An Efficient Indexing Model for the Fog Layer of Industrial Internet of Things

Dejun Miao, Lu Liu, Rongyan Xu, John Panneerselvam, Yan Wu, Wei Xu

Abstract—Given the recent proliferation in the number of smart devices connected to the Internet, the era of Internet of Things (IoT) is challenged with massive amounts of data generation. Fog Computing is gaining popularity and is being increasingly deployed in various latency-sensitive application domains including industrial IoTs. However, efficient discovery of services is one of the prevailing issues in the fog nodes of industrial IoTs which restrain their efficiencies in availing appropriate services to the clients. To address this issue, this paper proposes a novel efficient multilevel index model based on equivalence relation, named the DM-index model, for service maintenance and retrieval in the fog layer of industrial IoTs to eliminate redundancy, narrow the search space, reduce both the traversed number of services and retrieval time, ultimately to improve the service discovery efficiency. The efficiency of the proposed index model has been verified theoretically and evaluated experimentally, which demonstrates that the proposed model is effective in achieving much better service discovery and retrieval performance than the sequential and inverted index models.

Index Terms—multilevel, indexing, industrial IoT, fog nodes, efficiency

1 Introduction

Fog computing, as an extension of cloud computing, is one of the emerging technologies of distributed computing which incorporates a fog layer between the datacentre and the end devices to provide supplements for Internet-based smart devices, transforming such resource-constrained devices into more powerful computing utilities. The globally deployed IoT smart objects are expected to reach 212 billion by the end of 2020 [1]. Beyond these predictions, McKinsey Global Institute reported that the number of connected machines (units) has grown by 300% in the last five years [1]. Cisco projects the expected number of interconnected devices to reach 50 billion by 2020 along with an estimation of the IoT market at $14.4 trillion. The impact of the IoT market on the global GDP has been estimated by McKinsey between $2.7 trillion and $6.2 trillion annually by 2025 [2][3]. In general, a multi-scale distributed service paradigm comprises massive number of data/services generated by heterogeneous smart devices, achieving effective service discovery is even complicated in such environments. Realising the most appropriate services for a given enquiry is often complicated in industrial IoT platforms comprising large numbers of fog nodes with numerous connected smart devices [4][5]. Thus, enhancing the resource discovery efficiency within the industrial IoT has been studied in many recent research works [6-12]. Most of these existing methodologies [6-8][10][12] mainly focus on efficient routing methods without considering efficient services index model, which restrains the efficiency of service discovery in industrial IoTs with massive stored services.

With this in mind, this paper proposes an efficient multilevel indexing model for fog nodes of industrial IoTs in order to enhance the time management during the service discovery and retrieval process. The performance and dependability of the developed index model have been evaluated both theoretically and experimentally against the sequential index and inverted index models. The main contributions of this article in terms of the attributes of the proposed model are listed as follows:

- The proposed model effectively reduces the redundancy among the fog nodes of industrial IoTs which narrows the searching space and the number of traversed services is maintained at a minimum level.
- The proposed indexing strategy offers adaptive deployment models for fog nodes under dynamic industrial IoT domains.
- Importantly, the service discovery efficiency among the fog nodes in a multilevel index model is significantly improved.

The rest of this paper is organised as follows: Section II summarises the related work and Section III describes the proposed index model for fog nodes in industrial IoTs. Section IV presents the theoretical evaluation, and the preliminary assessments and discussions are covered in Section V. Section VI concludes this paper along with our future research directions.

2 Related Works

2.1 Industrial Internet of Things

Given the fact that the recent proliferation of smart devices has resulted in extravagant generation of data and ser-

D. Miao, L. Liu (corresponding author) and J. Panneerselvam are with the College of Engineering and Technology, University of Derby, Derby, DE22 1GB, United Kingdom. E-mail: {d.miao, l.liu, j.panneerselvam}@derby.ac.uk
R. Yan is with School of Tourism, Yangzhou Polytechnic College, Yangzhou, 225009, P.R.China. xurongyan@hotmail.com
Y. Wu and W. Xu are with School of Computer Science and Telecommunication Engineering, Jiangsu University, Zhenjiang, 212013, P.R.China. wuyan04418@ujs.edu.cn; 501329385@qq.com
D. Miao is also with School of Automation and Automobile, Yangzhou Polytechnic College, Yangzhou, 225009, P.R.China

vices [13], different IoT architectures for service discovery have been recently proposed [14] – [18] to achieve better efficiency. Recent researches in IoTs have postulated the addition of an abstraction layer to the existing 3-layer IoT architecture [14] – [19]. A 5-layer architecture has been proposed by a few other recent research works [13][15 – 17] , which encompasses a business layer, application layer, middleware layer, transport abstraction layer and perception layer [17] accordingly.

Recently, the application of IoT in the industrial domain is gaining momentum. In this direction, the operation of complex industrial physical machinery is integrated with networked sensors and software applications for enabling "Industrial Internet" based on IoTs [20][21]. Furthermore, the Industrial Internet Consortium has been founded in March 2014 by AT&T, CISCO, General Electric, Intel, and IBM. Its objective is to foster and promote innovation-driven and industrial oriented use cases, which in turn drives the definition and development of a reference architecture for the Industrial IoTs [3]. Moreover, new strategic initiatives like Industry 4.0 has been proposed by the German government, which laid the foundation for the fourth industrial revolution with an estimated interconnection of over 21 billion devices with the Internet 2020 [22].

Data generated by the smart industrial devices are usually transmitted over the network infrastructure to the fog layer storage facility [14]. Therefore, it is pivotal to integrate efficient indexing mechanisms into the fog nodes of industrial IoT to minimise the traffic load exerted onto the communication infrastructure, and also to apportion the load among the integrated elements depending up on their purpose, capabilities and availabilities of the resources.

2.2 Fog of Everything

Based on the integration of the physical infrastructures of IoT with fog computing, fog of everything has emerged as a self-orchestrating eco-systems paradigm, aimed at providing the technical capabilities to the applications services such as industrial IoT, smart city and so on [23][24]. In general, the underlying architecture of a virtualised technological platform should necessarily encompass at least three main components such as IoT layer, fog layer and remote cloud layer, as shown in Fig. 1 [23][25].

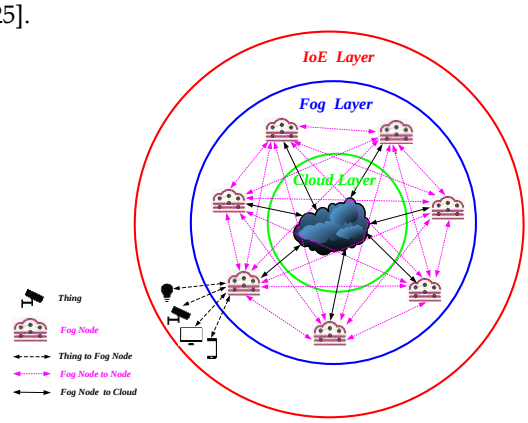


Fig.1 Underlying Architecture of Fog of Everything

It can be observed from Fig.1 that numerous heterogeneous devices like tablets, smartphones, RFID-based frequency tuners and smart industrial appliances operate over the multiple spatial clusters in the IoT layer [23][24][26]. Distributed fog nodes stay connected to the fog layer in order to facilitate a feasible and horizontal dynamic scaling of the inter-fog resource pooling. A fog node covers a spatial geographical area and serves as a cluster of things [27]. Finally, the remote cloud datacentre is connected to a set of fog nodes via Internet for facilitating more powerful computing and storage capability [25].

