Optimized Composition for Multiple User Service Requests Based on Edge-Cloud Collaboration

JINE TANG\textsuperscript{1}, TAO LIN\textsuperscript{1}, DONG WANG\textsuperscript{2,3}, AND ZHANGBING ZHOU\textsuperscript{4,5}, (Member, IEEE)

\textsuperscript{1}School of Artificial Intelligence, Hebei University of Technology, Tianjin 300401, China
\textsuperscript{2}Tianjin Institute of Aerospace Mechanical and Electrical Equipment, Tianjin 300458, China
\textsuperscript{3}Tianjin Key Laboratory of Aerospace Intelligent Equipment Technology, Tianjin 300458, China
\textsuperscript{4}School of Information Engineering, China University of Geosciences, Beijing 100083, China
\textsuperscript{5}Computer Science Department, TELECOM SudParis, 91000 Évry, France

Corresponding author: Dong Wang (23375510@qq.com)

This work was supported in part by the Open Project Program of Tianjin Key Laboratory of Aerospace Intelligent Equipment Technology, Tianjin Institute of Aerospace Mechanical and Electrical Equipment, under Grant TJYHZN2020KT001, and in part by the National Natural Science Foundation of China under Grant 61702232.

ABSTRACT The Internet of Things (IoT) has become the infrastructure to widely support ubiquitous services. In this setting, a large number of smart object devices provide IoT services for people to carry on day-to-day activities. These services are easy to access, and thus, can be easily used by different users with different requirements through service allocation and composition. Nevertheless, with the increasing adoption and presence of IoT services, it becomes more difficult to find the appropriate IoT service compositions that can maximize the service utility of all the users’ requests. Moreover, the IoT devices are usually resource constrained and cannot effectively finish some complex task of service requests, thereby reducing the service quality. To address this problem, we develop a novel IoT service allocation mechanism in collaborative device-edge-cloud architecture. First, prevalence of frequent function interaction is mined to help construct IoT service allocation index. Second, service requests’ priority and their preferences towards branch nodes are further set and estimated to help maximize the service composition utility of multiple user requests. Third, edge device is exploited to help service execution and guarantee service allocation utility by offloading task from IoT devices. The results are demonstrated by conducting a set of extensive experiments, which show that the proposed method performs better than the alternative techniques in reducing the total energy consumption and promoting the service utility maximum.

INDEX TERMS Internet of Things, collaborative device-edge-cloud architecture, frequent function interaction, IoT service allocation index, service requests’ priority.

I. INTRODUCTION

The Internet of Things (IoT) [1] is one of the hottest research fields nowadays, and has become the infrastructure to widely support ubiquitous applications [2]–[4]. An IoT system is expected to connect and manage huge numbers of heterogeneous devices, such as sensors, actuators, smartphones, and other smart object devices (SODs), which are capable of continuously providing real-time IoT services in different delay-sensitive applications at the network edge, including healthcare monitoring, automotive, smart buildings and so on [5], [6]. In most cases, the IoT services provided by different SODs are functionally correlated with each other and able to work together for supporting user’s social activity [7], [8], which motivates the idea of exploiting social-aware function interaction to help IoT services allocation. As such, function correlated IoT services are playing a crucial role in providing a composition service for objects to engage in a series of social activities. However, techniques are still under development to allow us to most effectively make use of the ubiquitous ”smart things” to accomplish the desired goals of various human activities, especially to maximize the satisfaction degree for all users’ service requests.

Due to increasing application demands and rapid technological advances in IoT systems, most of research on IoT service discovery and selection support service composition for single user’s service request, where the quality of services and the middleware development are the main concerns.
A particle swarm optimization method [9] aims to search for optimal IoT service compositions considering the factors, such as spatial-temporal constraints, energy efficiency and the configurability of IoT services. A peer-to-peer service discovery solution [10] is considered for very large-scale IoT network with a mix of static and mobile IoT nodes. Each IoT peer maintains a routing information cache to guide the peer-to-peer routing of the service discovery queries. Paper [11] proposes a social network based approach for IoT device management and service composition. Taking the social relationships between IoT services into account, it allows for the automatic collaboration of heterogeneous IoT devices in order to meet user’s requirements. Paper [12] provides an algorithm to solve IoT-enabled service optimal composition problems with the discovered empirical knowledge of IoT services based on the concept of service domain features. Notice that edge computing emerges recently as a reinforcement of cloud computing to optimize the resource usage of IoT devices and wireless network for providing context-aware services. Inspired by this, paper [13] further considers the cost of bringing services to the edge and proposes a heuristic algorithm to optimize the average service response time by considering the resource consumption, popularity and the service composition graph. Generally, current techniques adopt service-oriented architectures to support IoT service discovery and composition for a certain service request, where the services quality and spatial-temporal constraints, as well as the energy efficiency, are the main concerns. For multiple service requests, some works [14], [15] consider to identify the common sub-requests, and they can be executed once for fulfilling partial of these concurrent requests. Through identifying sharable sub-compositions and optimizing these concurrent service compositions, it can efficiently improve the functional shareability of IoT device among requests, which induces the reduction of energy consumption in the network. Generally, these works focus on providing the solutions of service compositions for multiple service requests which have sharable sub-compositions. However, in most cases, the sharable sub-compositions are only a part of multiple service requests, which still contain a lot of non-sharable sub-compositions with different spatial-temporal-resource-function constraints. It is required to consider all the service requests so as to maximize the whole service composition utility.

To address this challenge, this paper develops a novel allocation method for service composition with the help of collaborative device-edge-cloud architecture. The contributions of this paper are summarized as follows:

- A collaborative device-edge-cloud architecture based service allocation method is proposed in this paper for multiple user service requests, which aims to manage the IoT services in a practice and effective way.
- A novel IoT service allocation index, denoted STSC-tree, is built based on grid division and merge, as well as function interaction prevalence between IoT services.
- A ranked list of priority groups is constructed based on spatial-temporal-resource domination and service request’s importance. Leveraging this, a preference-aware service search method is executed along STSC-tree through estimating the preference of request group to branch nodes.
- Edge device is exploited to help service execution and guarantee service allocation utility by offloading task from IoT devices.
- Extensive experiments are conducted, and evaluation results demonstrate our approach improves the performance of IoT service composition with regard to energy consumption, composition time, response time and matching degree.

The remainder of this paper is organized as follows. Section II provides a review of related works. In Section III, we outline the collaborative device-edge-cloud allocation architecture. Section IV presents how to derive the frequent function interaction patterns between IoT services. Section V explores the construction process of STSC-tree. Section VI describes the search process for multi-user service composition based on group division and priority setting of service requests. Section VII presents evaluation results. Section VIII concludes this work.

II. RELATED WORK AND COMPARISON

A. IoT SERVICE ALLOCATION AND COMPOSITION

Most of research on IoT service discovery and selection support service composition for a certain user service request, where the quality of services and the middleware development are the main concerns.

Mishra et al. [16] propose a meta heuristic-based service allocation framework, which uses the following meta heuristic techniques, (binary) particle swarm optimization and bat algorithm. These proposed techniques can deal with the heterogeneity of resources in the fog computing environment. He et al. [17] propose a dynamic network slicing mechanism including virtual network mapping and reconfiguration to provide network slices for services. Liu et al. [18] present a service scheduling method, that includes the architecture, components and functional requirements, to balance the business load of Power Distribution Internet of Things based on edge computing. Boulakbech et al. [19] propose a new mashup service model, called IoT big services. These IoT services are organized in a multi-root tree data structure where the nodes represent big service and act as an integrator that collects data from lower level, processes them and delivers the results to higher level in the architecture. However, the index tree does not explore the spatial-temporal constraints for service composition in hierarchy. Cheng et al. [20] propose a situation-aware IoT service coordination platform based on the event-driven and service-oriented architecture paradigm, which integrates the advantages of SOA effectively to support the coordination of IoT services. Sun et al. [9] propose a particle swarm optimization method to search for optimal IoT service compositions considering the factors, such as
spatial-temporal constraint, energy efficiency, and the configurability of IoT services. Moeini et al. [10] consider a peer-to-peer service discovery solution for very large-scale IoT network with a mix of static and mobile IoT nodes. Each IoT peer maintains a routing information cache to guide the peer-to-peer routing of the service discovery queries. Guo et al. [11] propose a social network based approach for IoT device management and service composition. Taking the social relationships between IoT services into account, it allows for the automatic collaboration of heterogeneous IoT devices in order to meet user requirements. Li et al. [12] provide an algorithm to solve IoT-enabled service optimal composition problems with the discovered empirical knowledge of IoT services based on the concept of service domain features. Notice that edge computing is widely adopted nowadays to support applications especially when they are delay-sensitive. It emerges recently as a reinforcement of cloud computing to optimize the resource usage of IoT devices and wireless network for providing context-aware services. Motivated by this, Deng et al. [13] further consider the cost of bringing services to the edge and propose a heuristic algorithm to optimize the average service response time by considering the resource consumption, popularity and the service composition graph.

