Decision-Making Method

LONGCHANG ZHANG^{1,2}, (Member, IEEE), AND JING BAI³

¹School of Information Engineering, Suqian University, Suqian, Jiangsu 223800, China

²Shenzhen Research Institute, Beijing University of Posts and Telecommunications, Shenzhen, Guangdong 518000, China

³School of Management Science and Engineering, Dongbei University of Finance and Economics, Dalian, Liaoning 116025, China

Corresponding author: Longchang Zhang (zlc_041018@163.com)

This work was supported by the National Social Science Fund of China under Grant 19BTQ028.

ABSTRACT Selecting the QoS-optimized software-as-a-service (SaaS) from a large number of services with the same functionality and different Quality of Service (QoS) is still a hot issue. Massive QoS feedback forms big QoS data, which exhibits ambiguity and randomness increasing the uncertainty of service selection. Starting from the characterization of big QoS data, the Uncertain Big QoS data-driven Efficient SaaS Decision Making Method (UBQoS_ESDM) is proposed. The method firstly utilizes cloud model to portray QoS in order to solve the problem of inaccurate description of uncertain big QoS data; then draws on the idea of Skyline query to establish uncertain service Skyline set, which reduces the search space and improves the efficiency of QoS-optimal SaaS selection; and draws on the Technique for Order Preference by Similarity to an Ideal Solution (TOPSIS) to design two decision-making algorithms to evaluate alternative SaaSs, and to obtain the QoS-optimal SaaS that reflects user requirements. In addition, two types of backward QoS cloud generators are introduced to convert big QoS data to QoS cloud models, and the QoS cloud model adaptive adjustment mechanism is introduced too, which can adapt to the dynamic changes of QoS. Finally, theoretical proofs and experiments verify the superiority and efficiency of the method.

INDEX TERMS SaaS, big QoS data, cloud model, Skyline, TOPSIS.


I. INTRODUCTION

With the rapid development of service computing and cloud computing, SaaS products have increased dramatically [1], SaaS with the same function and different quality of service (QoS) have become a massive development trend, and the optimal SaaS selection or SaaS recommendation based on QoS has become a hot issue. The QoS of a service is composed of several attributes, some attributes are stable (i.e., facing different users with the same attribute values), such as price, reputation, etc.; while some attributes are user-dependent (i.e., invocations by different users with different attribute values), such as response time, invocation success rate, etc. [2]. If SaaS returns QoS data from the user's end every time it is invoked, the data will grow into a massive amount, which is called big QoS data. In addition, big QoS data is also manifested in the diversity of data types of

attributes, which may be real numbers, interval numbers, linguistic phrases, and so on. For example, the corresponding time is mostly represented by real numbers (e.g., 400 milliseconds), availability is mostly described using language phrases (very good, good, fair, poor, etc.), and price may be described using interval numbers (e.g., the price of the monthly service fee is between \$10 and \$20). In the context of big QoS data and massive SaaS, the following key issues need to be addressed for QoS-based SaaS selection.

(1) Sequences of QoS data obtained from the user's end, with both connections and differences between the data, and what methods can be used to more accurately and simply characterize the big QoS data as a whole is the first problem to be solved (see Example 1).

Example 1: Table 1 shows the response time of 10 user feedbacks of the SaaS with the name WSInvokerService in the WS-DREAM dataset mentioned in the Ref. [3]. In order to portray the QoS status of the SaaS more accurately, scholars have conducted some research on the methods of describing

The associate editor coordinating the review of this manuscript and approving it for publication was Porfirio Tramontana .

QoS attributes, such as real numbers (mainly referring to the mean value) [4], fuzzy numbers [5], interval numbers [6], mean and standard deviation [9], and so on. The results of converting the data series in Table 1 to mean, interval numbers, mean and standard deviation are shown in Table 2 (fuzzy numbers are a favorable tool for dealing with linguistic phrases and are not suitable for the description of data series).

TABLE 1. WSInvokerService response time.

User IP	Response Time	User IP	Response Time
12.108.127.136	5352	128.42.6.143	5516
128.10.19.52	506	35.9.27.26	447
64.151.112.20	598	72.36.112.71	406
128.59.20.226	311	128.83.122.179	408
128.2.223.63	420	128.84.154.49	405

TABLE 2. Response time after converting.

Mean	Interval	Mean variance
1436.9	(311,5516)	(1436.9,2108.3)

Describing the data series with a real number (mean 1436.9) does not reflect the range and frequency of change in the data series; describing the data series with an interval number (311,5516) does not reflect the central tendency and frequency of change in the data series; and describing the data series with a mean and a standard deviation (1436.9,2108) is able to reflect the central tendency and the degree of deviation in the data series, but does not reflect the frequency of changes in the data series. For example, if the QoS data sequence is changed to 5452, 5416, 498, 406, 420, 606, 447, 311, 408, 405, the mean and standard deviation is (1436.9,2108), which has the same mean and standard deviation as that of the data sequences in Table 1, but the standard deviation of the latter group of QoS data sequences is more stable, and the performance is more stable compared to the former group of QoS data sequences. more stable compared to the former set of QoS data sequences. The cloud model [10] describes the data sequence through three numerical features: expectation-Ex, entropy-En, and hyperentropy-He, where expectation reflects the central tendency of the data sequence, entropy describes the range of change of the data sequence, hyperentropy describes the frequency of change of the data sequence, and the larger the data volume the more stable the cloud model. Given that the cloud model has a better ability to describe the data sequence, it is necessary to explore the feasibility of the cloud model to describe the QoS sequence (called QoS cloud model).

(2) The size of the QoS data is increasing, and it is not only impractical for all the QoS history information of a service to be considered in service selection, but it also does not reflect the current state of the service's QoS. For example, assuming in Example 1 that the 10th value is the feedback value for

the next time the service is invoked, the mean of the first 9 values is 1551.6, and the mean of the 3rd to 9th values is 442.3. If these two means are used to select the optimal SaaS, it is clear that the latter is closer to the 10th QoS value, and hence the result obtained is more plausible. The mean needs to be recalculated when the difference between the current feedback value and the previous mean is large. This problem also exists in the QoS cloud model and when to update the QoS cloud model needs to be addressed, called the QoS cloud model adaptive adjustment problem.

(3) There are always some SaaSs in a large number of SaaSs that perform poorly in various attributes of QoS relative to some SaaSs, and culling these SaaSs before service selection reduces the search space and thus improves the selection efficiency (see Example 2), which is called Service Skyline Query. This in turn gives rise to the new problem of Service Skyline Query based on QoS cloud models.

Example 2: There are four existing services with QoS attributes including average response time and call failure rate: $s_1 = (498, 0.013)$, $s_2 = (311, 0.022)$, $s_3 = (423, 0.025)$, $s_4 = (301, 0.018)$. s_4 is better than s_2 across the board, and s_2 is better than s_3 across the board. Let the user weights be $\omega = (\omega_1, \omega_2)$, where $\omega_1 + \omega_2 = 1$. After weighting, we get $s_2 = (311\omega_1, 0.022\omega_2)$, $s_3 = (423\omega_1, 0.025\omega_2)$, and there is s_2 is better than s_3 and s_4 is better than s_2 . No matter how the user weights change, there is always s_4 better than s_2 and s_3 ; therefore, we do not need to consider s_2 and s_3 in service selection, which can effectively solve the problem of large number of SaaS and improve the efficiency of SaaS selection.

(4) Selection of QoS-optimal SaaS in Skyline service set based on QoS cloud model requires designing a new algorithm.

Currently, there exists no effective solution to resolve the above problems. Therefore, this paper proposes the uncertain big QoS data-driven efficient SaaS decision-making method (UBQoS_ESDM), which relies on the historical QoS data submitted by users. UBQoS_ESDM consists of three modules (see Fig.1): (1) The QoS Model Converting (refer to section III) module first defines the QoS of the service, then describes each QoS attribute using the three numerical characteristics of the cloud model (Ex, En, He), and then uses the Backward QoS Cloud Generator to convert uncertain big QoS data sequences into QoS cloud models. Finally, the QoS Cloud Model Adaptive Computing is deployed to ascertain the adjustment timing of the QoS cloud model. (2) The Uncertain Service Skyline Computation (refer to section IV) module eliminates redundant services with poor performance in each QoS attributes among the candidate services, which restricts the number of candidate services and reduces the execution time of the service selection algorithm. (3) The SaaS Decision Algorithm Based on Cloud Model (refer to section V) module ranks the candidate services based on the QoS cloud model to obtain the QoS-optimized service.

This article contribution covers four points: ① Using cloud model to depict the QoS data, and introducing two kinds

of backward QoS cloud generator. ② Proposing an adaptive judgment method of QoS cloud model to determine whether adaptive adjustment of QoS cloud models is required. ③ Presenting the uncertain Skyline service calculation method based on QoS cloud model to reduce the SaaS search space. ④ A SaaS selection algorithm is proposed based on QoS cloud model and TOPSIS method while considering user's QoS preference.

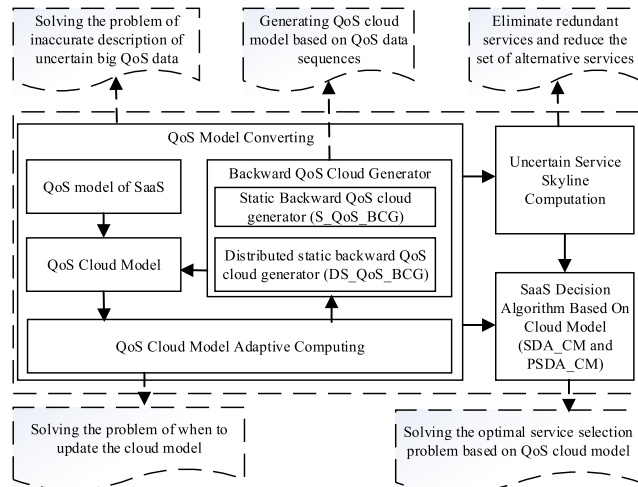


FIGURE 1. The architecture diagram of UBQoS_ESDM.

II. RELATED WORK

A. QoS MODEL

1) REAL QoS MODEL

A service QoS contains multiple attributes and the attribute values are exact real numbers. Zeng et al. [11] described in detail earlier the factors that should be considered in evaluating the quality of service, including price, duration, reputation, availability, and successful rate, and the QoS attribute was described as a real number and the calculation function for each attribute value was given. In the cloud-edge-end collaboration system, the differences in servers determine the variability in QoS values for the same candidate service. Chen et al. [12] divided the service execution time into three parts: the time required for the service request to be offloaded to the server, the time required for the server to execute the service, and the time required for the server to return the service result. The state of QoS attributes is affected by many factors (such as server environment, network environment, terminal environment, time, location, etc.). Simple QoS calculation functions cannot reflect the true state of QoS attributes. QoS data can be collected from user terminals. Since users cannot call services in all scenarios to obtain corresponding QoS data, QoS prediction becomes an optional solution, such as in [13].

2) INTERVAL NUMBER QoS MODEL

The QoS attribute values are interval numbers. Real QoS model is designed for certain QoS. For example, the response

time of a service is 506 ms. Unfortunately, QoS attribute may be uncertain during service execution, many QoS attributes prefer to be presented by intervals [6], such as the response time is in the range of [311-5516] ms in Table 1. In view of the limitations of traditional interval numbers in measuring service credibility, Ma and Hu [14] proposed a method for describing QoS using interval number of four parameters (INF). An INF is $V = [v^-, v^{e-}, v^{e+}, v^+]$. v^+ and v^- represent the upper limit and lower limit of original interval respectively; v^{e+} and v^{e-} are the upper limit and lower limit of the eigenvalue interval.

3) RANDOM QoS MODEL

The QoS attribute values are random variables. Interval numbers can reflect the uncertainty of QoS data sequences to a certain extent. Unfortunately, it is difficult to determine the probability density function of interval numbers. It is usually assumed that all intervals follow a uniform distribution or Gaussian distribution. However, these general mathematical distributions may not be consistent with the actual probability distribution of QoS intervals, resulting in inaccurate QoS descriptions. The collection and prediction of QoS attribute values can never be precise, and there were always some unobserved random effects, so it was reasonable to define QoS attribute values as random variables, as in literature [9], [15], [16].

4) FUZZY NUMBER QoS MODEL

The QoS attribute values are fuzzy sets. With the prevalent issues like vagueness, uncertainty, ambiguity, and inconsistency in the assessment data, recent works in MADM have employed fuzzy concepts like Neutrosophic fuzzy sets [7], Interval-Valued Intuitionistic Fuzzy Sets (IVIFSs) [8], Picture Fuzzy Sets [17], etc. to deal with the same.

5) HYBRID QoS MODEL

The QoS attribute value can be a precise real number, interval number, or fuzzy set. Among the QoS attributes [18], some stable attributes are suitable for depicting with precise numbers, such as price; some attributes that are greatly affected by the subjective factors of users are suitable for depicting with fuzzy sets, such as reliability and reputation; some attributes with nearly uniform distribution of attribute value sequences can be characterized by interval numbers, such as availability.

The current researches on QoS models can be classified into the five categories mentioned above. The different focuses of the models determine the differences in their ability to describe data. In describing the ability of series data, we compared the five models mentioned above with our QoS model from three dimensions: central tendency, variation range and frequency of variation. The results are shown in Table 3, where 'yes' and 'no' represent the ability to describe and the inability to describe, respectively.

TABLE 3. Comparison of data description capabilities of QoS models.

QoS Model	Central tendency	Variation range	Frequency of variation
Real QoS Model	Yes	No	No
Interval number QoS model	Yes	Yes	No
Random QoS model	Yes	Yes	No
Fuzzy number QoS model	Yes	Yes	No
Hybrid QoS model	Yes	Yes	No
Our QoS model	Yes	Yes	Yes

B. SERVICE SELECTION APPROACHES

1) THE SERVICE SELECTION METHODS BASED ON MULTI-ATTRIBUTE DECISION-MAKING

Multi-attribute decision-making (MADM) techniques are useful in helping decision-makers determine the weight of each attribute and rank the services provided by different service providers based on different selection criteria [19]. First, the influential QoS attributes of the candidate services should be identified and measured for selecting the services. Second, the QoS attributes are assigned weights based on user preferences and MADM techniques. Then, the candidate services are evaluated using MADM techniques to select the service with the best QoS. Mona et al. [5] proposed a systematical and succinct QoS-aware service selection model based on Analytical Hierarchy Process (AHP) and Simple Additive Weight (SAW), which collects QoS data from different sources including cloud providers, users' reviews, and cloud monitoring tools in order to ensure the credibility of the cloud service selection. And the utility of each QoS attribute for each service is calculated by using a SAW. Zeng et al. [11] also used the SAW method to select the optimal service, based on a real-valued QoS model. Based on Technique for Order of Preference by Similarity to ideal solution (TOPSIS), Tiwari et al. [20] proposed a novel Gaussian TOPSIS (G-TOPSIS), which is robust to rank reversal in different scenarios reported in exiting literatures. The proposed G-TOPSIS has been used to design an MCDM-based cloud service selection framework to assist cloud users to find the best cloud service. In addition to the above-mentioned common MADM methods used in service selection, there are also studies that apply MADM methods such as Elimination and Choice Expressing Reality (ELECTRE) [21], Best Worst Method (BWM) [22], Preference Ranking Organization Method for Enrichment of Evaluations (PROMETHEE) [23] to service selection. There have also been some cloud service MADM methods for interval number QoS model [14], fuzzy number QoS model [7], [8], [17], and hybrid QoS model [18].

