

Article

A Multi-Criteria Multi-Cloud Service Composition in Mobile Edge Computing

Beibei Pang ^{1,2}, Fei Hao ^{1,2,*} , Doo-Soon Park ³ and Carmen De Maio ⁴

¹ Key Laboratory of Modern Teaching Technology, Ministry of Education, Xi'an 710062, China; pangbb@snnu.edu.cn

² School of Computer Science, Shaanxi Normal University, Xi'an 710119, China

³ Department of Computer Software Engineering, Soonchunhyang University, Asan 31538, Korea; parkds@sch.ac.kr

⁴ Department of Information Engineering, Electrical Engineering and Applied Mathematics, University of Salerno, Fisciano 84084, Italy; cdemaio@unisa.it

* Correspondence: feehao@gmail.com; Tel.: +86-29-8531-0161

Received: 17 July 2020; Accepted: 11 September 2020; Published: 16 September 2020



Abstract: The development of mobile edge computing (MEC) is accelerating the popularity of 5G applications. In the 5G era, aiming to reduce energy consumption and latency, most applications or services are conducted on both edge cloud servers and cloud servers. However, the existing multi-cloud composition recommendation approaches are studied in the context of resources provided by a single cloud or multiple clouds. Hence, these approaches cannot cope with services requested by the composition of multiple clouds and edge clouds jointly in MEC. To this end, this paper firstly expands the structure of the multi-cloud service system and further constructs a multi-cloud multi-edge cloud (MCMEC) environment. Technically, we model this problem with formal concept analysis (FCA) by building the service–provider lattice and provider–cloud lattice, and select the candidate cloud composition that satisfies the user's requirements. In order to obtain an optimized cloud combination that can efficiently reduce the energy consumption, money cost, and network latency, the skyline query mechanism is utilized for extracting the optimized cloud composition. We evaluate our approach by comparing the proposed algorithm to the random-based service composition approach. A case study is also conducted for demonstrating the effectiveness and superiority of our proposed approach.

Keywords: mobile edge computing; multi-cloud multi-edge cloud; formal concept analysis; skyline

1. Introduction

The rapid development of cloud computing has gradually facilitated the popularity of data-intensive services. The issue of cloud service portfolios has attracted much attention in recent years from both industry and academia. Cloud computing environments have also become the best computational paradigm for satisfying the requirements of users by using multiple resources simultaneously.

To fulfill user requirements, the process of service execution on clouds is usually modeled with a multi-provider multi-service system architecture. After users' service requests, multiple clouds and services provided by multiple providers can be combined into a set of interacting services to provide a cloud-based solution that meets specific quality standards [1].

A centralized approach to big data processing, with the cloud computing model as its core, centralizes and unifies the management of a large number of resources. The dynamic allocation of the requests is efficient and adapted to business needs. However, the fully centralized model suffers from

network latency and energy consumption problems during data transfer and processing. However, its key technology cannot meet the requirements of strong real-time performance and large amounts of data interaction and processing [2,3]. To address these disadvantages, a distributed computing paradigm is emerging nowadays, mobile edge computing (MEC), which brings computation and data storage closer to the location needed to improve response times and save bandwidth.

An agent-based approach has been proposed to compose services in a multi-cloud environment [4]. Different kinds of services (e.g., virtualization services, vertical services, etc.) are considered. Researchers have adopted a semi-recursive contractual network protocol, but it still had some limitations in terms of processing and communication costs. Yu et al. [5] proposed two cloud composition algorithms, called Greedy-WSC (web service composition) and ACO-WSC (Ant Colony Optimization for web service composition) to obtain a valid service composition with the minimum number of clouds. Mezni et al. [6] utilized the clustering capabilities of formal concept analysis (FCA) to filter relevant providers to find the minimal set of clouds that hold the optimal services. However, these works discussed above clearly do not explore the issue of cloud service composition in an MEC environment.

Edge computing [7] is the fusion of network, computation, storage, and application core capabilities on the edge of the network near the data source. A distributed open platform is used to provide edge intelligence services near the edge to meet the industry's digital needs for agile connectivity, business implementation, and data optimization. It processes data at the edge of the network by pushing applications, data, and services from centralized nodes to the edge of the network, where edge computing can effectively shorten response time, improve processing efficiency, and reduce network traffic so as to meet real-time requirements. Li et al. [8] proposed a skyline community model for capturing communities not dominated by the other communities in a multi-valued network. In order to accommodate the need for a composition of cloud services in real-life mobile edge computing, our work targets the construction of multi-cloud multi-edge cloud environments to recommend the cloud/edge cloud composition that can provide the required services to the target users.

However, integrating the distributed services across multiple clouds raises several issues, such as the inter-cloud communication cost, the increase of monetary cost, the security and privacy issues, and so forth. Hence, a challenging task is to properly select the cloud composition with the considerations of services' constraints. The main contributions of this paper are summarized as follows:

- **Novel Problem Formulation:** Mobile edge computing is a key emerging technology in the 5G era, providing cloud computing while users are at the edge of the network, and it expands the architecture of cloud service composition recommendation systems. This paper pioneers a novel problem on energy-efficient multi-cloud composition in MEC.
- **FCA-Based Modeling:** This paper proposes a multi-cloud multi-edge cloud (MCMEC) environment by using FCA. We describe the MCMEC with a set of clouds and edge clouds, a group of providers, and their offered services. Importantly, two kinds of concept lattices are generated by FCA theory for better characterizing the hosting relationships between them.
- **Skyline-Based Cloud Composition Optimization:** By presenting a skyline query in a multi-value environment and an approach for skyline layer construction, we devise an algorithm for determining the optimal cloud composition with the joint considerations of energy, network latency, and cost.

The rest of this paper is organized as follows. Section 2 details the preliminary knowledge of FCA and skyline queries. The multi-cloud multi-edge cloud service composition problem is provided in Section 3. Section 4 presents our approach/algorithm and the way that FCA and the skyline query are applied to the multi-cloud multi-edge cloud service composition problem. Section 5 provides the performance evaluation and a case study for revealing the effectiveness of the proposed algorithm. The related work is surveyed in Section 6. Finally, Section 7 concludes the paper and outlines the future work.

