Journal of Systems Engineering and Electronics
Vol. 31, No. 4, August 2020, pp.743 – 750

A nonlinear service composition method based on the Skyline operator

HUO Ying* and ZHANG Jiande

School of Computer Engineering, Nanjing Institute of Technology, Nanjing 211167, China

Abstract: The concept of service composition can provide the complex functionality for users. As the widespread application of cloud computing, the number of services grows exponentially. It becomes more difficult to find out the optimal service composition solution quickly. This paper proposes a nonlinear service composition method based on the Skyline operator. The Skyline operator is to find a collection of data that cannot be dominated by others, which is used to prune the redundant services to reduce the search space. Then the service composition problem is formulated as a nonlinear integer programming model by a mathematical programming language (AMPL), and solved by the existing nonlinear solvers Bonmin. The experiments show that the proposed method can effectively improve the efficiency of service composition, while ensuring the quality of solution.

Keywords: quality of service, service composition, Skyline service, branch-and-bound Skyline.

DOI: 10.23919/JSEE.2020.000049

1. Introduction

The concept of service oriented architecture (SOA) solves the problem of cross-platform resource sharing. In order to ensure the modularity and reusability, the functionality of individual services is often limited, thus they need to be composited to satisfy the complex demand of users. The widespread application of cloud computing promotes the development of SOA. As the number of services grows exponentially, the massive services with similar functionality present new challenges to the effectiveness and efficiency of the existing service composition methods. It has been a research focus to find out a service composition solution as quickly as possible, which has the best properties and also satisfies all the functional requirements [1,2].

The existing service composition methods include the local selection, the global optimization, the intelligent optimization, and the exhaustive method. The local selection method can only obtain a local optimization solution, but its efficiency is high. The global optimization method transforms the global service composition problem into the mixed integer linear programming problem, and it can get the solution with high quality in a relatively short time, but it requires all the objective functions and constraints to be linearized, which limits the application to a certain extent. Although the intelligent optimization method also has a high efficiency, it can only obtain the approximate optimal solution because of its randomness. Moreover, the exhaustive method can obtain the optimal solution certainly, but its time complexity is too high to be practical.

The surge of candidate services is the most important factor to reduce the efficiency of service composition [3]. If we can select the potential services in advance, the search space can be reduced and the efficiency will be improved effectively. The concept of Skyline [4] first appeared in the field of data engineering, which is a collection of data that cannot be dominated by others. Based on this, we can select the non-dominated services from each service group, then just composite them to improve the efficiency.

The remainder of this paper is organized as follows. In Section 2, the related work is discussed. Section 3 gives the definition of the Skyline service and analyzes the calculation method. The service composition problem is modeled as a nonlinear integer programming problem in Section 4, and be solved by basic open-source nonlinear mixed integer programming (Bonmin). Experiments to validate the proposed model are presented in Section 5. Finally, Section 6 gives our conclusion.

2. Related work

2.1 Service composition

Service composition is selecting the appropriate service

Manuscript received April 17, 2019.
*Corresponding author.
This work was supported by the National Natural Science Foundation for Youth of China (61802174), the Natural Science Foundation for Youth of Jiangsu Province (BK20181016), the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (18KJB520019), and the Scientific Research Foundation of Nanjing Institute of Technology of China (CXY201922).

components in each service group, then compositing them to satisfy the relevant constraints and make the service quality optimal. Currently, there are four existing service composition methods.

(i) Exhaustive method. Traversing all the composite service solutions, and find out the one with the maximal quality of service (QoS). This can obtain the optimal solution certainly, but its time complexity is too high to be practical.

(ii) Local selection method. Different QoS attribute values are mapped into a certain value, and the multiple criteria decision-making (MCDM) process is used to select the appropriate service from each group [5]. It can only obtain a local optimization solution, whereas its efficiency is quite high.

(iii) Global optimization method. It transforms the global QoS service composition problem into the mixed integer linear programming problem, then be solved by the linear solvers, such as CPLEX [6,7] or IPSOLVE [5]. It can get the solution with high quality in a relatively short time, but it requires all the objective functions and constraints to be linearized, which limits the application of this method to a certain extent.

(iv) Intelligent optimization method. It also can be called the global optimization method. The difference is that the problem is nonlinear and is solved by the intelligent optimization algorithms, such as genetic algorithm (GA) [8], particle swarm optimization (PSO) algorithm [9], ant colony optimization (ACO) algorithm [10], artificial bee colony (ABC) algorithm [11,12], and so on. Although the intelligent optimization method also has a high efficiency, it can only obtain the approximate optimal solution because of its randomness.

In some applications, the absolute optimal service composition solution needs to be obtained in a short period of time, and the model is complex and cannot be linearized. Therefore, this paper formulates the problem as a nonlinear integer programming model, and solves the problem by the existing nonlinear solvers.

