A multi-layer Internet of things database schema for online-to-offline systems

Cai, H. , Luan, S. , Jiang, L. , Shah, N. , Farmer, R. and Chao, K.-M.

Original citation:
Cai, H. , Luan, S. , Jiang, L. , Shah, N. , Farmer, R. and Chao, K.-M. (2016) A multi-layer Internet of things database schema for online-to-offline systems. International Journal of Distributed Sensor Networks, volume 12 (8): 1550147716664248

URL: http://dx.doi.org/10.1177/1550147716664248
DOI: 10.1177/1550147716664248

Publisher: SAGE

Research Article

International Journal of Distributed Sensor Networks

International Journal of Distributed Sensor Networks
2016, Vol. 12(8)

DOI: 10.1177/1550147716664248
ijdsn.sagepub.com
SAGE

A multi-layer Internet of things database schema for online-to-offline systems

Hongming Cai[1], Shuai Luan[1], Lihong Jiang[1], Nazaraf Shah[2], Ray Farmer[2], Kuo-Ming Chao[2] and Boyi Xu[3]

Abstract

Due to the widespread usage of Internet of things devices in online-to-offline businesses, a huge volume of data from heterogeneous data sources are collected and transferred to the data processing components in online-to-offline systems. This leads to increased complexity in data storage and querying, especially for spatial–temporal data processing in online-to-offline systems. In this article, first, we design a multi-layer Internet of things database schema to meet the diverse requirements through fusing spatial data with texts, images, and videos transferred from the sensors of the Internet of things networks. The proposed multi-layer Internet of things database schema includes logical nodes, geography nodes, storage nodes, and application nodes. These data nodes cooperate with each other to facilitate the data storing, indexing, and querying. Second, a searching algorithm is designed based on pruning strategy. The complexity of the algorithm is also analyzed. Finally, the multi-layer Internet of things database schema and its application are illustrated in a smart city construction project in Shanghai, China, recommending available charging points to the customers who need to charge their electric energy–driven cars.

Keywords

Spatial data, Internet of things application, data storage, query processing, multi-layer Internet of things database schema

Date received: 31 March 2016; accepted: 5 July 2016

Academic Editor: Antonio Lazaro

Introduction

Online to offline (O2O) is a business model, which tries to attract the customers from surfing online to the physical stores to make their transactions offline.[1] The usage of Internet of things (IoT) technologies in O2O can achieve more accurate results by targeting customers through analyzing their offline data. It can also improve the loyalty of the consumers by providing more frequent interactions between consumers and business scenarios.

However, in IoT-based O2O applications, complex geography-related contexts are usually involved in the systems to communicate with the customers from which the products/services could be obtained. As most geography-related data in O2O applications are collected by IoT sensors, the data are usually heterogeneous and weak in semantic details which can hardly meet the requirements of complex enquiries with rich semantics. Therefore, it is important to propose methods to combine the IoT data with context data to support complicated O2O applications.

[1]School of Software, Shanghai Jiao Tong University, Shanghai, China
[2]Faculty of Engineering and Computing, Coventry University, Coventry, UK
[3]College of Economics and Management, Shanghai Jiao Tong University, Shanghai, China

Corresponding author:
Lihong Jiang, School of Software, Shanghai Jiao Tong University, Shanghai 200240, China.
Email: jianglh@sjtu.edu.cn

In this article, a multi-layer IoT database schema is proposed to represent the data involved in O2O applications from the viewpoints of geography, application, and storage-related data to achieve flexible information composition. Our proposed approach takes into account the properties of the IoT data used in O2O applications. A storage solution combining SQL, NoSQL, and distributed files is designed to support variable granulation of data accessing. The storage solution will be more flexible in scalability to support complicated user queries.

Related work

In recent years, many efforts have been performed in the field of IoT data representation and storage. Related work could be divided into three areas: (1) IoT data representation and collection, (2) heterogeneous data fusion, and (3) data storage supporting systems.