Generally, current techniques adopt service-oriented architectures to support IoT service discovery and composition for a certain service request, where the quality of services and the spatial-temporal constraint, as well as the energy efficiency, are the main concerns. However, these works generally do not consider the IoT service allocation search and composition for maximizing the service utility of multiple user service requests that arrive simultaneously.

**B. SERVICE RECOMMENDATION**

Recently, with the increasing demand of web services used in the Internet of Things, there has been a growing interest in the study of efficient service quality evaluation approaches. Luo et al. [21] propose an approximate dynamic programming based parameter tuner for collaborative IoT service quality prediction. Shao et al. [22] propose a hybrid collaborative filtering method for IoT service recommendation. The average service quality value of adjacent time intervals is used to fill part of the data, and then the filled data and the enhanced similarity method are used to make the prediction and produce recommendation results. Li et al. [23] propose a novel temporal influences-aware collaborative filtering method which designs an enhanced temporal influences-aware similarity measurement to predict service quality values. Wang et al. [24] propose an integrated service quality prediction approach which unifies the modeling of multi-dimensional service quality data via multilinear-algebra based concepts of tensor and enables efficient service recommendation for mobile clients via tensor decomposition and reconstruction optimization algorithms. Zhou et al. [25] propose a novel service recommender model using collaborative filtering to improve the prediction of Quality-of-Services. Benefiting from the accuracy of hybrid recommenders, the authors extend the idea of optimized predicting order and design the PGraph to describe the neighborhood. During the predicting process, a new algorithm using adjusted topological sorting for PGraph is further proposed to generate the optimized order. Pan and Wu [26] propose a personalized online-to-offline service recommendation method based on a novel frequent service-set network. This method integrates the service network’s attributes, which include popularity and commonality of service itself, and then generates the service recommendation list.

As the advantages of collaborative filtering based method for service prediction and recommendation, we use collaborative filtering score to estimate the service utility for composition service selected from STSC-tree so as to effectively reduce search space.

**C. RELATED SPATIAL AND SPATIAL-TEXTUAL INDEX**

Indexing as grid or tree structures is a popular and effective method to answer many spatial queries like range search or visit the k-nearest neighbors, often under some query constraints. Chung et al. [27] construct an extended index that reflects multiple non-spatial grid index properties of moving objects. Leveraging this index structure, the authors give an efficient query method for moving objects that simultaneously considers spatial attributes and non-spatial attributes. Li et al. [28] propose an efficient index called IR-tree, which supports top-k document retrievals leveraging the unified representation of textual and spatial relevances. Choudhury et al. [29] extend IR-tree and construct text-first index structure SIF for batch processing top-k spatial-textual queries with the help of block based inverted file. Zhong et al. [30] present a height-balanced and scalable index, namely G-tree, to efficiently support (keyword-based) KNN queries based on the assembly-based method. Kucuktunc and Ferhatosmanoglu [31] introduce a diverse browsing method for diverse k-nearest neighbor search based on the popular distance browsing feature of R-tree. Cao et al. [32] further propose to retrieve a group of spatio-textual objects such that the group’s keywords cover the query’s keywords, and such that the objects are nearest to the query location and have the smallest inter object distances, with the help of an IR-tree. You et al. [33] aim at evaluating the potentials of accelerating both R-tree bulk loading and spatial window query processing on GPUs using R-tree. Several parallel spatial window query processing techniques are implemented on GPUs using both dynamically generated R-trees constructed on CPUs and bulk loaded R-trees constructed on GPUs. Tang et al. [34] strive to construct an energy-efficient index tree for multi-region sensing attributes aggregation queries, which mainly adopts density-based clustering model. With the help of index routing tree, multi-attribute data that satisfy the query requirement can be efficiently forwarded to base station for further processing.

As the advantages of spatial index in coping with spatial query, we construct a new STSC-tree based on R-tree, grid
division and merge, as well as function interaction prevalence, to represent IoT service and answer multiple user service requests.

### III. SYSTEM DESIGN

We are now ready to describe the three-tier collaborative device-edge-cloud architecture (see Fig. 1) and its tiers, as well as the involved allocation process of IoT service. Intuitively, the tiers should be closely associated with each other to finish the IoT service allocation and composition, as well as task execution efficiently and accurately. The major notations used in this paper are listed in Table 1.

- **Device tier**: It is the lowest level of system architecture. This tier involves various smart object devices, such as sensors and/or monitoring devices, which continuously sample the behavior activity states of real-world objects and provide the IoT service for object behavior activity. The IoT services generated from smart object devices always fall into some social activity region at current timestamp, and intend to engage in providing some social activity services in future timestamps. These devices carry on local computing and have no transmission delay or extra costs for service execution. However, due to the limited capabilities of the devices, it might put a heavy burden on them, which might even be unable to effectively finish the complex service request.

- **Edge tier**: An edge device maintains a function interaction frequently occurred in the historical log of social activities. It is responsible for collecting and maintaining the IoT service allocation and composition, together with spatial-temporal proximity. If there is no enough resource for preferent IoT devices to efficiently execute the corresponding request tasks, the edge device would offload the computation of IoT devices according to the priority of respective service request. The higher the priority of service request, the earlier it is offloaded to the edge device from corresponding IoT devices for service execution. This utilizes the available computational, communication, energy resources and storage capabilities of edge devices, but suffering extra communication and delay costs to transmit offloading data.

- **Cloud tier**: It is the highest level of system architecture. Each cloud data center is responsible for storing and processing the IoT service allocation in its corresponding edge IoT network. The cloud sends the requests from multiple user service chains to the edge tier in the form of a query. If a certain edge node detects the service query request from cloud, it further collects the most relevant IoT service allocation from smart object devices.

### IV. SOCIAL-AWARE FUNCTION CHAIN PATTERN MINING

The allocation process of IoT services for future service requests is based on a learning approach, such that the node log files from previously successful function matching processes can be used to recreate similar function matching for similar social activities in the future. In the following, we focus on the combinations’ constraints of sequent social function properties of IoT services to form a function service chain pattern that is satisfactory.

**Definition 1 Sequential Functional Matching Relationship $R_f$**: Notice that the sequent services allocation might
be functionally mismatched. To allow for constraints on sequential functional matching of two IoT services \( s_1 \) and \( s_2 \), a threshold for the minimum functional matching degree between IoT services involved in building a potential sequential service chain pattern needs to be specified, i.e. \( F_{\text{deg}}(s_1 \otimes s_2) \geq t_f \).

To complete the mining basis for our social-aware function service chain pattern, we combine the sequential matching degree between IoT services’ functions to form a function instance chain pattern as follows.

**Definition 2 Service Allocation Function Instance Chain Pattern (SAFICP):** Given a sequentially functional matching relationship \( R_f \) and an IoT service composition’s function chain pattern \( S = \{s_1, s_2, \ldots, s_k\}(s_i \rightarrow s_j, i < j, i, j \in [1, 2, \ldots, k]) \). A tuple of \( k \) IoT service \( T = \{s_1, s_{t_2}, \ldots, s_{t_k}\} \) is an instance of \( S \) if and only if (1) \( \{s_1, s_{t_2}, \ldots, s_{t_k}\} \) forms an IoT service composition chain under the constraint of \( R_f \), (2) \( s_i \) is an instance of \( s_i \), \( \forall i \in [1, 2, \ldots, k] \).

To measure whether or not a function chain pattern \( f \) is statistically significant, we give the participation ratio defined in Eq. 1 to derive its prevalence.

\[
P_f = \frac{N_{s, \text{st}}}{N_{s', \text{st}}} \tag{1}
\]

\( N_{s, \text{st}} \) denotes the set of service instances participating in \( T(S) \), where \( S \) corresponds to the set of all generalizations of pattern \( f \). \( N_{s', \text{st}} \) denotes the set of service instances participating in \( T(S') \), where \( S' \) corresponds to the set of all generalizations of the emerging social function chain patterns.

To this end, the prevalent IoT service function chain pattern is mined and formed. Consider that an IoT service’s composition and allocation may be necessary to different users in different extents. In the following, we give a spatial-temporal IoT service allocation index based on the service function chain pattern that is interesting and significant in order to maximize the requirements of different users.

**V. SPATIAL-TEMPORAL SERVICE ALLOCATION INDEX**

**A. OVERVIEW OF STSC-TREE**

Notice that grouping the IoT services having a higher functions interaction prevalence into a hierarchical tree structure can facilitate the allocation and composition of correlated IoT services, since the IoT services satisfying the user’s request functions can be effectively collected along as fewer branch nodes as possible. Inspired by this, we propose to construct the STSC-tree that adapts to each local IoT network managed by edge nodes based on R-tree [35] and functions interaction prevalence. Specifically, we first divide the service region into grid cells with the same geographical size. Then, any two neighbor grid cells are merged into one sub-region, when the prevalence is the biggest between paired functions interaction of IoT services contained in them (see Eq. 2, Eq. 3). Likewise, the formed sub-regions may be merged again based on the prevalence of function interaction between them (see Eq. 4, Eq. 5, Eq. 6) if the number of current sub-regions is beyond the node capacity of R-tree. In this paper, we use grid cell to denote the one that was initially divided, and use sub-region to denote the merge region from several nearby grid cells or sub-regions.