2.2 Skyline operator

Skyline is a collection of data that cannot be dominated by others [4]. Some algorithms have been proposed to compute the Skyline set, like the branch and bound Skyline (BBS) algorithm [13]. BBS is simple to implement and can be efficiently applied to a variety of alternative Skyline queries.

In the service composition problem, although all services in each service group are candidate services, not all of them are potential candidates for the optimal solution. Based on the notion of Skyline, Alrifai proposed a new service composition method [14]. The Skyline operator is attached into the process of service composition, to filter out candidate services that cannot dominated by others in each service group, which can reduce search space and improve the efficiency. Yu presented a new concept, called p-dominant service Skyline, to deal with the uncertainty of the QoS [15]. A p-R-tree indexing structure and a dual-pruning scheme are presented to efficiently compute the p-dominant Skyline. For the cloud-base web services, a fast service composition approach based on the Skyline operator and PSO is proposed, which is the combination of Skyline and the intelligent algorithm [16]. For the large scale services with high-dimensional QoS attributes, Moradi applied the parallelism technique into the Skyline computing, to increase the speed of service composition [17]. For building up cloud mashup applications, Zhang proposed an integrated Skyline query processing method, using a similarity test to achieve optimal localized Skyline [18]. Guo proposed a service selection approach based on QoS prediction [19], the Skyline is also applied to prune redundant services and perform Skyline service selection using 0-1 mixed-integer programming.

According to the existing research work, the Skyline operator is widely used to prune the redundant services and improve the efficiency of the service composition. In this paper, the Skyline operator and the efficient algorithm BBS are used to compute the Skyline service set.

3. Service composition based on Skyline operator

3.1 Definition of Skyline service

Alrifai gave the definition of Skyline service [14]. Consider a service group S_i includes n_i candidate services $S_i = \{ws_{i,1}, ws_{i,2}, \dots, ws_{i,n_i}\}$, and each candidate service considers r QoS attributes $q(ws_{i,j}) = \{q_1, q_2, \dots, q_r\}$. In this paper, we suppose each attribute value preference for smaller values. We define the service dominance and Skyline services as follows.

Definition 1 Service dominance

If $ws_{i,j}$ is as good as or better than $ws_{i,k}$ in all parameters of attributes and better in at least one attribute, i.e., $\forall\, l \in [1,r], ws_{i,j}(q_l) \leqslant ws_{i,k}(q_l)$ and $\exists\, l \in [1,r]$, $ws_{i,j}(q_l) < ws_{i,k}(q_l)$, then $ws_{i,j}$ dominates $ws_{i,k}$, denoted as $ws_{i,j} \prec ws_{i,k}$.

Definition 2 Skyline services

In the service group $S_i = \{ws_{i,1}, ws_{i,2}, \dots, ws_{i,n_i}\}$, the collection of all the candidate services that cannot be dominated by any other service is the Skyline services set, denoted by $SkyS_i$, i.e., $SkyS_i = \{ws_{i,j} \in S_i | \neg\exists\, ws_{i,k} \in S_i : ws_{i,k} \prec ws_{i,j}\}$.

An example of reservation services is given below. In this case, the reservation service group S_i consists of nine candidate services $ws_{i,1}$ to $ws_{i,9}$, such as booking, ctrip, agoda, hotelclub, and so on. Each candidate service considers two QoS attributes, the price and the response time. The price is the booking fee, ranging from 0 to \$10. And the response time is the time for feedback after the service request be sent, ranging from 0 to 10 s. The QoS information for all candidate services is identified in a two-dimensional space, as shown in Fig. 1.

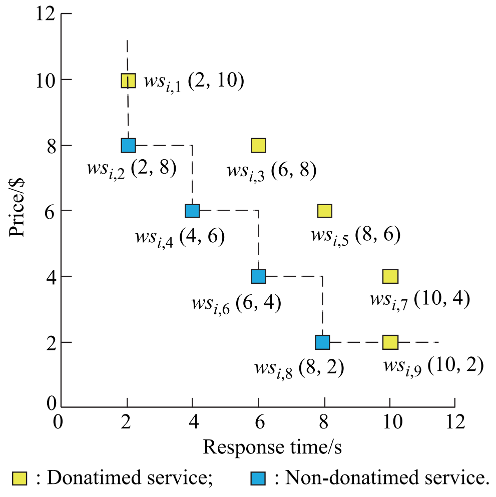


Fig. 1 Example of Skyline services in the reservation service group

As shown in Fig. 1, because of $2 = 2$ and $8 < 10$, then $ws_{i,2} \prec ws_{i,1}$. Besides, $2 < 4$ but $8 > 6$, thus there is no domination relationship between $ws_{i,2}$ and $ws_{i,4}$. In addition, there are no other services having a lower price and a shorter response time than $ws_{i,2}$, thus $ws_{i,2} \in SkyS_i$. In the same way, the Skyline service set can be obtained as $\{ws_{i,2}, ws_{i,4}, ws_{i,6}, ws_{i,8}\}$.