IoT data representation and collection

The IoT sensor data have the following features: (1) due to the quick data generation through sensors, IoT systems have huge data throughputs; (2) IoT systems usually need high scalability to process huge volume of data; and (3) since IoT data are often collected from multiple sources and collected data could be structured and unstructured data, IoT application systems need to be able to handle the challenges related to the data collection, transmission, and storage stages.[2]

Compressed sensing is one of the popular methods used in the recent years in data collection stage. In compressed sensing method, data are directly compressed by the sensors while collecting, instead of compressing after collection. Compressed sensing is more suitable for data processing of large number of cheap and simple sensors. S Li et al.[3] agreed that this kind of data compression is an effective approach and built a compressed sensing architecture for IoT applications. In this architecture, terminal nodes are used to compute, transmit, and store sensor data. Li proposed sparse representation classification algorithm to improve energy usage efficiency and data collection accuracy.[3] In Fazel et al.,[4] random access compressed sensing method is proposed to obtain greater throughput and to save energy in long-term deployed sensor networks. In Wang et al.,[5] wireless sensor network (WSN) is integrated with radio frequency identification (RFID) using a five-layer architecture to reduce data redundancy.

In addition to the compressed sensing model, there is research work on data representation and data fusion to support IoT data collection. Some data representation models support flexible IoT data collection by defining a unified data format. For example, heterogeneous event processing (HEP) is one of the data representation frameworks that integrate data relations with Extensible Markup Language (XML) event flows to achieve higher system scalability.[6] Virtual object (VO) is another data representation model that adds contexts to physical objects in IoT systems.[7]

Other than data representation models which aim to describe more information about the objects in IoT systems, data fusion methods intend to provide a more understandable and meaningful data view to IoT users through data integration and analysis. The research work in Liu et al.[8] describes a model called heterogeneous data integration model (HDIM) which is designed not only to integrate multi-source data but also to provide the users whole data views without heterogeneity.

In order to realize application systems focusing on different geography zones, some information exchange languages are proposed to facilitate the communication between different IoT devices. For example, Physical Markup Language (PML) is designed to support information exchanges between physical objects and their environments.[9] SensorML is an XML coding model to improve the interoperability of the sensors in IoT.[10] In order to achieve heterogeneity support between wireless sensors, service-oriented middleware (SOM) is also used in Mohamed and Al-Jaroodi[11] to design WSN-oriented applications.

The above related work discussion shows that data collection is one of the important components in IoT systems. IoT data collection is a challenging task due to the data heterogeneity among IoT devices. With the application of IoT technology in O2O systems, data representation methods combining semantic description and context analysis need to be researched in more detail.

Heterogeneous data fusion

In order to consider geography information with context information simultaneously, many researchers have paid attentions to data fusion methods. Method of searching keywords with location information organizes spatial data and non-spatial data separately. In this kind of method, R-tree is used to retrieve spatial data, while reverse index is built to retrieve context data. However, keyword search method has a low efficiency, no matter which kind of information is searched first, spatial, or context.

Following the idea of keyword searching, in Li et al.,[12] IR-tree is combined with R-tree to calculate context and location correlation. The advantage of IR-tree is that it could sufficiently create joining of context data and location data. But the complexity of IR-tree retrieval is comparatively high. In Rocha-Junior et al.,[13] a R-tree is built to reduce the I/O operation time for highly accessed keywords. In Zhang et al.,[14]

the I3 method is proposed to organize spatial and content data in a unified storage space. A pruning algorithm is implemented to improve the efficiency of data accessing. Although I3 method strictly divides the storage space using a quadtree to improve the algorithm efficiency, it still has some limitations in its practical use.

Zhang and Li[15] proposed a grid-based geography data engine to search spatial data in a distributed environment. Both Liao and Peng[16] and Lu et al.[17] studied the methods of synchronously accessing geography information and content information. In Liao and Peng,[16] a cluster method is proposed to process contexts with specific location restrictions. In Lu et al.,[17] a hybrid index tree named Intersection-Union R-tree (IUR-tree) is designed to compute content similarity. In their method, location information and content information are considered simultaneously. In addition to spatial dimension, temporal dimension is also involved in Shen et al.[18] to search static and dynamic WSN data by calculating spatial and temporal similarities.

From the abovementioned research efforts, it could be seen that in IoT applications, how to index and retrieve content and spatial information together has attracted the attention of many researchers. Solutions have been proposed to store and access these two kinds of information flexibly according to the user requirements. However, the efficiencies of these methods still need to be improved.