MADM technology has been proven to be very useful in selecting services with multiple attributes and different attribute weights, especially TOPSIS, which has been successfully applied to real-time decision-making problems. However, none of the above service selection methods can

effectively handle QoS data sequences, and cannot simultaneously reflect the central tendency, range of variation, and frequency of change in QoS data sequences.

2) THE SERVICE SELECTION METHODS BASED ON RECOMMENDATION

This class of methods is based on intelligent computing techniques to find out enough services from a large range of services to form a service recommendation list that satisfies all the needs of the user. The user then selects the service from the service recommendation list instead of selecting from a large number of available services. Where Collaborative filtering (CF) is a popular approach for service recommendation to be used for predicting missing QoS values [24], the basic assumption is that if two users observed similar QoS values in the past, this means that the network conditions of the two users are likely to be similar, so the QoS values observed by the two users in the future are likely to be similar again. Traditional similarity modelling and QoS prediction methods rarely consider the impact of temporal information, which is an important factor affecting the QoS performance of Web services. Hu et al. [4] proposed an improved time-aware collaborative filtering method for high-quality Web service recommendation. The method integrates temporal information into similarity measurement and QoS prediction. The memory-based CF method fully utilizes local information but ignores global information. The model-based CF algorithm uses all QoS values (global information) in the user-service matrix to construct a global model for QoS value prediction, which compensates for the above shortcomings [24]. For example, the method in Ref [25] performs both a QoS prediction of the current time interval using a flexible matrix factorization (MF) technique and a QoS prediction of the future time interval using a time series forecasting method based on an Auto Regressive Integrated Moving Average (ARIMA) model. Dynamic relationships are frequently encountered in service computing related applications. They are modeled as high-dimensional sparse tensors containing rich knowledge of temporal patterns, and non-negative latent factor tensors (NLFT) models are employed to extract these patterns. However, inappropriate regularization schemes can lead to the occurrence of overfitting. To address this problem, NLFT models with multiple regularization schemes were investigated by Wu et al [26]. In recent years, neural network algorithms have been widely applied, and some research has applied them to service recommendation. Existing service recommendation methods fail to recommend services with reliable QoS in the MEC environment. To tackle this issue, Liu et al [27]. proposed an accurate and reliable service recommendation (ARSR) approach based on bilateral perception. First, a user's service demand is estimated by an improved online deep learning model. Then, multiple QoS attributes of candidate services are forecasted by an improved multi-task deep neural network. Finally, the optimal service is recommended to the user based on the predicted QoS.

TABLE 4. Summary of the previous literature related to service selection.

Literature	Objective	Data type	Data representation	Selection Method	DataSet Source	Existing problems
Mona et al. [5]		Qualitative QoS	Fuzzy Number QoS Model	AHP+SAW	Amazon cloud platform	
Zeng et al. [11]		Quantitative QoS	Real QoS Model	SAW	Simulation	
Tiwari et al. [20]		Quantitative QoS	Real QoS Model	TOPSIS	CloudHarmony	Cannot effectively handle QoS data sequences and cannot simultaneously reflect the central tendency, range of change and frequency of change in QoS data sequences.
Ma et al. [21]		Uncertain QoS	Real QoS Model	ELECTRE	WS-DREAM	
Youssef [22]		Quantitative QoS	Real QoS Model	TOPSIS+ BWM	Simulation	
Purohit et al. [23]	According to the user's needs and preferences, help users choose the shortlisted alternatives or a single alternative.	Quantitative QoS	Real QoS Model	PROMETHEE	QWS	
Ma et al. [14]		Uncertain QoS	Interval Number QoS Model	AHP	WS-DREAM	
Tiwari et al. [7]		Qualitative QoS	Fuzzy Number QoS Model	TOPSIS	CloudHarmony	
Gireesha et al. [8]		Qualitative QoS	Fuzzy Number QoS Model	WASPAS	Cloud Armor project (University of Adelaide)	
Obulaporam et al. [17]		Qualitative QoS	Fuzzy Number QoS Model	MARCOS	QWS	
Zhang et al. [18]		Qualitative QoS	Hybrid QoS Model	TOPSIS	Simulation	
Hu et al. [4]	The historical QoS records of the user's access to the services are processed with knowledge filtering to give the user recommendations for potentially more suitable services.	Uncertain QoS	Real QoS Model	CF	WS-DREAM	
Ngaffo et al. [25]		Uncertain QoS	Real QoS Model	MF+ ARIMA	WS-DREAM	The universality and prediction accuracy are insufficient, and cannot reflect the range and trend of changes in QoS sequences.
Wu et al. [26]		Uncertain QoS	Real QoS Model	NLFT	Simulation	
Liu et al. [27]		Uncertain QoS	Real QoS Model	DNN	Simulation	
Ding et al. [28]		Quantitative QoS	Real QoS Model	GA	Simulation	
Sadeghiram et al. [29]		Uncertain QoS	Real QoS Model	MA	QWS, WS-DREAM	
Dahan et al. [30]		Quantitative QoS	Real QoS Model	ACO	Simulation	
Mohamed et al. [31]	To find the best cloud service that minimizes or maximizes the objective function or decision criteria for the consumer without violating the constraints.	Quantitative QoS	Fuzzy Number QoS Model	SA, GA, PSO	Simulation	No optimal solution can be found, trade-offs between multiple objectives are required, and uncertain QoS sequences are understudied.
Wang et al. [32]		Quantitative QoS	Real QoS Model	ABC	QWS	
Wang et al. [33]		Quantitative QoS	Real QoS Model	PSO	QWS	
Fan et al. [9]		Uncertain QoS	Random QoS Model	Markov	Simulation	
Hosseinnezhad et al. [16]		Uncertain QoS	Random QoS Model	Bayesian	CloudArmor	
He et al. [34]		Quantitative QoS	Real QoS Model	Combinatorial Auction	QWS	
Wang et al. [35]		Uncertain QoS	Random QoS Model	Incentive Mechanism	Simulation	
Richang et al. [36]		Uncertain QoS	Random QoS Model	Portfolio Theory	Simulation	
Our Method	Help users select QoS-optimized services based on their needs and preferences.	Uncertain QoS	QoS Cloud Model	TOPSIS	WS-DREAM	The complexity of selecting composite services is high.

Although recommendation-based service selection algorithms have achieved great success, there is still a lot of room for improvement in their pervasiveness and prediction

accuracy due to the constraints of data sparsity and network environment. The prediction results of this type of algorithms are often real-valued, and cannot predict the overall trend, the

range of changes, the frequency of changes and other laws of the QoS sequence as a whole.

3) THE SERVICE SELECTION METHODS BASED ON OPTIMIZATION

In the optimization-based service selection approaches, the service selection problem is mathematically modeled. Then, the optimization models are used to solve the given model optimally. The goal of optimization-based cloud service selection is to find the best cloud service that minimizes or maximizes the objective function or decision criteria for the consumer without violating the constraints. Ding et al. [28] first introduced the transaction attributes of individual Web services and composite services, as well as transaction rules for service composition. Then, they conducted a performance analysis of basic workflow patterns and developed an algorithm to calculate the execution time of complex composite services. Finally, based on genetic algorithm (GA), they proposed a global optimal service selection algorithm that considers execution time, price, transaction attributes, stability, and penalty factors. Data-intensive web services focus on providing and updating data through extensive data manipulation and exchange. However, current service composition methods ignore the impact of data communication and service distribution. Although recent methods have revealed the usefulness of local search, they completely ignore the problem of selecting appropriate composition solutions in local search. To address this issue, Sadeghiram et al. [29] proposed a priority-based local search selection method that can be integrated consistently with any Memetic Algorithm (MA). Dahan et al. [30] proposed a more efficient neighboring selection process and multi-pheromone distribution method named enhanced flying Ant Colony Optimization (ACO) to solve QoS-aware Web service composition. The goal is to improve the computation complexity of the flying ACO algorithm by introducing three different enhancements. To select qualified cloud service providers, Mohamed et al. [31] proposed a constrained multicriteria multi-cloud provider selection mathematical model. Three metaheuristics algorithms (simulated annealing (SA), GA, and particle swarm optimization algorithm (PSO)) were implemented to solve the model, and their performance was studied and compared using a hypothetical case study. Existing population-based service selection algorithms are generally complicated to use, and often used as a general approach to solving different optimization problems. Wang et al. [32] developed specialized algorithms for QoS-aware service selection, based on the artificial bee colony algorithm (ABC). An approximate approach for the neighborhood search of ABC was developed, which enables effective local search in the discrete space of service selection in a way that is analogical to the search in a continuous space. In order to quickly find an appropriate composition of services that meet the individual user's requirements in the Internet big data, Wang et al. [33] proposed an improved particle swarm service composition method based on prior knowledge. This improved particle

swarm algorithm has a mechanism to escape from the local optima. Heuristic algorithm-based service selection requires balancing the accuracy of the optimal solution and algorithm performance, but it is also a feasible solution. Regarding the distribution characteristics of QoS sequences, some studies have adopted methods such as Markov decision models and Bayesian networks in service selection. Without considering the inherent stochastic and dynamic nature of Web service, the existing composition methods mostly generate static plans. As a result, Web service composition often terminates with failure inevitably. Fan et al. [9] presented metrical methods of several random QoS dimensions and QoS Management Architecture, and one reliable Web service composition algorithm based on Markov decision process. Hosseinneshad et al. [16] proposed a new trust model for cloud services using a Bayesian network, which is a probabilistic graphical model that can be used as one of the best methods to control uncertainty. Using Bayesian networks makes it possible to infer more accurate QoS values, which leads to the selection of highly trustworthy services by several cloud service requesters. In addition, there are also some service selection methods based on economic game perspectives, such as combinatorial auction [34], incentive mechanism [35], and portfolio theory [36].

The optimization-based service selection algorithms cannot find the theoretical optimal solution, and needs to weigh between multiple objectives. There are insufficient researches on uncertain QoS sequences, which cannot reflect the central trend, distribution, and frequency of QoS sequences.

C. SERVICE SKYLINE COMPUTING

To improve the efficiency of QoS-based service selection, Skyline query is often used to obtain a small candidate set from a large number of services. Zhang et al. [37] proposed an integrated Skyline query processing method for building Cloud Mashup applications based real QoS model. The method uses similarity tests to achieve an optimal local skyline that scales well with the number of cloud sites involved in the application. Faster skyline selection, reduced composition time, dataset sharing, and resources integration assure the QoS over multiple clouds. Shu et al. [38] proposed a novel dimension-based partition on incomplete QoS, which can quickly divide all services into partitions based on the QoS probability distribution. The computation process of skyline probability is improved by reducing the number of skyline candidates and revising the computation algorithm, which greatly improves the computation efficiency of skyline probability. Existing interval-valued skyline service selection methods assume that the probability density function (PDF) of the QoS intervals follows a general mathematical distribution, which leads to inaccurate dominant relationships between QoS intervals, and when the QoS values of certain services are missing or invalid. For this reason, Shu et al. [6] developed a new incomplete QoS-based skyline service selection method that combines probabilistic skyline queries and missing QoS predictions. A probability distribution of QoS

intervals is constructed using valid QoS values, and early termination and sorting techniques are employed to accelerate the probabilistic skyline computation. Service performance may fluctuate due to dynamic environments, so service quality is essentially uncertain. Yu and Bouguettaya [39] proposed a new concept called *p*-dominated skyline based on random QoS model to address the optimization problem of service queries for uncertain QoS information. Benouaret et al. [40] represented each QoS attribute of a Web service using a possibility distribution and introduced two skyline extensions on uncertain QoS called *pos*-dominant skyline and *nec*-dominant skyline. They then developed appropriate algorithms to efficiently compute both the *pos*-dominant skyline and *nec*-dominant skyline.

The above skyline query methods can handle real QoS model, interval number QoS model and random QoS model, which mitigate the impact of uncertain QoS on query effectiveness to some extent. However, they lack the ability to handle the fuzziness of QoS data sequences and does not support QoS cloud models. Our approach has strong processing ability for the fuzziness and randomness of QoS data sequences, and is more accurate for Skyline queries for services with uncertain QoS.

D. OUR MODEL

Compared with the existing research results, our work has the following advantages. (1) The QoS cloud model has stronger QoS data sequence description capability than existing QoS models, and can reflect the central trend, change range and change frequency of QoS sequences; the QoS cloud model can change in time with the QoS dynamics of the service, and more accurately reflect the current QoS state of the service; the distributed static backward QoS cloud generator can efficiently process large-scale QoS data sequences. (2) Uncertain Service Skyline Computing is able to process QoS data sequences described by cloud models and obtain service dominating sets more accurately than previous service skyline queries that can only process QoS data sequences described by real numbers, intervals or random numbers; The parallel Service Skyline query framework can efficiently handle large-scale service search. (3) Compared with previous service selection algorithms that can only handle QoS data sequences described by real numbers, interval numbers, and random numbers, SaaS Decision Algorithm Based on Cloud Model (SaaS Decision Algorithm Based on Cloud Model) is able to handle QoS data sequences described by the cloud model and can more accurately rank alternative services; Parallel SaaS Decision Framework is able to efficiently rank large-scale alternative services.

Fig.2 depicts the process of our method, including the following steps:

① Users who have invoked SaaS and provided feedback on QoS data are called historical users, and the QoS collected from several invocations and feedbacks are saved in the QoS database to form historical QoS data, which is used as the data basis for service selection.

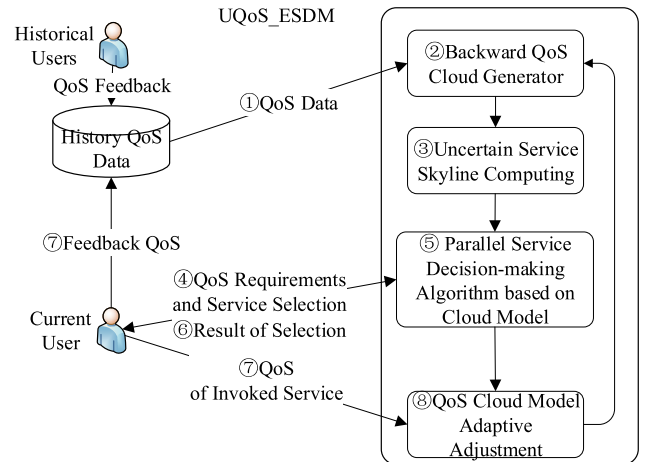


FIGURE 2. QoS-optimized SaaS selection process for UBQoS_ESDM.