**Definition 3 Paired Service Interaction Prevalence (STP):** Let \( N_{s_1, s_2} \) represent the number of function interactions between IoT service \( s_1 \) and \( s_2 \), \( P_{f_{s_1, s_2}} \) be the prevalence of their paired functions interaction pattern (see Eq. 1), as well as \( H(s_1) \) and \( H(s_2) \) (\( |H(s_1)| = |H(s_2)| = n \)) be their sequential neighbors. The quantified paired functions interaction of \( s_1 \) and \( s_2 \) can be derived as:

\[
P_{s_1, s_2} = \frac{1}{n} \left( \sum_{x \in H(s_1)} I_{s_1, s_2} + \sum_{y \in H(s_2)} I_{s_1, s_2} \right)
\]

\[
= \frac{1}{n} \left( \frac{N_{s_1, s_2}}{n} \sum_{x \in H(s_1)} P_{f_{s_1, f_{s_2}}} + \sum_{y \in H(s_2)} P_{f_{s_1, f_{s_2}}} \right) \tag{2}
\]

**Definition 4 Grid Cell Interaction Prevalence (GIP):** Let \( g_1 = \{s_1, s_{12}, \ldots, s_{1m}\} \) and \( g_2 = \{s_2, s_{22}, \ldots, s_{2n}\} \) represent a set of IoT services contained in grid cells \( g_1 \) and \( g_2 \), respectively. Let \( P_{f_{g_1, g_2}}(t \in [1, h]) \) be the interaction prevalence of paired frequent functions \( \{f_i, f_j\}(t \in [1, h]) \), and subject to \( P_{f_{g_1, g_2}} \geq P_{f_{g_1, g_2}^*}(t \in [1, h - 1]) \). Let \( f_i(i \in [1, m]) \) and \( f_j(j \in [1, n]) \) be the functions of \( s_1(i \in [1, m]) \) and \( s_2(j \in [1, n]) \), respectively. As IoT devices have limited resource, we only consider the relatively higher prevalence of paired functions interaction to derive the interaction prevalence between grid cells if an IoT device involves in more than one frequent function interactions. This facilitates to search the IoT service composition from neighbor region since the frequent function combinations are desired by many users to engage in important social activity. Inspired by this, the probability of \( g_1 \) and \( g_2 \) belonging to the same service composition is derived as follows:

\[
P_{g_1, g_2} = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} P_{s_1, s_2}
\]

\( s.i.
\)

\[(f_i, f_j) \in \{f_i, f_j')(t \in [1, h'], h' \leq h)\]

\[
P_{f_{g_1, g_2}}(t \in [1, h']) \geq P_{f_{g_1, g_2}}(t \in [h' + 1, h])
\]

\[
\exists f' = [1, n], f' \neq j, (f_i, f_{i'}) \in \{f_i, f_{i'})(t \in [1, h'])
\]

\[
\exists h' = [1, m], i \neq i, (f_{i'}, f_{j'}) \in \{f_{i'}, f_{j'})(t \in [1, h'])
\]

\[
\tag{3}
\]

**Definition 5 Sub-Region Interaction Prevalence (SIP):** Let \( R_1 = [R_{11}, R_{12}, \ldots, R_{1m}] \) and \( R_2 = [R_{21}, R_{22}, \ldots, R_{2n}] \) represent a set of sub-regions contained in region \( R_1 \) and \( R_2 \), respectively. Consider that a higher prevalence of function interaction between neighbor grid cells ensures low energy consumption, transmission delay as well as desired function composition. We only select the paired grid cells that are the nearest in distance although more pairs of grid cells with
regions. Inspired by this, the probability of frequent function interactions may be contained in two sub-regions. Inspired by this, the probability of \( R_1 \) and \( R_2 \) belonging to the same service composition is derived as follows (three situations are considered):

- \( R_1(i \in [1, m]) \) and \( R_2(j \in [1, n]) \) are grid cells:
  \[
P_{R_1,R_2} = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} P_{R_1,R_2}
  \]
  \[
  s.t.
  \]
  \[
  Dist(R_{1i}, R_{2j}) \leq Dist(R_{1i}, R_{2j})(j \neq j')
  \]
  \[
  Dist(R_{1i}, R_{2j}) \leq Dist(R_{1i'}, R_{2j})(i \neq i')
  \]

- \( R_1(i \in [1, m]) \) is a grid cell while \( R_2(j \in [1, n]) \) is a sub-region:
  \[
P_{R_1,R_2} = \frac{1}{mnr} \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{r'=1}^{r} P_{R_1,R_2}
  \]
  \[
  s.t.
  \]
  \[
  Dist(R_{1i}, R_{2j}) \leq Dist(R_{1i}, R_{2j'}) \text{ if } r' \neq r''
  \]
  \[
  Dist(R_{1i}, R_{2j}) \leq Dist(R_{1i'}, R_{2j}) \text{ if } i \neq i'
  \]

- \( R_1(i \in [1, m]) \) and \( R_2(j \in [1, n]) \) are sub-regions:
  \[
P_{R_1,R_2} = \frac{1}{mnr} \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{r'=1}^{r} P_{R_{1i}, R_{2j}}
  \]
  \[
  s.t.
  \]
  \[
  Dist(R_{1i}, R_{2j}) \leq Dist(R_{1i}, R_{2j^d})(r' \neq r'')
  \]
  \[
  Dist(R_{1i}, R_{2j}) \leq Dist(R_{1i'}, R_{2j^d})(s' \neq s'')
  \]

where \( R_{1i} \) \((i \in [1, s])\), \( R_{1i'} \) \((i' \in [1, s])\), and \( R_{2j^d} \) \((r' \in [1, s])\) represent the grid cells contained in \( R_{1i} \) and \( R_{2j} \), respectively. Next, we formally give the properties of STSC-tree:

- The root node is the whole MBR region.
- The interior parent nodes are the merge regions, whose sub-regions’ number is within \([m_0, M_0]\) \((1 \leq m_0 \leq M_0/2)\). A window summary \( ws_i \) about a service composition \( s_i \) at level \( l_i \) is stored in its edge node \( ED_i \):
  \[
  ws_i = \{sl_{i}, st_{i}, sf_{i}, sa_i\}
  \]
  where \( sl_{i}, st_{i}, sf_{i} \) and \( sa_i \) denote the service region location, service time, service function and chain, as well as service resource availability of \( s_i \), respectively.
- The interior child nodes \( ED_{i+1,j} \) and \( ED_{i+1,j+1} \) of \( ED_i \) are also the merge regions, whose sub-regions’ number is within \([m_0, M_0]\) \((1 \leq m_0 \leq M_0/2)\). Two window summaries \( ws_{i+1,j} \) and \( ws_{i+1,j+1} \) about two child service compositions \( s_{i+1,j} \) and \( s_{i+1,j+1} \) at level \( l_{i+1} \) are stored in their edge nodes \( ED_{i+1,j} \) and \( ED_{i+1,j+1} \):
  \[
  ws_{i+1,j} = \{sl_{i+1,j}, st_{i+1,j}, sf_{i+1,j}, sa_{i+1,j}\}
  \]
  \[
  ws_{i+1,j+1} = \{sl_{i+1,j+1}, st_{i+1,j+1}, sf_{i+1,j+1}, sa_{i+1,j+1}\}
  \]
  \[
  s.t.
  \]
  \[
  sl_{i+1,j} \in sl_i, st_{i+1,j} \in st_i, sf_{i+1,j} \in sf_i
  \]
There is no sequent relationship for IoT devices in spatial index. The non-leaf node can store the sequent function relationship, which means that the sub-tree in it contains the IoT devices with these functions, but not with the execution sequence due to the limitation of spatial-temporal relationship. It is desired to search the IoT device that satisfies the specific spatial-temporal-resource-function constraints in the neighbor sibling node of the local region.

- The non-leaf node is also a big service composition therefore acts as an integrator, collecting data from the big service compositions in lower level, processing them, and delivering the result to the big service in upper level.
- The leaf nodes are the grid cells. A window summary \(w_{m}\) about a leaf service composition \(s_m\) at level \(l_m\) is stored in its edge node \(E_{d_m}\):

\[
w_{m} = \{s_{l_m}, s_{m}, s_{f_m}, s_{a_m}\}
\]

such that

\[
s_{l_m} \in s_{l_{m-1}}, s_{m} \in s_{m-1}
\]

\[
s_{f_m} \in s_{f_{m-1}}, s_{a_m} \leq s_{a_{m-1}}
\]

\[
d_{l_{m}} \in s_{l_{m}}, d_{f_{m}} \in s_{f_{m}}, d_{a_{m}} \leq s_{a_{m}}
\]

where \(d_{l_{m}}, d_{f_{m}}, \) and \(d_{a_{m}}\) denote the service region location, service time, service function and resource availability of the \(i\)th IoT device in \(s_m\), respectively.