The network encompassed by the fog nodes opens up a few research issues. The current operational efficiency of the service discovery architecture in the fog layer is still far from entirely satisfying the user requirement, especially for latency-sensitive application requirements [28] – [30].

2.3 Indexing Model

Inverted index is regarded as a core data structure of information retrieval systems, especially in search engines [31]. Inverted index models can enhance the efficiency of the service discovery process by offering better response times than those of the sequential matchmaking processes, and by providing better recall rates than those approaches without an inverted index. However, it suffers from the disadvantages of extra space consumption for maintaining the frequency counts and constant reordering in the lookup table.

The current operational efficiency of the service discovery architecture in the fog layer is still far from entirely satisfying user satisfaction, especially for latency-sensitive requirements. This paper proposes a novel efficient indexing model that enables less latency incurred service discovery and easy service operations in fog computing for industrial IoT applications.

3 Multilevel Indexing Model

3.1 Definitions

In order to better explain the proposed multilevel indexing model and relevant concepts in a more, all the notations mentioned in this paper are presented as follows in Table 1.

Table 1 List of Notations

Notation	Meaning
s	service
P_{si}	input parameters set of service
P_{so}	output parameters set of service
P_{sr}	remaining parameters
Q	User request
Q_p	the parameters set provided by user
Q_r	the parameters set required by user
S	services set
L	a set of constraints for any attributes
R	relation
C_s	same-class set
P_{csi}	input parameters set of same-class
P_{cso}	output parameters set of same-class
C_{is}	semi-class set
P_{csi}	input parameters set of semi-class

P_{cso}	output parameters set of semi-class
P_{rs}	all the randomly selected input parameters set
I_r	the randomly selected input parameters list
E_s	the expectation of the traversed services for the sequential index
E_i	the expectation of the traversed services for the inverted index
P	the input parameter pool of all stored services and retrieval requests
P_i	the average number of input parameters of each stored service
E_{fl}	the expectation of the traversed services for the full DM-index model
E_{pr}	the expectation of the traversed services for the primary deployment model of DM-index
E_{pt}	the expectation of the traversed services for the partial deployment model of DM-index

Before leading into the description of the proposed novel indexing model, some basic definitions and theories should be introduced and analysed. As numerous heterogeneous smart devices are usually deployed in the perception layer of industrial IoT, it is essential to adopt a generalised definition for services generated by the involved smart devices, as shown in equation 1.

$$s = (P_{si}, P_{so}, P_{sr}) \qquad (1)$$

The input and output parameters set of service in industrial IoT are represented as P_{si} and P_{so} respectively, and all the remaining parameters are represented by A.

For a given user requirement, effectively retrieving the relevant resources from the fog nodes of industrial IoT is crucial. Thus, it is necessary to unify the definitions of user requirements, service retrieval and service discovery. Firstly, a user request can be simply denoted as in equation 2.

$$Q = (Q_p, Q_r) \qquad (2)$$

where, Q_p is the parameters set provided by the user, and Q_r is the parameters set required by the user.

Since service retrieval is an operation that accepts a set of parameters (Q_p) provided by users and returns a set of services those can be invoked by the user. Thus, service retrieval can be defined as shown in equation 3.

$$R(Q_p, S) = \{s | P_{si} \subseteq Q_p \wedge s \in S\} \qquad (3)$$

Similarly, service discovery is an operation that discovers required services from the invoked services; it can be defined as shown in equation 4.

$$D(Q_p, Q_r, L, S) = \{s | R(Q_p, S) \wedge Q_r \subseteq P_{so} \wedge s \angle L \wedge s \in S\} \qquad (4)$$

where, L is a set of constraints for any attributes, S is a service set, $s \angle L$ reflects that s can satisfy the constraints and the symbol "$\subseteq$" denotes their subset relationship.

3.2 Equivalence Theory

Relationship is a very general definition. In general, ordered pairs is simply a list of elements those are related. For instance, the notation $(a, b) \in A$, can also be denoted as aRb and it simply means that a is in relation with b, and whatever the relation R can be.

In mathematics, an equivalence relation may take various perspectives such as a binary relation, a reflexive relation, a symmetric relation and a transitive relation defined as below [32].

(a) $\forall x \in \mathrm{A}$, if xRx, then relation R on A is reflexive [32]. In other words, every element is in relation with itself.

(b) $\forall x, y \in A$ if xRy implies yRx, then relation R on A is symmetric [32]. It means that x is related to y, then y is related to x.

(c) $\forall x, y, z \in A$, if xRy and yRz implies xRz, then relation of R on A is transitive [32].

In terms of digraphs, reflexivity is the characteristics of including at least a loop on each vertex; symmetry means that any arrow from one vertex to another will always be accompanied by another arrow in the opposite direction; and transitivity insists that there must be a direct arrow from one vertex to another if one can walk from that vertex to the other through a list of arrows, always travelling along the direction of the arrows [33].

As a consequence, an equivalence relation provides a partition of a set into equivalence classes. A partition of set A is a collection of non-empty, pairwise disjoint subclasses which should cover all the components of set A [32]. To prove R has an equivalence relation on a set A, the equivalence classes of R should form a partition of set A. For a given element x in set A, the equivalence class of x to be the set of all elements of A that are equivalent to x, which can be notated as $[X]$. In symbols, $[X] = \{ y \in \mathrm{A} | xRy \}$.

Firstly, for $x \in [X]$, it is obvious that no equivalence class is empty.

Secondly, if x and y are in different equivalence sub classes, then $(x, y) \notin R$ and then $[X] \cap [y] = \emptyset$.

Finally, for any element $x \in A$, it is obvious that $x \in [X]$, and so the union of all equivalence classes covers A.

The equivalence partition should ensure that every element contained in set S can be classified into a unique equivalence subclass, and any two different equivalence subclasses are disjointed. Existence of zero-intersection between equivalent subclass ensures that there is no redundancy duplicate; full coverage of equivalent subclass upon equivalent elements ensures integrity in the subset classes.

3.3 Classifying the Same Service

From the works of [4][5], services in a fog node can be invoked according to the input or output parameter set. It is often common that a couple of services are always associated with the same input and output parameters stored in the fog nodes of industrial IoTs. The number of services retrieved in industrial IoTs are usually more than those actually required by the user. This can significantly reduce the service retrieval and discovery efficiencies.

Based on the prior definitions, a service set S in a fog node of industrial IoTs can be divided into several subclasses by the equivalence relation R_1, and each subclass is named as same-class C_s. Thus, there is a surjection function between S and C_s, as indicated in equation 5.

$$f_1: S \rightarrow C_s \qquad (5)$$

Therefore, each same-class C_s has a unique pair of input and output parameters set, denoted as P_{csi} and P_{cso}. The same-class C_s set and the original service set S of a fog node can be indexed as shown in Fig. 2.