- ② UBQoS_ESDM selects the most recent n feedbacks from the historical big QoS data and converts the n feedbacks into a QoS cloud model using the Backward QoS Cloud Generator. Selecting part of the historical QoS data can reduce the computation amount and reflect the current QoS state of SaaS. Generating a QoS cloud model once can be used in multiple service selections until the current QoS state of service cannot be reflected.
- ③ UBQoS_ESDM utilizes the Uncertain Service Skyline Computing to generate an alternative set of SaaSs that are not dominated from a large number of SaaSs described by the QoS cloud model, thus narrowing the search space for service selection. Generating an Uncertain Service Skyline set once can be used in multiple SaaS selections until the QoS cloud model does not accurately represent the current QoS state of the service.
- ④ The user who submits a SaaS selection request to UBQoS_ESDM is called the current user (referred to as user), and the user is required to submit a SaaS selection request with QoS requirements. There are two types of QoS requirements: SLAs describing the lower bounds of QoS attributes; and the degree of user's preference for the QoS attributes, i.e., weights. In our study, QoS requirements refer to weights (called user weights).
- ⑤ Based on the user's QoS needs and requests, UBQoS_ESDM invokes the SaaS Decision Algorithm Based on Cloud Model to select the QoS-optimal SaaS from the uncertain service Skyline set.
- ⑥ UBQoS_ESDM returns the selection result of the QoS-optimal SaaS to the user, who invokes and runs the service and monitors its QoS.
- ⑦ The user submits the QoS information of this service call to the QoS database to form historical QoS data, which serves as the data base for generating the QoS cloud model; at the same time, it passes the QoS information back to UBQoS_ESDM.

- ⑧ UBQoS_ESDM invokes the QoS cloud model adaptation calculation and adjusts the QoS cloud model depending on the result of the calculation.

III. QoS MODEL CONVERTING

A. QoS MODEL OF SAAS

QoS describes the non-functional attributes of SaaS. There are various definitions of QoS, but they all agree that QoS is a multivariate group consisting of several attributes, which can be defined as $QoS(s) = (q_1(s), q_2(s), \dots, q_n(s))$. These attributes may be response time, call failure rate, price, reputation, availability, reliability, etc., which can be adapted according to the actual application. There are three characteristics of QoS attributes as follows: (1) there are benefit-type attributes (denoted by O) and cost-type attributes (denoted by I); the larger the benefit-type attributes, the more favorable the evaluation results; the smaller the cost-type attributes, the more favorable the evaluation results. For example, the shorter the response time, the better the SaaS, and the response time is a cost-type attribute; the higher the availability of the SaaS, the better the SaaS, and the availability is a benefit-type. (2) There is a different scale between attributes, for example, response time is expressed in seconds, call failure rate is expressed as a percentage, and values with different units of measurement cannot be directly used for comparison. (3) Some attribute values can be obtained by implanting program methods on the user side, such as response time, call failure rate, etc.; some attribute values come from user feedback (e.g., reputation), and this type of attribute is mostly given by user ratings, which can be set up with 11 grades in the order of 10 points, 9 points, 8 points, ..., 1 point, and 0 points.

In order to solve the three problems existing in the QoS attributes, it is necessary to normalize the attribute values, the commonly used normalization methods are linear change and standard 0-1 transformation, in order to be able to connect with the form of user scoring, this paper adopts the form of 0-1 transformation, and the calculation method is shown in Eq. (1). Let the original SaaS decision matrix be $Y = (y)_{m \times n}$, the transformed decision matrix be $Z = (z)_{m \times n}$, z_{ij} be an attribute value ($i = 1, \dots, m$ and $j = 1, \dots, n$), let y_i^{max} be the maximum of the values in column j of the decision matrix, let y_i^{min} be the minimum of the values in column j of the decision matrix, and k be the multiplier of the user ratings. For example, the QoS (attributes are response time and availability) of two SaaS, $s_1 = (600, 0.90)$ and $s_2 = (400, 0.95)$. Here $k = 10$ is taken (because the user ratings set in this paper differ by a factor of 10 from the 0-1 transformed values). The QoS of the two services are transformed using equation (1) to get $s_1 = (0, 0)$ and $s_2 = (10, 10)$.

$$\begin{cases} z_{ij} = k \left(\frac{y_{ij} - y_j^{min}}{y_j^{max} - y_j^{min}} \right), & j \in O \\ z_{ij} = k \left(\frac{y_j^{max} - y_{ij}}{y_j^{max} - y_j^{min}} \right), & j \in I \end{cases} \quad (1)$$

B. QoS CLOUD MODEL

Cloud modeling is a kind of mutual transformation between qualitative concepts to quantitative representations, which mainly reflects the ambiguity and randomness of human understanding of a concept, and provides a new method for studying uncertainty artificial intelligence [10]. It has been widely applied and achieved good results in intelligent control, data mining, knowledge discovery, signal recognition, and decision analysis.

Definition 1 (Cloud): Let U be a quantitative thesis expressed in exact numerical terms, and C be a qualitative concept on U . If for an element $x(x \in U)$, there exists a random number $\mu(x) \in [0, 1]$ with stable tendency, called the degree of affiliation of x to C , then the distribution of x on the domain U is called a cloud, and each x is called a cloud drop.

A cloud is composed of a number of cloud droplets, which are one random realization of a certain qualitative concept, and a sufficient number of cloud droplets can be synthesized to reflect the overall characteristics of this qualitative concept. The overall characteristics of a certain qualitative concept can be represented by the numerical characteristics of the cloud, i.e., the cloud model.

Definition 2 (Cloud Model): A model that represents the numerical characteristics of a cloud by three parameters (Ex, En, He) is called a cloud model, denoted as $C(Ex, En, He)$. Among them, the cloud expectation Ex, which represents the expected value of the spatial distribution of the cloud droplets in the thesis space, is the point that best represents the qualitative concept (defined in the literature [10] as the value of the thesis space that corresponds to the form center $G(x = Ex, \mu = \sqrt{2}/4)$ of the area under the coverage of the cloud model); the entropy En of the cloud, which represents a measure of the uncertainty of the qualitative concept, can be used to characterize the span of the cloud, reflecting the degree of discretization of the cloud droplets. From the two numerical eigenvalues of Ex and En, the equation $\mu(x) = e^{-(x-Ex)^2/2En^2}$ of the cloud expectation curve with a normal distribution form is then determined. Hyperentropy He is an uncertainty measure of entropy that responds to the frequency of change of the cloud (defined in the literature [10] as the variance of the random distribution of affiliations corresponding to the point $M(x = Ex + \sqrt{\ln 8}En, \mu = \sqrt{2}/4)$ on the cloud model expectation curve, reflecting the degree of discrete affiliation to the cloud). Fig.3 depicts an example of a cloud model, where Ex = 0 is the mean of the sequence of values (the point that best represents all the values), En = 1 is the variance of the sequence of values (portraying the range of the distribution of the sequence of values), and He = 0.5 is the variance of the variance of the sequence of values (portraying the frequency of changes in the sequence of values).

Definition 3 (QoS Cloud Model): Each QoS attribute of SaaS is described by a cloud model called QoS cloud model. That is, the three parameters of QoS attribute q_i of service s are (Ex, En, He). Expectation Ex represents the expected

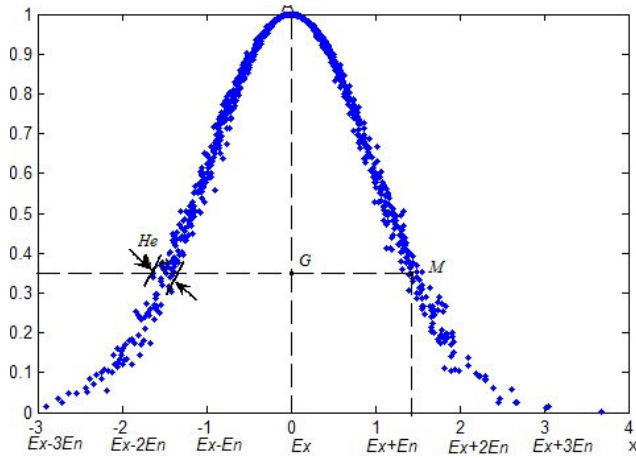


FIGURE 3. Cloud and its numerical characteristics $C(0, 1, 0.05)$.

value of the distribution of the numerical sequence of attribute q_i in the domain space and is the point that best represents the qualitative concept; entropy En describes the span of the distribution of the numerical sequence, reflecting the degree of discretization of the numerical sequence; and superentropy He is the uncertainty measure of entropy, responding to the frequency of change of the numerical sequence.

For example, the cloud model $C(3, 0.5, 0.1)$ describes the response time. 3 describes the mean of the numerical sequence of response time, which best represents the numerical sequence; 0.5 describes the span of the numerical sequence, i.e., the coverage of the numerical sequence; and 0.1 describes the frequency of changes in the span of the numerical sequence, i.e., the stability of the distribution of the numerical sequence.

C. BACKWARD QoS CLOUD GENERATOR

Backward QoS cloud generator converts history QoS numerical sequence to cloud model, then obtained the three numerical characteristics Ex, En, He of the QoS attributes. The algorithm designed based on the literature [10] is as follows.

The sequence of values 5352, 5516, 598, 406, 420, 506, 447, 311, 408, 405 in Table 1 is converted to the cloud model as (1436.9,2003.8,655.515) using S_QoS_BCG, and the altered sequence of values 5452, 5416, 498, 406, 420, 606, 447, 311, 408, 405, converted to the cloud model as (1436.9,2003.8,654.5). Both have the same expectation and entropy, but the latter has better superentropy than the former, which is more stable relative to the former.

While the real number (mean), the number of intervals, the mean and standard deviation cannot describe the frequency of change of the QoS data sequence (Table 2), relative to these methods the cloud model is able to describe the distribution characteristics (i.e., randomness) and the frequency of change (i.e., ambiguity) ability of the QoS data sequence very well.

Based on the Ref. [10], the distributed static back QoS cloud generator is designed as follows.

Algorithm 1 Static Backward QoS Cloud Generator (S_QoS_BCG)

Input: m services, n QoS attributes, QoS feedback matrix for N users is $(x)_{m \times n \times N}$

Output: Cloud model matrix is $(c)_{m \times n}$ of m services and n QoS attributes

Steps: (1) Calculate the mean $\bar{x}_{ij} = \frac{1}{N} \sum_{k=1}^N x_{ijk}$ of the sample $x_{ijk}, k = 1 \dots N$, the first order sample absolute central moment $|\bar{x}_{ij}| = \frac{1}{N} \sum_{k=1}^N |x_{ijk} - \bar{x}_{ij}|$, and the sample variance $S_{ij}^2 = \frac{1}{N-1} \sum_{k=1}^N (x_{ijk} - \bar{x}_{ij})^2$;

(2) The estimated value $\hat{E}x_{ij} = \bar{x}_{ij}$ of the expectations value c_{ij}, Ex ;

(3) The estimated value $\hat{E}n_{ij} = \sqrt{\frac{\pi}{2}} |\bar{x}_{ij}|$ of the super entropy value c_{ij}, En ;

(4) The estimated value $\hat{H}e_{ij} = \sqrt{|S_{ij}^2 - \hat{E}n_{ij}^2|}$ of the entropy value c_{ij}, He ;

Algorithm 2 Distributed Static Backward QoS Cloud Generator (DS_QoS_BCG)

Input: m services, n QoS attributes, QoS feedback matrix for N users is $(x)_{m \times n \times N}$

Output: Cloud model matrix is $(c)_{m \times n}$ of m services and n QoS attributes

Steps: (1) Assign user feedback matrix to l processors, if $l \leq m \times n$, then each processor handle $\lceil (m \times n) / l \rceil$ QoS attributes and call S_QoS_BCG; If $m \times n < l < 2 \times m \times n$, then each processor handle $\lceil (m \times n) / l \rceil$ QoS attributes and call S_QoS_BCG; If $2 \times m \times n \leq l$, then there are $k = \lfloor l / (m \times n) \rfloor$ processors to jointly computing one cloud model, performing steps (2), (3), and (4).

(2) Calculate the sample mean value $\bar{x}_{ij} = 1 / \sum_{k=1}^N x_{ijk}$, and then gain the estimate value of expectations Ex_{ij} .

(3) Divide the original samples $x_{ij1}, x_{ij2}, \dots, x_{ijN}$ into T groups, and there are r samples in each group ($N = r \times T$). Compute the sample variance within a group $y=12$, where $x=1$. According to the forward cloud generator, it can be assumed that y_1, y_2, \dots, y_T are a set of samples from $N(En_{ij}, He_{ij}^2)$.

(4) Estimate $\hat{E}n_{ij}$ and $\hat{H}e_{ij}$ from the sample $y_1^2, y_2^2, \dots, y_T^2$.

The calculating formula is $\hat{E}n_{ij} = \sqrt{\frac{1}{2} \sqrt{4 (\hat{E}Y)^2 - 2 \hat{D}Y^2}}$ and $\hat{H}e_{ij}^2 = \hat{E}Y^2 - \hat{E}n^2$, where $\hat{E}Y^2 = \sum_{i=1}^T \hat{y}_i^2 / T$ and $\hat{D}Y^2 = \sum_{i=1}^T (\hat{y}_i^2 - \hat{E}Y^2)^2 / (T - 1)$.