- Unlike R-tree, STSC-tree is often an unbalanced tree. Thus, the leaf nodes are generally not at the same level.

**Example 1:** Fig. 3(b) gives an example of STSC-tree constructed from the four kinds of IoT services (colored with blue, green, yellow, purple), as shown in Fig. 2, by setting \(M_0\) as 4. The construction process is based on the grid division and merge (see Fig. 3(a)), as well as the function interaction prevalence (see Fig. 2). Each non-leaf node corresponds to a sub-region of \(R_i\) \((i \in [0, 6])\). Each leaf node corresponds to a grid region of \(g_i\) \((i \in [0, 9])\). For example, the node \(R_2\) corresponds to the bottom-left sub-region region shown in Fig. 3(a), which includes IoT services \(d_{16}, d_{17}, d_{18}, d_{19}\). The four IoT services have adjacent distance and higher function interaction prevalence. Therefore, they are merged into one parent node in upper level. In this example, we consider one-to-one function interaction matching when giving the statistic number of paired functions provided by IoT services due to the resource constraint of IoT devices.

**B. STSC-TREE CONSTRUCTION**

Algorithm 1 shows how to construct STSC-tree based on grid division and merge, as well as function interaction probability. We first divide the whole region evenly into grid cells, where each grid cell has the same side-length of \(\sqrt{2} r\) (line 1). \(r\) is the communication radius of smart object devices.

The communication radius of smart object is assumed the same in this paper. We set the side-length as \(\sqrt{2} r\), since according to the experiments conducted in [36], the optimal region size is \(l\)-hop. Then, the function interaction prevalence between a grid cell \(g_i\), and another neighbor grid cell \(g_j\), is computed according to Eq. 3 (line 5). When more than one sub-regions are unmerged, we select the two sub-regions (denoted \(n_{ri}\) and \(n_{rj}\)) with the biggest interaction prevalence and merge them into one (line 10). The new formed region \(n_r\) is the parent of \(n_{ri}\) and \(n_{rj}\) in hierarchy. Thereafter, the function interaction prevalence between this new formed sub-region \(n_r\) and its neighbor sub-regions are calculated according to Eq. 4, 5 and 6. The procedure of lines 10-17 iterates until there is only one sub-region left. Consequently, these merging sub-regions form a tree hierarchy called STSC-tree, and initially divided grid cells form the leaf nodes of this tree. When all sub-regions are merged into one, a tree hierarchy is constructed (line 19). Finally, line 20 selects head nodes for each sub-region by means of LEACH [37].

**Algorithm 1:** STSCTreeBuilt

**Require:** \(S_{gi} \): the set of IoT services

**Ensure:** \(n_{dti} \): the root node of constructed STSC-tree

1: dividing the whole region evenly into grid cells with the side length of each grid cell as \(\sqrt{2} r\)

2: foreach grid cell \(g_i\) do

3: \(N_{g} \leftarrow \) the grid cell in the neighborhood of \(g\)

4: foreach \(g_j \in N_{g}\) do

5: \(p^\prime_{g_{i}, g_{j}} \leftarrow \) computing the function interaction prevalence of \(g_i\) and \(g_j\) according to Eq. 3

6: end for

7: end for

8: \(N_{sr} \leftarrow \) the number of sub-regions in the whole region

9: while \(N_{sr} > 1\) do

10: \(n_{r} \leftarrow \) merge two adjacent sub-regions \(n_{ri}\) and \(n_{rj}\) with the biggest function interaction prevalence

11: set \(n_{ri}\) and \(n_{rj}\) as the child nodes of \(n_{r}\)

12: \(a_{nr} \leftarrow \) window summary for the composition service in \(n_r\)

13: \(H_{nr} \leftarrow \) the sub-regions in the neighborhood of \(n_r\)

14: foreach \(s_{ri} \in H_{nr}\) do

15: \(p^\prime_{sr_{i}, nr} \leftarrow \) computing the function interaction prevalence of \(s_{ri}\) and \(nr\) according to Eq. 4, 5 and 6

16: end for

17: \(N_{sr} \leftarrow N_{sr} - 1\)

18: end while

19: \(n_{dti} \leftarrow n_r\)

20: head node selection according to [37]

**Time Complexity of STSC-Tree:** Given \(n\) as the number of initially divided grid cells, \(p\) as the reducing scale of the number of sub-regions in each merging process, the number of sub-regions in the following merging process is approximately \(\frac{n}{p^i}\) \((i = 0, 1, \ldots)\) from low level to high level. The time
complexity of merging grid cells is $O\left(\frac{n^2}{t} \times k \times t\right)$, where $k$ is the merging number and $t$ is the iterative time. Usually, $k \ll n$ and $t \ll n$. In our construction algorithm, $k$ is a constant and $t$ is typically quite small. Therefore, the time complexity of Algorithm 1 is $O(n)$.

VI. MULTI-USER SERVICE ALLOCATION
A. DIVISION AND PRIORITY SETTING OF SERVICE REQUEST GROUP

It is noteworthy that when a lot of users request for service allocation, some analysis and decision are required as soon as possible by sharing and reducing computations. Motivated by this, we consider to divide the user service requests into priority groups using the domination relationship of request conditions, including service request timestamp, service request region, service request resource availability, as well as service request importance, before searching IoT services.

Definition 6 User Request Timestamp Domination (URTD): The earlier time request has a higher priority because it may consume a longer time for service waiting and execution. Given a timestamp-varying service request $sr_i$ with the request timestamp $st_i$, and another service request $sr_j$ with the request timestamp $st_j$, $sr_i$ dominates $sr_j$ in terms of request timestamp $st$, denoted as $sr_i \geq_{st} sr_j$, if and only if $st_i \leq st_j$.

Definition 7 User Request Geographical Domination (URGD): The service spatial location that is farther has a higher priority because it may consume a lot of energy for message transmission to the next IoT service. Given a location-varying service request $sr_i$ with the request location $sl_i$, and another service request $sr_j$ with the request location $sl_j$, $sr_i$ dominates $sr_j$ in terms of request location $sl$, denoted as $sr_i \geq_{sl} sr_j$, if and only if $|sl_i - sl_p| \geq |sl_j - sl_p|$, where $sl_p$ is the nearest service region away from them.

Definition 8 User Request Resource Domination (URRD): The service request that requires more resources has a higher priority because the service request may consume a lot of energy for service execution. Given a resource-varying service request $sr_i$ with the resource request $sa_i$, and another service request $sr_j$ with the resource request $sa_j$, $sr_i$ dominates $sr_j$ in terms of service resource $sa$, denoted as $sr_i \geq_{sa} sr_j$, if and only if $sa_i \geq sa_j$.

Considering the service function’s importance, we further fuse the degree, diversity and betweenness of IoT service requests into priority group division. In this work, we assume that history data on the criteria ”degree”, ”betweenness” and ”diversity” of service functions are available through server logs saved in the repository or through other means, so that a personalized ranking model can be learned. Here, a user’s service request is a given service request chain, that is composed with some sequent function requirements in it.

- The degree of IoT service request functions: It is defined as the number of social activity function links where it engages in the historical service activity process. The higher the degree of IoT service request function is, the more likely it is associated with a function service chain path to finish the desired IoT service composition.
- The diversity of IoT service request functions: An IoT service that has a high diversity is involved in many different types of social service activities, and may have a broad importance in the service composition for different user service requests, thus having a higher chance to connect with other desired IoT service functions for different user requests.
- The betweenness of IoT service request functions: The betweenness reflects the probability that a given IoT service function stands at the critical intersection of multiple IoT service function chains. The higher the local betweenness is, the higher chance the IoT service function is located at the intersection that connects different function composition chains in the neighborhood.

To maximize the service utility for all users who have this kind of IoT service request function in their service compositions, the user’s service request function that has a higher weight in its service composition will be given a higher priority to be first satisfied by assigning an IoT service. Motivated by this, we consider the following importance for service request:

- Service Request’s Importance in Multiple Service Compositions: Check which important and urgent criteria with regard to their weights are wined by the service requests in the log records of service composition. The more weighted criteria summation the service request contains, the higher demand the request service is, thus more resource energy is consumed on the service that satisfies the request.
- Service Request’s Importance in Service Composition of Oneself: Check how important a criterion in terms of the request service is to someone’s whole service composition. A request service is ranked higher if it satisfies a higher weighted preference criterion in someone’s service composition.
- Service Request’s Importance in Both: If a request service contributes more to the final ranking score in term of both the importances mentioned above, it means it is preferred by the user’s service composition compared to other request services.

Therefore, to maximize the service utility for all the users, the request service that is higher in both importances should be first assigned an IoT device with enough energy resource to finish it. In the following, we give the specific definition to formalize the three kinds of importances.