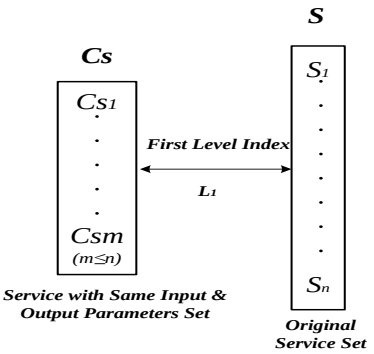


Fig. 2. The First Level Index

For a service retrieval request $Q(Q_p, Q_r)$, it can be clearly observed that only same-class C_s set needs to be retrieved $R(Q_p, C_s)$ instead of all the stored services in the fog nodes of industrial IoT. Normally, the count of same-class C_s is smaller than that of the original service set S, thus the retrieval efficiency should be better. However, if there are no or very few services sharing the same parameter set, the first level index would have no or little effect to improve efficiency. Thus in such a case, the deployment of the first level index should be avoided in practice.

3.4 Classifying the Partial Same Service

A clustered same-class C_s set may contain several elements characterising the same input parameters. Similar to the earlier discussion, assuming R_2 as an equivalence relation on same-class C_s set, which could be further divided into several subclasses and each subset is called as semi-class C_{is} containing at least one or more same-classes having common input parameters in this step. Thus a surjection function is formed between $C_s$and C_{is}, which is denoted in equation 6.

$$f_2: C_s \rightarrow C_{is} \quad (6)$$

The second level index is postulated to be integrated between C_{is} and C_s in this step, as shown in Fig. 3. It can be observed that from Fig. 3 that the retrieval request $R(Q_p, S)$ only focuses on semi-class C_{is} set rather than same-class C_s set with the help of the second level index. Generally, the count of semi-class C_{is} set is smaller than that of the same-class C_s set. The searching space can be further narrowed without redundancy. If there are no or only a few services sharing the same input parameter set, the second level index would have no or little effect on efficiency. In this case, the deployment of the second level index should be ignored [4][5].

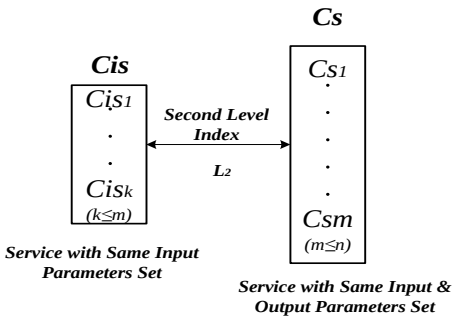


Fig. 3. The Second Level Index

3.5 Selecting Input Parameter

Since the fog layer of industrial IoT is a distributed paradigm, an optimised Distributed Hash Table (DHT) algorithm should be deployed to provide efficient services across the entire fog layer. Generally, one input parameter (I_r) for a given service will be randomly selected to generate its hash ID. Although the hash matching and lookup process are out of scopes of this paper, the randomly selected input parameter (I_r) does satisfy $I_r \in P_{cisi} \in P_{csi} \in P_{si}$. For the fog nodes of industrial IoT, various services might be added with same randomly selected input parameters. Therefore, the semi-class C_{is} set should be further divided into many subclasses based on randomly selected parameters. Thus, the third level index is formed between the semi-class C_{is} set and the new P_{rs} set including all the randomly selected input parameters. The elements in the P_{rs} set should be indexed to at least one semi-class C_{is}. As all the elements of C_{is} could be mapped by P_{rs}, there is a surjection function between C_{is} and P_{rs} as denoted in equation 7.

$$f_3: C_{is} \rightarrow P_{rs} \quad (7)$$

The overlay view of all the three levels of indexes developed is shown in Fig. 4.

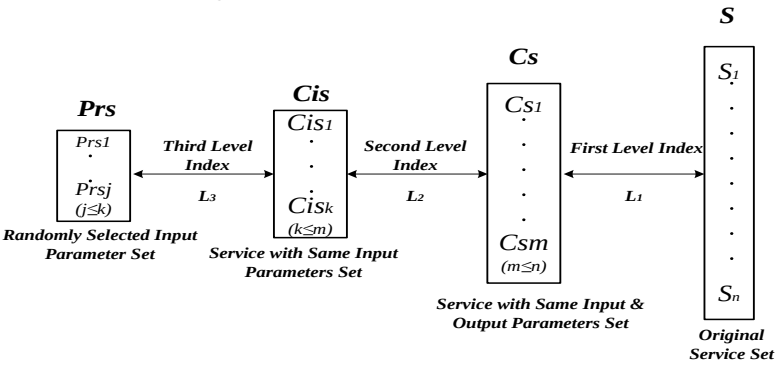


Fig. 4. The Third Level Index

3.6 DM-index Model

As the P_{rs} set may not be used for retrieval directly, an inverted index is structured between the randomly selected input parameters of I_r and P_{rs} set as a fourth level index. Redundancy in the proposed inverted index model is maintained at the minimum possible level, since every input parameter is indexed with only one element in the

P_{rs} set. Therefore, the function f_4 between I_r and P_{si} is a surjection function which can be denoted as in equation 8.

$$f_4: P_{rs} \rightarrow I_r \qquad (8)$$

The complete structure of the developed multilevel indexing model, named DM-index, for the fog nodes of industrial IoT is illustrated in Fig. 5.

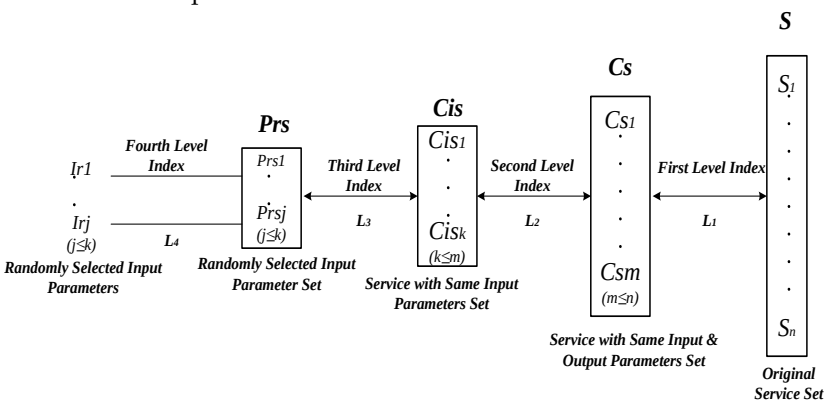


Fig. 5. The Multilevel Full Index

3.7 DM-index Operations

For a given service discovery request, the retrieval process in the fog nodes with the DM-index model is summarised as follows.

Step 1: Check whether Q_p can match at least one of the listed I_r or not. If matched, go to step 2, otherwise return 'not found'.

Step 2: Retrieve elements in P_{rs} set indexed by I_r.

Step 3: Retrieve semi-classes C_{is} indexed by P_{rs}, only those can satisfy $P_{cisi} \subseteq Q_p$ are chosen. If found, go to step 4, otherwise, return 'not found'.

Step 4: Similar to Step 3, only the indexed same-class C_s those can satisfy $Q_r \subseteq P_{cso}$ are chosen. If found, go to step 5, otherwise, return 'not found'.

Step 5: Discover and return s from the indexed same-class C_s.

Based on the service retrieval definition and $f_1 - f_4$ four functions, the computation of $D(Q_p, Q_r, L, S)$ algorithm with the DM-index is illustrated in Algorithm 1.