In the calculation of the Skyline service of each service group, each QoS attribute of every candidate service needs to be compared. When there are a large number of candidate services, this calculation will cost too much time.

3.2 Calculation of Skyline service

3.2.1 Non-dominate algorithm

Non-dominate is a method of judging the domination of the solutions in the multi-objective optimization problem. The quality of solutions is evaluated by non-dominate sorting, to provide a reference for evolution and selection [20]. We improve it and design the non-dominate algorithm for solving the Skyline set, as shown Algorithm 1.

Algorithm 1 Non-dominate algorithm for solving the Skyline set

Input: Matrix $\boldsymbol{A}$ (It represents the QoS value of the whole services in a group, which has $|N\_\text{candidate}|$ rows and $|\text{ATTRIBUTE}|$ columns. $|N\_\text{candidate}|$ indicates the number of the candidate services in the group. $|\text{ATTRIBUTE}|$ represents the number of attributes considered for each service.)

Output: The Skyline service set Sky of this group

```
1: FOR $i$ = 1 : 1 : $|N\_\text{candidate}|$ //Traverse all the candidate services
2:    individual($i$).$n$ = 0; // Initialize the size of the dominated set
3:    FOR $j$ = 1 : 1 : $|N\_\text{candidate}|$ //Traverse all the candidate services again
4:       dom_less = dom_equal = dom_more = 0;
5:       FOR $k$ = 1 : 1: $|ATTRIBUTE|$ //Traverse all the attributes
6:          IF $(A(i,k) < A(j,k))$
7:             dom_less = dom_less + 1;
8:          ELSEIF $(A(i,k) == A(j,k))$
9:             dom_equal = dom_equal + 1;
10:         ELSE
11:            dom_more = dom_more + 1;
12:         END
13:      END
14:      IF dom_less == 0 && dom_equal $\sim=$ $|ATTRIBUTE|$;
15:         individual($i$).$n$ = individual($i$).$n$ + 1;
16:      END
17:   END
18: IF individual($i$).$n$ == 0
19:   $SkyA = SkyA \cup A(i,:)$;
20: END
21: END
```

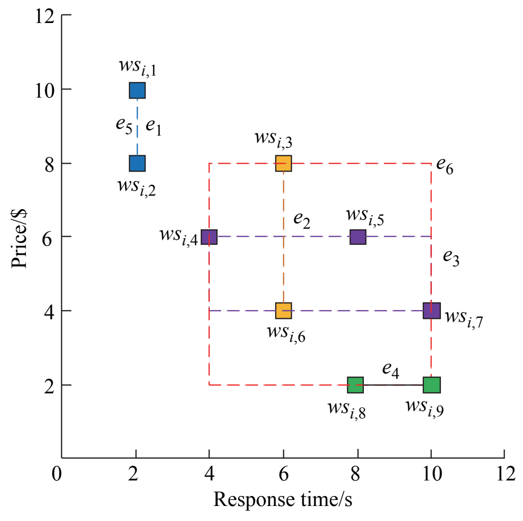
In the non-dominate algorithm, all the candidate services need to be traversed twice, and all the attributes need to be compared, thus its time complexity is $O(|N\_\text{candidate}|^2 \cdot |\text{ATTRIBUTE}|)$.

3.2.2 BBS algorithm

The general procedure of the BBS algorithm is as follows. First, the R tree [21] is constructed, which is a balance tree with two types of nodes: the leaf node and the non-leaf node. Based on space segmentation, the minimum bounding rectangle (MBR) is used to divide the space from the leaf node to the rectangle. And the concept of node capacity is presented to represent the largest number of submatrix or points included in each matrix in the R tree. Let the node capacity be 3. The R tree of Fig. 1 can be constructed as Fig. 2.

Definition 3 Minimum distance $Mindist$

Every node of the R tree can be a matrix or a point. For each node, $Mindist$ of every point is equal to the sum of its attributes value. And $Mindist$ of an MBR equals $Mindist$ of the point in the lower left.



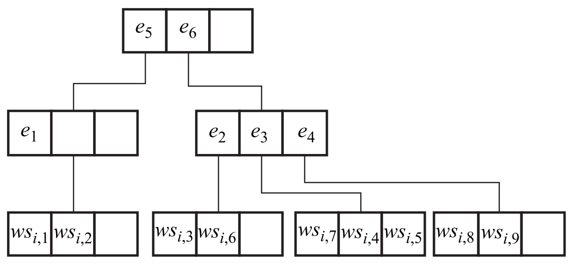


Fig. 2 Construction of R tree

The root node of the R tree is inserted into the heap. After computing the $Mindist$ of every node, the node with the minimum $Mindist$ would be selected from heap, and judge whether it is dominated. If non-dominated, it expands all its sub MBR and inserts them into the heap one by one. If the selected node does not contain sub MBR and cannot be dominated by any other node in $SkyR$, it can be added into the Skyline set $SkyR$.