Definition 9 Service Request’s Importance in Multiple Service Compositions (SRIMSC): Given a service request $sr_i$ and a set of attributes $A = \{a_1 : degree, a_2 : diversity, a_3 : betweenness\}$ that $sr_i$ has, let $W_{a_1,a_2}$ be a weighted factor of user $u_i$ ($i \in [1, m]$) semantic importance on $a_j$ ($j \in [1, 3]$). Some possible semantic weights are extremely important, very important, important, and somewhat important. The overall attribute aggregation summation endowed by the $m$
users can be described by an aggregation operator \( \otimes \) given as

\[
Ims(sr) = \sum_{i=1}^{m} I(u_i, sr) = \sum_{i=1}^{m} \otimes (W_{u_i, a_i} \cdot a_i)
\]  

(7)

If a service request has a higher \( Ims \) in server logs saved in the repository, it indicates that the service satisfying the request is needed by more service compositions, leading to more energy consumption of itself. Thus, the more weighted criteria summation the service request has, the higher importance the service request’ demand possesses.

**Definition 10 Service Request’s Importance in Service Composition of Oneself (SRISCO):** Given a set of service requests \( Q = \{sr_1, sr_2, \cdots, sr_n\} \) that a user \( u_i \)’s service composition has, let \( W_{u_i, sr_j} \) be a weighted factor of \( u_i \)’s semantic composition importance on \( sr_j \). Some possible semantic weights are extremely important, very important, important, and somewhat important. The importance of \( sr_i (i \in [1, N]) \) endowed by \( u_i \) can be described as

\[
Iss(sr_j) = \frac{W_{u_i, sr_j} \cdot sr_j}{\sum_{j=1}^{n} W_{u_i, sr_j} \cdot sr_j}
\]  

(8)

If a service request has a higher importance in its service composition, it indicates that the service request is important compared to other service requests for service composition.

**Definition 11 Service Request’s Importance in Both (SRIB):** Given the weighted summation for SRIMSC and SRISCI, the overall ranking aggregation endowed by the user can be described as

\[
Ibs(sr) = \alpha \cdot Ims(sr) + (1 - \alpha) \cdot Iss(sr)
\]  

(9)

If a service request has a higher \( Ibs \), it indicates that the service request with a higher priority requires to be first satisfied so that the service composition can maximize the service quality of all the users’ requirements.

**Definition 12 User Request Importance Domination (URID):** As can be seen, the service request function that has a higher importance indicates that it contributes to more than one service composition in server logs saved in the repository, resulting in more energy consumption of themselves for service execution. Given a function-varying service request \( sr_i \) with the function request importance \( sc_i \), and another service request \( sr_j \) with the function request importance \( sc_j \), \( sr_i \) dominates \( sr_j \) in terms of service function importance \( sc \), denoted as \( sr_i \succeq_{sc} s_j \), if and only if \( sc_i \geq sc_j \).

**Definition 13 User Weighted Attributes Domination (UWAD):** Given the spatial-temporal-function-resource available attributes \( a_1 = sl, a_2 = st, a_3 = sf, a_4 = sa \) and their weights \( w_1, w_2, w_3, w_4 \), a service request \( sr_1 = \{a_{11}, a_{12}, a_{13}, a_{14}\} \) dominates another service request \( sr_2 = \{a_{21}, a_{22}, a_{23}, a_{24}\} \) in terms of weight in \( [\theta_{min}, \theta_{max}] \), denoted as \( sr_1 \succ_{w} sr_2 \in [\theta_{min}, \theta_{max}] \), if (1) \( \forall i \in \{1, 2, 3, 4\} : a_{1i} \geq a_{2i} \), (2) \( \exists j \in \{1, 2, 3, 4\} : a_{1j} > a_{2j} \), and (3) \( \sum_{i=1}^{4} w_i \times a_{1i} - \sum_{i=1}^{4} w_i \times a_{2j} \in [\theta_{min}, \theta_{max}] \).

**Definition 14 Service Request Priority Group Division (SRPGD):** Given a set of user service requests \( S = \{sr_1, \cdots, sr_m\} \), four attributes \( a_1 = sl, a_2 = st, a_3 = sf, a_4 = sa \) and their weights \( w_1, w_2, w_3, w_4 \), as well as the n groups and their respective ranges \( [\theta_i_{min}, \theta_i_{max}] \), \( i \in [1, n] \), \( \theta_i_{max} < \theta_{i+1, min} \), if \( (sr_{pivot} \succ w sr_i) \in [\theta_{i, min}, \theta_{i,max}] \), then \( sr_i \) is divided into the \( \theta_i \) group, where \( sr_{pivot} \) is the service request not dominated by any other service request in \( S \) with respect to the four attributes.

For each service request group in descending order of ranked priority list, the following is to predict whether the service request \( sr \) is spatial-temporal-functional most matched to an ongoing or recent IoT service composition, which is critical for better understanding and addressing the following service composition search intent and facilitates to find the service composition in an accurate and fast way.

**B. PREFERENCE-AWARE SERVICE ALLOCATION SEARCH**

The process of IoT service selection for composition begins with a service request to the edge node. The composition process might involve a number of IoT devices in which each offers a particular type of function corresponding to each request function in the request chain. Moreover, the edge nodes are deployed practically in a certain region, which includes massive sensors or other devices. Usually, these sensors and devices are the same or similar types with close distance and approximate resource availability since the important and frequent social activities are usually provided by the IoT devices within the close regions. Therefore, it is desirable to seek one or more IoT devices within the management region of edge node to provide service request execution. A Plan Generator will then query the network’s available resources to determine their capabilities of enforcing the plan. The branch preference estimation, composition and adaptation score, and service quality score enable our solution to efficiently reduce the service search space by eliminating invalid branch nodes. In the following, we consider to query STSC-tree leveraging the preference, composition and adaptation score, and service quality score.

Given the current IoT service in the service function chain with specific spatial-temporal constraints, resource constraint, sequent correlated functional constraints, as well as personalized service features, the next IoT service or IoT service sub-chain is sought from the STSC-tree that is updated in the current period according to the frequent and important function interaction detection. When \( n \) service chains are given, it is required to seek \( m(\leq n) \) IoT services simultaneously or in a proximate time interval so as to maximize the service utility of all the service compositions. According to the priority of service requests in ranked list, the service requests in the front should be first satisfied to maximize the users’ service request utility because it gives more importance in reducing energy consumption, time delay and improving service quality in service composition. In the following, we use geographical-temporal preference model and recommendation technology to predict the performance
of service composition that is formed by combing the service request functions already allocated and one or several ongoing or recent IoT services in a certain branch node of STSC-tree.

**Definition 15 Group Branch Preference Estimation (GBPE):** Given a group of service requests \(sr(s_{r1}, \cdots, s_{rm})\) with validity in the service time interval \(s_r\), service region \(s_r\), and service resource amount interval \(s_a\) and a branch service region \(q(q_1, \cdots, q_d)\) with service availability in the service time interval \(q_t\), service region \(q_r\), and service resource amount interval \(q_a\), as well as another branch service region \(q(q'_{1}, \cdots, q'_{d})\) with service availability in the service time interval \(q'_t\), service region \(q'_r\), and service resource amount interval \(q'_a\), \(q\) is preferred to \(q'\) for \(sr\) if \(|s_r - q_t| \leq |s_r - q'_t|, |s_r - q_r| \leq |s_r - q'_r|\) and \((q_a - s_a) \geq (q'_a - s_a)\).

In this paper, we use collaborative filtering based score to calculate the composition and adaptation of IoT services formed through combining the function of \(u_i\)’s service request \(sr\) with an ongoing or recent IoT service functions \(sp\) from the preferent branch nodes of STSC-tree.

\[
S(sr, sp) = \sum_{u_i \in S(u_i)} \sim_{ij} \times \frac{1}{N_i} \sum_{h=1}^{l_p} \frac{i_{p_h} - t_s}{t_c - t_s} \times SF(fq_{j_p}, sr) \tag{10}
\]

where \(S(u_i)\) is the set of top-\(k\) similar users to \(u_i\). Two users are considered to be similar in high level if they select and invoke the same IoT service request, since they make the same decision on the weight of the criteria under similar service composition scenarios. \(N_i\) is the maximum number of invocations on an IoT service the system would save for the user. For each invocation on IoT service \(sp\) from \(u_i\), \(i_{p_h}\) represents the time of the \(h\)th invocation, \(t_s\) represents the starting time of the user log, \(t_c\) represents the current system time, \(l_p\) is the number of invocations on \(sp\) from \(u_i\), \(fq_{j_p}\) is the \(h\)th functional query for \(u_i\) over \(sp\). \(sr\) is \(u_i\)’s function query. \(SF(fq_{j_p}, sr)\) denotes the similarity of \(fq_{j_p}\) and \(sr\). The above calculation considers the invocations from the top-\(k\) similar users, and the current functional search.

If the composition and adaptation of the service request \(sr\) combined with an ongoing or recent IoT service functions \(sp\) satisfy the requirement, we strive to evaluate the service composition quality with regard to a certain user request.