D. QoS CLOUD MODEL ADAPTIVE COMPUTING

Frequent SaaS invocations, which constantly generate QoS data, will greatly increase the amount of computation if the QoS cloud model is updated upon receiving QoS feedback. If the QoS cloud model is not updated for a long time, it will not reflect the current QoS status of the service, which

will cause bias in SaaS selection. In this section, we design an adaptive computation method for QoS cloud model based on Chebyshev's inequality. Chebyshev's inequality is $P(\mu - k\delta < X < \mu + k\delta) \geq 1 - 1/k^2$, which describes that the proportion of samples in any given dataset that lie within k standard deviations from the mean is always at least $1 - 1/k^2$, where k is any positive number greater than one. For $k = 2$, at least 3/4 of all samples lie within 2 standard deviations from the mean. According to the definition of the forward cloud generator [10], Ex and En in the cloud model are the mean and standard deviation, respectively, and β is the degree of confidence, and the user returns the value of a certain attribute of QoS as x after calling SaaS, and if any of the conditions in Equation 2 are met, then the QoS cloud model needs to be adjusted (note: to ensure consistency, the cloud model of all attributes of QoS need to be updated here). In the formula, m_0 is the number of original samples which falls in the range of $\{Ex - \beta En, Ex + \beta En\}$, m_1 is the number of newly increased samples which falling into this area, n_0 is the number of original samples, n_1 is the number of newly increased samples.

$$\begin{cases} (1) x \notin \{Ex - \beta En, Ex + \beta En\} \\ (2) (m_0 + m_1)/(n_0 + n_1) > 1 - 1/\beta^2 \end{cases} \quad (2)$$

Example 3: Samples 598, 406, 420, 506, 447, 311, 408, 405 describe the response time, the generated cloud model is (437.625,74.6114,39.2827), set $\beta = 5$, then according to the definition of Chebyshev's Inequality there should be 96% of the samples fall in $\{64.5680,810.6820\}$, the cloud model best represents the current QoS state of the service. The value of $x = 5352$ is produced after calling SaaS, the sample as a whole becomes worse. Using the calculation method of condition (1) in equation (2), we can get $x \notin \{64.5680, 810.6820\}$. The original cloud model can no longer represent the current data sequence, take the last 8 samples and recalculate, get the cloud model as (1031.9,1353.6,1103.5).

Example 4: From the Example 3, $m_0=8$ and $n_0=8$ can be obtained; after the increase of the sample 450, $m_1=1$ and $n_1=1$ can be obtained, satisfied the condition (2) in the formula (2), and original cloud model cannot represent current data sequence, then take the last 8 samples recalculate and the cloud model (419.1250,55.3752) can be gained.

IV. UNCERTAIN SERVICE SKYLINE COMPUTATION

In Example 2, there are dominance relationship between SaaSs based on real QoS, and the same dominance relationship between SaaSs based on the QoS cloud model, which in turn constitutes the Uncertain Service Skyline set, and the process is said to be the Uncertain Service Skyline Computation.

Definition 4 (Dominate Relation): For the N dimensional space point $\mathbf{p}(p_1, p_2 \dots, p_N)$ and $\mathbf{q}(q_1, q_2 \dots, q_N)$, if $\forall i \in [1, N]$, $(p_i \approx q_i) \vee (p_i \gg q_i)$, (\gg mean better than, \approx mean

equal), and $\exists i \in [1, N]$, $p_i \gg q_i$, said \mathbf{q} dominated by \mathbf{p} , expressed with $\mathbf{q} \ll_N \mathbf{p}$.

Definition 5 (' \approx ' is Operator of QoS Cloud): In the services s_1 and s_2 , which QoS described as cloud model, if $(s_{1j}.Ex = s_{2j}.Ex) \wedge (s_{1j}.En = s_{2j}.En) \wedge (s_{1j}.He = s_{2j}.He)$, then $s_{1j} \approx s_{2j}$.

Definition 6 (' \ll ' is Operator of QoS Cloud): If $(s_{1j}.Ex \leq s_{2j}.Ex) \wedge (s_{1j}.En \leq s_{2j}.En) \wedge (s_{1j}.He \leq s_{2j}.He)$ and exist $(s_{1j}.Ex < s_{2j}.Ex) \vee (s_{1j}.En < s_{2j}.En) \vee (s_{1j}.He < s_{2j}.He)$, then $s_{1j} \ll s_{2j}$.

Definition 7 (uncertain service domination): The n QoS attributes of the services s_1, s_2 are all described by the cloud model and are said to be uncertain services. If $\forall j \in [1, n]$, $(s_{1j} \approx s_{2j}) \vee (s_{2j} \ll s_{1j})$ and $\exists j \in [1, n]$, $s_{2j} \ll s_{1j}$ then called $s_2 \ll_n s_1$. (' \ll_n ' said the uncertain service domination).

Definition 8 (Uncertain Skyline Service Set): Assuming the whole set of uncertain service \mathcal{S} , uncertain Skyline service set $\text{sky}(n, \mathcal{S}) = \{s | (s \in \mathcal{S}) \wedge (\mathbf{r} \in \mathcal{S}), s \ll_n \mathbf{r}\}$ is on n QoS attributes. The services in the uncertain Skyline Service set are the set consisting of those services that are not dominated by any of the services in the alternative service set \mathcal{S} .

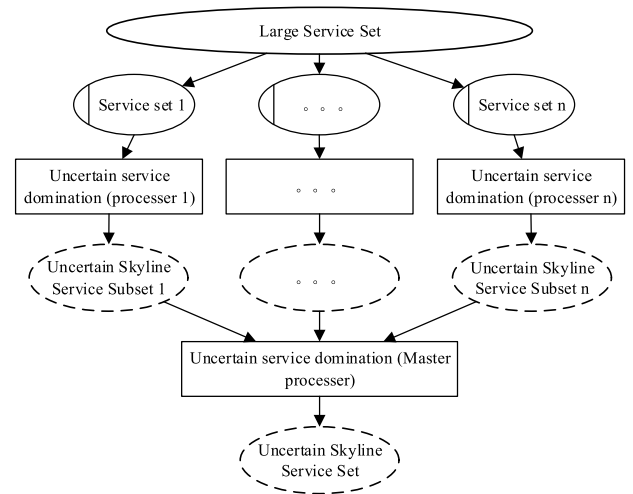


FIGURE 4. Parallel processing of uncertain service skyline queries.

Skyline query efficiency of services becomes a problem to be considered in large-scale alternative service environments. The design of parallel service Skyline query based on cloud model is shown in Fig. 4. The design idea can be applied to four common Skyline query algorithms: BNL (Block Nested Loop) algorithm; D&C (Divide-and-Conquer); NN (Nearest Neighbor); BBS (Branch and Bound Skyline).

Example 5: There are five alternative SaaSs in Table 5, QoS includes response time and throughput attributes, and attribute values are obtained from 10 user terminals, respectively, and Table 6 is generated by invoking the inverse QoS cloud generator (see S_QoS_BCG) after normalizing (see Eq. (1)) the data from Table 3. According to the definition of uncertain service domination (see Definition 7), there are $s_1 \ll_2 s_2, s_3 \ll_2 s_2$. According to the definition of

uncertain Skyline service set (see Definition 8) it is obtained that sky $(2, S) = \{s_2, s_4, s_5\}$.

TABLE 5. WSInvokerService response time.

SaaS	Response Time	Throughput
s_1	228 262 366 226 227	14.4 15.0 14.8 18.4 17.2 18.9 17.6
	251 245 247 212 207	13.8 13.8 14.9
s_2	278 266 271 217 233	18.7 18.2 18.9 19.2 19.4 18.3 15.2
	212 227 290 289 268	18.3 17.8 18.8
s_3	223 301 268 264 266	17.5 15.3 10.9 17.7 17.7 15.9 16.3
	205 207 195 194 295	16.2 18.9 19.3
s_4	298 283 218 228 234	13.4 14.1 18.3 17.5 17.1 16.9 18.0
	237 222 208 209 212	19.2 19.1 18.9
s_5	214 220 212 208 206	17.9 13.3 14.9 15.2 15.0 19.5 19.3
	218 263 218 225 213	20.5 20.6 13.6

TABLE 6. Alternative service QoS cloud model.

SaaS	Response Time	Throughput
s_1	(6.9128,2.0068,1.7017)	(5.1340,2.2172,0.9544)
s_2	(6.4477,1.9149,0.8100)	(7.6083,0.9406,0.7809)
s_3	(7.2209,2.6961,1.2041)	(5.8454,2.1319,1.2019)
s_4	(7.6221,1.6512,0.7404)	(6.5464,1.9381,0.7495)
s_5	(8.5058,0.7126,0.6183)	(6.2680,3.3336,1.5232)

V. SAAS DECISION ALGORITHM BASED ON CLOUD MODEL

The backward QoS cloud generator generates a SaaS decision-making matrix for a cloud model representation of QoS, set to $A(c) = (c)_{m \times n}$, where $c_{ij} = (Ex_{ij}, En_{ij}, He_{ij})$. The problem is a multi-attribute decision-making problem, and drawing on the basic idea of the TOPSIS method, SaaS decision-making algorithms based on cloud model (SDA_CM) is proposed. model, which is processed as follows (note: when $k = 10$, after normalization by Eq. (1), the elements c_{ij} in the decision matrix, $0 \leq Ex_{ij}, En_{ij}, He_{ij} \leq 10$, are in the same magnitude).

Step 1: Construct a weighted matrix.

User QoS weights reflect the user's preference for each attribute of QoS and need to be considered when evaluating SaaS. It is more difficult for users to accurately express the user's QoS preference, which can be expressed using linguistic phrases or using pairwise comparison matrix, and ultimately all of them will generate numerical user QoS weights, so let the user weight be $\omega = (\omega_1, \omega_2, \dots, \omega_n)$, $\sum_{i=1}^n \omega_i = 1$. According to the numerical eigenvalues of the QoS cloud model and the weight corresponding to attributes to compute the weighted decision matrix, then the computational formula is as follows:

$$\bar{c}_{ij} = (\omega_j Ex_{ij}, \omega_j En_{ij}, \omega_j He_{ij}) \quad (3)$$

Step 2: Determine the positive ideal solution and negative ideal solution.

In order to determine the advantages and disadvantages among the alternatives, it is necessary to define the comparison criteria, using the positive and negative ideal solutions of the weighted decision matrix as the comparison criteria. Considering the numerical characteristics of the QoS data series alone, when the values of an attribute are 10 and 0 are the maximum and minimum values respectively. If the samples are concentrated on 10 and 0, then it is the positive and negative ideal solution of an attribute. Extending the idea to SaaS with multidimensional QoS cloud model, then the positive and negative ideal solution calculation is shown in Equation (4).

$$\begin{cases} A^+ = \{(c_1^+, c_2^+, \dots, c_n^+)\} = \{(c_j^+ = (1, 0, 0))\} \\ A^- = \{(c_1^-, c_2^-, \dots, c_n^-)\} = \{(c_j^- = (0, 0, 0))\} \end{cases} \quad (4)$$

Equation (4) shows that when all samples are 10, it is the best result, i.e., $Ex = 1, En = 0, He = 0$; and when all samples are 0, it is the worst result, i.e., $Ex = 0, En = 0, He = 0$. Positive ideal solution is the best solution, which achieves the best value of each index value among the alternatives; and negative ideal solution is the worst solution, which achieves the worst of the alternatives.

Step 3: Calculate the distance.

The distance of the alternative SaaS from the positive ideal solution determines its degree of superiority, while the QoS criteria of the alternative and the criteria of the positive and negative ideal solutions are described in terms of a cloud model, so it is necessary to define the distance between the two cloud models, and two methods for calculating the distance between the cloud models are given below.

Define 9: Given vector \bar{c}_i and \bar{c}_j composed of two numerical characteristics of cloud model i and j , and the cosine angle between them is called cosine distance of cloud i and j :

$$\begin{aligned} d(i, j) &= d(\bar{c}_i, \bar{c}_j) = \frac{\bar{c}_i \cdot \bar{c}_j}{\|\bar{c}_i\| \|\bar{c}_j\|} \\ &= \frac{Ex_i Ex_j + En_i En_j + He_i He_j}{\sqrt{Ex_i^2 + En_i^2 + He_i^2} \sqrt{Ex_j^2 + En_j^2 + He_j^2}} \end{aligned} \quad (5)$$

Among them there is $\bar{c}_i = (Ex_i, En_i, He_i)$, $\bar{c}_j = (Ex_j, En_j, He_j)$, and $0 \leq d(i, j) \leq 1$. The larger the cosine distance, the closer i and j .

Define 10: Given vector \bar{c}_i and \bar{c}_j composed of two numerical characteristics of cloud model i and j , then the Euclidean distance between these two cloud models i and j :

$$\begin{aligned} d(i, j) &= d(\bar{c}_i, \bar{c}_j) \\ &= \sqrt{(Ex_i - Ex_j)^2 + (En_i - En_j)^2 + (He_i - He_j)^2} \end{aligned} \quad (6)$$

Among them there is $\bar{c}_i = (Ex_i, En_i, He_i)$, $\bar{c}_j = (Ex_j, En_j, He_j)$, and $0 \leq d(i, j) \leq 1$. the smaller the Euclidean distance, the closer i and j .

Three clouds $\bar{c}_1 = (0.1, 0.2, 0.01)$, $\bar{c}_2 = (0.3, 0.6, 0.03)$, $\bar{c}_3 = (0.1, 0.3, 0.01)$ are set up, and the distances from \bar{c}_2 and \bar{c}_3 to \bar{c}_1 are calculated using \bar{c}_1 as a reference. The

distance results calculated based on cosine distance are ((LICM method in literature [41] uses this method to calculate the distance of two cloud models)): $d(1, 2) = 1.0000$, $d(1, 3) = 0.9883$, which gives that \bar{c}_2 is closer to \bar{c}_1 than \bar{c}_3 . The distance calculated based on Euclidean distance results in $d(1, 2) = 0.4473$, $d(1, 3) = 0.1005$, which concludes that \bar{c}_3 is closer to \bar{c}_1 than \bar{c}_2 . $C1$ represents \bar{c}_1 , $C2$ represents \bar{c}_2 , and $C3$ represents \bar{c}_3 in Fig.5, and it can be intuitively seen that \bar{c}_3 is closer to \bar{c}_1 than \bar{c}_2 , which concludes that the distance of the cloud model calculated by the Euclidean distance is more reasonable, and therefore, in this paper, we use Eq.(6) to calculate the distance between cloud models.

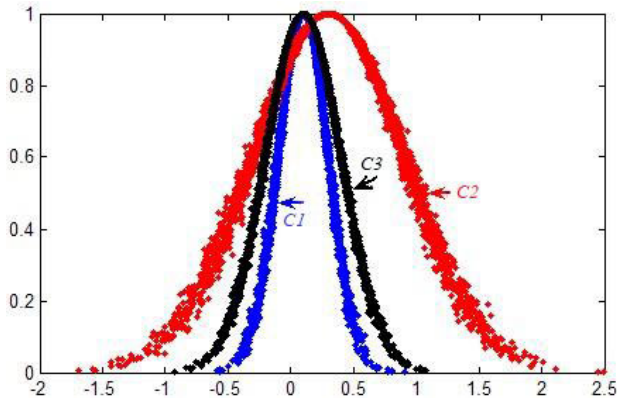


FIGURE 5. \bar{c}_1, \bar{c}_2 and \bar{c}_3 diagram.

Step 4: Calculate the distance from the alternative to the positive and negative ideal scenarios.