2. Preliminaries

This section mainly presents the preliminary knowledge about FCA [9–11] and skyline queries [12,13], which we are going to utilize for addressing our problem in this paper. FCA is often used in the field of data analysis as a classic computational intelligence technique. A formal concept is defined to characterize the relationships between objects and attributes in a domain. Specifically, the objects and attributes are grouped into concepts, and then a conceptual hierarchy of the concepts can be constructed.

Definition 1. Formal Context. A formal context is a triple $K = (O, A, R)$ where O denotes the set of objects, A indicates the set of attributes, and R is the set of binary relations between the objects and the attributes ($R = O \times A$). If $(x, a) \in R$, we know x has a relationship with attribute a , which could be defined the value as “1”, or “0” otherwise.

Definition 2. Formal Concept. Given a formal context $K = (O, A, R)$, a formal concept is represented as a pair of sets (E, I) such that $E \subseteq O$ and $I \subseteq A$. There are two parts in the grouping of the formal concept: The extent E congregates the objects that share a sub-set of common attributes, and the intent I is formed by these common attributes shared by all objects in E .

Definition 3. Concept Lattice. Given a formal context $K = (O, A, R)$, a concept lattice $L = (C(K), \leq)$ can be obtained by all formal concepts $C(K)$ of a context K with the partial order relation \leq . Its graphical representation is a Hasse diagram.

Definition 4. Dominance Relationship. If a point $p = \{p_1, p_2, \dots, p_n\}$ dominates a point $q = \{q_1, q_2, \dots, q_n\}$, it donates $p < q$ satisfying $\forall i \in \{1, 2, \dots, n\}, p_i \leq q_i, \forall j \in \{1, 2, \dots, n\}, p_j < q_j$. In other words, it shows that in any one dimension, values will be lower than or equal to the point q in the corresponding dimension value, while there is at least in one dimension in which the p value is superior to the q value.

Definition 5. Skyline. The skyline consists of the points that are not dominated by other points in the given dataset.

Example 1. Table 1 illustrates a set of points $G = \{g_1, g_2, \dots, g_9\}$ and their corresponding two-dimensional values. Clearly, we utilize the definition of the dominance relationship to obtain their relationships that exist $g_4 < g_1, g_4 < g_2, g_4 < g_3, g_8 < g_5, g_8 < g_6, g_8 < g_7, g_8 < g_9$. So, the skyline contains $\{g_4, g_8\}$.

Table 1. A set of points.

G	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9
d_1	14	27	36	8	25	40	36	16	20
d_2	500	390	300	260	250	200	170	80	120

3. Problem Statement

MEC architecture is mainly composed of edge cloud servers and cloud servers. As locations move, users who want to access the same services may switch to different cloud platforms in order to decrease energy consumption and shorten the time delay of transaction. In this section, the main purpose of this paper is to extract the cloud composition with the optimal reliability and service performance based on the service requests given by the user in an MCMEC environment to complete the response to the user.

First of all, some key definitions of MEC, service requests, energy consumption between clouds, and prices of clouds are formally defined. Then, the problem formalism is presented mathematically.

3.1. Basic Definitions

Definition 6. Multi-Cloud and Multi-Edge Cloud Environment. A multi-cloud and multi-edge cloud environment (MCMEC) is composed of a set of clouds $C = \{C_1, C_2, \dots, C_m\}$ and a set of edge clouds $EC = \{EC_1, EC_2, \dots, EC_n\}$, where $C_i/EC_j (1 \leq i \leq m, 1 \leq j \leq n)$ is a cloud/edge cloud that hosts a set P of providers, $P \subseteq \{P_1, P_2, \dots, P_h\}$. A provider may belong to more than one cloud or edge cloud, and it offers a set of services, $S \subseteq \{S_1, S_2, \dots, S_k\}$.

Example 2. Table 2 shows an example of the MCMEC environment, including 40 services with different functional capabilities that are offered by 10 providers hosted in the set of clouds and edge clouds. The cloud C_1 contains three providers that offer 11 services. Meanwhile, many providers might be hosted by some different clouds (e.g., providers P_1, P_2, P_3, P_6, P_8 , and P_{10}), and several services might be offered by different providers.

Table 2. A typical case of a multi-cloud multi-edge cloud (MCMEC) environment.

Clouds	C_1				C_2		C_3			C_4			
Providers	P_1	P_2	P_6	P_2	P_6	P_7	P_8	P_{10}	P_3	P_4	P_5	P_9	
Services	5	2	4	2	6	11	3	5	7	2	3	8	
Edge Clouds	EC_1		EC_2		EC_3		EC_4		EC_5		EC_6		EC_7
Providers	P_1	P_2	P_3	P_6	P_7	P_8	P_{10}	P_1	P_8	P_3	P_9	P_{10}	
Services	5	2	7	4	11	3	5	5	6	7	8	5	

Definition 7. Service request. A user's service request is defined as a composition of n services, i.e., $\mathbb{R} = \{S_1, S_2, \dots, S_n\}$. There are several candidate compositions of clouds that enable providers to fulfill the requested service at the same time.

From a practical point of view, most users will consider the price and quality of service (QoS) attributes of the clouds when purchasing cloud services; thus, this work will measure the excellence of the obtained candidate cloud compositions in terms of network latency, cloud price, and energy consumption.

To model the communication network between clouds/edge clouds, we formulate the MCMEC as a weighted undirected graph $G = (V, E, W, T)$ where $V = C \cup EC$ is a set of clouds and edge clouds, $E \in V \times V$, which is the set of communications between cloud/edge clouds, and $C_i \in V, C_j \in V$.

Definition 8. Energy Consumption between Clouds. The set of energy consumption caused by the communication between two clouds is expressed as W . We use \hat{W} to denote the energy consumption of a cloud composition obtained from the MCMEC. Its consumption can be categorized as follows.