Data storage supporting systems

In order to support the use of hybrid databases, system architectures need to be designed differing from those used to support single database. In Yaish et al.,[19] a multi-tenant-oriented data storage system is implemented to support traditional relational database and virtual relational database. Research by Su and Swart[20] supports the running of local Hadoop and connects rational database with MapReduce and realizes both kinds of databases through complex SQL commands. Although NoSQL databases have the advantages of high scalability, distributed storage, and dynamic structure, they have the disadvantage of weak ACID (Atomicity, Consistency, Isolation, Durability) property. Some researchers adopted extended NoSQL databases to store IoT data. For example, in Curé et al.,[21] NoSQL database incorporated with ontology is used to improve data query efficiency. Other kinds of databases such as Bigtable are used to handle continuously changing data from the sensor networks.[22]

Some issues such as data security and privacy have attracted attention of many researchers in the field of Internet-based cooperative data sharing and accessing, especially for cloud computing environment.[23–25] However, with the application of IoT in O2O commerce, mixed usage of structured databases together with unstructured data and sometimes distributed files systems has brought more challenges to researchers.[26] In Dede et al.,[27] MongoDB is deployed with Hadoop to improve the efficiency, scalability, and tolerance of the systems, merging structured databases with distributed files. In Jiang et al.,[2] rational databases, NoSQL databases, and Hadoop are used together to handle structured and unstructured information.

From the above analysis, it could be concluded that due to the special characteristics of the IoT data, hybrid solutions are the most used options to design IoT systems. In recent researches, methods to create indexes to the hybrid data storage systems are also explored to optimize the access efficiency.

Summarizing the literature from IoT data representation, heterogeneous data fusion, and data storage support, it could be seen that in recent years, storage and accessing methods combining location and content information are the important issues attracting more research attentions in designing O2O application systems. Although so far more efforts were focused on how to accurately handle geography information and how to process large amounts of heterogeneous data, little work have been done on how to utilize IoT data which are weak in semantic meanings in O2O-oriented applications which are rich in semantic contexts. Therefore, the motivation of this article is to propose a multi-layer IoT database schema to support complex information retrieval in O2O applications.

Multi-layer IoT architecture for O2O application

In order to deal with the heterogeneous data in O2O applications, we have designed a multi-layer IoT database schema and demonstrated how it could be used in O2O systems, as shown in data layer of Figure 1.

In Figure 1, O2O systems include components of IoT device interface, data storage, data management, and O2O services supporting.

Through IoT device interfaces, data are collected from sensors, RFID devices, and GPRS (General Packet Radio Service) devices by IoT terminals and transferred through the Wi-Fi, Ethernet, and 4G wireless networks into the IoT data management components.

O2O services supporting components are the front ends of IoT data presentation. Various kinds of business scenarios, such as smart communities, smart buildings, smart shopping districts, and smart cities, could be supported by providing functions such as data analysis, data accessing priority management, smart scene operation, and message pushes.