Algorithm 1: *Service Discovery with DM-index Model*

$I_r^{'} = Q_p \cap I_r$
$P_{rs}^{'} = \{P_{rs} | P_{rs} \in f_4^{-p}(I_r^{'})\}$
$C_{is}^{'} = \{C_{is} | C_{is} \in f_3^{-p}(P_{rs}^{'}) \wedge P_{cisi} \subseteq Q_p\}$
$C_s^{'} = \{C_s | C_s \in f_2^{-p}(C_{is}^{'}) \wedge Q_r \subseteq P_{cso}\}$
$s = D(Q_p, Q_r, L, f_1^{-p}(C_s^{'}))$

To insert a newly generated service into the matched fog node with the DM-index model, the first step is to identify whether the service s is existing or not. If there is no matching, the new service s is added to the existing service set S, same-class C_s set, semi-class C_{is} set, P_{rs} set and I_r list respectively in a step by step fashion. If there is a matching, the algorithm will check P_{si} set and I_r list for further operations. The algorithm for service addition in the fog nodes with the DM-index is illustrated in Algorithm 2.

Algorithm 2: *Service Addition with DM-index Model*

I_{rs} *is the randomly selected input parameter for service s hashing*
if $s \in S$
  service already existing
else if $s \notin S$ && $P_{si} \in P_{cisi}$ && $I_{rs} \in I_r$
  {
  $S = S \cup s$
  $C_s = C_s \cup s$
  }
else if $s \notin S$ && $P_{si} \notin P_{cisi}$ && $I_{rs} \in I_r$
  {
  $S = S \cup s$
  $C_s = C_s \cup s$
  $C_{is} = C_{is} \cup P_{si}$
  }
else if $s \notin S$ && $P_{si} \notin P_{cisi}$ && $I_{rs} \notin I_r$
  {
  $S = S \cup s$
  $C_s = C_s \cup s$
  $C_{is} = C_{is} \cup P_{si}$
  $P_{rs} = P_{rs} \cup I_{rs}$
  $I_r = I_r \cup I_{rs}$
  }

The delete operation of a service s from the fog nodes with a randomly selected input parameter I_{rs}, is illustrated in Algorithm 3.

Algorithm 3: *Service Deletion with DM-index Model*

$S = S - s$
if $|S| == 0$
  delete all C_s *set,* C_{is} *set,* P_{rs} *set and* I_r *list*
else $C_s = f_1(S)$
  $C_{is} = f_2(C_s)$
  if $\exists P_{rsi} \in P_{rs}$ $f_3^{-p}(P_{rsi}) \cap C_{is} == \emptyset$
  {
  $I_r = I_r - f_4(P_{rsi})$
  $P_{rs} = P_{rs} - P_{rsi}$
  }

3.8 Adaptive Deployment

As mentioned in prior sections, the first level index of the proposed DM-index model aims to remove redundancy induced by services characterising the same input and output parameters. If the fog nodes of industrial IoTs con-

tain only a few services characterising the same input and output parameters, the first level index yields no or less effect. In some cases, the first level index is not needed which means that services are directly indexed to semi-classes (C_{is}) according to their input parameters as shown in Fig.6, this is termed as the partial deployment.

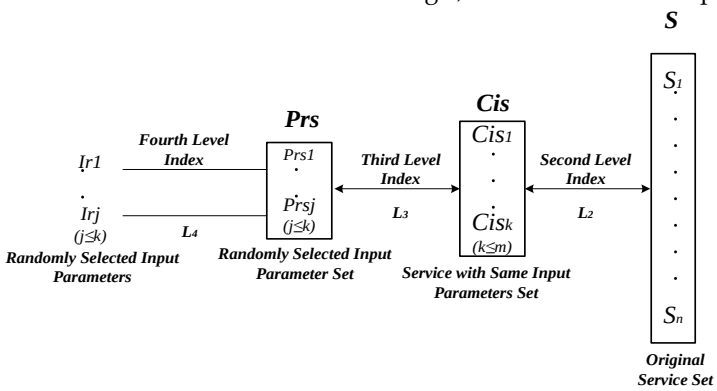


Fig. 6. The Multilevel Indexing Model Partial Deployment

Similar to the first level index, the second level index aims to remove redundancy induced by services characterising the same input parameters. If the fog nodes of industrial IoT contain very few services characterising same input parameters, the second level index yields no effect. Thus, services directly link to P_{rs} set according to their input parameters as shown in Fig.7. This is named as the primary deployment.

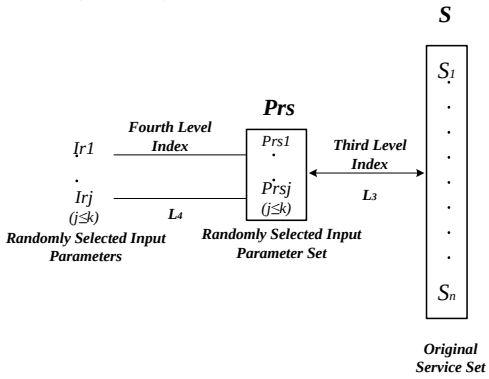


Fig. 7. The Multilevel Indexing Model Primary Deployment

If a fog node comprises only a very few services, those share the same input parameters, the partial index may be slightly slower than the primary index due to the complicated structures of former. For the same reason, the efficiency of the full DM-index may be slightly lower than those of the partial indexes. The deployment model selection is explained as follows.

Step 1: Deploy full DM-index model

Step 2: If $|S| \approx |C_s| \gg |C_{is}|$, deploy partial model,

Step 3: If $|S| \approx |C_s| \approx |C_{is}|$, deploy primary model.

Determination of the threshold is considered to be out of scope of this paper. Algorithm 4 illustrates the adaptive deployment model selection process.

***Algorithm 4*:** Adaptive Deployment

1 *Deploy full DM-index model*

2 *If* $|S| \approx |C_s| \gg |C_{is}|$ *choose partial DM-index model*

3 *If* $|S| \approx |C_s| \approx |C_{is}|$ *choose primary DM-index model*

4 Theoretical Validation

The sequential index model is also known as a non-index model, in which all the services are stored in a sequential structure, such as a list. For a given retrieval request, all the stored services $S = \sum_1^n s_i$ in a fog node will be traversed and checked whether they match the request or not. The expectation of the traversed services in the sequential index (E_s) equals to the number of all stored services $|S|$ in the fog node of industrial IoTs, as shown in equation 9.

$$E_s = |S| \qquad (9)$$

Let E_i denote the expectation of the traversed services for the inverted index and $P = P_{i1} \cup P_{i2} \dots P_{in-1} \cup P_{in} \cup Q_{r1} \cup Q_{r2} \dots Q_{rm-1} \cup Q_{rm}$ denotes the input parameter pool of all the stored services and the retrieval requests, $P_i \in P$ denotes the average number of input parameters of each stored service s_i in the fog node and $Q_r \in P$ denotes the average number of input parameters of each retrieval request. The expectation of the inverted index (E_i) can be defined as in equation 10.

$$E_i = \frac{P_i \times Q_p}{|P|} \times |S| \qquad (10)$$

where, $P_i \times |S|/P$ implies that the average number of services linked by one input parameter, and $P_i \times Q_p \times |S|/|P|$ denotes that the number of services expected to be traversed for a retrieval request. If $P_i \times Q_p > |P|$, its efficiency will be less than the sequential index.