- (1) The energy consumption for cloud-only composition is calculated by

$$\hat{W} = \sum_{1 \leq i, j \leq m} w(C_i, C_j). \quad (1)$$

- (2) The energy consumption for edge-cloud-only composition is calculated by

$$\hat{W} = \sum_{1 \leq i, j \leq n} w(EC_i, EC_j). \quad (2)$$

- (3) Cloud composition is composed of both clouds and edge clouds, where the energy consumption includes the sum of the energy consumption between each cloud, the sum of the energy consumption between each edge cloud, and the sum of the energy consumption between the cloud and edge cloud. Its energy consumption is represented as:

$$\hat{W} = \sum_{1 \leq i, j \leq m} w(C_i, C_j) + \sum_{1 \leq i, j \leq n} w(EC_i, EC_j) + \sum_{1 \leq i \leq m, 1 \leq j \leq n} w(C_i, EC_j). \quad (3)$$

Definition 9. Monetary Cost of Cloud Sets. Monetary expense is a necessary consideration for people in choosing a composition of clouds. Therefore, the price of each cloud/edge cloud will be investigated and listed to work out the monetary cost of a cloud set.

Let us assume that the candidate cloud composition is C_i, \dots, EC_j . Its monetary cost is calculated as follows.

$$\hat{P} = \sum_{1 \leq i \leq m} P_{C_i} + \sum_{1 \leq j \leq n} P_{EC_j}, \quad (4)$$

where P_{C_i} is the price of cloud C_i and P_{EC_j} is the price of edge cloud EC_j .

Definition 10. Network Latency of Cloud Sets. Adding up all the values in the network latency matrix composed of cloud and edge clouds is the network latency of the cloud composition. It is formulated as

$$\hat{T} = \sum_{1 \leq i \leq m, 1 \leq j \leq n} t(C_i, EC_j). \quad (5)$$

3.2. Problem Description

In this work, the candidate cloud/edge cloud compositions are firstly obtained through the FCA model once a user's request comes into the MCMEC environment. When we want to select proper multi-cloud composition, three dimensions are jointly considered, i.e., energy, monetary cost, and network latency.

Skyline querying is a very important point query technique in multidimensional database.s Here is a motivating example: "Nassau Hotel" is based on the two objectives of "cheap" and "close to the beach", assuming that when visiting Nassau Beach, most tourists must first look for a cheap hotel close to the beach. However, hotels that are generally closer to the beach are more expensive, so there will not be a definitive result, only a series of results returned for the user to choose from. The distances and prices of these hotels are not "worse" than those of other hotels, i.e., there are no points closer to the same price. Therefore, our problem can be regarded as a skyline problem.

The multi-criteria, multi-cloud, and multi-edge cloud service composition problem is described as: In an MCMEC environment, $G = (V, E, W, T)$; when a service request \mathbb{R} is given, the target of the problem is to recommend cloud/edge cloud composition \tilde{CC} with the multi-criteria optimizations, i.e., minimized monetary cost $Min(\hat{P})$, minimized energy consumption $Min(\hat{W})$, and minimized network latency $Min(\hat{T})$.

Service request \mathbb{R} is an input into the concept lattices of the MCMEC to acquire the providers and multi-cloud/edge cloud compositions. We consider some key factors like price, energy consumption, and network latency related to an optimal cloud composition. Obviously, our problem is viewed as a multi-criteria optimization problem [14]. The coordinate system is established for every two factors, and the cloud composition obtained in the previous step is subjected to skyline layer computation; then, the cloud/edge cloud composition should minimize monetary cost $Min(\hat{P})$, energy consumption $Min(\hat{W})$, and network latency $Min(\hat{T})$. Therefore, the intersection of the first layer is the optimal cloud combination we are looking for.

In the MCMEC environment, a solution to the selection of recommended clouds based on the skyline query is attempted. This approach considers energy consumption, price, and network latency to be equally important, and skyline hierarchical calculations select the optimal cloud composition under multi-factor optimization, successfully achieving trade-offs between different optimization goals.

4. Proposed Approach

Before clarifying the concrete steps of FCA-based candidate cloud/edge cloud composition and skyline layer computation for obtaining the optimal cloud/edge cloud composition, the framework of our problem is first introduced in this section.

4.1. Big Picture

Figure 1 shows a framework of energy-efficient multi-cloud service composition in mobile edge computing.

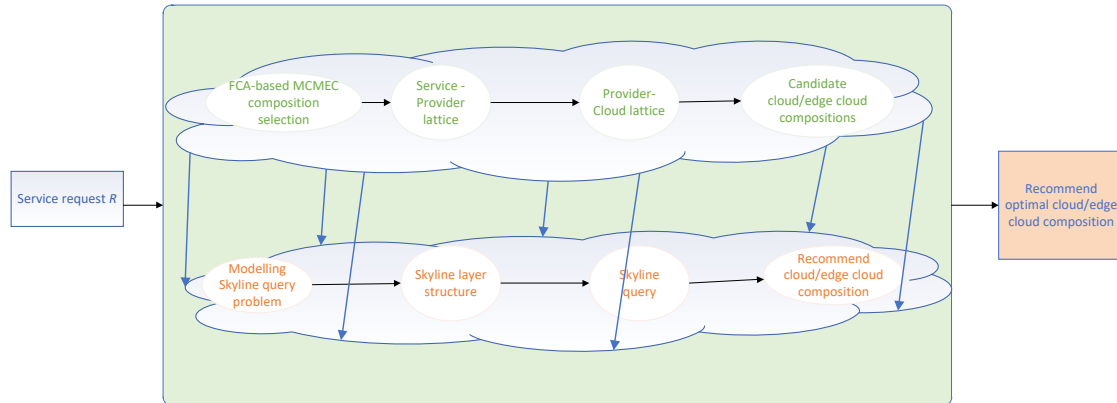


Figure 1. A framework of energy-efficient multi-cloud service composition in mobile edge computing.

The main steps for screening and recommending multi-cloud multi-edge cloud compositions are laid out in this section. A brief description is given below.