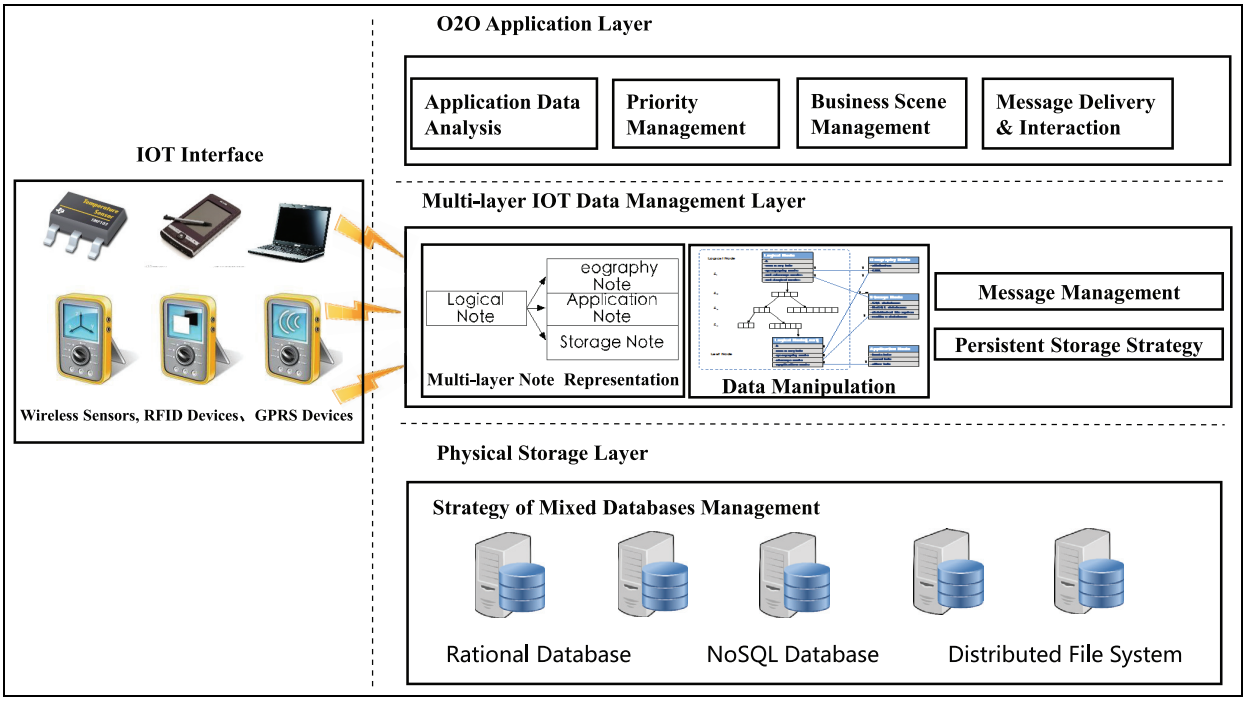

Figure 1. Architecture of IoT data management and application based on the multi-layer database schema.

IoT data management components correspond to integration of data from various IoT devices connected to data storage layer and meet the requirements from application layer. A multi-layer database schema is designed to handle the problem of data heterogeneity in O2O systems. The multi-layer IoT database schema will be discussed in more detail in section "The implementation of the multi-layer IoT database schema and the searching algorithm." Methods of data manipulations, O2O services supporting, and message pushing are also developed in the data management layer.

In the data storage layer of Figure 1, a hybrid solution is adopted to organize multiple types of databases, such as relational databases, NoSQL databases, and distributed file systems, to meet the requirements of processing large amounts of multi-source heterogeneous data in IoT applications.

As three kinds of IoT data storage problems need to be addressed in O2O-oriented applications, which are how to synchronously support the retrieval of geography information and content information, how to support diverse IoT applications, and how to process multi-source heterogeneous data, these problems are addressed in the proposed multi-layer IoT database schema, as shown in Figure 1. In the proposed database schema, IoT data from various O2O applications are integrated and then distributed to different processing nodes according to the data properties. A retrieval method is also designed to search the entire data storage space.

The architecture shown in Figure 1 has the following advantages:

1. Information from different aspects of the application is uncoupled with each other and organized more clearly.

Because, in the applications, geography data and application data have different characteristics, they need different processing methods. If the data are stored in different processing nodes, they could be uncoupled from each other. In the multi-layer IoT database schema, logical nodes, location nodes, application nodes, and storage nodes process different kinds of information. All the nodes are connected with the logical nodes so that the information involved in the O2O applications could be searched as a whole.

2. Location data and content data are processed synchronously so that they could be conveniently searched.

In multi-layer IoT database schema, location data are collected to the location nodes. By calculating the correlation of two locations, more accurate results could be obtained in O2O applications.

In the logical nodes, location information is used to create indexes for data accessing. When searching information with location data, multi-layer IoT database schema helps retrieve the information more quickly.

Compared with traditional methods that access location information same as the content information, processing geography information separately could obtain more efficient search results.

3. The architecture shown in Figure 1 provides flexibility for diverse IoT applications.

In recent years, Internet applications especially IoT-related applications have evolved quickly. Facing diverse IoT applications, multi-layer IoT database schema aims to provide high flexibility in geography area division, information search, and display. In the proposed database schema, application nodes are defined as objects so that location information could be described more flexibly. For information searching, not only methods of keyword-based searching but also correlation-based searching are designed to support various IoT applications.

By defining logical nodes, users could implement their own geography area divisions. The structure of the logical nodes could be chosen according to the IoT application fields. Therefore, the proposed multi-layer database schema could adapt to diverse IoT applications.

Nevertheless, in order to develop the architecture shown in Figure 1, the following challenging tasks need to be carried out:

1. The high flexibility of the architecture brings some difficulties in its implementation.

As discussed above, in order to support diverse IoT applications, high flexibility is sought in the system architecture, so that system configuration is needed when the system architecture is implemented. The configuration includes setting geography area division and the schema of information indexing. Although the configuration costs time and spaces for computation, the efficiency of the application system could be improved after configuration.