Based on Eq. (6), the formula for the distance from each alternative to the positive ideal solution and the negative ideal solution are given below, and the distance calculation still uses the n-dimensional Euclidean distance. Let the distance from the alternative to the positive ideal solution A^+ be $d_{s_i}^+$ and the distance to the negative ideal solution A^- be $d_{s_i}^-$, and the formulas are as follows:

$$\begin{cases} d_{s_i}^+ = \left\{ \sum_{j=1}^n d(\bar{c}_j, c_j^+)^2 \right\}^{\frac{1}{2}} \\ d_{s_i}^- = \left\{ \sum_{j=1}^n d(\bar{c}_j, c_j^-)^2 \right\}^{\frac{1}{2}} \end{cases}, i = 1, \dots, m \quad (7)$$

Step 5: Calculate the optimal degree of alternative.

The basic idea of TOPSIS is that the closer the positive ideal solution and the farther the negative ideal solution is, the better the alternative is; conversely the closer the negative ideal solution is and the farther the positive ideal solution is, the worse the alternative is. Based on the basic idea of TOPSIS, the optimal degree of alternative is given in the following formula:

$$R_i = \frac{d_{s_i}^-}{d_{s_i}^+ + d_{s_i}^-}, i = 1, \dots, m \quad (8)$$

Among them that $R_i \in [0, 1]$; When $R_i = 0$, $s_i = A^-$, the scheme is the worst scheme; When $R_i = 1$, $s_i = A^+$, the scheme is the optimal scheme. In practical multi-attribute

decision making, the likelihood of A^+ and A^- existing is small. The evaluation schemes are ranked according to the value of R_i in order from largest to smallest. The larger the value of R_i for the closeness of the ranking result, the better the scheme is, and the one with the largest value of R_i is the optimal scheme.

Taking single alternative SaaS as the basic unit, the optimal selection process of m alternative SaaS can be formed into the structure of Fig.6, which links each SaaS is the user weight and the ideal solution, which are known conditions in the SDA_CM algorithm, so that the various schemes of SDA_CM can be executed in parallel, which is called PSDA_CM.

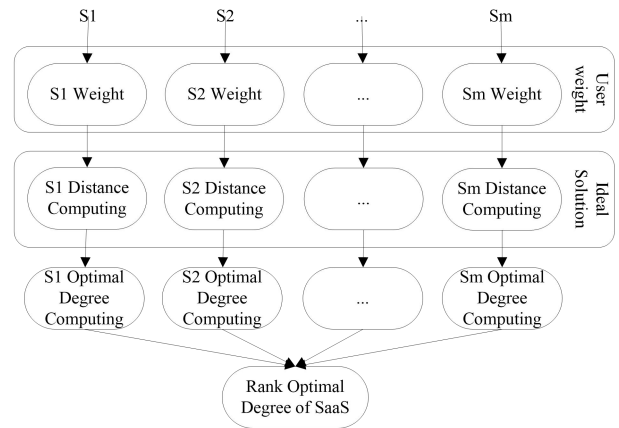


FIGURE 6. Parallel processing of SDA_CM.

VI. EXPERIMENTAL ANALYSIS

A. EXPERIMENTAL ENVIRONMENT AND DATASET

We run our experiments on a PC with Intel (R) Core (TM) i7-10750H with 2.60 GHz CPU, 16.0 GB of RAM and Windows 10 operation system. The algorithms are implemented in Java. To evaluate our method in the real world, the WS-DREAM dataset #2 [3] is used in the experiments. The dataset collects real QoS data, including response time and throughput, for 4532 services invoked over 64 time periods from 142 users around the world. A user-side light weight middleware for service users was designed to automatically record QoS values of invocations and to contribute the local records to the server for obtaining more invocation results from other service users. Compared with other QoS datasets, this dataset is derived from user calls from different locations at different times, which is more suitable for analyzing the uncertainty of QoS of services. The QoS of services in this dataset varies significantly due to fluctuations in server load and network instability. In addition, the problems and solutions presented in the paper are based on the analysis of the QoS of the services in the WS-DREAM dataset, see Section I. Here, it should be further clarified that UBQoS_ESDM does not limit the number of QoS attributes of a service, as long as the attributes are described by numerical sequences can be processed; the QoS attributes of a service are selected according

to the actual scenarios, and the number of attributes is stable and approximated to be a constant, and it does not affect the results of the evaluation of the UBQoS_ESDM method in this experiment.

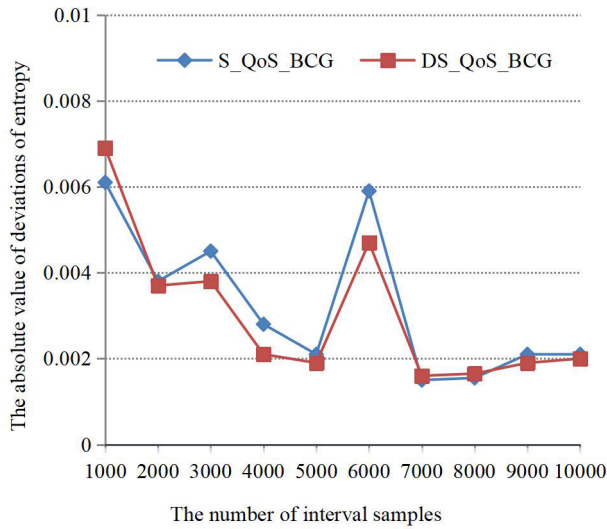


FIGURE 7. Comparison of mean deviations of entropy.

B. BACKWARD QoS CLOUD GENERATOR ANALYSIS

Take the response time of a SaaS with a cloud model $E_x = 8$, $E_n = 0.9$, $E_e = 0.3$ and run the forward cloud generator [10] 10 times, generating n cloud drops each time. Two algorithms S_QoS_BCG and DS_QoS_BCG are called to compute the numerical features \hat{E}_n and \hat{E}_e of the cloud model, both algorithms are called repeatedly for 10 times, and the average error and the average running time of the 10 calls are computed. Fig.7 and Fig.8 illustrate that the error difference between the two algorithms is not much, and DS_QoS_BCG is slightly better than S_QoS_BCG. The experiment in Fig.9 is a comparison of the time of S_QoS_BCG and DS_QoS_BCG with a fixed number of processors of 10. The experiment in Fig.10 is to derive the variation of DS_QoS_BCG’s running time with the number of processors for a sample size of 1000.

We can get the following conclusions (1) The errors of both algorithms are acceptable. (2) The running time of the two algorithms is very close when the number of samples is 1000; when the number of samples becomes smaller, the running time of S_QoS_BCG is better than that of DS_QoS_BCG; (3) The infinite increase of the number of processors cannot reduce the running time of DS_QoS_BCG, and the number of processors must be proportional to the number of samples to get the minimum running time.

C. THE ANALYSIS OF THE QoS CLOUD MODEL ADAPT COMPUTING

Our QoS cloud model adaptation computation is called adaptive QoS cloud model (AQoS_CM) here, and two other

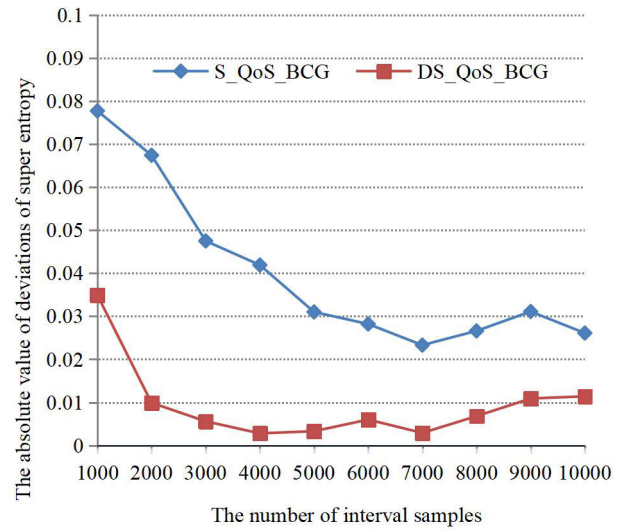


FIGURE 8. Comparison of mean deviations of super entropy.

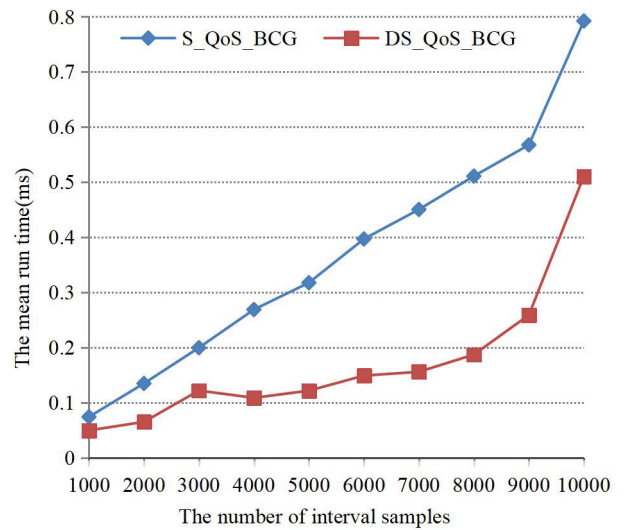


FIGURE 9. Comparison of mean of run time.

methods for generating the QoS cloud model are described below. (1) During the continuous collection of QoS feedbacks, each time the number of QoS feedbacks collected reaches n , the n QoS feedbacks closest to the current time are taken to compute the QoS cloud model, followed by the collection of the next n QoS feedbacks, which is referred to as n -interval QoS cloud model (NIQoS_CM). (2) As soon as a new QoS feedback is collected, the current nearest n QoS feedbacks are taken to compute the QoS cloud model, and the method is called real-time QoS cloud model (RTQoS_CM). Real-time QoS cloud model is the ideal goal to pursue, but its computation is too frequent to be realistic. In this experiment, we analyze the accuracy of AQoS_CM and NIQoS_CM relative to RTQoS_CM and the number of times of generating

cloud models, and we take the response time data sequence of service number 1 in QoSDataSet2 in WSDream dataset from literature [3] as the experimental data. To facilitate the processing, the samples in the dataset with failed QoS feedback (remaining 333 samples) are excluded; the values are expanded by 1000 times in milliseconds; and the degree of confidence is set to $\beta = 5$.

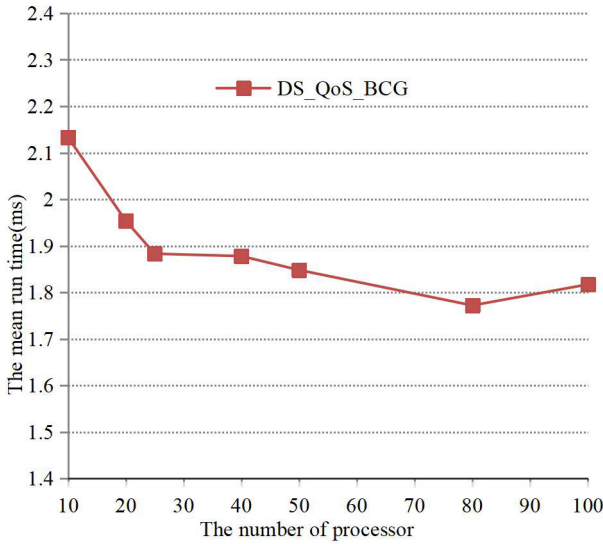


FIGURE 10. The relationship between the running time and the number of processors.

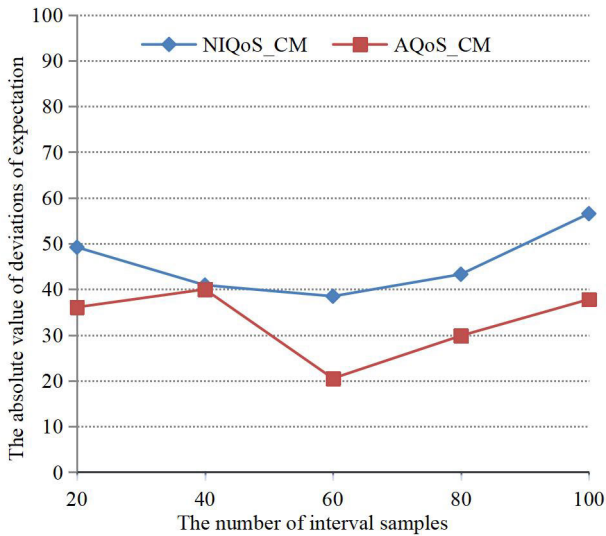


FIGURE 11. Comparison of deviations of expectation.

Fig. 11, Fig. 12, and Fig. 13 illustrate that the after introduced QoS cloud model adaptation calculated that the error of QoS cloud model produced by adaptive and the QoS cloud model generated in real time relative to the NIQoS_CM has much better precision. Fig. 14 shows that on the number of executions that AQoS_CM is lot less than the RTQoS_CM and gradually approaching NIQoS_CM; In addition, the

AQoS_CM run before the SaaS selection, does not affect the user experience.

D. SKYLINE SET CORRECTNESS PROOF

The alternative service set is computed by the Uncertain Service Skyline to obtain the uncertain service Skyline set, which can effectively reduce the SaaS search scope, and the SaaS selection algorithm can simply select the optimal SaaS in the Uncertain Service Skyline set. Therefore, it is necessary to prove the correctness of SaaS selection in the Uncertain Service Skyline set, and here it is only necessary to prove that the optimal SaaS is in the Uncertain Service Skyline set, i.e., it is necessary to prove that the dominated SaaS (i.e., the service that is not in the Uncertain Service Skyline set) must not be the optimal SaaS.

Proposition 1: Let any $s_i \in S$ and $s_i \notin \text{sky}(n, S)$, prove that s_i must not be the optimal SaaS obtained via SDA_CM.