- **Step 1: FCA-based MCMEC composition selection:** This step mainly applies FCA theory to construct a Service–Provider lattice and Provider–Cloud lattice stemming from the formal contexts of the MCMEC. When a user makes a service request \mathbb{R} , the Service–Provider lattice will filter candidate providers who provide acquired services according to the relationship between service and provider. Then, depending on the provider compositions, the cloud compositions in the Provider–Cloud lattice that have those providers are also acquired. With this step, we can get candidate cloud/edge cloud compositions that are likely to satisfy the customer’s request.
- **Step 2: Skyline query for cloud/edge cloud recommendation:** In the previous step, we obtained the candidate cloud/edge cloud compositions. For each cloud combination, three values can be calculated from each of the three dimensions of energy consumption, price, and network latency. This implies that a list of cloud/edge cloud compositions and properties can be obtained. After that, we take the values of each two dimensions and make the skyline layer computation, and then we interact their first layer, which results in an optimal cloud/edge cloud composition.

4.2. FCA-Based Multi-Cloud Service Composition Selection

Two types of formal contexts that characterize the connection between the providers and the services provided by the providers in a cloud, as well as the relationship between all cloud platforms and the resident providers, respectively, are built initially. Next, a concept lattice generation algorithm [15,16] is utilized to construct the Service–Provider lattice and Provider–Cloud lattice.

Example 3. Assume that there are four cloud/edge cloud hosting 10 providers, in which the Service–Provider lattice and Provider–Cloud lattice are modeled as follows. We take the clouds/edge clouds $\{C_1, C_2, EC_1, EC_2\}$ as the objects and the providers $\{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$ as the attributes in the formal context, and then the concept lattice (Provider–Cloud lattice) is generated using the concept lattice generation algorithm, as shown in Figure 2. Similarly, we take the services as the objects and the providers as the attributes in the formal context, and further construct the Service–Provider lattice of Cloud 1, which is shown in Figure 3.

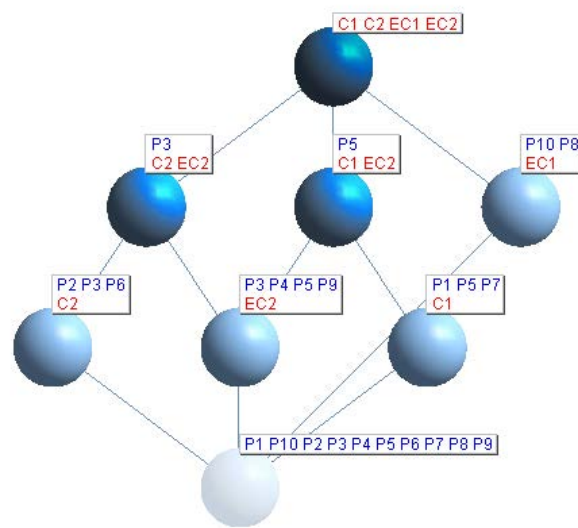


Figure 2. Provider–Cloud lattice.

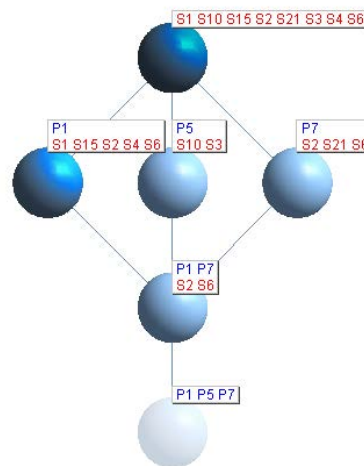


Figure 3. Service–Provider lattice.

After obtaining the Service–Provider lattice of each cloud, the service request \mathbb{R} enters the first Service–Provider lattice appropriately. After that, \mathbb{R} is checked with the extent of each concept c_j for matching. While intersection is not empty, the corresponding intent of c_j is deposited into a collection with the intersection, removing it from \mathbb{R} . All concepts are then iterated until all individuals of \mathbb{R} get paired. The obtained candidate provider compositions are deemed as input to the Provider–Cloud lattice. Similarly to the former process, we make the intersection operation on candidate provider compositions and concept c_j . The intents that have the set of providers will come into a collection, and we can gain the candidate cloud/edge cloud compositions.

According to the algorithm of previous works [17,18], we assume that $\mathbb{R} = \{S_2, S_{10}, S_{18}, S_{23}\}$, as well as that the algorithm can be applied to obtain candidate provider combinations $\{\{P_1, P_5, P_6, P_8\}, \{P_1, P_5, P_6, P_{10}\}, \{P_1, P_6, P_9, P_8\}, \{P_1, P_6, P_9, P_{10}\}, \{P_5, P_7, P_6, P_8\}, \{P_6, P_7, P_9, P_{10}\}, \{P_5, P_6, P_7, P_{10}\}\}$ and to further obtain candidate service combinations $\{\{C_1, C_2, EC_1\}, \{C_1, C_2, EC_1, EC_2\}, \{C_1, C_2, EC_1, EC_2\}\}$.

4.3. Skyline Layer Construction

For each cloud composition, three values can be calculated from each of the three dimensions of energy consumption, price, and network latency, which means that a list of cloud/edge cloud compositions and properties can be obtained. Inspired by the classic skyline hotel problem, our

solution idea is to treat the cloud/edge cloud composition as a point where properties can be used to build coordinate systems. In this way, the cloud/edge-cloud composition optimization problem is abstracted to a skyline problem.

We take the values of every two dimensions and make the skyline layer computation, then repeat the above steps. Finally, we interact with the first layer of each time, which results in optimal cloud/edge cloud composition. We are obviously in possession of candidate cloud/edge cloud compositions that are treated as points. For example, the points below $g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9$ represent candidate cloud/edge cloud compositions $CC_1, CC_2, CC_3, CC_4, CC_5, CC_6, CC_7, CC_8, CC_9$, respectively. Based on the data of price (d_1), energy consumption (d_2), and network latency (d_3) in the MCMEC, we figure out the multi-valued attributes of points in Table 3. Moreover, a set of two-dimensional data points for representing cloud/edge cloud compositions is represented in Figure 4.

Table 3. Multi-valued attributes of candidate cloud/edge cloud compositions.