E_{fl} denotes the expectation of the traversed services for the full DM-index model. It can be defined as in equation 11.

$$E_{fl} = |Q_p \cap I_r| \times \frac{\overline{|C_s|}}{|C_{is}|} \qquad (11)$$

where, $\frac{\overline{|C_s|}}{|C_{is}|}$ represents the average amount of services stored in a semi-class C_{isj}. For a given retrieval request $Q(Q_p, Q_r)$, $|Q_p \cap I_r|$ reflects that the number of matched input parameters is between the user-provided parameters Q_p and the existing parameters contained in the fourth index. All the parameters of Q_p will be retrieved in the worst circumstance. For $\frac{\overline{|C_s|}}{|C_{is}|} \leq S$ and $|Q_p \cap I_r| \leq Q_p$, the proposed multilevel indexing model should achieve much better service retrieval efficiency than the sequential index and traditional inverted index models when the

fog nodes comprise massive number of partially similar or same services.

As described in section 3, the full DM-index model can be replaced by the primary model or partial model when the fog nodes contain no or only a few services sharing the same input and output parameters sets or input parameters sets, respectively. For the primary deployment model, the expectation is denoted by E_{pr} and can be calculated using equation 12.

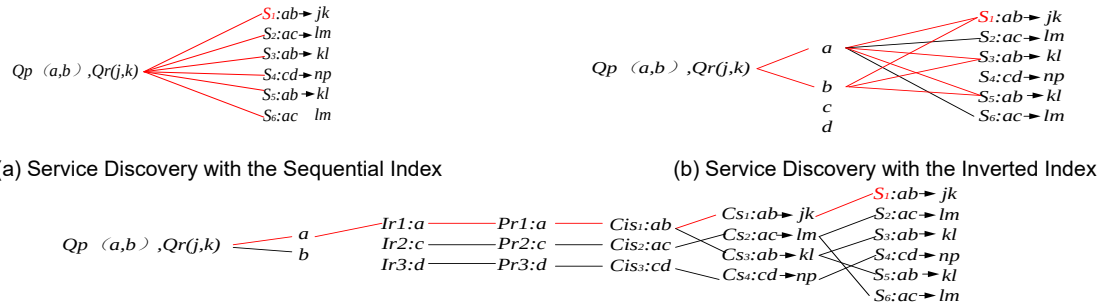
$$E_{pr} = |Q_p \cap I_r| \times \frac{|S|}{|I_r|} \qquad (12)$$

As there are no same-class C_s set or semi-class C_{is} set contained in the primary deployment model, the expectation of the primary deployment model E_{pr} only relates to the number of randomly selected input parameters list I_r. The average number of services indexed by each input parameter is donated as $\frac{|S|}{|I_r|}$. Thus, the number of randomly selected input parameters in I_r list is equal to the size of P_{rs} set in the third level. For a given retrieval request $Q(Q_p, Q_r)$, we have $|Q_p \cap I_r| = |Q_p \cap P_{rs}| \leq |I_r|$ even in the worst circumstance. Therefore, the efficiency of the primary index is not less than the sequential model.

For the partial deployment model, the expectation can be represented by E_{pt} and estimated using equation 13.

$$E_{pt} = |Q_p \cap I_r| \times \frac{|S|}{|C_{is}|} \qquad (13)$$

As there are no same-class C_s set in the partial model, the average number of services maintained by each input-similar subclass is $\frac{|S|}{|C_{is}|}$. The efficiency of the partial index may range between the primary index E_{pr} and the full level E_{fl}, since $|C_{is}| \geq |I_r|$ and $|S| \geq |C_s|$.



(a) Service Discovery with the Sequential Index

(b) Service Discovery with the Inverted Index

(c) Service Discovery with DM-Index

Fig. 8 Example of Service Discovery with Different Index Models

For a given user request $[Q_p(a,b), Q_r(j,k)]$, Fig. 8 illustrates an example of the service retrieval progress of the fog nodes in industrial IoTs with different index models. Fig. 8(a) proves that all the stored services are retrieved by the sequential index. Fig. 8(b) shows that a few services are traversed more than the one by the inverted index. Fig. 8(c) presents that only one relevant service is traversed by the proposed DM-index. Thus, the proposed DM-index is efficient in reducing the number of traversed services for a given query than the sequential and inverted index models.

5 Experimental Evaluation

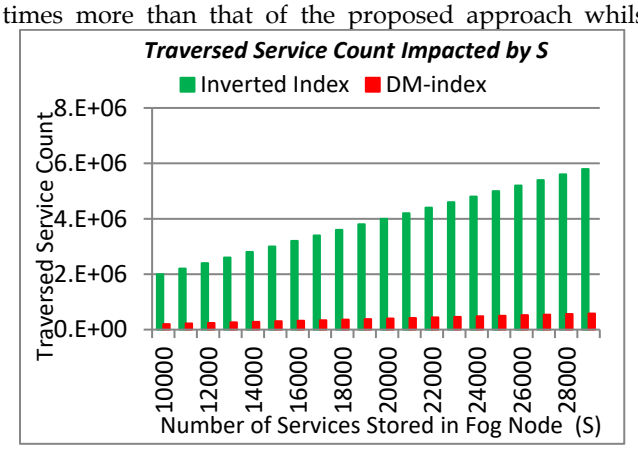
This section evaluates the efficiencies of the proposed DM-index model by validating the correctness of the previous theoretical analysis of expectation in terms of the traversed service counts and retrieval time. ICEBE05 [35] is a publically available test set that is generated based on WSDL, as used in [36]–[39]. It contains two test sets, composition1 and composition2 that both include 81464 services. From prior formulas, the expectations of the traversed service count of the sequential index is only based on S, inverted index and the full DM-index are based on S, P, P_i and Q_p. Therefore, the efficiencies of the sequential index has not been considered in experiments. The initial values for the number of stored services (S) , the average number of input parameters of each stored services (P_i), the average number of input parameters of retrieval request (Q_p) and the input parameter pool of all stored services and retrieval requests (P) are set as 10000, 100, 10, and 10, respectively. We run every experiment for 50 times based on 200 retrieval requests.

5.1 Impact of the Number of Stored Services in Fog Node

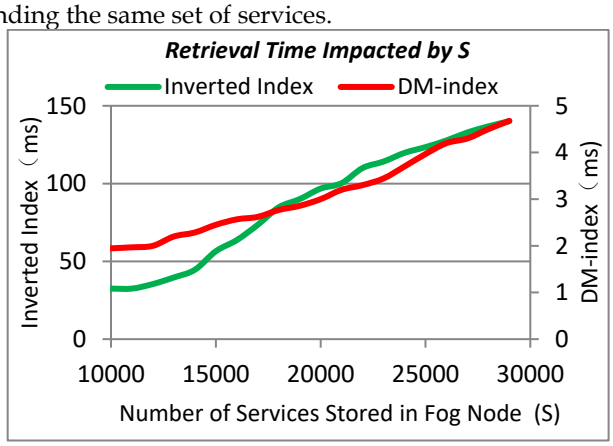
To evaluate the impact of the number of stored serivces in the fog node, the value of S is changed from 10,000 through to 29,000 with an increase of 1,000 in each iteration, and all the other parameters remain constant. Fig. 9(a) presents that the expectation of the inverted index $E_i \propto S \times 200$ is simple and directly proportional to S. Since the average number of services stored in each semi-class ($\frac{|C_s|}{|C_{is}|}$) is impacted by S ($\frac{|C_s|}{|C_{is}|} \propto |S|$), E_{fl} increases accordingly depending upon S. These results are in accordance with the theoretical analysis and further demonstrate that the DM-index model traverses much fewer number of services than those of the sequential index and inverted index model, despite the increase in the value of S.