Proof: Since any $s_i \in S$ and $s_i \notin \text{sky}(n, S)$, according to the definition of Uncertain Skyline Service Set (see Definition 8), then $s_k \in \text{sky}(n, S)$ and $s_i \ll_n s_k$ must be found. Assuming that the user weights $\omega = (\omega_1, \omega_2, \dots, \omega_n)$ and $\sum_{i=1}^n \omega_i = 1$, The computational procedure for s_i and s_k are computed by SDA_CM as follows:

(1) After weighted, we can get

$$\begin{cases} s_{ij} = (\omega_j \text{Ex}_{ij}, \omega_j \text{En}_{ij}, \omega_j \text{He}_{ij}) \\ s_{kj} = (\omega_j \text{Ex}_{kj}, \omega_j \text{En}_{kj}, \omega_j \text{He}_{kj}) \end{cases}$$

(2) Based on the calculation of positive and negative ideal solutions (see Eq. (4)), we can get

$$\begin{cases} A^+ = \{c_1^+, c_2^+, \dots, c_n^+\} = \{c_j^+ = (10, 0, 0)\} \\ A^- = \{c_1^-, c_2^-, \dots, c_n^-\} = \{c_j^- = (0, 0, 0)\} \end{cases}$$

(3) The following equations are obtained from the distance formulas (6) and (7):

$$\begin{cases} d(s_i, A^+) = \sqrt{(\text{Ex}_{i1} - 10)^2 + (\text{En}_{i1})^2 + (\text{He}_{i1})^2 + (\text{Ex}_{i2} - 10)^2 \dots} \\ d(s_i, A^-) = \sqrt{(\text{Ex}_{i1})^2 + (\text{En}_{i1})^2 + (\text{He}_{i1})^2 + (\text{Ex}_{i2})^2 \dots} \\ d(s_k, A^+) = \sqrt{(\text{Ex}_{k1} - 10)^2 + (\text{En}_{k1})^2 + (\text{He}_{k1})^2 + (\text{Ex}_{k2} - 10)^2 \dots} \\ d(s_k, A^-) = \sqrt{(\text{Ex}_{k1})^2 + (\text{En}_{k1})^2 + (\text{He}_{k1})^2 + (\text{Ex}_{k2})^2 \dots} \end{cases}$$

From $s_i \ll_n s_k$, we can get $d(s_k, A^-) > d(s_i, A^-)$, $d(s_i, A^+) > d(s_k, A^+)$ and $d(s_k, A^-) > d(s_i, A^+) > d(s_i, A^-) > d(s_k, A^+)$.

(4) Calculate the optimal degree of scheme according to the formula (8):

$$R_i = \frac{d(s_i, A^-)}{d(s_i, A^+) + d(s_i, A^-)},$$

$$R_k = \frac{d(s_k, A^-)}{d(s_k, A^+) + d(s_k, A^-)},$$

TABLE 7. The process of the three algorithms.

	SDA_R	SDA_F	SDA_CM
Matrix	$A(c) = (c)_{m \times n}$	$A(c) = (c)_{m \times n}, c_{ij} = (l_{ij}, u_{ij})$	$A(c) = (c)_{m \times n}, c_{ij} = (Ex_{ij}, En_{ij}, He_{ij})$
Step 1	$\bar{c}_{ij} = (\omega_j c_{ij})$	$\bar{c}_{ij} = (\omega_j l_{ij}, \omega_j u_{ij})$	$\bar{c}_{ij} = (\omega_j Ex_{ij}, \omega_j En_{ij}, \omega_j He_{ij})$
Step 2	$A^+ = \left\{ \left\{ c_j^+ = \max_{1 \leq i \leq m} (\bar{c}_{ij}) \right\} \right\}$ $A^- = \left\{ \left\{ c_j^- = \min_{1 \leq i \leq m} (\bar{c}_{ij}) \right\} \right\}$	$A^+ = \left\{ \left\{ c_j^+ = \left(\max_{1 \leq i \leq m} (\bar{l}_{ij}), \max_{1 \leq i \leq m} (\bar{u}_{ij}) \right) \right\} \right\}$ $A^- = \left\{ \left\{ c_j^- = \left(\min_{1 \leq i \leq m} (\bar{l}_{ij}), \min_{1 \leq i \leq m} (\bar{u}_{ij}) \right) \right\} \right\}$	$A^+ = \left\{ \left\{ c_j^+ = (1, 0, 0) \right\} \right\}$ $A^- = \left\{ \left\{ c_j^- = (0, 0, 0) \right\} \right\}$
Step 3	$d(i, j) = \sqrt{(c_i - c_j)^2}$	$d(i, j) = \sqrt{(l_i - l_j)^2 + (u_i - u_j)^2}$	$d(i, j) = \sqrt{(Ex_i - Ex_j)^2 + (En_i - En_j)^2 + (He_i - He_j)^2}$
Step 4	$d_{s_i}^+ = \left\{ \sum_{j=1}^n d(\bar{c}_{ij}, c_j^+) \right\}^{\frac{1}{2}}, i = 1, \dots, m$ $d_{s_j}^- = \left\{ \sum_{j=1}^n d(\bar{c}_{ij}, c_j^-) \right\}^{\frac{1}{2}}, i = 1, \dots, m$	$d_{s_i}^+ = \left\{ \sum_{j=1}^n d(\bar{c}_{ij}, c_j^+) \right\}^{\frac{1}{2}}, i = 1, \dots, m$ $d_{s_j}^- = \left\{ \sum_{j=1}^n d(\bar{c}_{ij}, c_j^-) \right\}^{\frac{1}{2}}, i = 1, \dots, m$	$d_{s_i}^+ = \left\{ \sum_{j=1}^n d(\bar{c}_{ij}, c_j^+) \right\}^{\frac{1}{2}}, i = 1, \dots, m$ $d_{s_j}^- = \left\{ \sum_{j=1}^n d(\bar{c}_{ij}, c_j^-) \right\}^{\frac{1}{2}}, i = 1, \dots, m$
Step 5	$R_i = \frac{d_{s_i}^-}{d_{s_i}^+ + d_{s_i}^-}, i = 1, \dots, m$	$R_i = \frac{d_{s_i}^-}{d_{s_i}^+ + d_{s_i}^-}, i = 1, \dots, m$	$R_i = \frac{d_{s_i}^-}{d_{s_i}^+ + d_{s_i}^-}, i = 1, \dots, m$

$$\frac{R_k}{R_i} = \frac{d(s_k, A^-) (d(s_i, A^+) + d(s_i, A^-))}{d(s_i, A^-) (d(s_k, A^+) + d(s_k, A^-))}$$

$$= 1 + \frac{d(s_k, A^-) d(s_i, A^+) - d(s_i, A^-) d(s_k, A^+)}{d(s_i, A^-) (d(s_k, A^+) + d(s_k, A^-))} > 1$$

Thus, the proposition is proved.

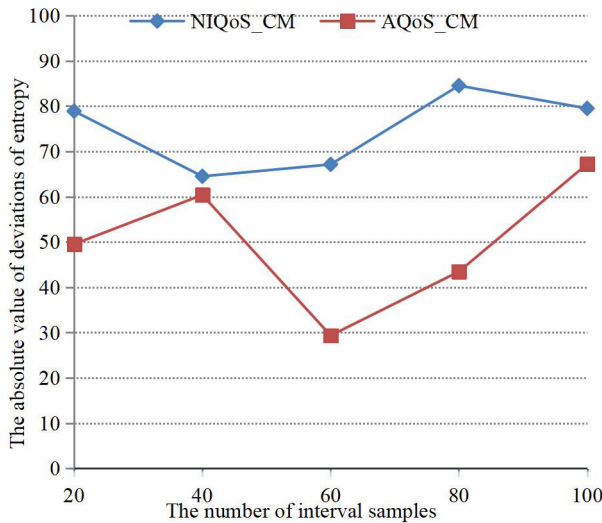


FIGURE 12. Comparison of deviations of entropy.

E. THE CORRECTNESS ANALYSIS OF THE DECISION-MAKING ALGORITHM

Existing TOPSIS based service selection algorithms can support real number QoS [20], [22], interval number QoS [18], fuzzy number QoS [7]. In the WS-DREAM dataset, the QoS of a service is a sequence of data at different times in different locations, which is easier to convert to real number QoS model (taking the mean value) and interval number

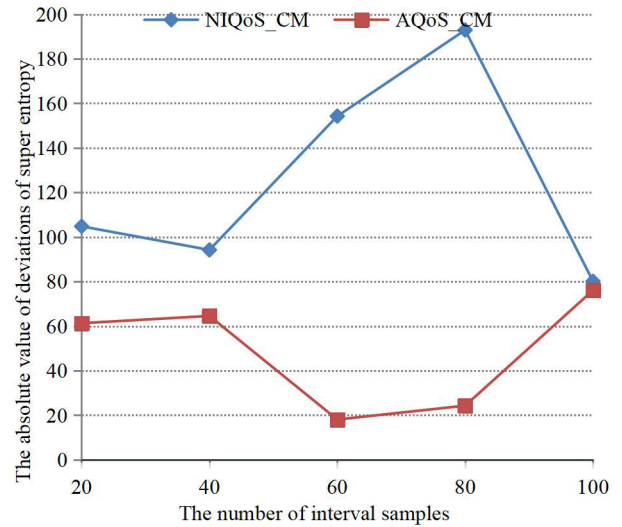


FIGURE 13. Comparison of deviations of super entropy.

QoS (taking the minimum and maximum value). Therefore, our method-SDA_CM is compared with real number-based method (SDA_R) and interval number-based method (SDA_F). After normalizing the QoS data sequences (see Eq. (1)), the decision matrices of the three algorithms are constructed, and the steps of the three algorithms for the decision matrices are as follows. Step 1: Construct the weighting matrix, Step 2: Determine the positive and negative ideal solutions, Step 3: Distance calculation, Step 4: Calculate the distance from the alternative services to the positive and negative ideal solutions; Step 5: Calculate the optimal degree of the alternative services. The process of the three algorithms is shown in Table 7.

The data examples are shown in Table 5, and after transformation, the real QoS and interval QoS are shown in Table 8, and the QoS cloud model is shown in Table 6. The sorting

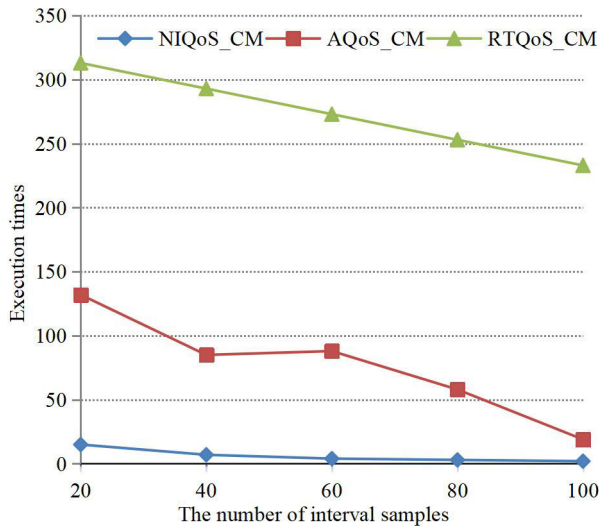


FIGURE 14. Comparison of execution times.

result of SDA_CM is $s_5 > s_2 > s_4 > s_3 > s_1$; and the sorting result of SDA_R is $s_5 > s_4 > s_2 > s_3 > s_1$; the sorting result for SDA_F is $s_5 > s_2 > s_4 > s_3 > s_1$. On this data example, the 3 algorithms sorting results differ only on s_2 and s_4 , and the reasons for the differences are analyzed below. After weighted normalization $s_2 = (0.2377, 0.2403)$ and $s_4 = (0.2188, 0.2268)$, the response time of s_2 is worse than that of s_4 , while the throughput of s_2 is better than that of s_4 , and the difference between them is very small. From Table 5, it can be seen that the stability of s_2 and s_4 is more or less the same in the response time attribute; however, in the throughput attribute, the stability of s_2 is much better than that of s_4 ; therefore, the sorting results obtained by SDA_CM are more reasonable. SDA_F obtains the same results as SDA_CM because the data distribution is more uniform in this instance, and it happens to obtain relatively good sorting results; if there are bursts of high values and low values, the accuracy of the calculation results of this method is greatly reduced. In summary, SDA_CM can obtain better sorting results because the algorithm considers the central tendency, distribution range and change frequency of the data series at the same time. SDA_R only considers the central tendency of the data series and SDA_F simply considers the range of variation of the data series.

F. PRACTICAL APPLICATION SCENARIO DATA ANALYSIS

Application Scenario: A university library has contracted with 10 book retrieval service providers to provide book retrieval SaaS for the students and faculty of the university, which are defined as s_1, s_2, \dots, s_{10} , and the QoS attributes include cost, response time, trustworthiness, availability, and reliability. There are 40,000 students and faculty members in the university, among which 10,000 students and faculty members submit their QoS scores for the 10 book

TABLE 8. Real numbers and interval numbers of data instances.

SaaS	Real number (Mean)		Interval number	
	Response Time	Throughput	Response Time	Throughput
s_1	247.1	15.88	(207,366)	(13.8,18.9)
s_2	255.1	18.28	(212,290)	(15.2,19.4)
s_3	241.8	16.57	(194,301)	(10.9,19.3)
s_4	234.9	17.25	(208,298)	(13.4,19.2)
s_5	219.7	16.98	(206,263)	(13.3,20.6)

retrieval SaaSs, and the scoring criteria are set up with 11 grades in the order of 10, 9, 8, ..., 1, and 0. Some of the scoring data are randomly selected to make an experimental dataset. Now user u makes an application for service selection, and the user weight given by this user is $\omega = (0.25, 0.25, 0.125, 0.25, 0.125)$. From Fig. 15, it is concluded that SDA_CM and SDA_R are more stable in this experimental scenario, and the optimal degree of the optimal service obtained by SDA_R and SDA_F is lower than that of SDA_CM.

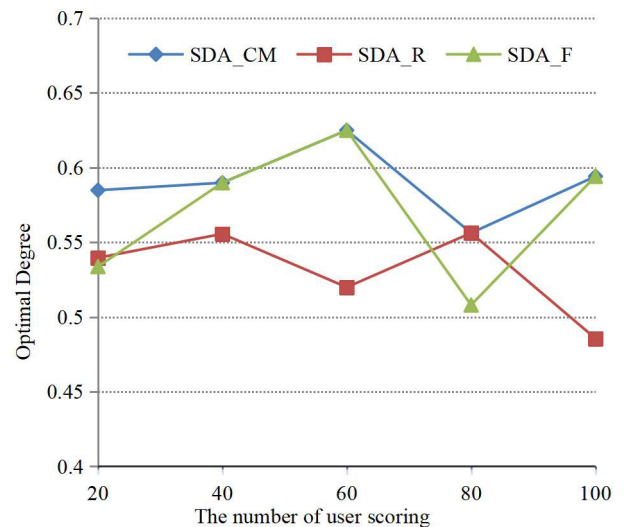


FIGURE 15. The optimal degree changes with the number of users scoring.

G. ALGORITHM PERFORMANCE ANALYSIS

In the above practical application scenario, the time complexity comparison of the three algorithms, SDA_CM, SDA_R, and SDA_F, is shown in Fig. 16, and all the three algorithms show a linear growth trend; the time complexity of SDA_CM and SDA_F is similar and lower than that of the SDA_R method; the extra time of SDA_CM is used to compute the cloud model. However, after generating the QoS cloud model from users' scoring information, adding users' scoring information has little effect on the numerical characteristics of the

cloud model, which is called a steady state. Fig.17 shows that the cost of s_1 in this application scenario is basically in a steady state for the 3 cloud numerical characteristics generated after about 1500 user scores. There is no need to perform QoS cloud model computation after the three numerical characteristics of the cloud model reach the stable state, and the time complexity comparison of the three algorithms after the stable state is shown in Fig. 18.