G	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	g_9
d_1	14	27	36	8	25	40	36	16	20
d_2	500	390	300	260	250	200	170	80	120
d_3	7	8	6	6	9	7	5	4	8

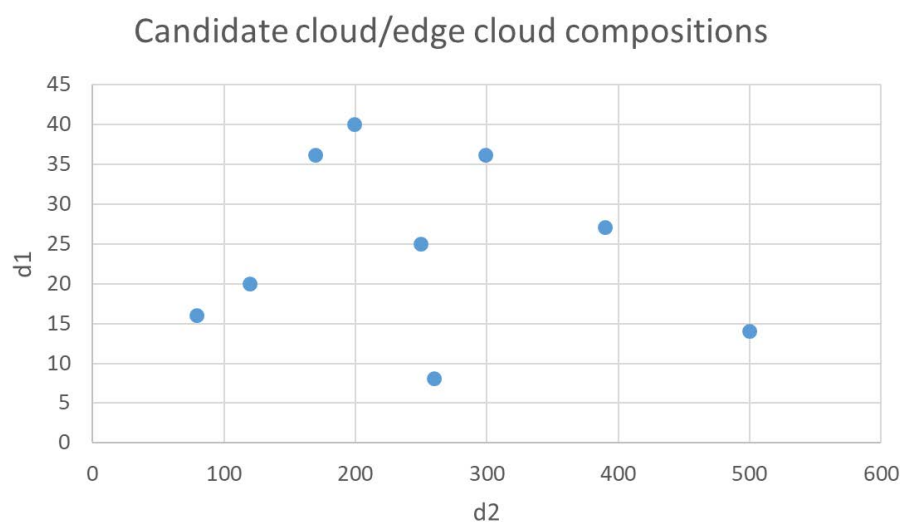


Figure 4. A set of two-dimensional data points for representing cloud/edge cloud composition.

Definition 11. Skyline Layer Computation. A skyline layer structure (SLS) is introduced to organize objects and represent dominance relations. A skyline layer structure is introduced to group objects and indicates dominance relations. Given a set of objects P in two-dimensional space, a skyline object of P is enclosed in the skyline layer L_1 , i.e., $L_1 = \text{Skyline}(P)$. The second skyline layer, L_2 , maintains the skyline object of P with the removal of the object in L_1 , i.e., $L_2 = \text{Skyline}(P - L_1)$. Basically, the i th skyline layer consists of what dominates $P - \sum_{k=1}^{i-1} L_k$. Repeat the above process iteratively until all points in P are assigned to the skyline layer structure. Figure 4 demonstrates the skyline layer structure of the object in Table 1.

We considered three dimensions, as presented in Table 3. Let us take Figure 4 as an example; d_1, d_2 are selected as a given two-dimensional space. Each point is compared with the other points in turn, and if the values of both dimensions match the dominance relationship, the skyline object g_4, g_8 of the entire set of points is derived, which forms the skyline layer L_1 . After removing the points of L_1 , we repeat the above steps to select g_1, g_9 , which form the skyline layer L_2 . Similarly, the skyline layer L_3 and skyline layer L_4 are constructed. Eventually, the directed skyline graph is constructed, as shown in Figure 5.

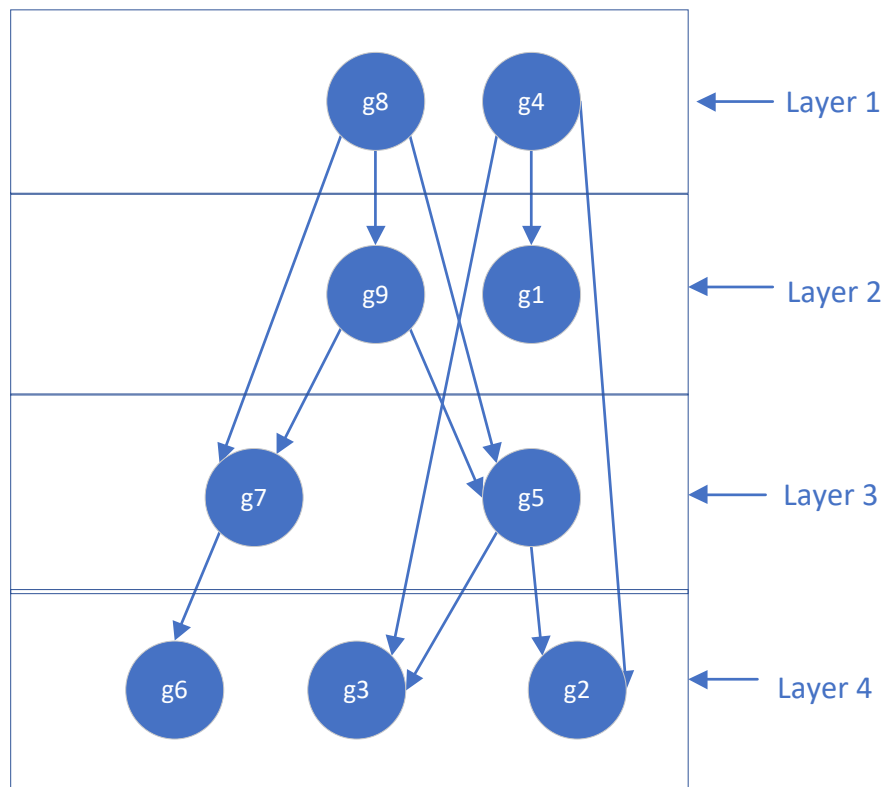


Figure 5. Directed skyline graph.

The above process demonstrates the two-dimensional space d_1, d_2 skyline layer construction; however, this cloud/edge cloud composition optimized recommendation must be conducted over a three-dimensional space. Hence, we adopt the same method to perform skyline layer construction for the d_2, d_3 dimensional point set and d_1, d_3 dimensional point set. Finally, three directed skyline graphs are easily obtained.

4.4. Multi-Cloud Service Composition Recommendation Based on a Skyline Query

In the previous section, three directed skyline graphs are obtained. It can be seen from the previous example that the points of Layer 1 in SLS_1 keep the monetary cost smaller, but no other point of energy consumption is less than it, or energy consumption is smaller, but no other point of monetary cost is less than it in some context when price and energy consumption are equally important. Skyline Layer 1 is what we need and is optimal, satisfying the needs of users with multiple properties. Taking out the points of the first layer, candidate cloud/edge cloud compositions are reduced from 9 to 2 numerically. For instance, g_8, g_4 are probably the best choice, g_4 has the lowest price with relatively low energy consumption, while the g_8 has both minimal monetary cost and energy consumption.