2. The complexity of information description brings low efficiency of information retrieval.

Compared to the comparatively simple method of information description through keywords, in the proposed multi-layer database schema, complex application nodes are used to describe semantic meanings of IoT applications. The semantic meanings of these application nodes could not be extracted in advance and defined as keywords, while retrieving contents each location node needs to be accessed entirely to recognize their meanings. Since the node accessing is a traversal of the trees, it would bring low efficiency compared to keyword mapping methods. In the section "The implementation of the multi-layer IoT database schema and the searching algorithm," the traversal algorithm and its complexity will be discussed in detail.

The implementation of the multi-layer IoT database schema and the searching algorithm

As multi-layer IoT data storing and accessing are the core components to solve the problem of heterogeneous data fusion in O2O applications, in this section, the definition of the multi-layer IoT database schema and the searching algorithm built on it will be discussed in detail.

Nodes definition in the multi-layer IoT database schema

Following the idea that geography information usually varies from rough to fine granularity, the multi-layer IoT database schema is designed as a tree structure, meaning that nodes on the up level of the tree have rough granularity so that they could cover larger range of geography areas than those on the low level. However, on the low level, data nodes with finer granularities represent more detailed information about the entities in the O2O applications.

Scattered in the different levels of the tree structure, four kinds of nodes are defined in the multi-layer IoT database schema to describe heterogeneous information, which are logical nodes, geography nodes, storage nodes, and application nodes. These four kinds of nodes, respectively, encapsulate different types of data transferred from various sensors into the IoT networks. The logical nodes fuse the data according to the user requirements. The relationships between these four kinds of nodes are demonstrated in Figure 2.

As shown in Figure 2, logical nodes are the center of the database schema. All the other three kinds of nodes are connected with the logical nodes. They cooperate with each other to manage different aspects of O2O applications. The logical nodes are not only the coordinators but also the bridges between the nodes with different granularities. They are composed of non-leaf nodes and leaf nodes, as shown in Figure 2.

The definition of non-leaf logical node is as follows

$$\{K,\ \text{summary info},\ \text{geography node},\ \text{set}<\text{storage node}>,\ \text{set}<\text{logical node}>\}$$

In our solution, the non-leaf logical nodes are stored in MongoDB systems. An example of the non-leaf logical node is illustrated in Table 1.

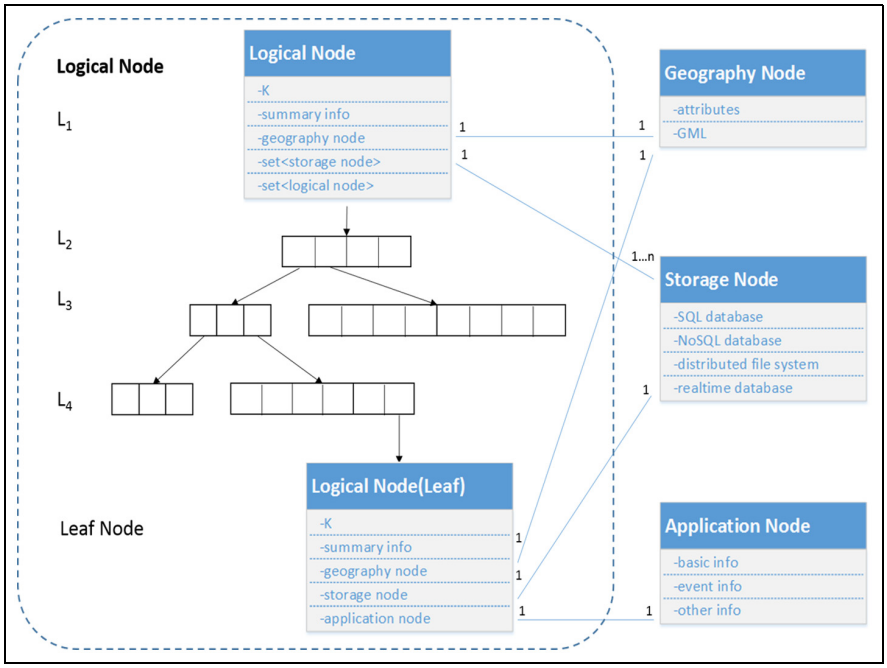

Figure 2. Relationships between four kinds of nodes in the multi-layer IoT database schema.

Table 1. The logical node object (non-leaf node).