As for Fig. 2, the element $< e_5, 10 >$ and $< e_6, 6 >$ in the root node would be inserted into the heap first. Then based on $Mindist$, e_6 should be shifted out first. Expand e_6 since it is an MBR, and its child nodes $< e_2, 10 >$, $< e_3, 8 >$, $< e_4, 10 >$ would be added into the heap and be sorted. After that, the element e_3 with $Mindist$ also should be expanded and sorted. Now the element with $Mindist$ would be $ws_{i,4}$, which does not need to expand since it is a point, and it can be added into $SkyR$ since $SkyR$ is empty now. At the next step, e_2 is expanded. Its child node $ws_{i,6}$ is not dominated by $ws_{i,4}$ and has been added into $SkyR$, whereas $ws_{i,3}$ is discarded since it is dominated by $ws_{i,4}$. Similarly, e_4, e_5, e_1 would be expanded. Finally the $SkyR$ is $\{ws_{i,2}, ws_{i,4}, ws_{i,6}, ws_{i,8}\}$.

Though the quick method of Skyline has been applied, it still costs time. Because service composition and the Skyline calculation are independent with each other, the calculation of Skyline can be completed in a off-line way. When the service list updates, only the Skyline set in the corresponding service group needs to be recomputed. In addition, Skyline services reflect the tradeoff between the different QoS attributes and can satisfy the diverse QoS requirements of the users.

4. Nonlinear integer programming modeling

4.1 A mathematical programming language (AMPL) modeling

In this section, we formulate the service composition problem as a nonlinear integer programming problem by AMPL. AMPL is a kind of modeling language to describe and solve complex mathematical problems. Its main characteristic is to use simple mathematical expressions to describe the optimization problem, and solve the problem through the existing solvers [22]. Now AMPL supports many popular solvers, such as CPLEX, Bonmin, and so on.

4.1.1 Definition of set

In AMPL, $N\_\mathrm{candidate}$ indicates the number of the total services. And these services are divided into N_class service groups. For each service, the QoS model is used to evaluate its quality. The quality attributes considered in this paper are response time, availability, throughput, price and reputation.

The service is evaluated according to these attributes from different aspects. The original information of evaluation can be obtained directly from the service provider such as price, or be derived from the users feedback such as reputation, or be calculated from the historical records such as response time, availability and throughput.

4.1.2 Definition of parameters

The parameter matrix $\boldsymbol{QOS}$ is defined to describe the evaluation value of service, where $QOS_{i,j}$ represents the evaluation value of the ith candidate service at the jth attribute.

For each attribute, there are three parameters: Weight, $\mathrm{Com\_status}$ and $\mathrm{Posi\_status}$, where Weight represents the different importance of each attribute, which can be set beforehand by users according to their preference. The sum of the weight of all attributes should be 1, i.e., $\sum_{j=1}^{|\mathrm{ATTRIBUTE}|} \mathrm{Weight}_j = 1$.

When composing the services, the composite quality of each attribute can be aggregated by the attribute value of each service. According to the difference of attributes, there are four types of aggregation operations: AVERAGE, MIN, SUM and PRODUCT. In the AMPL model, the parameter $\mathrm{Com\_status}$ is defined to identify the composition state of each attribute. The meaning is shown in Table 1.

Table 1 QoS attribute aggregation

QoS attribute	Com_status	Aggregation
Response time	1	SUM
Availability	2	PRODUCT
Throughput	3	MIN
Price	1	SUM
Reputation	4	AVERAGE

In the normalization phase, the service attributes can be classified as positive and negative by their semantic meanings. The higher the value is, the better the quality of positive attributes is, such as availability, throughput and reputation. For negative attributes, the lower the value is, the better the quality is, such as the response time and the price. They should be handled separately when being normalized. In this paper, the parameter Posi_status is defined as an attribute notation. When $\text{Posi\_status}_j = 1$, the jth attribute is a positive attribute, while $\text{Posi\_status}_j = -1$ means it is negative.