However, SLS_1 is extracted when the factors of price and energy consumption are considered as a skyline layer computing structure. In order to achieve the trade-off between price, energy consumption, and network latency, the point sets of the first layers in SLS_1, SLS_2 , and SLS_3 obtained in the previous section all become the new candidate cloud/edge cloud compositions, and then they are brought out together for an intersection. Subsequently, the final recommended cloud/edge cloud compositions are the points that repeat the most after the intersection. This can be illustrated briefly by the data of Table 3.

Based on the way of constructing the skyline layers in Algorithm 1, three SLSs were obtained. The hierarchy of SLS_1 is shown in Figure 4. The first layer of SLS_1 has two points at g_4, g_8 , which carry forward the advantages of the skyline query theory. It performs better in both the d_1 and d_2 dimensions. Similarly, the first layer of SLS_2 is constructed to be g_4 and g_8 in the dimensions of d_2 and

d_3 , and the first layer of SLS_3 contains a point of g_8 below the dimensions between d_1 and d_3 . Three lists, respectively expressed as $\{g_4, g_8\}$, $\{g_4, g_8\}$, and $\{g_8\}$, would be intersected to search for the one that repeats the most. Consequently, the optimal cloud/edge cloud composition is g_8 .

It is necessary to compare multiple QoS attributes of all candidate cloud/edge cloud compositions when recommending cloud/edge cloud compositions. The approach of skyline layer construction eliminates redundant candidate points, which obviously reduces the selection space.

Furthermore, the skyline embodies the trade-off between diverse factors of cloud composition. Skyline cloud/edge cloud compositions are not comparable without explicitly giving priority to QoS attributes. The results of our methodological selection preserve more valuable QoS information. With the intersection of different new candidate cloud/edge cloud compositions, the user's multi-dimensional QoS requirements can be satisfied in all aspects.

Based on the FCA and skyline query theories, we finally calculate and recommend the optimal cloud/edge cloud composition for current users via Algorithm 1.

Algorithm 1: Optimal Cloud/Edge Cloud Composition Recommendation Algorithm

Input: MCMEC, a weighted graph $G = (V, E, W, T)$;
 A service request \mathbb{R} ;
 A set of cloud prices A ;
 A user U ;
Output: Recommended cloud/edge cloud composition (\widetilde{CC})

- 1 Construct Service–Provider formal context and Provider–Cloud formal context from MCMEC;
 Build Service–Provider lattice L_1 and Provider–Cloud lattice L_2 ;
- 2 Extract Candidate cloud/edge cloud compositions;
- 3 **while** (d_1 and d_2)
- 4 **begin**
- 5 Construct directed skyline graph for cloud/edge cloud composition CC_k
- 6 **return** SLS_1 ;
- 7 **end**
- 8 **while** (d_1 and d_3)
- 9 **begin**
- 10 Construct directed skyline graph for cloud/edge cloud composition CC_k
- 11 **return** SLS_2 ;
- 12 **end**
- 13 **while** (d_2 and d_3)
- 14 **begin** Construct directed skyline graph for cloud/edge cloud composition CC_k
- 15 **return** SLS_3 ;
- 16 **end**
- 17 $L_{11} = \text{getL1}(SLS_1)$;
- 18 $L_{21} = \text{getL1}(SLS_2)$;
- 19 $L_{31} = \text{getL1}(SLS_3)$;
- 20 $\widetilde{CC} = L_{11} \cap L_{21} \cap L_{31}$
- 21 **return** \widetilde{CC}

The above algorithm works as follows. It takes MCMEC G , a service request \mathbb{R} , a set of cloud prices A , and a user U as the inputs of the algorithm. The output of the algorithm is cloud/edge cloud composition \widetilde{CC} . Lines 1–2 are used to construct the Service–Provider formal context and Provider–Cloud formal context, and then generate the corresponding concept lattices L_1 and L_2 , respectively. Line 3 is to obtain the possible candidate cloud/edge cloud composition. Lines 4–17 are in charge of constructing the directed skyline graphs under different combinations of two-dimensional spaces. Then, the algorithm goes to extract the first layers from the obtained directed skyline graphs

(Lines 18–20). Line 21 is to make the intersection among the first layers L_{11}, L_{21}, L_{31} . After that, it finally returns the optimized cloud/edge cloud composition \widehat{CC} (Line 22).

5. Evaluation

In this section, we evaluate the effectiveness of the proposed approach through comparison experiments and a concrete case study.

5.1. Evaluation

To demonstrate the effect of the approach proposed in Section 4, we adopt a random approach to select the cloud composition for the comparison with our approach. To guarantee the fairness and stability of the random approach, our experiment adopts the data in Table 3 to yield the 10 random cloud compositions $g_6, g_1, g_6, g_8, g_4, g_5, g_1, g_1, g_1$, as shown in Table 4, and evaluate their performance by calculating the average values of these cloud compositions in terms of different dimensions d_1, d_2 , and d_3 , as shown in Table 5. Figure 6 depicts the comparison results of two approaches. It is easily found that our approach can significantly reduce the monetary cost, latency, and energy consumption compared to the random approach.

Table 4. Ten random cloud compositions.

	d_1	d_2	d_3
g_6	40	200	7
g_1	14	500	7
g_1	14	500	7
g_6	40	200	7
g_8	16	80	4
g_4	8	260	6
g_5	25	250	9
g_1	14	500	7
g_1	14	500	7
g_1	14	500	7

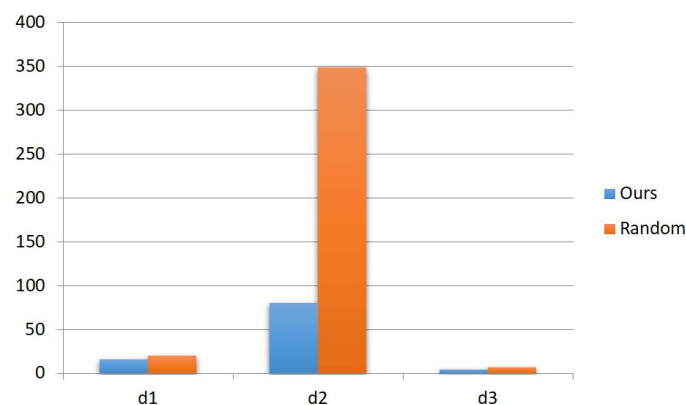


Figure 6. Performance evaluation results.