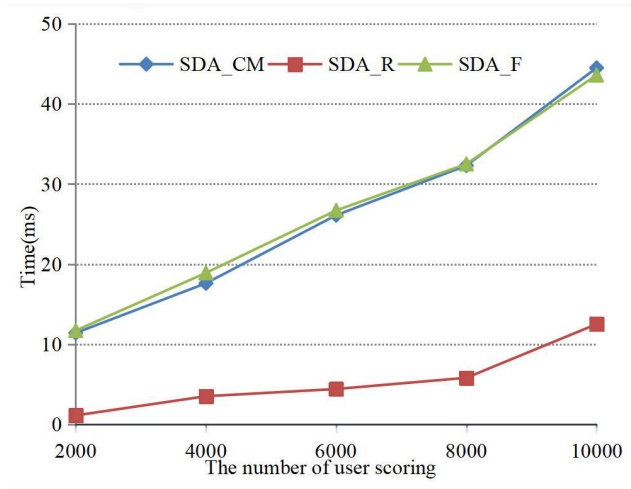


FIGURE 16. Comparison of time complexity.

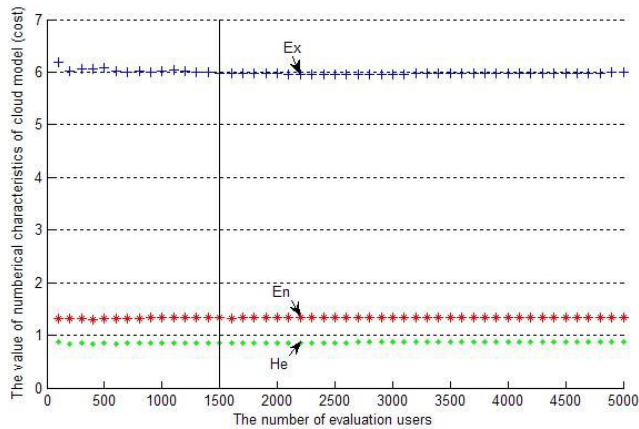


FIGURE 17. Cost of s_1 cloud numerical characteristics change curve.

QoS contains 5 attributes, QoS values are randomly generated, the number of processors is 10, and the two algorithms are repeatedly executed 1000 times to compare the average time consumption of the two algorithms SDA_CM and PSDA_CM. From Fig. 19, it can be seen that as the number of services increases, the average time consumption of SDA_CM shows a linear growth trend, and the average time consumption of PSDA_CM grows less, and PSDA_CM has a better execution efficiency in large-scale service selection. When the number of alternative services is 10000 and the algorithm is repeatedly executed 1000 times, the variation of the average time consumption of PSDA_CM with the number

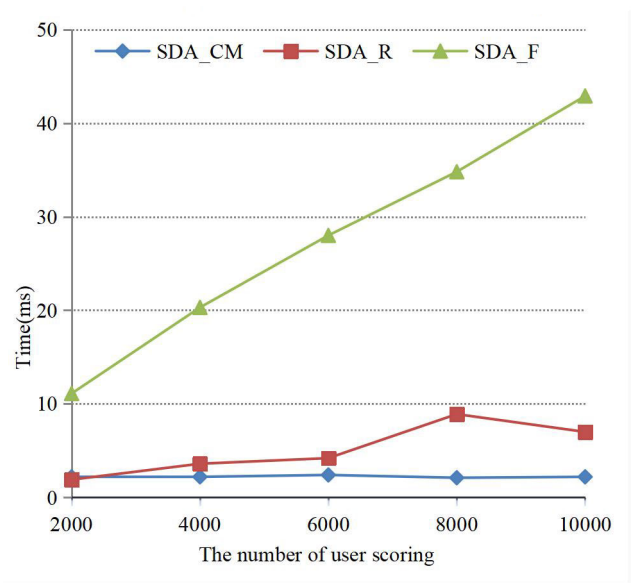


FIGURE 18. comparison of time complexity of three kinds of algorithms under steady state.

of processors is shown in Fig. 20. From Fig. 20, it can be seen that in this example, when the number of processors is less than or equal to 4, the increase in the number of processors can significantly improve the algorithm execution time; when the number of processors is greater than 4, the variation of the algorithm execution time with the increase in the number of processors is not significant.

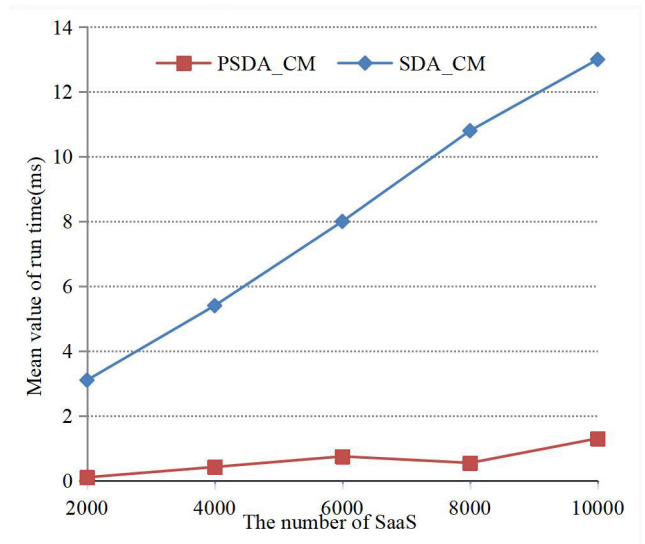


FIGURE 19. Run time comparison of SDA_CM and PSDA_CM.

Comparing the performance that based on Skyline set (SSDA_CM) and non-Skyline set (NSSDA_CM) to choose the optimal SaaS, we need to set up three different types of data sets [42]: (1) Forward QoS data sets, arbitrarily attribute value of QoS is preferably and any other attribute values

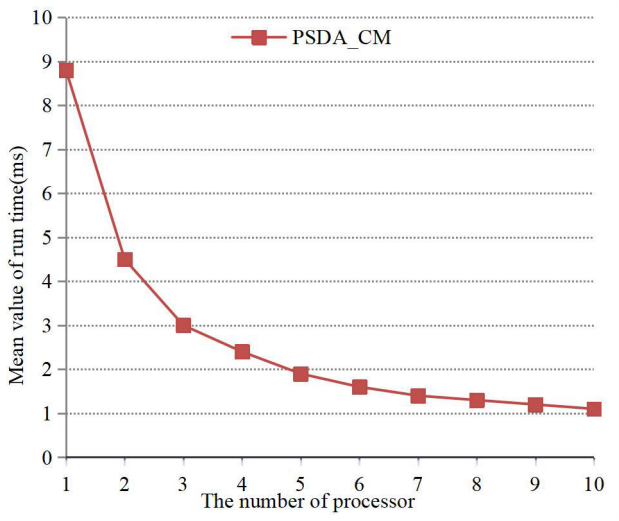


FIGURE 20. PSDA_CM time consumption curve.

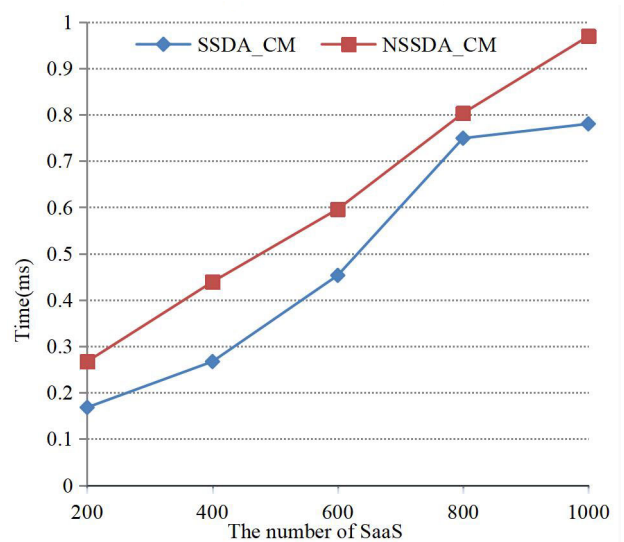


FIGURE 22. Time consumption curve on reverse QoS data sets.

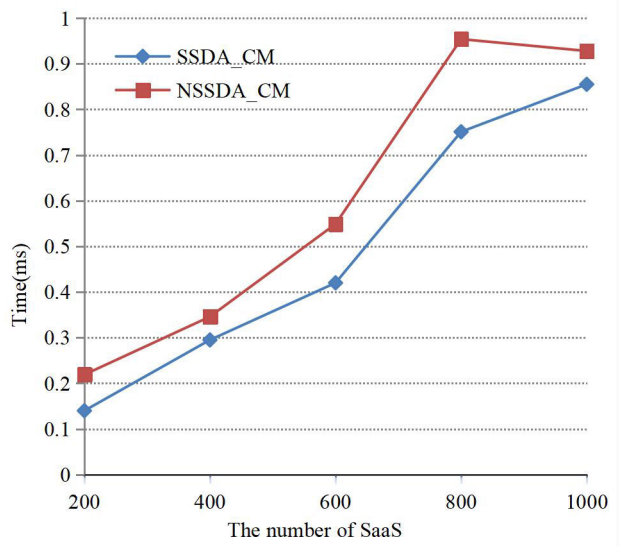


FIGURE 21. Time consumption curve on forward QoS data sets.

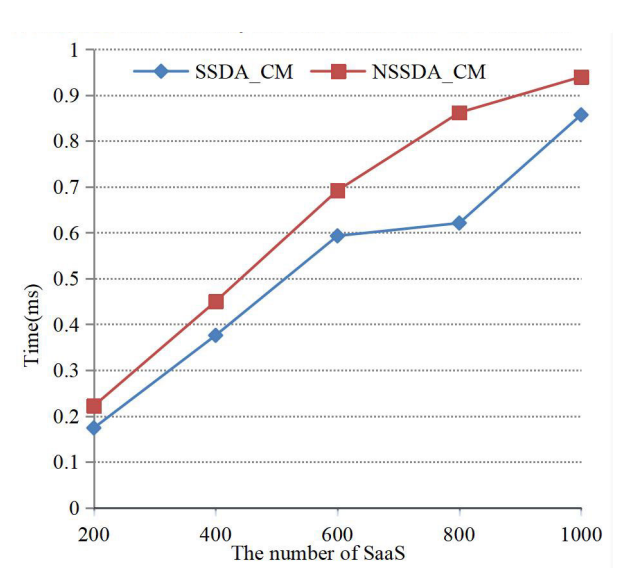


FIGURE 23. Time consumption curve on independent QoS data sets.

also is better; (2) Reverse QoS data sets, arbitrarily attribute value of QoS is preferably and in other attributes there at least one attribute value is poor; (3) Independent QoS data sets, QoS attribute values randomly generated. Taking the response time and throughput of two QoS attributes, and taking the average time of 1000 times execution.

Fig.21 depicts the comparison of the average execution time of the two algorithms in the forward QoS dataset, it can be seen that as the number of services increases, the average execution time of the Skyline set-based SDA_CM is better than that of the full set-based SDA_CM. Fig.22 depicts the comparison of the average execution time of the two algorithms in the reverse QoS dataset, from the figure it can be seen that as the number of services increases, the average execution time of the Skyline set based SDA_CM is better

than the full set based SDA_CM. Fig.23 depicts the comparison of the average execution time of the two algorithms in the independent QoS dataset, from which it can be seen that as the number of services increases, the average execution time of the Skyline set based SDA_CM is better than the full set based SDA_CM. Overall, the average execution time of the SSDA_CM based on the forward dataset is better than the other two datasets, and the performance is not significant here because the number of services is relatively small.

VII. DISCUSSION

In the environment of big QoS data and massive alternative services, UBQoS_ESDM has higher ability to describe big QoS data and higher efficiency of optimal service selection.

However, applying UBQoS_ESDM to engineering practice, there may be the following issues that need further discussion. (1) The number of samples for generating QoS cloud models should be determined according to specific application scenarios; too few samples cannot reflect the overall state of QoS, and too many samples cannot effectively reflect the local characteristics of QoS. (2) The selection of sample dimensions may also affect the descriptive ability of the QoS cloud model, e.g., the QoS cloud model generated from samples in the time dimension and samples in the location dimension will reflect the QoS state differently, and it may be necessary to select the sample dimensions based on the actual application scenarios. (3) The QoS cloud model has a certain degree of uncertainty, and the following two situations may exist: the uncertain service skyline computation may have fewer dominated services appearing in the uncertain service skyline set; the service decision algorithm based on the cloud model has fewer alternative services with incorrect ordering results. However, both algorithms have higher accuracy when the QoS cloud model is in a steady state. (4) In uncertain service skyline computing, no uncertain service skyline query algorithm based on QoS cloud model has been designed, and the commonly used skyline query algorithms BNL, D&C, NN, BBS need to be improved in engineering applications.

VIII. CONCLUSION

In this paper, we analyze the randomness and ambiguity of big QoS data, and propose to use cloud model to portray QoS attributes, which has better descriptive ability compared with existing methods. The backward QoS cloud generator is proposed to convert big QoS data to QoS cloud model, which characterize the current SaaS state through local QoS data; the QoS cloud model is dynamically adjusted through QoS cloud model adaptive adjustment mechanism to adapt to the dynamic changes of QoS. Through the uncertain service Skyline calculation method to eliminate the dominated SaaS in the alternative service set, narrow the search space of SaaS decision-making algorithm and improve the efficiency of the algorithm. The SaaS decision algorithm based on the cloud model can obtain the QoS-optimized SaaS reflecting the user's preference, while the parallel algorithm is designed to improve the execution efficiency of the algorithm. Experiments demonstrate that UBQoS_ESDM is an efficient and reliable method for dealing with optimal SaaS selection based on uncertain large QoS data.

The next research work focuses on three aspects: (1) In big QoS data, there are always some users who submit QoS for some purposes that do not match with their monitored QoS, and methods to identify and exclude such QoS data will be investigated to enhance the credibility of the QoS cloud model. (2) Feedback QoS information needs to occupy users' resources, and a mechanism will be established to mimic users' free-riding behavior to obtain more and more accurate QoS information. (3) Composite services consist of shared atomic services combined in a certain logical way

(often using control structures to orchestrate these shared atomic services, such as sequential, parallel, choosing, etc.) to provide more powerful and complete value-added services to users. Composite services are more complex than atomic services, and the difficulty of service composition is exacerbated by the uncertainty of the QoS of atomic services, so efficient uncertain service composition algorithms will be further investigated.

ACKNOWLEDGMENT

The authors are grateful to a senior editor and three anonymous referees for their constructive comments and suggestions, which have helped them significantly improve the quality and exposition of this article.