4.1.3 Definition of objective function and variables

The goal of the service composition is to find a solution to maximizing the QoS of the composite service ComSAW. Therefore, the objective function is as follows:

$$\mathrm{ComSAW} = \sum_{j=1}^{|\mathrm{ATTRIBUTE}|} \mathrm{UniComScore}_j \times \mathrm{Weight}_j \tag{1}$$

where Weight_j represents the importance of each attribute, $\mathrm{UniComScore}_j$ is the aggregation value of each attribute after being normalized, which is computed based on $\mathrm{Posi\_status}$, the equation is as follows:

$$\mathrm{UniComScore}_j = \begin{cases} \dfrac{\mathrm{ComScore}_j - \mathrm{MIN}_j}{\mathrm{MAX}_j - \mathrm{MIN}_j}, & \mathrm{Posi\_status}_j = 1 \\ \dfrac{\mathrm{MAX}_j - \mathrm{ComScore}_j}{\mathrm{MAX}_j - \mathrm{MIN}_j}, & \mathrm{Posi\_status}_j = -1 \end{cases} \tag{2}$$

where $\mathrm{MAX}_j(\mathrm{MIN}_j)$ represents the maximum (minimum) value of the jth attribute for all composition paths. If they are equal, i.e., $\mathrm{MAX}_j = \mathrm{MIN}_j$, then the normalized value is 1. $\mathrm{ComScore}_j$ is the aggregation value of the jth attribute of the composite service, which is calculated by different aggregation operations according to $\mathrm{Com\_status}$. The equation is as follows:

$$\mathrm{ComScore}_j = \begin{cases} \sum_{i=1}^{|N\_\mathrm{class}|} \mathrm{ComService}_{i,j}, & \mathrm{Com\_status} = 1 \\ \prod_{i=1}^{|N\_\mathrm{class}|} \mathrm{ComService}_{i,j}, & \mathrm{Com\_status} = 2 \\ \min_{i=1}^{|N\_\mathrm{class}|} \mathrm{ComService}_{i,j}, & \mathrm{Com\_status} = 3 \\ \left(\sum_{i=1}^{|N\_\mathrm{class}|} \mathrm{ComService}_{i,j}\right) / |N\_\mathrm{class}|, \\ \quad \mathrm{Com\_status} = 4 \end{cases} \tag{3}$$

In (3), $\mathrm{ComService}_{i,j}$ is a matrix of $|N\_\mathrm{class}|$ rows and $|\mathrm{ATTRIBUTE}|$ columns, to store the $\boldsymbol{QOS}$ information of the selected composition path. Let $|N\_\mathrm{candidate}|$ dimension vector $\boldsymbol{use}$ describe whether the service is selected, when $use_j = 1$, the jth service is selected in the composite path, while $use_j = 0$ is not. The selected composite path ComService can be obtained by multiplying the matrix $\boldsymbol{QOS}$ by $\boldsymbol{use}$.

4.2 Model solving

After the model is defined, it can be solved by the existing solvers. We choose the nonlinear integer programming solver Bonmin [22] to solve the problem.

5. Experiments

In this section, we verify the effectiveness of the proposed model and method through the experiments. The experimental environment is as follows: Intel Core i5-4590 (3.3 GHz), 4 GB RAM, Windows 7 (32 bit) and MATLAB R2010b.

5.1 Comparison of model

In order to verify the model proposed in this paper, we compare it with the existing methods described in Section 2.1, exhaustive, local, discrete gbest-guided artificial bee colony algorithm (DGABC), and Bonmin. DGABC is a kind of the intelligent optimization method proposed in our previous work [11]. The colony size is 20 and the iterations number is 1 000. And the Bonmin method is proposed in this paper.

The experiments in this section are conducted on the Random dataset. Each service has five attributes and the values are randomly generated assuming a uniform distribution in the interval [0,1]. Let $|N\_\mathrm{class}|$ be 5, and the number of candidate services in each service group $|N\_\mathrm{candidate}|/|N\_\mathrm{class}|$ increases from 5 to 60. The weight of each attribute is equal to 0.2. The results of ComSAW based on the above model are shown in Fig. 3. The exhaustive method only calculates the case of $|N\_\mathrm{candidate}|/|N\_\mathrm{class}| = 5, 10, 15$ because the time complexity is too high.

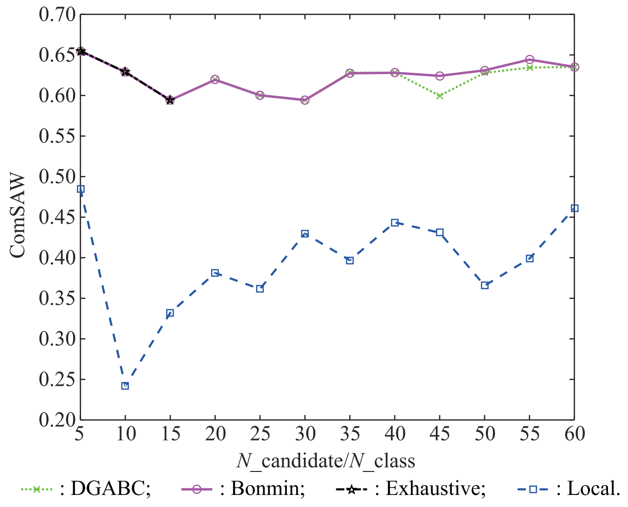


Fig. 3 Comparison of ComSAW

As shown in Fig. 3, the solution obtained by the local method is the worst. Because the local optimal service in each group cannot guarantee that the composite solution is the best. Although only three cases are calculated, the Bonmin method can obtain the same optimal solution as the exhaustive method. The DGABC method can obtain the optimal solution in most cases except two. The computation time of these methods is shown in Table 2.