Table 5. Multi-valued attributes of cloud compositions under a random approach and our approach.

	d_1	d_2	d_3
Random	19.9	349	6.8
Ours	16	80	4

5.2. Case Study

Suppose that our MCMEC environment is shown in Table 2; when we enter the service request $\mathbb{R} = \{S_1, S_3, S_4, S_7, S_8, S_9\}$ into the MCMEC environment, it is the provider's turn to provide services based on the information about the provider and the services they provide. The Service–Provider lattice plays a role of confirming candidate provider compositions, then $\{[P_1], [P_2], [P_1, P_3], [P_3], [P_1, P_2], [P_3, P_6], [P_1, P_6], [P_1, P_2, P_3], [P_1, P_2, P_3, P_6]\}$ are taken out. These provider sets are brought into the Provider–Cloud lattice, and the candidate provider combinations are intersected with the concept lattice to obtain cloud/edge cloud combinations, $\{[C_3, C_4, C_2], [C_3, C_1, C_2], [C_3, C_4, C_1, C_2], [C_3, C_2]\}$, which all satisfy the basic needs of the user. However, we expect to recommend to our users a cloud/edge cloud composition that achieves the lowest and most balanced state in terms of monetary cost, energy consumption, and network latency. As the skyline query theory works particularly well, similarly to the classic hotel recommendation problem, the cloud/edge cloud combination is viewed as a hotel in our case study.

Table 3 shows the data related to the nine cloud/edge cloud compositions. Based on the way of constructing the skyline layers in the algorithm, three SLSs were obtained. The hierarchy of SLS_1 is shown in Figure 4. The first layer of SLS_1 has two points at g_4 and g_8 , which carry forward the advantages of the skyline query theory. It performs better in both the d_1 and d_2 dimensions. Similarly, the first layer of SLS_2 is constructed to be g_4 and g_8 , and the first layer of SLS_3 contains a point of g_8 .

6. Related Work

Since this research covers two directions, service composition in a multi-cloud environment and skyline query processing, we will provide an overview of the state of the art of these two categories.

6.1. Service Composition in a Multi-Cloud Environment

Web service composition technology integrates multiple web services and forms a function that can satisfy a single user's new services for providing more complex services to meet the complex needs of different users. Service composition technology has also attracted widespread attention from academia and industry. Particularly, service composition in multi-cloud environments is becoming more and more complex. Mezni et al. [6] proposed a multi-cloud service composition approach based on FCA, and they used FCA to represent and combine information of multiple clouds, then extracted the optimal cloud composition with the smallest number of clouds. Pang et al. [17,18] adopted FCA to develop a sustainable strategy for multi-cloud service composition by considering multiple criteria, such as energy consumption between clouds, monetary cost, and trust between users and clouds. Tang [19] proposed a holistic model, Function, Auditability, Governability and Interoperability (FAGI) to help a cloud service consumer to engage and select a trusted cloud service provider. Gutierrez-Garcia [20] presented a self-organizing and agent-based cloud service composition approach. Specifically, cloud participants and resources are implemented and instantiated by agents, which sustain a three-layered self-organizing multi-agent system. Recently, a collaborative Service Level Agreements (SLA) and reputation-based trust management (RTM) solution for federated cloud environments was developed in [21].

In multi-cloud environments, the optimization of service composition with multiple criteria is a key research issue. However, there exists a research gap about service composition in an MCMEC environment.

6.2. Skyline Query Processing for Service Composition

Zhang et al. [22] attempted to discover appropriate services and compose them with guaranteed quality of service (QoS); they further proposed an integrated skyline query processing method for building up cloud-based applications. Alrifai et al. [23] utilized a skyline query processing technique to extract the candidate services that cannot be dominated by others in each service group, which can reduce search space and improve the efficiency. Wang et al. [24] devised a fast cloud-based web

service composition approach. Their approach utilizes skyline query processing and particle swarm optimization to achieve cloud-based web service composition. Guo et al. [25] adopted skyline query processing to prune redundant services, and performed skyline service selection using 0–1 mixed-integer programming.

In summary, the skyline query processing technique has been broadly used to prune the redundant services from the global perspective and improve the efficiency of the service composition. To fill in the above research gap on service composition in an MCMEC environment, this work adopted an FCA methodology to obtain the candidate cloud/edge cloud compositions, then filtered out the optimal cloud/edge cloud composition with the advantage of skyline query processing.

7. Conclusions

This paper aims to provide a solution to the cloud service composition problem in an MCMEC environment. In order to select a sustainable cloud composition with lower energy consumption and monetary cost as well as short network latency, we propose an algorithm that effectively recommends the optimal cloud/edge cloud composition. After the user makes a service request, we select a possible candidate provider's composition and then determine the cloud/edge cloud candidate compositions based on the internal correlation between the clouds and providers. In addition, considering that there might be more candidate cloud/edge cloud candidate compositions, it is worth considering whether the cloud/edge cloud combination is a sustainable one that meets the optimization constraints on reducing energy consumption, monetary cost, and network latency. Existing research only focuses on reducing the number of clouds, and cannot flexibly provide the best cloud composition in a dynamic network environment. Our work integrates the skyline query theory and extracts the cloud/edge cloud composition in terms of different dimensions. An evaluation and a case study are provided to validate that our proposed cloud/edge cloud composition is the most sustainable and can be recommended to users in terms of monetary cost, latency, and energy.

In future work, we will conduct service composition by considering the dependency of complicated requests for services.