Fig. 9(b) depicts the total retrieval time for inverted index and DM-index under a changing value of S. It can be observed from Fig. 9(b) that the proposed approach only consumes less than 6ms to retrieve all the services in any circumstances. Meanwhile, the inverted index consumes significantly higher retrieval time, witnessed at 25-30

times more than that of the proposed approach whilst finding the same set of services.



(a) Traversed Service Impacted by S



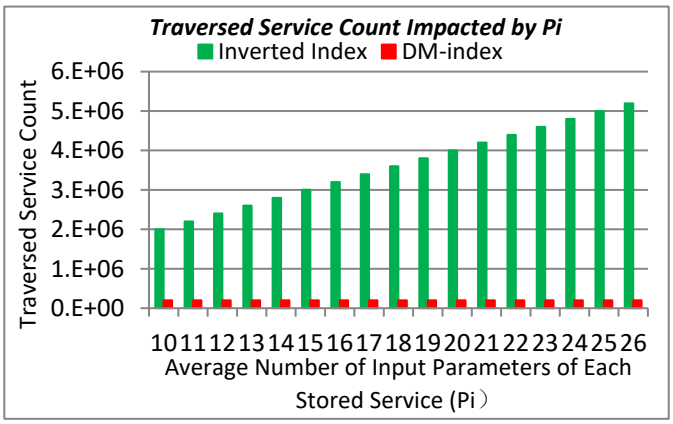
(b) Retrieval Time Impacted by S

Fig. 9 Impact of the Number of Stored Services in Fog Nodes

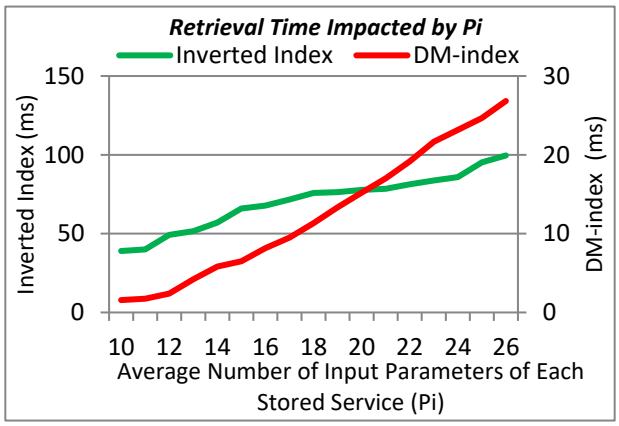
5.2 Impact of the Average Number of Input Parameters of Each Stored Service

To evaluate the impact of the average number of input parameters of each stored service, the value of P_i is changed from 10 through to 29 with an increase of 1 in every iteration and all the other parameters remain constant. Experimental results presented in Fig. 10(a) further confirm that E_i is directly proportional to P_i. Since P_i increases with constant S, and $\frac{|C_S|}{|C_{iS}|}$ further reduces, the obtained results prove that P_i does not affect E_{fl} to any notable level. Similar to the first experiment, the DM-index model traverses much fewer services than the inverted index models when P_i keeps increasing.

From Fig. 10(b), it is evident that the DM-index is exhibiting an increasing trend than that of the inverted index in terms of the incurred retrieval time. This suggests that the inverted index may achieve better retrieval time efficiency than the DM-index model because of its less maintenance burden. The results of this experiment presented in Fig. 10(b) further confirm that the retrieval time of the two models are directly proportional to P_i. Similar to the prior experiments, the proposed efficient indexing model costs much less time than the inverted index when there is an increase in the value of P_i.



(a)Traversed Service Impacted by P_i



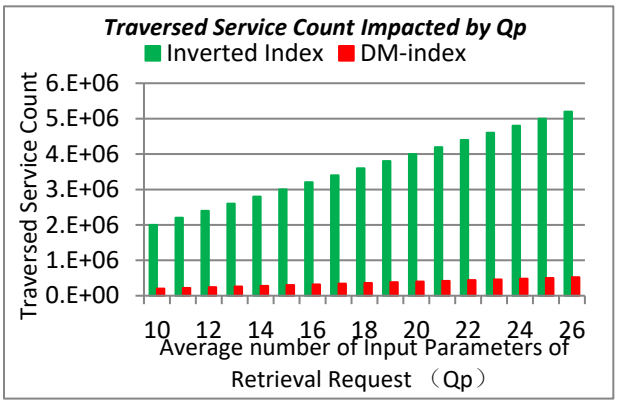
(b) Retrieval Time Impacted by P_i

Fig. 10 Impact of the Average Number of Input Parameters of Each Stored Service

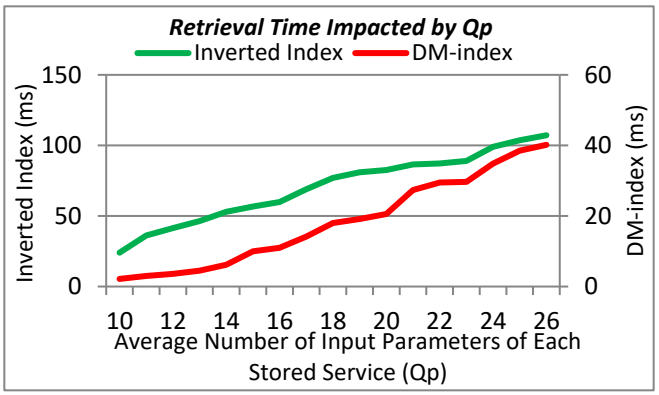
5.3 Impact of the Average Number of Input Parameters of Each Retrieval Request

To evaluate the impact of the average number of input parameters of each retrieval request, in the third experiment, the value of Q_p is changed from 10 through to 29 with an increase of 1 in every iteration and all the other parameters remain constant. The experiment results presented in Fig. 11(a) confirm that E_i and E_{fl} impacted by Q_p is very relevant to the theoretical expectations and the DM-index model traverses much fewer services.

It can be observed from Fig. 11(b) that the retrieval time of both the DM-index model and the inverted index model are exhibiting increasing trends for increasing values of P_r. However, the retrieval time of the proposed approach is much lower than that of the inverted index model. Similar to the two previous experiments, the proposed DM-index model achieves better retrieval efficiency than the sequential index and the inverted index models, despite the changes in the value of Qp.