Table 2 Comparison of computation time s

$\frac{\lvert N\_candidate\rvert}{\lvert N\_class\rvert}$	Bonmin	Local	DGABC	Exhaustive
5	0.13	0.005 8	0.544 5	0.087 2
10	0.46	0.004 9	0.548 9	97.191 8
15	1.34	0.004 8	0.555 9	5 983.208 1
20	1.12	0.004 8	0.562 8	—
25	2.79	0.004 9	0.557 5	—
30	3.96	0.005 0	0.544 2	—
35	7.44	0.005 0	0.555 7	—
40	10.67	0.005 0	0.545 3	—
45	30.1	0.004 8	0.546 7	—
50	21.28	0.004 9	0.560 1	—
55	29.21	0.004 9	0.546 2	—
60	19.04	0.004 9	0.544 0	—

As shown in Table 2, the time complexity of the exhaustive method is too high to be practical. While that of the local method is the best, but its ComSAW is the worst. The computation time of DGABC is controlled by the number of iterations, thus it always stays around 0.55 s because of the same iterations. The Bonmin method proposed in this paper can obtain the optimal solution as the exhaustive method in the limited time, thus can be effectively applied in the service composition.

5.2 Comparison of Skyline calculation

The following experiments are conducted to analyze the two Skyline calculation methods described in Section 3 from the effectiveness and efficiency aspects.

For more real data of QoS, the quality of web service (QWS) dataset is used in this section, which was collected by Eyhab Al-Masri [23–25]. 2507 real Web services are included and each one considers nine quality attributes.

5.2.1 Skyline filtering service

As shown in Table 3, the remaining services increase as the considering attributes increase. The reason is that the dominance relation becomes less apparent as the attributes increase. Usually, five attributes would be considered in the service composition.

Table 3 Skyline filtering service on different numbers of attributes

Attribute	Before filtering	After filtering
2	2 507	4
3	2 507	12
4	2 507	12
5	2 507	51
6	2 507	72
7	2 507	163
8	2 507	278
9	2 507	475

In addition, we analyze the Skyline filtering when the candidate services increase. The number of attributes is 5, and the candidate services increase from 250 to 2 500. The remaining number of services after filtering is shown in Table 4.

Table 4 Skyline filtering service as the number of services increases

Before filtering	After filtering
250	36
500	26
750	28
1 000	35
1 250	31
1 500	33
1 750	46
2 000	46
2 250	49
2 500	51

As shown in Table 4, there is a little growth of the number after filtering as the candidate services increase. Therefore, when the number of QoS attributes is determined, even if there are a huge number of candidate services, the Skyline set would not be too large after filtering.

5.2.2 Efficiency comparison of different Skyline methods

In this section, we analyze the efficiency of the two different Skyline methods. The calculation time is shown in Fig. 4, where BBS_Node represents the node capacity of the R tree, and we carry out the experiments respectively when BBS_Node is 3 and 4. In addition, because the R tree can be constructed offline, the time of constructing the tree is not included.

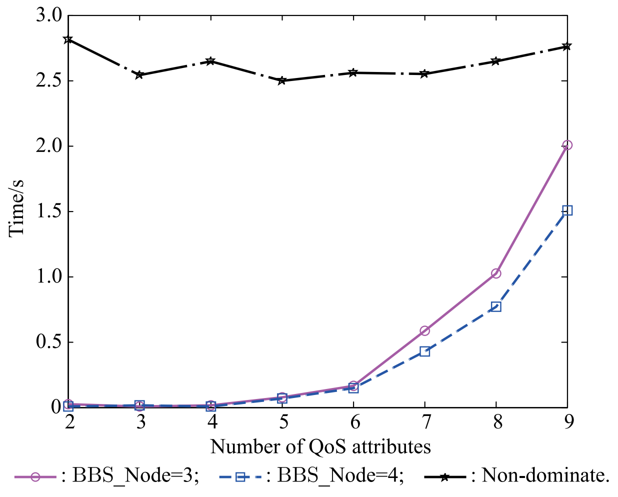


Fig. 4 Calculation time with the number of QoS attributes

As shown in Fig. 4, the computation time increases with the increase of the QoS attributes. When the number of QoS attributes is less than 6, the computation time can be within 0.3 s. However, when the QoS attributes are more that six, the BBS method performs worse because the performance of the R tree decreases in the high dimensional space. In contrast, the calculation time of the non-dominate method is stable, and its average time is around 2.6 s, but is worse than the BBS method.