Logical node object (non-leaf node)
{"_id": ObjectId("51b23a9db424b3094d3e"), "Usage_ID": "62po09akdm0a", "K": k, "bool_conditions": {"cond_1": 1,"cond_2": 0,"cond_3": 1,…} "quantity_conditions": {"content_score": 47,"cond_1": "description_1","cond_2": "description_2",} "timestamp":{"time_stamp"} "geography_node": geo_id, "storage_node": {"node_1": store_tag_1, "node_2": store_tag_2,…} "child_node": {"node_1": log_id_1, "node_2": log_id_2, …}}

Table 2. An example of usage node.

Usage node
{ "_id": ObjectId("51b23sfs9b4234b3094d3c4"), "Usage_ID": "62po09akdm0a", "K": 1, "bool_conditions": {"cond_1": 1,"cond_2": 0,"cond_3": 1,…} "quantity_conditions": {"content_score": 47,"cond_1": "description_1","cond_2": "description_2",} "timestamp":{"time_stamp"} "attributes":{"attr_1": "description_1", "attr_2": "description_2", "attr_3": "description_3"…} "other_data": {"file_1": "hdfs_addr_1","file_2": "hdfs_addr_2",…}}

The definition of leaf logical node is as follows

$$\{K,\ \text{summary info},\ \text{geography node},\ \text{storage node},\ \text{usage node}\}$$

Same as the non-leaf logical nodes, leaf logical nodes are also stored in MongoDB systems. Differing from logical nodes, which mainly focus on the coordination of data fusion, application nodes organize data from viewpoints of application domains, which classify data into master data and transaction data. An example of application node is shown in Table 2.

As the application nodes may contain data such as pictures and videos, Hadoop Distributed File System

(HDFS) together with MongoDB systems are used to store application nodes.

For geography nodes, we use Geography Markup Language (GML)[28] to describe the locations and the shapes of the entities in the O2O applications.

Searching algorithm based on the multi-layer IoT database schema

Problem definition. A spatial data query problem could be defined as follows

$$Query\ (locations,\ conditions,\ contents)$$

where *locations* refer to where the query occurs, *conditions* mean restriction on the searching, and *contents* are used to calculate the similarity between the data nodes and the searching targets. The searching algorithm is depicted in Table 3.

The parameters in the algorithms are described as follows:

Cx is the current node being visited.
sumsig() is an array of the candidates.
FinalScore() is an array of the similarity scores considering both location and content.

The main idea behind the search algorithm is pruning. The steps of the algorithm are as follows:

Step 1. Cx is initialized to the root of the logical node tree.

Table 3. Searching algorithm.

Algorithm search (Cx,query)

```
1. if Cx has no child
2.     Calculate FinalScore and check the conditions
3.   If FinalScore > current score and fulfill all the conditions
4.     candidate node = Cx
5.     check another unchecked and unpruned node
5.   Else
6.     prune(Cx)
7.   else
8.   check every sig_i in sumsig(Cx)
9.     if it cannot fulfill the conditions
10.        prune(Cx)
11.     else
12.        calculate LocationScore
13.        calculate ContentScore
14.        FinalScore = ∂
           LocationScore + (1 − ∂)ContentScore
15.        if FinalScore > current score
16.           Cx_u is the u-th child of Cx
17.           Search(Cx_u,query)
18.        Else
19.           Prune(Cx)
```

Step 2. If Cx is a leaf node and if Cx satisfies the conditions, then Cx is set to one of the candidates and put into the array of sumsig(). Otherwise, the node is pruned.
Step 3. If Cx is not a leaf, then check its child nodes till reaching a leaf node.
Step 4. Repeat steps 2 and 3 to travel the whole tree till all the non-leaf nodes are checked or pruned.
Step 5. For each node in the array of sumsig(), calculate the value of FinalScore().

The best one in the FinalScore() will be the result of the searching. The complexity of the searching algorithm depends on the cost of traveling the logical nodes tree. The whole tree will be visited if no node is pruned. In this case, the worst complexity will be O(n), given n is the number of the data nodes.

By analyzing the searching algorithm shown in Table 3, it could be found that building indexes of the logical nodes can improve the efficiency of the algorithm. As the logical nodes are the bridges for linking the data nodes with different spatial granularities, if indexes are created between the parent nodes and the child nodes, they would guide the pruning of the child nodes which will limit the searching range and reduce the complexity of the traversal.