Author Contributions: Conceptualization, F.H.; Methodology, F.H.; Software, D.-S.P.; Supervision, F.H.; Validation, B.P.; Writing—Original draft, B.P.; Writing—Review and editing, D.-S.P. and C.D.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded in part by the National Natural Science Foundation of China (Grant No. 61702317), the Natural Science Basic Research Plan in Shaanxi Province of China (Grant No. 2019JM-379), the Fund Program for the Scientific Activities of Selected Returned Overseas Professionals in Shaanxi Province (Grant No. 2017024), and the National Research Foundation of Korea (No. NRF-2020R1A2B5B01002134).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jula, A.; Sundararajan, E.; Othman, Z. Cloud computing service composition: A systematic literature review. *Expert Syst. Appl.* **2014**, *41*, 3809–3824.
2. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge computing: Vision and challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. [\[CrossRef\]](#)
3. Gupta, H.; Vahid Dastjerdi, A.; Ghosh, S.K.; Buyya, R. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. *Softw. Pract. Exp.* **2017**, *47*, 1275–1296. [\[CrossRef\]](#)
4. Gutierrez-Garcia, J.O.; Sim, K.M. Agent-based cloud service composition. *Appl. Intell.* **2013**, *38*, 436–464. [\[CrossRef\]](#)
5. Yu, Q.; Chen, L.; Li, B. Ant colony optimization applied to web service compositions in cloud computing. *Comput. Electr. Eng.* **2015**, *41*, 18–27. [\[CrossRef\]](#)
6. Mezni, H.; Sellami, M. Multi-cloud service composition using formal concept analysis. *J. Syst. Softw.* **2017**, *134*, 138–152.

7. Zhu, D.; Wang, X.D. Mobile network edge computing and caching technology. *Railway Comput. Appl.* **2017**, *26*, 51–54.
8. Li, R.H.; Qin, L.; Ye, F.; Yu, J.X.; Xiao, X.; Xiao, N.; Zheng, Z. Skyline community search in multi-valued networks. In Proceedings of the 2018 International Conference on Management of Data, Houston, TX, USA, 10–15 June, 2018; pp. 457–472.
9. Abid, A.; Rouached, M.; Messai, N. Semantic web service composition using semantic similarity measures and formal concept analysis. *Multimed. Tools Appl.* **2020**, *79*, 6569–6597. [\[CrossRef\]](#)
10. Hao, F.; Pang, G.; Pei, Z.; Qin, K.; Zhang, Y.; Wang, X. Virtual machines scheduling in mobile edge computing: A formal concept analysis approach. *IEEE Trans. Sustain. Comput.* **2020**, *5*, 319–328. [\[CrossRef\]](#)
11. Hao, F.; Yau, S.S.; Min, G.; Yang, L.T. Detecting k-balanced trusted cliques in signed social networks. *IEEE Internet Comput.* **2014**, *18*, 24–31. [\[CrossRef\]](#)
12. Choi, J.H.; Hao, F.; Nasridinov, A. HI-Sky: Hash Index-Based Skyline Query Processing. *Appl. Sci.* **2020**, *10*, 1708. [\[CrossRef\]](#)
13. Wang, S.; Ooi, B.C.; Tung, A.K.H.; Xu, L. Efficient skyline query processing on peer-to-peer networks. In Proceedings of the 2007 IEEE 23rd International Conference on Data Engineering, Istanbul, Turkey, 11–15 April 2007; pp. 1126–1135.
14. Rosa, R.V.; Rothenberg, C.E. The pandora of network slicing: A multicriteria analysis. *Trans. Emerg. Telecommun. Technol.* **2020**, *31*, E3651. [\[CrossRef\]](#)
15. Hao, F.; Min, G.; Pei, Z.; Park, D.-S.; Yang, L.-T. K-clique community detection in social networks based on formal concept analysis. *IEEE Syst. J.* **2015**, *11*, 250–259. [\[CrossRef\]](#)
16. Hao, F.; Yang, Y.; Pang, B.; Yen, N.-Y.; Park, D.-S. A fast algorithm on generating concept lattice for symmetry formal context constructed from social networks. *J. Ambient. Intell. Humaniz. Comput.* **2019**, *69*. [\[CrossRef\]](#)
17. Pang, B.; Hao, F.; Yang, Y.; Park, D.-S. An efficient approach for multi-user multi-cloud service composition in human–land sustainable computational systems. *J. Supercomput.* **2020**, *76*, 5442–5459. [\[CrossRef\]](#)
18. Pang, B.; Yang, Y.; Hao, F. A sustainable strategy for multi-cloud service composition based on formal concept analysis. In Proceedings of the 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS), Zhangjiajie, China, 10–12 August 2019; pp. 2659–2665.
19. Tang, C.; Liu, J. Selecting a trusted cloud service provider for your SaaS program. *Comput. Secur.* **2015**, *50*, 60–73. [\[CrossRef\]](#)
20. Gutierrez-Garcia, J.O.; Sim, K.M. Self-organizing agents for service composition in cloud computing. In Proceedings of the 2010 IEEE Second International Conference on Cloud Computing Technology and Science, Indianapolis, IN, USA, 30 November–3 December 2010; pp. 59–66.
21. Papadakis-Vlachopapadopoulos, K.; González, R.S.; Dimolitsas, I.; Dechouniotisa, D.; Ferrerc, J.A.; Papavassiliou, S. Collaborative SLA and reputation-based trust management in cloud federations. *Future Gener. Comput. Syst.* **2019**, *100*, 498–512. [\[CrossRef\]](#)
22. Zhang, F.; Hwang, K.; Khan, S.U.; Malluhi, Q.M. Skyline discovery and composition of multi-cloud mashup services. *IEEE Trans. Serv. Comput.* **2015**, *9*, 72–83. [\[CrossRef\]](#)
23. Alrifai, M.; Skoutas, D.; Risse, T. Selecting skyline services for QoS-based web service composition. In Proceedings of the 19th International Conference on World wide Web, Raleigh, CA, USA, 26–30 April 2010; pp. 11–20.
24. Wang, S.; Sun, Q.; Zou, H.; Yang, F. Particle swarm optimization with skyline operator for fast cloud-based web service composition. *Mob. Netw. Appl.* **2013**, *18*, 116–121. [\[CrossRef\]](#)
25. Guo, Y.; Wang, S.; Wong, K.S.; Kim, M.H. Skyline service selection approach based on QoS prediction. *Int. J. Web Grid Serv.* **2017**, *13*, 425–447. [\[CrossRef\]](#)