Let the number of QoS attributes be 5, the number of candidate services increases from 250 to 2 500, and the step size is 250. The computation time is shown in Fig. 5.

As shown in Fig. 5, with the increase of the candidate services, the computation time of the BBS method is stable, while that of the non-dominate method increases significantly. It is because the time complexity of the non-dominate method is $O(|N\_\text{candidate}|^2 \cdot |\text{ATTRIBUTE}|)$, thus there is an exponential growth of the running time with the increase of $|N\_\text{candidate}|$.

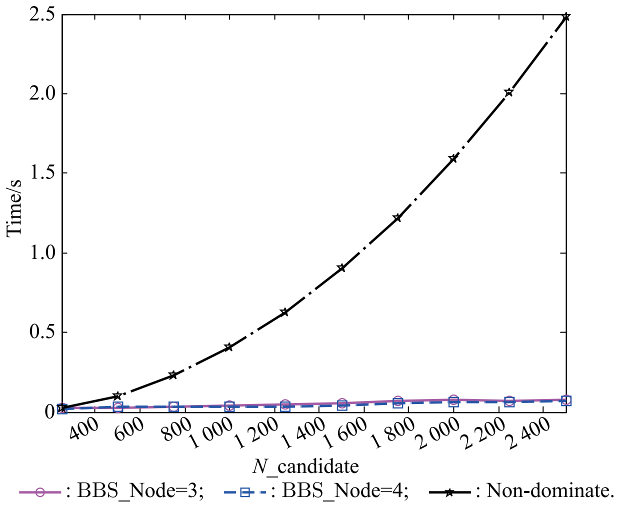


Fig. 5 Calculation time with the number of candidate services

Therefore, the Skyline computing methods can be selected flexibly. When the number of QoS attributes is over 9, we can choose the non-dominate method to get a higher efficiency. When there are a huge number of candidate services, the BBS algorithm can calculate the Skyline set more quickly.

5.3 Comparison of service composition based on Skyline

The Bonmin method is used to calculate the original services set and the filtered Skyline set separately, and the computation time is shown in Fig. 6.

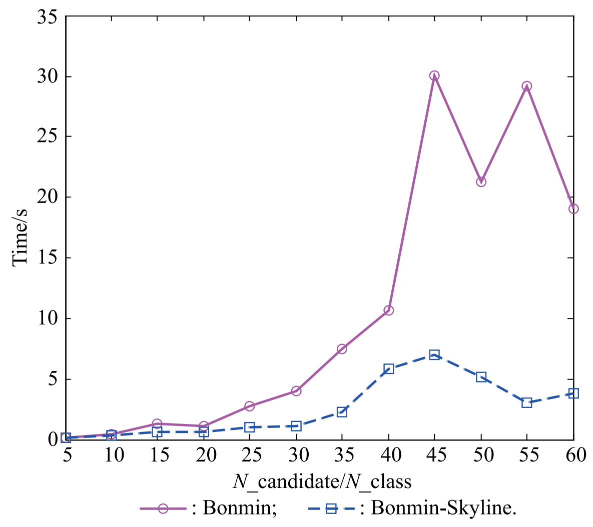


Fig. 6 Bonmin method on the original set and the Skyline set

As shown in Fig. 6, when the number of candidate services in a group is less than 40, the efficiency of the two methods is close. However, when it is over 40, the time of the Skyline operator can decrease by at least 3/4. Therefore, the Bonmin method base on Skyline can effectively improve the efficiency of service composition, while ensuring the quality of solution.

6. Conclusions

With the development of cloud computing, the number of services is exponentially increasing, which brings new challenges for the existing service composition technology. In order to find out a service composition solution satisfying all the functional requirements as quickly as possible, this paper proposes a nonlinear service composition method based on the Skyline operator. The dominated services in each service group would be filtered out by Skyline, which can reduce the search space and improve the efficiency effectively. Then the service composition problem is formulated as a nonlinear integer programming problem by AMPL, and solved by Bonmin. The experimental results show that the proposed method can obtain the optimal solution similar to the exhaustive algorithm, and the efficiency is also excellent. Therefore, the service composition method based on Skyline can significantly improve the efficiency of service composition.

In the future work, we will conduct the service composition in the parallel architecture to further improve the efficiency.

References

[1] WANG P W, MENG J, CHEN J C, et al. Smart contract-based negotiation for adaptive QoS-aware service composition. IEEE Trans. on Parallel Distributed Systems, 2019, 30(6): 1403–1420.

[2] YE D Y, HE Q, WANG Y C, et al. An agent-based integrated self-evolving service composition approach in networked environments. IEEE Trans. on Services Computing, 2019, 12(6): 880–895.

[3] JATOTH C, GANGADHARAN G R, FIORE U, et al. QoS-aware big service composition using MapReduce based evolutionary algorithm with guided mutation. Future Generation Computer Systems, 2018, 86: 1008–1018.