APPENDIX

LIST OF ABBREVIATIONS

Abbreviation	Full form
SaaS	Software-as-a-service
QoS	Quality of Service
UBQoS_ESDM	Uncertain Big QoS Data-driven Efficient SaaS Decision-making Method
MADM	Multi-attribute decision-making
AHP	Analytical Hierarchy Process
SAW	Simple Additive Weight
TOPSIS	Technique for Order of Preference by Similarity to ideal solution
ELECTRE	Elimination and Choice Expressing Reality
BWM	Best Worst Method
PROMETHEE	Preference Ranking Organization Method for Enrichment of Evaluations
WASPAS	Weighted Aggregate Sum and Product Assessment
MARCOS	Measurement of Alternatives and Ranking according to Compromise Solution
CF	Collaborative Filtering
DNN	Deep Neural Network
MEC	Multi-access Edge Computing
MF	Matrix Factorization
ARIMA	Auto Regressive Integrated Moving Average
NLFT	non-negative latent factor tensors
GA	Genetic Algorithm
MA	Memetic Algorithm
ACO	Ant Colony Optimization
SA	Simulated Annealing
PSO	Particle Swarm Optimization
ABC	Artificial Bee Colony Algorithm
SDA_CM	Service Decision Based on Cloud Model
PSDA_CM	Parallel Processing of SDA_CM
SDA_R	Service Decision Based on Real QoS Model
SDA_F	Service Decision Based on Interval Number QoS Model
SSDA_CM	SDA_CM Based on Skyline Set

NSSDA_CM	SDA_CM Not Based on Skyline Set
AQoS_CM	Adaptive QoS Model
NIQoS_CM	n Interval of QoS Cloud Model
RTQoS_CM	Real-time QoS Cloud Model
S_QoS_BCG	Static Backward QoS Cloud Generator
DS_QoS_BCG	Distributed Static Backward QoS Cloud Generator

REFERENCES

- N. Limam and R. Boutaba, "Assessing software service quality and trustworthiness at selection time," *IEEE Trans. Softw. Eng.*, vol. 36, no. 4, pp. 559–574, Jul. 2010, doi: [10.1109/TSE.2010.2](#).
- Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-aware web service recommendation by collaborative filtering," *IEEE Trans. Services Comput.*, vol. 4, no. 2, pp. 140–152, Apr. 2011, doi: [10.1109/TSC.2010.52](#).
- Y. Zhang, Z. Zheng, and M. R. Lyu, "WSPred: A time-aware personalized QoS prediction framework for web services," in *Proc. IEEE 22nd Int. Symp. Softw. Rel. Eng.*, Hiroshima, Japan, Nov. 2011, pp. 210–219, doi: [10.1109/ISSRE.2011.17](#).
- Y. Hu, Q. Peng, X. Hu, and R. Yang, "Time aware and data sparsity tolerant web service recommendation based on improved collaborative filtering," *IEEE Trans. Services Comput.*, vol. 8, no. 5, pp. 782–794, Sep. 2015, doi: [10.1109/TSC.2014.2381611](#).
- M. Eisa, M. Younas, K. Basu, and I. Awan, "Modelling and simulation of QoS-aware service selection in cloud computing," *Simul. Model. Pract. Theory*, vol. 103, Sep. 2020, Art. no. 102108, doi: [10.1016/j.simpat.2020.102108](#).
- Y. Shu, J. Zhang, D. Zuo, and Q. Z. Sheng, "Interval-valued skyline web service selection on incomplete QoS," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Barcelona, Spain, Jul. 2022, pp. 361–366, doi: [10.1109/ICWS55610.2022.00060](#).
- R. Kumar Tiwari and R. Kumar, "A framework for prioritizing cloud services in neutrosophic environment," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 34, no. 6, pp. 3151–3166, Jun. 2022, doi: [10.1016/j.jksuci.2020.05.009](#).
- O. Gireesha, N. Somu, and K. Krithivasan, "IIVIFS-WASPAS: An integrated multi-criteria decision-making perspective for cloud service provider selection," *Future Gener. Comput. Syst.*, vol. 103, pp. 91–110, Feb. 2020, doi: [10.1016/j.future.2019.09.053](#).
- X.-Q. Fan, C.-J. Jiang, J.-L. Wang, and S.-C. Pang, "Random-QoS-aware reliable web service composition: Random-QoS-aware reliable web service composition," *J. Softw.*, vol. 20, no. 3, pp. 546–556, Mar. 2010, doi: [10.3724/SP.J.1001.2009.03339](#).
- G. Wang, D. Li, and Y. Yao, *Cloud Model and Granular Computing*. Beijing, China: Sci. Press, 2012.
- L. Zeng, B. Benatallah, A. H. H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang, "QoS-aware middleware for web services composition," *IEEE Trans. Softw. Eng.*, vol. 30, no. 5, pp. 311–327, May 2004, doi: [10.1109/TSE.2004.11](#).
- H. Chen, S. Deng, and H. Zhao, "Composite service selection and optimization for mobile edge systems," *Chin. J. Comput.*, vol. 45, no. 4, pp. 82–97, Jan. 2022, doi: [10.11897/SP.J.1016.2022.00082](#).
- D. Wu, P. Zhang, Y. He, and X. Luo, "A double-space and double-norm ensemble latent factor model for highly accurate web service QoS prediction," *IEEE Trans. Services Comput.*, vol. 16, no. 2, pp. 802–814, Mar. 2023, doi: [10.1109/TSC.2022.3178543](#).
- H. Ma and Z. Hu, "Recommend trustworthy services using interval numbers of four parameters via cloud model for potential users," *Frontiers Comput. Sci.*, vol. 9, no. 6, pp. 887–903, Dec. 2015, doi: [10.1007/s11704-015-4532-0](#).
- S.-Y. Hwang, C.-C. Hsu, and C.-H. Lee, "Service selection for web services with probabilistic QoS," *IEEE Trans. Services Comput.*, vol. 8, no. 3, pp. 467–480, May 2015, doi: [10.1109/TSC.2014.2338851](#).
- M. Hosseinezhad, M. A. Azgomi, and M. R. E. Dishabi, "A probabilistic trust model for cloud services using Bayesian networks," *Soft Comput.*, vol. 28, no. 1, pp. 509–526, May 2023, doi: [10.1007/s00500-023-08264-z](#).
- O. Gireesha, A. B. Kamalesh, K. Krithivasan, and V. S. Shankar Sriram, "A fuzzy-multi attribute decision making approach for efficient service selection in cloud environments," *Expert Syst. Appl.*, vol. 206, Nov. 2022, Art. no. 117526, doi: [10.1016/j.eswa.2022.117526](#).
- L. Zhang, X. Zhang, and Y. Yang, "Hybrid QoS-aware web service composition strategies for group Pareto optimal plan," *J. Internet Technol.*, vol. 16, no. 2, pp. 255–266, 2015, doi: [10.6138/JIT.2015.16.2.20120827](#).
- N. Thakur, A. Singh, and A. L. Sangal, "Cloud services selection: A systematic review and future research directions," *Comput. Sci. Rev.*, vol. 46, Nov. 2022, Art. no. 100514, doi: [10.1016/j.cosrev.2022.100514](#).
- R. K. Tiwari and R. Kumar, "G-TOPSIS: A cloud service selection framework using Gaussian TOPSIS for rank reversal problem," *J. Supercomput.*, vol. 77, no. 1, pp. 523–562, Jan. 2021, doi: [10.1007/s11227-020-03284-0](#).
- H. Ma, H. Zhu, Z. Hu, K. Li, and W. Tang, "Time-aware trustworthiness ranking prediction for cloud services using interval neutrosophic set and ELECTRE," *Knowl.-Based Syst.*, vol. 138, pp. 27–45, Dec. 2017, doi: [10.1016/j.knsys.2017.09.027](#).
- A. E. Youssef, "An integrated MCDM approach for cloud service selection based on TOPSIS and BWM," *IEEE Access*, vol. 8, pp. 71851–71865, 2020, doi: [10.1109/ACCESS.2020.2987111](#).
- L. Purohit and S. Kumar, "A classification based web service selection approach," *IEEE Trans. Services Comput.*, vol. 14, no. 2, pp. 315–328, Mar. 2021, doi: [10.1109/TSC.2018.2805352](#).
- Z. Zheng, X. Li, M. Tang, F. Xie, and M. R. Lyu, "Web service QoS prediction via collaborative filtering: A survey," *IEEE Trans. Services Comput.*, vol. 15, no. 4, pp. 2455–2472, Jul. 2022, doi: [10.1109/TSC.2020.2995571](#).
- A. N. Ngaffo, W. E. Ayeb, and Z. Choukair, "Service recommendation driven by a matrix factorization model and time series forecasting," *Int. J. Speech Technol.*, vol. 52, no. 1, pp. 1110–1125, Jan. 2022, doi: [10.1007/s10489-021-02478-0](#).
- H. Wu, X. Luo, and M. Zhou, "Advancing non-negative latent factorization of tensors with diversified regularization schemes," *IEEE Trans. Services Comput.*, vol. 15, no. 3, pp. 1334–1344, May 2022, doi: [10.1109/TSC.2020.2988760](#).
- Z. Liu, Q. Z. Sheng, Z. Zhang, X. Xu, D. Chu, J. Yu, and S. Wang, "Accurate and reliable service recommendation based on bilateral perception in multi-access edge computing," *IEEE Trans. Services Comput.*, vol. 16, no. 2, pp. 886–899, Mar. 2023, doi: [10.1109/TSC.2022.3155448](#).
- Z. Ding, J. Liu, Y. Sun, C. Jiang, and M. Zhou, "A transaction and QoS-aware service selection approach based on genetic algorithm," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 7, pp. 1035–1046, Jul. 2015, doi: [10.1109/TSMC.2015.2396001](#).
- S. Sadeghiram, H. Ma, and G. Chen, "Priority-based selection of individuals in memetic algorithms for distributed data-intensive web service compositions," *IEEE Trans. Services Comput.*, vol. 15, no. 5, pp. 2939–2953, Sep. 2022, doi: [10.1109/TSC.2021.3066322](#).
- F. Dahan, K. E. Hindi, A. Ghoneim, and H. Alsalman, "An enhanced ant colony optimization based algorithm to solve QoS-aware web service composition," *IEEE Access*, vol. 9, pp. 34098–34111, 2021, doi: [10.1109/ACCESS.2021.3061738](#).
- A. M. Mohamed and H. M. Abdelsalam, "A multicriteria optimization model for cloud service provider selection in multicloud environments," *Softw., Pract. Exper.*, vol. 50, no. 6, pp. 925–947, Jun. 2020, doi: [10.1002/spe.2803](#).
- x. wang, X. Xu, Q. Z. Sheng, Z. Wang, and L. Yao, "Novel artificial bee colony algorithms for QoS-aware service selection," *IEEE Trans. Services Comput.*, vol. 12, no. 2, pp. 247–261, Mar. 2019, doi: [10.1109/TSC.2016.2612663](#).
- H. Wang, Y. Ding, and H. Xu, "Particle swarm optimization service composition algorithm based on prior knowledge," *J. Intell. Manuf.*, vol. 35, no. 1, pp. 35–53, Jan. 2024, doi: [10.1007/s10845-022-02032-w](#).
- Q. He, J. Yan, H. Jin, and Y. Yang, "Quality-aware service selection for service-based systems based on iterative multi-attribute combinatorial auction," *IEEE Trans. Softw. Eng.*, vol. 40, no. 2, pp. 192–215, Feb. 2014, doi: [10.1109/TSE.2013.2297911](#).
- P. Wang and X. Du, "QoS-aware service selection using an incentive mechanism," *IEEE Trans. Services Comput.*, vol. 12, no. 2, pp. 262–275, Mar. 2019, doi: [10.1109/TSC.2016.2602203](#).
- X. Peng, B. Chen, W. Zhao, and R. Lin, "Web services dynamic combination method which support risk appetite," *Scientia Sinica Inf.*, vol. 44, no. 1, pp. 130–141, Jan. 2014, doi: [10.1360/N112013-00099](#).
- F. Zhang, K. Hwang, S. U. Khan, and Q. M. Malluhi, "Skyline discovery and composition of multi-cloud mashup services," *IEEE Trans. Services Comput.*, vol. 9, no. 1, pp. 72–83, Jan. 2016, doi: [10.1109/TSC.2015.2449302](#).

- [38] Y. Shu, J. Zhang, W. E. Zhang, D. Zuo, and Q. Z. Sheng, "IQSrec: An efficient and diversified skyline services recommendation on incomplete QoS," *IEEE Trans. Services Comput.*, vol. 16, no. 3, pp. 1934–1948, May 2022, doi: [10.1109/TSC.2022.3189503](https://doi.org/10.1109/TSC.2022.3189503).
- [39] Q. Yu and A. Bouguettaya, "Computing service skyline from uncertain QoS," *IEEE Trans. Services Comput.*, vol. 3, no. 1, pp. 16–29, Jan. 2010, doi: [10.1109/TSC.2010.7](https://doi.org/10.1109/TSC.2010.7).
- [40] K. Benouaret, D. Benslimane, and A. Hadjali, "Selecting skyline web services from uncertain QoS," in *Proc. IEEE 9th Int. Conf. Services Comput.*, Honolulu, HI, USA, Jun. 2012, pp. 523–530, doi: [10.1109/SCC.2012.84](https://doi.org/10.1109/SCC.2012.84).
- [41] G.-W. Zheng, "A collaborative filtering recommendation algorithm based on cloud model," *J. Softw.*, vol. 18, no. 10, p. 2403, Oct. 2007, doi: [10.1360/jos182403](https://doi.org/10.1360/jos182403).
- [42] S. Borzsony, D. Kossmann, and K. Stocker, "The skyline operator," in *Proc. 17th Int. Conf. Data Eng.*, Berlin, Germany, 2001, pp. 421–430, doi: [10.1109/ICDE.2001.914855](https://doi.org/10.1109/ICDE.2001.914855).



LONGCHANG ZHANG (Member, IEEE) was born in Xiuyan, Liaoning, China, in 1978. He received the B.S. degree in computer science and technology from Bohai University, Jinzhou, China, in 2001, and the M.S. and Ph.D. degrees in computer science and technology from the Beijing University of Posts and Telecommunications, Beijing, in 2008 and 2011, respectively.

From 2011 to 2012, he was a Lecturer with the College of Information Science and Technology, Bohai University, where he was an Associate Professor, from 2013 to 2018.



JING BAI was born in 1987. She received the master's degree in business administration from the Beijing University of Posts and Telecommunications, in 2020. She is currently pursuing the Ph.D. degree with the School of Management Science and Engineering, Dongbei University of Finance and Economics, China. Her paper has appeared in computer-integrated manufacturing systems. Her current research interests include cloud computing and inventory optimization.

• • •