(a) Traversed Service Impacted by Q_p



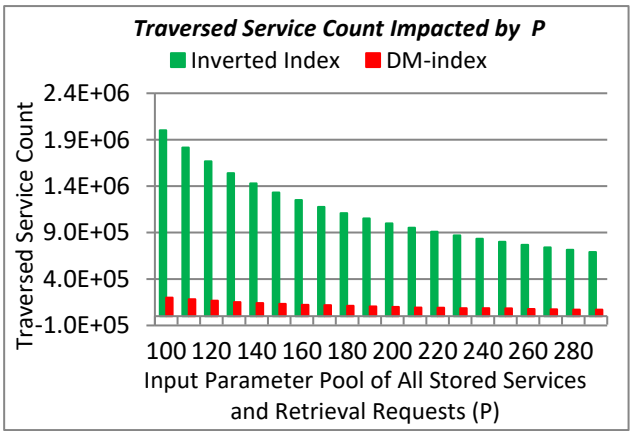
(b) Retrieval Time Impacted by Q_p

Fig. 11 Impact of the Average Number of Input Parameters of Each Retrieval Request

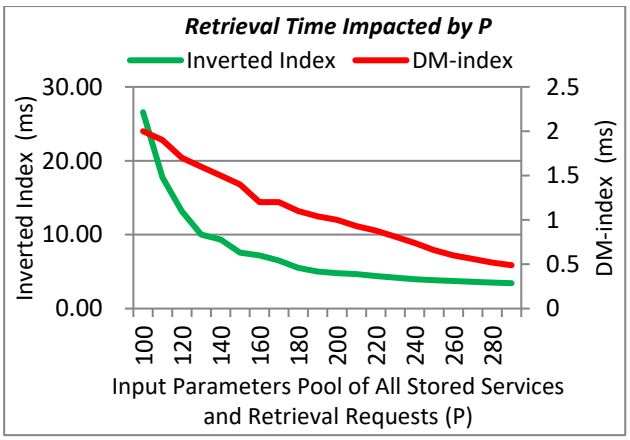
5.4 Impact of the Input Parameter Pool of All Stored Services and Retrieval Requests

In the final experiment, the impact of the input parameter pool of all stored services and retrieval requests is analysed by changing the values of P from 100 through to 290 with an increase of 10 in each iteration. Fig. 12(a) presents that E_i and E_{fl} are inversely proportional to P and the proposed index model achieves better efficiency with a less number of traversed services.

Fig. 12(b) presents that the changes in the value of P does not affect the efficiency of the proposed approach to any notable level. Further the retrieval time of the inverted index model is inversely proportional to P. Thus, it can be concluded that the proposed DM-index model is outperforming the inverted index model.



(a) Traversed Service Impacted by P



(b) Retrieval Time Impacted by P

Fig.12 Impact of the Number of Input Parameter Pool of All Stored Services and Retrieval Requests

6 Conclusion

This paper proposes a novel efficient multilevel indexing model for fog nodes of industrial IoT. Both the conducted theoretical and experimental evaluations demonstrate that the developed DM-index model is more efficient for fog nodes comprising massive stored data, in comparison with the sequential and inverted index models. The study concludes that the DM-index model can achieve much better efficiency than the inverted index and the sequential index. Specifically, the inverted index model incurs increased traversed service counts and higher retrieval time. Both are significantly worse than the performance of the DM-index. As a future work, we plan to extend our developed indexing model in order to balance the load among the nodes in the fog layer in the industrial fog computing systems, to specify the users' requirements and to identify the threshold for multilevel indexing model adaptive deployment.

Acknowledgement

This work was partially supported by the National Natural Science Foundation of China under Grants No. 61502209 and 61502207, Natural Science Foundation of Jiangsu Province under Grant BK20170069 and UK-China Knowledge Economy Education Partnership. Lu Liu is the corresponding author.

References

[1] U. S. Profile, 'The Digital Universe In 2020: Big Data, Bigger Digital Shadows, and Biggest Growth in the Far East — United States', pp. 1–7, 2013.

[2] J. Manyika, M. Chui, J. Bughin, R. Dobbs, P. Bisson, and Marrs, 'Disruptive technologies: Advances that will transform life, business, and the global economy', *McKinsey Glob. Insitute*, no. May, p. 163, 2013.

[3] V. Gazis, A. Leonardi, K. Mathioudakis, K. Sasloglou, P. Kikiras, and R. Sudhaakar, 'Components of fog computing in an industrial internet of things context', *2015 12th Annu. IEEE*

Int. Conf. Sensing, Commun. Netw. - Work. SECON Work. 2015, pp. 37–42, 2015.

[4] Y. Wu, C. G. Yan, L. Liu, Z. J. Ding, and C. J. Jiang, 'An Adaptive Multilevel Indexing Method for Disaster Service Discovery', *IEEE Trans. Comput.*, vol. 64, no. 9, pp. 2447–2459, 2015.

[5] Y. Wu, C. Yan, Z. Ding, G. Liu, P. Wang, C. Jiang, and M. C. Zhou, 'A Multilevel Index Model to Expedite Web Service Discovery and Composition in Large-Scale Service Repositories', *IEEE Trans. Serv. Comput.*, vol. 9, no. 3, pp. 330–342, 2016.

[6] I. Stoica, I. Stoica, R. Morris, D. Karger, R. Morris, D. Karger, M. F. Kaashoek, H. Balakrishnan, M. F. Kaashoek, and H. Balakrishnan, 'Chord: A scalable peer-to-peer lookup service for internet applications', *Proc. 2001 Conf. Appl. Technol. Archit. Protoc. Comput. Commun. - SIGCOMM '01*, pp. 149–160, 2001.

[7] S. Ratnasamy, P. Francis, M. Handley, S. Shenker, and R. Karp, 'A Scalable Content-Addressable Network'.

[8] A. Rowstron and P. Druschel, 'Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems', *Middlew. 2001*, vol. 2218, no. November 2001, pp. 329–350, 2001.

[9] S. Misra and S. Sarkar, 'Theoretical modelling of fog computing: a green computing paradigm to support IoT applications', *IET Networks*, vol. 5, no. 2, pp. 23–29, 2016.

[10] M. Aazam and E. N. Huh, 'Dynamic resource provisioning through Fog micro datacenter', *2015 IEEE Int. Conf. Pervasive Comput. Commun. Work. PerCom Work. 2015*, pp. 105–110, 2015.

[11] M. Aazam and E. N. Huh, 'Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT', *Proc. - Int. Conf. Adv. Inf. Netw. Appl. AINA*, vol. 2015–April, pp. 687–694, 2015.

[12] C. T. Do, N. H. Tran, C. Pham, M. G. R. Alam, J. H. Son, and C. S. Hong, 'A proximal algorithm for joint resource allocation and minimizing carbon footprint in geo-distributed fog computing', *Int. Conf. Inf. Netw.*, vol. 2015–Janua, pp. 324–329, 2015.

[13] A. Al-fuqaha, S. Member, M. Guizani, M. Mohammadi, and S. Member, 'Internet of Things : A Survey on Enabling', vol. 17, no. 4, pp. 2347–2376, 2015.

[14] S. Krco, B. Pokric, and F. Carrez, 'Designing IoT architecture(s): A European perspective', *2014 IEEE World Forum Internet Things, WF-IoT 2014*, pp. 79–84, 2014.

[15] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, 'Future internet: The internet of things architecture, possible applications and key challenges', *Proc. - 10th Int. Conf. Front. Inf. Technol. FIT 2012*, pp. 257–260, 2012.

[16] Zhihong Yang, Yufeng Peng, Yingzhao Yue, Xiaobo Wang, Yu Yang, and Wenji Liu, 'Study and application on the architecture and key technologies for IOT', *2011 Int. Conf. Multimed. Technol.*, pp. 747–751, 2011.

[17] M. Wu, T. J. Lu, F. Y. Ling, J. Sun, and H. Y. Du, 'Research on the architecture of Internet of Things', *ICACTE 2010 - 2010 3rd Int. Conf. Adv. Comput. Theory Eng. Proc.*, vol. 5, pp. 484–487, 2010.