[4] BORZSONY S, KOSSMANN D, STOCKER K. The Skyline operator. Proc. of the 17th International Conference on Data Engineering, 2001: 421–430.

[5] ALRIFAI M, RISSE T. Combining global optimization with local selection for efficient QoS-aware service composition. Proc. of the 18th International Conference on World Wide Web, 2009: 881–890.

[6] ZENG L Z, BENATALLAH B, NGU A H H, et al. QoS-aware middleware for web services composition. IEEE Trans. on Software Engineering, 2004, 30(5): 311–327.

[7] ARDAGNA D, PERNICI B. Adaptive service composition in flexible processes. IEEE Trans. on Software Engineering, 2007, 33(6): 369–384.

[8] MISTRY S, BOUGUETTAYA A, DONG H, et al. Metaheuristic optimization for long-term IaaS service composition. IEEE Trans. on Services Computing, 2016, 11(1): 131–143.

[9] ZHANG A, SUN H Y, TANG Z L, et al. Service composition based on discrete particle swarm optimization in military organization cloud cooperation. Journal of Systems Engineering and Electronics, 2016, 27(3): 590–601.

[10] YU Q, CHEN L, LI B. Ant colony optimization applied to web service compositions in cloud computing. Computers and Electrical Engineering, 2015, 41: 18–27.

[11] HUO Y, ZHUANG Y, GU J J, et al. Discrete gbest-guided artificial bee colony algorithm for cloud service composition. Applied Intelligence, 2015, 42(4): 661–678.

[12] HUO Y, ZHUANG Y, GU J J, et al. Elite-guided multi-objective artificial bee colony algorithm. Applied Soft Computing, 2015, 32(1): 199–210.

[13] PAPADIAS D, TAO Y F, FU G, et al. An optimal and progressive algorithm for skyline queries. Proc. of the ACM SIGMOD International Conference on Management of Data, 2003: 467–478.

[14] ALRIFAI M, SKOUTAS D, RISSE T. Selecting Skyline services for QoS-based web service composition. Proc. of the 19th International Conference on World Wide Web, 2010: 11–20.

[15] YU Q, BOUGUETTAYA A. Computing service Skyline from uncertain qows. IEEE Trans. on Services Computing, 2010, 3(1): 16–29.

[16] WANG S G, SUN Q B, ZOU H, et al. Particle swarm optimization with skyline operator for fast cloud-based web service composition. Mobile Networks and Applications, 2013, 18(1): 116–121.

[17] MORADI M, EMADI S. Reducing the calculations of quality-aware web services composition based on parallel Skyline service. International Journal of Advanced Computer Science and Applycations, 2016, 7(7): 306–311.

[18] ZHANG F, HWANG K, KHAN S U, et al. Skyline discovery and composition of multi-cloud mashup services. IEEE Trans. on Services Computing, 2016, 9(1): 72–83.

[19] YAN G, WANG S G, WONG K S, et al. Skyline service selection approach based on QoS prediction. International Journal of Web and Grid Services, 2017, 13(4): 425–447.

[20] DEB K, PRATAP A, AGARWAL S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. on Evolutionary Computation, 2002, 6(2): 182–197.

[21] GUTTMAN A. R-trees: a dynamic index structure for spatial searching. SIGMOD Record, 1984, 14(2): 47–57.

[22] GAY D M. The AMPL modeling language: an aid to formulating and solving optimization problems. Springer Proceedings in Mathematics & Statistics: Numerical analysis and optimization. Berlin: Springer, 2015, 134: 95–116.

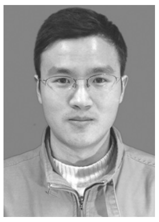
[23] AL-MASRI E, MAHMOUD Q H. QoS-based discovery and ranking of web services. Proc. of the 16th International Conference on Computer Communications and Networks, 2007: 529–534.

[24] AL-MASRI E, MAHMOUD Q H. Discovering the best web service. Proc. of the 16th International Conference on World Wide Web, 2007: 1257–1258.

[25] AL-MASRI E, MAHMOUD Q H. Investigating web services on the world wide web. Proc. of the 17th International Conference on World Wide Web, 2008: 795–804.

Biographies

HUO Ying was born in 1988. She received her Ph.D. degree from College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics in 2016. Now she is an associate professor at the School of Computer Engineering, Nanjing Institute of Technology. Her research interests include service composition and intelligent computing.
E-mail: huoy@njit.edu.cn



ZHANG Jiande was born in 1982. He received his Ph.D. degree in mechanical engineering from Nanjing Tech University. Now, he is an associate professor at the School of Computer Engineering, Nanjing Institute of Technology. His research interests include intelligent optimization algorithm, image measurement and pattern recognition.
E-mail: zhangjd@njit.edu.cn