**Definition 16 Service Quality Utility (SQU):** Let \(Q_s = (q_{dc}, q_{td}, q_{ts})\) represent the service quality values of a service composition \(s\) in terms of remaining energy consumption, sustainable time delay, and service matching. Let \(P_s = (w_1, w_2, w_3)\) be a user’s request preferences given to different service quality criteria of \(s\) with \(w_j \in R^+_0\) and \(\sum_{j=1}^{3} w_j = 1\). Then the service quality utility of \(s\) is derived as

\[
U(s) = Q_s \times P^T_s = \sum_{j=1}^{m} w_j \times q_j \tag{11}
\]

To this end, when a group of service requests with a higher priority arrives, it is required to seek the preferred branch nodes with regard to the minimization in terms of energy consumption and time delay by using Eq. 10 and Eq. 11.

**Algorithm 2 : CompositionServiceSearch**

**Require:** \(sr\): a group of user service requests
\(nd_{st}\): the root of STSC-tree
\(Q_r\): the group’s service region
\(Q_t\): the group’s service time interval
\(Q_a\): the group’s service resource interval
\(\theta_{score}\): threshold of the composition and adaptation score as well as service quality score

**Ensure:** \(S_{chain}\): the IoT service chain having the highest utility to \(sr\)

1: if \(nd_{st}\) is a leaf node then
2: \(S_{chain} \leftarrow \) allocate the composition services in \(nd_{st}\) according to the composition and adaptation score as well as service quality score
3: else
4: \(N_{nxc} \leftarrow \) ChildCompositionServiceSearch \((sr, nd_{st}, Q_r, Q_t, Q_a, \theta_{score})\)
5: \(N_{nxb} \leftarrow \) SiblingCompositionServiceSearch \((sr, N_{nxc}, Q_r, Q_t, Q_a, \theta_{score})\)
6: \(NN_{chain} \leftarrow (N_{nxc} \rightarrow N_{nxb})\)
7: \(S_{chain} \leftarrow \) CompositionServiceSearch\((sr, NN_{chain}, Q_r, Q_t, Q_a, \theta_{score})\)
8: end if

**Algorithm 3 : ChildCompositionServiceSearch**

**Require:** \(sr\): a group of user service requests
\(N_{nx}:\) a branch node of STSC-tree
\(Q_r\): the group’s service region
\(Q_t\): the group’s service time interval
\(Q_a\): the group’s service resource interval
\(\theta_{score}\): threshold of the composition and adaptation score as well as service quality score

1: \(C_{set} \leftarrow\) the child nodes of \(N_{nx}\)
2: \(PC_{set} \leftarrow \) GBPE\((c_i \in C_{set}, ws, sr, Q_r, Q_t, Q_a)\) ( \(i \in [1, |C_{set}|]\))
3: foreach \(N_{pi} \in PC_{set}\) do
4: \(\) Score\((cc) \leftarrow S(sr, N_{pi}) + U(sr, N_{pi})\)
5: if \(\) Score\((cc) \geq \theta_{score}\) then
6: \(\theta_{score} \leftarrow\) Score\((cc)\)
7: \(N_{nx} \leftarrow N_{pi}\)
8: end if
9: end for

Algorithm 2 presents main steps for searching the IoT services corresponding to a group of service requests. The parent nodes in each level of STSC-tree are used to carry on judgment and pruning (line 4-5) until the leaf nodes with the highest adaption and utility score are found (line 6).
Algorithm 4: SiblingCompositionServiceSearch

Require: sr: a group of user service requests
        \( N_{nx} \): a branch node of STSC-tree
        \( Q_r \): the group’s service region
        \( Q_t \): the group’s service time interval
        \( Q_u \): the group’s service resource interval
        \( \theta_{score} \): threshold of the composition and adaptation score as well as service quality score

Ensure: \( \text{c}_{nxb} \): the sibling node having the highest utility to sr

1. \( N_n \leftarrow \text{the sibling node of } N_{nx} \)
2. \( \text{Score}(cc) \leftarrow S(sr, N_{nx}, N_s) + U(sr, N_{nx}, N_s) \)
3. if \( \text{Score}(cc) > \theta_{score} \) then
4. \( N_{nxb} \leftarrow N_s \)
5. end if

FIGURE 4. User service request.

In the search process, the preferent child node is determined by spatial-temporal-resource-function adaptation (see Algorithm 3), while the preferent sibling node is determined by the function adaptation (see Algorithm 4) since it is the adjacent node in spatial-temporal-resource dimension. The time complexity of Algorithm 2 is \( O(\log m n) \), where \( n \) is the total number of IoT services and \( m \) is the minimum number of child nodes in a branch node.

Example 2: Here we use an example to illustrate the traversal process over STSC-tree, as shown in Fig. 3. When group division and priority of user service requests are determined, we traverse STSC-tree to achieve the service compositions of current service request \( sr \), as shown in Fig. 4. Suppose that these user service requests are divided into two groups \( gp_1 = \{sr_1, sr_2, sr_3, sr_4\} \) with a relatively higher priority and \( gp_2 = \{sr_5, sr_6, sr_7, sr_8, sr_9, sr_{10}\} \) with a relatively lower priority. When initiating the service composition search, the first is to consider the group’s service locations, timestamp and resource. We assume that \( R_1 \) is the preferent branch node for \( gp_1 \) compared to \( R_2 \) and \( R_3 \) through the estimated GBPE(\( gp_1, R_1, R_2, R_3 \), \( S(gp_1, R_1) \), \( U(gp_1, R_1) \)), while \( R_2 \) and \( R_3 \) are the preferent branch nodes for \( gp_2 \) compared to \( R_1 \) through the estimated GBPE(\( gp_2, R_1, R_2, R_3, S(gp_2, R_2, R_3) \), \( U(gp_2, R_2, R_3) \)). In what follows, leveraging the preference and estimation score, we can conclude that (1) \( sr_1, sr_2 \) prefer \( R_4 \), (2) \( sr_3, sr_4 \) prefer \( g_3 \), (3) \( sr_5, sr_6 \) prefer \( R_2 \), (4) \( sr_7, sr_8 \) prefer \( R_5 \), (5) \( sr_9, sr_{10} \) prefer \( R_6 \). Finally, we acquire the optimal service composition for each user service request from the leaf nodes. The symbol \( \rightarrow \) in Fig. 4 shows the search result for the initial service in the service chain corresponding to each service request \( sr \).

C. SERVICE EXECUTION WITH EDGE DEVICE

Based on the priority of service requests, all the high priority requests are first identified and IoT devices with a higher spatial-temporal-resource-function match are allocated to appropriate service requests. If there are any residual resources that remain unassigned, they are allocated to serve the low priority requests. It is worthy nothing that cooperation among a group of edge devices can adapt and manage the available services and resources of IoT nodes to reach the goal of composing a particular service and help execute it. Thus, if there are no enough energy resources, these service requests should be executed with the help of edge device by offloading some burden to it but suffering some transmission latency. It is able to ensure the normal IoT device allocation and make the service composition finished without impacted by resource availability. Due to more than one IoT device may offload some burden to edge device for service execution, there is a queue in sequence maintained in the edge device according to service request priority. In such a case, the goal of IoT device allocation is to maximize the service composition utility via finding appropriate pairs of IoT devices and service requests as well as increase the overall productivity in terms of resource utilization with the help of edge device.

VII. IMPLEMENTATION AND EVALUATION

A. EXPERIMENTAL SETUP

We use Beijing taxi cabs dataset [38] and the publicly available IoT dataset [39] to evaluate our proposed service allocation and composition method. Beijing taxi cabs dataset contains 2,325,708 check-in records at 920,023 activity regions from 207,232 taxi cabs. Therefore, the spatial-temporal information of taxi cabs and activity region is imported in our dataset. We divide the corresponding spatial region of activity regions into \( N \times N \) squared grid cells with the left-bottom coordinate as (100, 100) and right-upper coordinate as (1000, 1000). IoT dataset consists of several types of sensors/devices whose invoking data for nearly 2 weeks is recorded. The dataset contains basic information such as the types, locations and cooperation relationships of IoT devices in different scenarios. We place the IoT devices in different scenarios to different activity regions with location relativity and assign an edge device to manage them. Therefore, we can augment the amount of IoT devices to
9000 with the data format in [39] and extract top-20 frequent function interactions and their prevalences according to the check-in records at different activity regions. Moreover, we assume that each taxi cab has a service chain request with no more than 10 function services when it arrives at an activity region.