[18] M. A. Chaqfeh and N. Mohamed, 'Challenges in middleware solutions for the internet of things', *Proc. 2012 Int. Conf. Collab. Technol. Syst. CTS 2012*, pp. 21–26, 2012.

[19] F. Paganelli and D. Parlanti, 'A DHT-Based Discovery Service for the Internet of Things', *J. Comput. Networks Commun.*, vol. 2012, pp. 1–11, Oct. 2012.

[20] 'Challenges for database management in the internet of things', vol. 26, no. March, pp. 320–329, 2013.

[21] P. Zhang, Z. Chen, J. K. Liu, K. Liang, and H. Liu, 'An efficient access control scheme with outsourcing capability and attribute update for fog computing', *Futur. Gener. Comput. Syst.*, 2016.

[22] H. (Deutsche P. A. Henning, Kagermann(National Academy of Science and Engineering). Wolfgang, Wahlster (German Research Center for Artificial Intelligence). Johannes, 'Recommendations for implementing the strategic initiative INDUSTRIE 4.0', *Final Rep. Ind. 4.0 WG*, no. April, p. 82, 2013.

[23] E. Borgia, 'The internet of things vision: Key features, applications and open issues', *Comput. Commun.*, vol. 54, pp. 1–31, 2014.

[24] S. V. Vandebroek, 'Three pillars enabling the Internet of Everything: Smart everyday objects, information-centric networks, and automated real-time insights', *Dig. Tech. Pap. - IEEE Int. Solid-State Circuits Conf.*, vol. 59, pp. 14–20, 2016.

[25] M. Scarpiniti, 'Fog of Everything : Energy-efficient Networked Computing Architectures , Research Challenges , and a Case Study Fog of Everything : energy-efficient networked computing architectures , research challenges , and a case study', vol. 5, no. May, 2017.

[26] J. Bradley, 'Internet of Everything (IoE) Value Index from IoE in 2013 ?', pp. 1–25, 2013.

[27] A. Asadi, Q. Wang, and V. Mancuso, 'A Survey on Device-to-Device Communcation in Cellular Networks', *Commun. Surv. Tutorials, IEEE*, vol. 16, no. 4, pp. 1801–1819, 2014.

[28] D. Bernstein, 'Containers and cloud: From LXC to docker to kubernetes', *IEEE Cloud Comput.*, vol. 1, no. 3, pp. 81–84, 2014.

[29] S. Soltesz, H. Pötzl, M. E. Fiuczynski, A. Bavier, and L. Peterson, 'Container-based operating system virtualization', *ACM SIGOPS Oper. Syst. Rev.*, vol. 41, no. 3, p. 275, 2007.

[30] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M. F. Kaashoek, F. Dabek, and H. Balakrishnan, 'Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications', *{IEEE} Trans. Netw.*, vol. 11, pp. 1–14, 2003.

[31] W. Jung, H. Roh, M. Shin, and S. Park, 'Inverted index maintenance strategy for flashSSDs : Revitalization of in-place index update strategy', *Inf. Syst.*, vol. 49, pp. 25–39, 2015.

[32] 'Kolman, Busby & Ross, Discrete Mathematical Structures, 6th Edition'. [Online]. Available: https://www.pearson.com/us/higher-education/program/Kolman-Discrete-Mathematical-Structures-6th-Edition/PGM104913.html. [Accessed: 03-Oct-2017].

[33] 'Discrete Mathematics'. [Online]. Available: http://staff.scem.uws.edu.au/cgi-bin/cgiwrap/zhuhan/dmath/dm_readall.cgi?page=18&part=2. [Accessed: 06-Dec-2017].

[34] K. E. S. Desikan, M. Srinivasan, and C. S. R. Murthy, 'A Novel Distributed Latency-Aware Data Processing in Fog Computing-Enabled IoT Networks', *Proc. ACM Work. Distrib. Inf. Process. Wirel. Networks - DIPWN'17*, pp. 1–6, 2017.

[35] 'http://www.comp.hkbu.edu.hk/~ctr/wschallenge/'. [Online]. Available: http://www.comp.hkbu.edu.hk/~ctr/wschallenge/. [Accessed: 23-Dec-2017].

[36] S. C. Oh and S. R. T. Kumara, 'Effective Web Services Composition in Diverse and Large-Scale Services Networks', *Smrr*, vol. 1, no. 1, pp. 15–32, 2006.

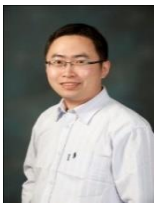
[37] D. Lee, J. Kwon, S. Lee, S. Park, and B. Hong, 'Scalable and efficient web services composition based on a relational database', *J. Syst. Softw.*, vol. 84, no. 12, pp. 2139–2155, 2011.

[38] R. Hewett, P. K. Sanayothin, and B. Nguyen, 'Scalable optimized composition of web services with complexity analysis', *2009 IEEE Int. Conf. Web Serv. ICWS 2009*, pp. 389–396, 2009.

[39] S.-C. Oh, D. Lee, and S. R. T. Kumara, 'Web Service Planner (WSPR)', *Int. J. Web Serv. Res.*, vol. 4, no. 1, pp. 1–22, 2007.

Dejun Miao received the MSc degree from University of Southampton, UK, and the BSc degree from Nanjing University of Science & Technology, China. He is currently working towards the Ph.D. degree at the Department of Electronics, Computing and Mathematics, the University of Derby, UK. He is also an Associate Professor in the School of Automation and Automobile at Yangzhou Poly-

technic College His research interests include Internet of Things, Fog Computing , Service Computing and P2P computing.



Lu Liu is the Professor of Distributed Computing in the University of Derby, UK and adjunct professor in Jiangsu University, China. Prof. Liu received his Ph.D. degree from University of Surrey, UK and M.S. degree from Brunel University, UK. Prof. Liu's research interests are in the areas of Cloud computing, social computing, service computing and peer-to-peer computing. He is the Fellow of British Computer Society (BCS).

Rongyan Xu is an Associate Professor in the School of Tourism at Yangzhou Polytechnic College. Mrs Xu received her first MSc degree in E-tourism from University of Surrey and second MSc degree in Computing Science from University of Bath. Her current research interests include Information Management, E-business in Tourism, Intelligent Tourism and Big Data Analytics.

John Panneerselvam received his Ph.D. from the University of Derby and currently is a Lecturer in Computing at the University of Derby, United Kingdom. His current research is focused on energy efficient cloud systems and he has published his recent research works in notable peer reviewed international conferences, journals and as book chapters. He is an active member of IEEE and British Computer Society, and his research interests include Cloud Computing, Big Data Analytics, Opportunistic Networking and P2P Computing.

Yan Wu received the M.S. degree from Shandong University of Science and Technology, Qingdao, China, in 2009, and the PhD degree from Tongji University, Shanghai, China, in 2014. He is currently a Lecturer with the School of Computer Science and Telecommunication Engineering in Jiangsu University, China. His research interests include formal methods, service-oriented Computing, and Cloud Computing.

Wei Xu is a graduate research student in the School of Computer Science and Telecommunication Engineering at Jiangsu University. She received her BSc degree from Binzhou University in China. Her research interests include Distributed Computing, Cloud Computing and P2P Computing.