To model a realistic collaborative device-edge-cloud environment, we utilize the iFogSim [40] to model device-edge-cloud and network links environments, and measure the impact of service composition techniques. The major application model considered in iFogSim is the Sense-Process-Actuate model. The IoT object devices communicate with the edge devices via IEEE 802.11 a/g, and we also set the transmission rate to 54 Mb/s. The speed of links between edge devices in the same edge network is 100 Mb/s and the seed link from an edge device to cloud server is 10 Gb/s. All the index structures and algorithms are implemented by Java and performed on a laptop with an Intel(R) Core(TM) i5-5200U CPU 2.20GHZ, 8G memory, 64 bits operating system. Table 2 shows the parameters used in our experiments.

| Parameter | Value       |
|-----------|-------------|
| SOON      | 1000, 3000, 5000, 7000, 9000 |
| MSRN      | 100, 300, 500, 700, 900 |
| CSRN      | 10, 20, 30, 40, 50 |
| SCL       | 10, 20, 30, 40, 50 |
| 1         | [0.0, 30.10] |
| 2         | [30, 60], [60, 100] |
| 3         | [30, 60], [60, 100] |
| 4         | [30, 60], [60, 100] |
| 5         | [7:00-9:00] |
| 6         | [9:00-11:00] |
| 7         | [11:00-13:00] |
| 8         | [13:00-16:00] |
| 9         | [0.0, 30, 60], 8:00-9:00 |
| 10        | [30, 60], 11:00-13:00 |
| 11        | [0.0, 30, 60], 11:00-13:00 |
| 12        | [30, 60], 8:00-9:00 |
| 13        | [0.0, 30, 60], 8:00-9:00 |
| 14        | [30, 60], 11:00-13:00 |
| 15        | [0.0, 30, 60], 8:00-9:00 |
| 16        | [30, 60], 11:00-13:00 |

In the following, we carry on experiments for evaluating our approach in terms of energy consumption, composition time, response time and matching degree for both single user service request and multiple user service requests.

- The energy consumption for service composition is measured by Eq. 12, where $m$ is the energy consumption for message sending to the following IoT service, $rn$ is the energy consumption for messages receiving from the former IoT service, $n$ is the number of IoT services in the service composition chain, $a$ denotes the energy consumption constant of the transmit electronics, $b$ denotes the energy consumption constant of the transmit amplifier, $c$ denotes the energy consumption constant of the receiver electronics, $k$ denotes the number of bits in a packet, and $d$ denotes the transmission distance.

\[
E_s = \sum_{i=1}^{n} (m_i + r_i) = \sum_{i=1}^{n} (a + c) \times k \times b \times k \times d^n \tag{12}
\]

- The service composition time is measured as time interval required from the request sending to the result returning along STSC-tree.
- The respond time for service composition is measured by Eq. 13, where $tt$ is the messages transmission time to the following IoT service, $wt$ is the waiting time delay for the available following IoT service or the transmission delay to edge device, $et$ is the execution time of IoT service or edge device, $n$ is the number of IoT services in the service composition chain.

\[
T_s = \sum_{i=1}^{n} (tt_i + wt_i + et_i) \tag{13}
\]

- The satisfaction degree for service composition is measured as $D_{dg} = Mh(sr, sp)$, where $Mh$ is the matching degree between the functions of service composition $sp$ and service requirement $sr$.

Under these settings, this paper implements the following alternative techniques for the comparison purpose:

- **SL** [11]: It is a social network based approach for IoT device management and service composition.
- **CP** [10]: It is an efficient peer-to-peer service discovery solution for very large-scale IoT network with a mix of static and mobile IoT nodes.
- **PS** [9]: It is a particle swarm optimization method to search for optimal IoT service compositions considering the factors, such as spatial-temporal constraints, energy efficiency, and the configurability of IoT services.
- **MR** [14], [15]: It is a method to identify the common sub-requests for multiple user requests, and they can be executed once for fulfilling partial of these concurrent requests.
- **SI** [13]: It is a method that measures the cost of bringing services to the edge to optimize the average service response time by considering the resource consumption, popularity and the service composition graph.
- **SF**: It is the novel service composition search method proposed in this paper with the help of collaborative device-edge-cloud architecture.

### B. SERVICE COMPOSITION PERFORMANCE COMPARISON

#### 1) OVERALL EVALUATION

As can be seen from the figures shown in the experiment, our learned service composition allocation method leveraging spatial index performs best in service composition, especially for multiple user requests. The service requests’ priority and branch preference are more useful strategies benefiting the overall service composition performance. PS and MR are
approximately same in service composition performance for single user request, since there is no concurrent request. MR is more effective for multiple user requests since it considers the concurrent requests integration mechanism to optimize shared functionalities among concurrent requests. Meanwhile, they consume the most energy, composition time as well as response time on service composition compared with other approaches because the focus of them is the particle swarm optimization that requires to consider each service class chain as the particle and calculate the fitness function values for them. In what follows, SL generates better service composition performance than PS and MR, since it utilizes the location cluster, service type, and social correlation based approach to select and compose service in a parallel way for each user request. CP and ES further consider to utilize the edge cache to help service composition. As a result, they generate better service composition performance than SL, since the edge cache takes the role of caching the neighbors with proximity capability as well as reducing transmission delay.
2) EFFECT OF SMART OBJECT DEVICES NUMBER

Fig. 5 and Fig. 6 show the comparison of service composition performance for varying the number of smart object devices with single user request and multiple user requests, respectively. All the methods have an increasing trend on energy consumption and service composition time as the number of smart object devices grows up since more objects devices provide more selections for IoT service compositions. However, they have the approximate service response time since the service composition length is the same. As for the service matching degree, SL, CP and ES perform better since they consider the service capability most during the service composition no matter how many or what other service requests arrive. SI performs worst in service matching degree since it considers to maximize the service utility of all the service requests.
3) EFFECT OF SERVICE COMPOSITION LENGTH

As shown in Fig. 7 and Fig. 8, when varying the service composition length, these methods generate an increasing performance trend not only in terms of energy consumption and service composition time, but also with regard to service response time since the longer service composition chain contains more service functions. As for the service matching degree, service composition length generates important impact since more IoT services require to be chosen as the composition length grows up, resulting in low service function matching, especially when multiple user requests arrive.

4) EFFECT OF CONCURRENT REQUEST NUMBER

From the observation in Fig. 9, when varying the number of concurrent service requests, these methods except MR are not impacted by it since they do not distinguish concurrent or similar service requests. MR improves the service composition performance as the number of concurrent service requests.
increases since these service requests can be satisfied by the shared functionalities.

5) EFFECT OF SERVICE REQUEST REGION AND TIME
In this set of experiments, we test the service composition performance by distinguishing whether the given 100 service requests are in the same geographical region and time interval.

(1) All service requests are in the same geographical region with fixed time interval: When varying service request region (see Fig. 10 and Fig. 11), the farther request region suffers a longer transmission distance and requires more transmission time for service selection and composition, resulting in more energy consumption and response time. However, the service composition time and matching degree do not change with the request region since it is only a condition factor for service selection.

(2) All service requests are in the same time interval with fixed geographical region: Different from the performance trend changing with service request region, service request time does not have distance constraints on service composition. It is only a condition factor for service composition, resulting in the approximately same energy consumption, composition time, response time as well as matching degree (see Fig. 12 and Fig. 13).

(3) All service requests are in the same geographical region and time interval: Based on the performance analysis for changing request region and request time, the variant trend for changing request region and time simultaneously inherits the impacts from both request region and request time, thus having different energy consumption and response time, as well as the approximately same service composition time and matching degree (see Fig. 14 and Fig. 15).

(4) All service requests are divided into two groups and the service requests in the same group are within the same request region and time interval: In SI, the service requests group with a farther request region and an earlier request time is assigned with a higher priority and will be first allocated IoT services for maximizing the service utility of all user requests (see 13, 14, 16 in Table 2). With regard to these three kinds of service requests, SI generates the approximately optimized service composition performance compared to other methods. As for the service requests group with a farther request region and a later request time as well as a nearer request region and an earlier request time (see 15 in Table 2), their priorities are set through Definition 14. Compared to the request conditions 13, 14, 16, SI generates more approximately optimized service composition performance than that of other methods in terms of conditions 15 (see Fig. 16).

VIII. CONCLUSION
This paper proposes a novel IoT service composition method based on device-edge-cloud collaboration. Generally, the proposed composition mechanism is based on a spatial-temporal service allocation index, which is constructed according to the prevalence of function interactions of IoT services. During the allocation process, service request priority and branch preference estimation are used to help position the branch nodes with a higher utility as soon as possible. Edge device is also exploited to help service execution and guarantee service allocation utility by offloading task from IoT devices. Experimental results show that the proposed IoT service composition method promotes the service utility maximization in comparison with the alternative techniques. For the future work, we will combine the artificial intelligence technology with our proposed method to speed up the IoT service composition for multiple user service requests.

REFERENCES
[1] E. Sisinni, A. Saifullah, S. Han, U. Jenenchag, and M. Gidlund, “Industrial Internet of Things: Challenges, opportunities, and directions,” IEEE Trans. Ind. Informat., vol. 14, no. 11, pp. 4724–4734, Nov. 2018.
[2] E. Park, Y. Cho, J. Han, and S. J. Kwon, “Comprehensive approaches to user acceptance of Internet of Things in a smart home environment,” IEEE Internet Things J., vol. 4, no. 6, pp. 2342–2350, Dec. 2017.
[3] B. Xu, L. Da Xu, H. Cai, C. Xie, J. Hu, and F. Bu, “Ubiquitous data accessing method in IoT-based information system for emergency medical services,” IEEE Trans. Ind. Informat., vol. 10, no. 2, pp. 1578–1586, May 2014.
[4] S. Y. Shapsohugh and I. A. Zualkerman, “A generic IoT architecture for ubiquitous context-aware learning,” IEEE Trans. Learn. Technol., vol. 13, no. 3, pp. 449–464, Jul. 2020.
[5] Y. Ngoko and C. Cerin, “An edge computing platform for the detection of acoustic events,” in Proc. IEEE Int. Conf. Edge Comput. (EDGE), Jun. 2017, pp. 240–243.
[6] L. Xian, M. Wang, and W. Jiang, “Resource optimization service chain composition and deployment in IoT,” in Proc. IEEE World Congr. Services (SERVICES), Oct. 2020, pp. 29–32.
[7] W. Chen and I. Paik, “Toward better quality of service composition based on a global social service network,” IEEE Trans. Parallel Distrib. Syst., vol. 26, no. 5, pp. 1466–1476, May 2015.
[8] S. Wang, A. Zhou, M. Yang, L. Sun, C.-H. Hsu, and F. Yang, “Service composition in cyber-physical-social systems,” IEEE Trans. Emerg. Topics Comput., vol. 8, no. 1, pp. 82–91, Mar. 2020.
[9] M. Sun, Z. Shi, S. Chen, Z. Zhou, and Y. Duan, “Energy-efficient composition of configurable Internet of Things services,” IEEE Access, vol. 5, pp. 25609–25622, 2017.
[10] H. Mocini, I.-L. Yen, and F. Bastani, “Efficient caching for peer-to-peer service discovery in Internet of Things,” in Proc. IEEE Int. Conf. Web Services (ICWS), Jun. 2017, pp. 196–203.
[11] G. Chen, J. Huang, B. Cheng, and J. Chen, “A social network based approach for IoT device management and service composition,” in Proc. IEEE World Congr. Services, Jun. 2015, pp. 1–8.
[12] T. Li, T. He, Z. Wang, and Y. Zhang, “An approach to IoT service optimization for mass customization on cloud manufacturing,” IEEE Access, vol. 6, pp. 50572–50586, 2018.
[13] S. Deng, Z. Xiang, J. Yin, J. Taheri, and A. Y. Zomaya, “Composition-driven IoT service provisioning in distributed edges,” IEEE Access, vol. 6, pp. 54258–54269, 2018.
[14] Z. Zhou, J. Xu, Y. Zha, F. Lei, and W. Fang, “Energy-efficient optimization for concurrent compositions of WSN services,” IEEE Access, vol. 5, pp. 19994–20008, 2017.
[15] M. Sun, Z. Zhou, J. Wang, C. Du, and W. Gaolou, “Energy-efficient IoT service composition for concurrent timed applications,” Future Gener. Comput. Syst., vol. 100, pp. 1017–1030, Nov. 2019.
[16] S. K. Mishra, D. Puthal, J. I. P. C. Rodrigues, B. Sahoo, and E. Dutkiewicz, “Sustainable service allocation using a metaheuristic technique in a fog server for industrial applications,” IEEE Trans. Ind. Informat., vol. 14, no. 10, pp. 4497–4506, Oct. 2018.
[17] W. He, S. Guo, Y. Liang, R. Ma, X. Qiu, and L. Shi, “QoS-aware and resource-efficient dynamic slicing mechanism for Internet of Things,” Comput. Mater. Continua, vol. 61, no. 3, pp. 1345–1364, 2019.
[18] Z. Liu, X. Qiu, S. Zhang, S. Deng, and G. Liu, “Service scheduling based on edge computing for power distribution IoT,” Comput. Mater. Continua, vol. 62, no. 3, pp. 1351–1364, 2020.
[19] M. Boulakbech, N. Messai, Y. Sam, T. Devogele, and M. Hammoudeh, “IoT mashups: From IoT big data to IoT big service,” in Proc. Int. Conf. Future Netw. Distrib. Syst., Jul. 2017, pp. 1–5.

[20] B. Cheng, D. Zhu, S. Zhao, and J. Chen, “Situation-aware IoT service coordination using the event-driven SOA paradigm,” IEEE Trans. Netw. Service Manage., vol. 13, no. 2, pp. 349–361, Jun. 2016.

[21] X. Luo, H. Luo, and X. Chang, “Online optimization of collaborative Web service QoS prediction based on approximate dynamic programming,” in Proc. Int. Conf. Identificat., Inf. Knowl. Internet Things, Oct. 2014, pp. 80–83.

[22] R. Shao, H. Mao, and J. Jiang, “Time-aware and location-based personalized collaborative recommendation for IoT services,” in Proc. IEEE 43rd Annu. Comput. Softw. Appl. Conf. (COMPSAC), Jul. 2019, pp. 203–208.

[23] J. Li, J. Wang, Q. Sun, and A. Zhou, “Temporal influences-aware collaborative filtering for QoS-based service recommendation,” in Proc. IEEE Int. Conf. Services Comput. (SCC), Jun. 2017, pp. 471–474.

[24] S. Wang, Y. Ma, B. Cheng, F. Yang, and R. N. Chang, “Multi-dimensional QoS prediction for service recommendations,” IEEE Trans. Services Comput., vol. 12, no. 1, pp. 47–57, Jan. 2019.

[25] Z. Zhou, B. Wang, J. Guo, and J. Pan, “QoS-aware Web service recommendation using collaborative filtering with PGPT,” in Proc. IEEE Int. Conf. Web Services, Jun. 2015, pp. 392–399.

[26] Y. Pan and D. Wu, “Personalized online-to-offline (O2O) service recommendation based on a novel frequent service-set network,” IEEE Syst. J., vol. 13, no. 2, pp. 1599–1607, Jun. 2019.

[27] J. Chung, H. Jung, and U.-M. Kim, “An efficient top-K query processing method for location based services,” in Proc. Int. Conf. Adv. Image Process., Aug. 2017, pp. 201–204.

[28] Z. Li, K. C. K. Lee, B. Zheng, W.-C. Lee, D. Lee, and X. Wang, “IR-tree: An efficient index for geographic document search,” IEEE Trans. Knowl. Data Eng., vol. 23, no. 4, pp. 585–599, Apr. 2011.

[29] F. M. Choudhury, J. S. Culpepper, and Z. Bao, “Batch processing of top-k spatial-temporal queries,” in ACM Trans. Spatial Algorithms Syst., vol. 3, no. 4, pp. 13:1–13:40, 2018.

[30] R. Zhong, G. Li, K.-L. Tan, L. Zhou, and Z. Gong, “G-tree: An efficient and scalable index for spatial search on road networks,” IEEE Trans. Knowl. Data Eng., vol. 27, no. 8, pp. 2175–2189, Aug. 2015.

[31] O. Kucuktunc and H. Ferhatosmanoglu, “β-diverse nearest neighbors browsing for multidimensional data,” IEEE Trans. Knowl. Data Eng., vol. 25, no. 3, pp. 481–493, Mar. 2013.

[32] X. Cao, G. Cong, T. Guo, C. Jensen, and B. C. Ooi, “Efficient processing of spatial group keyword queries,” ACM Trans. Database Syst., vol. 40, no. 2, pp. 13:1–13:48, 2015.

[33] S. You, J. Zhang, and L. Gruenwald, “Parallel spatial query processing on GPUs using R-trees,” in Proc. 2nd ACM SIGSPATIAL Int. Workshop Anal. Big Geospatial Data (BigSpatial), 2013, pp. 23–31.

[34] J. Tang, B. Zhang, Y. Zhou, and L. Wang, “An energy-aware spatial index tree for multi-region attribute query aggregation processing in wireless sensor networks,” IEEE Access, vol. 5, pp. 2080–2095, 2017.

[35] A. Guttman, “R-trees: A dynamic index structure for spatial searching,” in Proc. ACM SIGMOD Int. Conf. Manage. Data, 1984, pp. 47–57.

[36] A. Forster, A. Forster, and A. L. Murphy, “Optimal cluster sizes for wireless sensor networks: An experimental analysis,” in Proc. Int. Conf. Ad Hoc Netw., 2009, pp. 1–15.

[37] S. Naeimi, H. Ghafraghizi, C.-O. Chow, and H. Ishii, “A survey on the taxonomy of cluster-based routing protocols for homogeneous wireless sensor networks,” Sensors, vol. 12, no. 6, pp. 7350–7409, May 2012.

[38] Z. Ding, Z. Chen, and Q. Yang, “IoT-SVKSearch: A real-time multimodal search engine mechanism for the Internet of Things,” Int. J. Commun. Syst., vol. 27, no. 6, pp. 871–897, Jun. 2014.

[39] E. M. Tapia, S. S. Intille, and K. Larson, “Activity recognition in the home using simple and ubiquitous sensors,” in Proc. Int. Conf. Pervas. Comput., 2004, pp. 158–175.

[40] H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, “IFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, edge and Fog computing environments,” Software, Pract. Exper., vol. 47, no. 9, pp. 1275–1296, Sep. 2017.