Case study and discussion

Case study

In this section, a multi-layer IoT database schema is used in a project to manage charging points that have been installed during smart city construction in Shanghai, China, to help electric energy–driven car drivers to locate available nearby charging points.

In order to encourage more people to purchase electric energy–driven cars, it is important to recommend available charging points to the drivers when they need to charge their cars. Therefore, the nearest location searching and status checking are critical for charging point recommendation.

As the charging point recommendation needs to combine online information query with offline car charging, it is a kind of O2O application. In this section, we take the charging point recommendation system as an example to illustrate the usefulness of the multi-layer IoT database schema in the construction of O2O systems.

The requirements of the charging point recommendation system include the following:

1. Compare the parameters of the charging points with those of the cars;
2. Check the location and the status of the charging points;

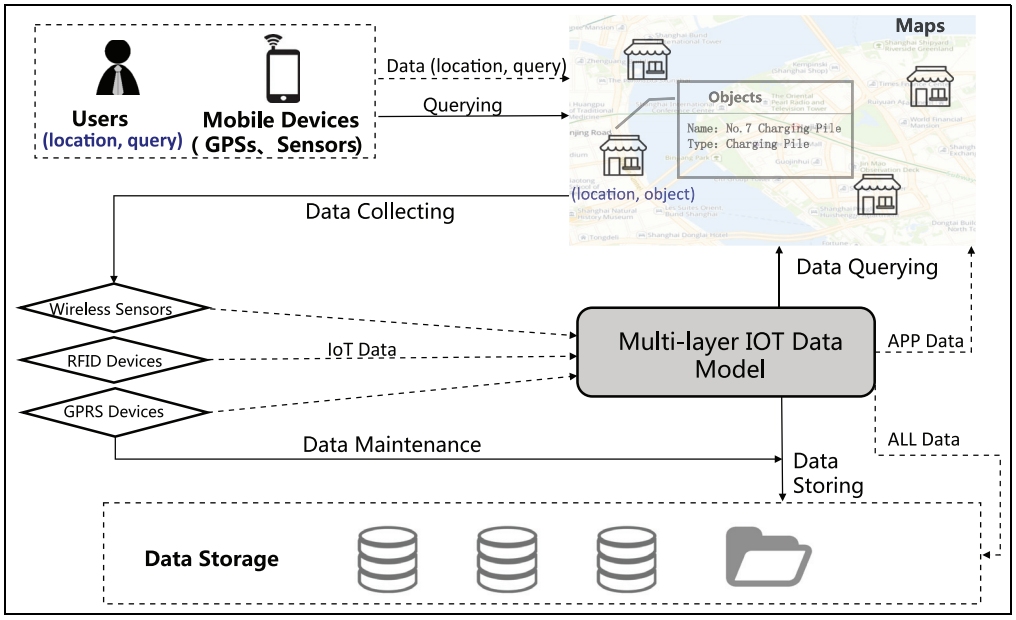

Figure 3. Charging point recommendation system.

3. Check the availability of the charging points;
4. Recommend suitable charging points to the customers considering both the distance and the availability of the charging points.

The solution of using the proposed multi-layer IoT database schema to implement the above functions is depicted in Figure 3.

In the data storage layer of Figure 3, master data of the charging points, transaction data of the car charging behaviors, and the advertisement pictures and videos are stored in MongoDB and HDFS systems, respectively.

In Figure 3, heterogeneous data from sensors, RFID devices, and GPRS devices will be collected based on the multi-layer IoT database schema. The location data are related to geography nodes. The distances between the points and the customers will be calculated to recommend the nearest point to the customers if its status is usable.

In the prototype of the project, we have used 2104 charging points because nearly 2000 charging points have been built in Shanghai. In our experiment, the longitudes and the latitudes of the charging points are created randomly. Figure 4 shows the multi-layer data management for the charging points, including 2306 geography nodes labeling the spatial information of the charging points, 2124 application nodes reflecting the status of the charging points, and 2241 logical nodes coordinating the geography area division and information fusing during charging point recommendation. It could be seen that the number of the nodes of each type is not exact as the number of the charging points. That is because extra nodes are needed to manage the common information, which are shared among the charging points. For example, in addition, each charging point needs a geography node to label its exact location, those charging points belonging to the same district may also need a common geography node to label their district locations in a relatively rough granularity.

In Figure 4, it could be noticed that the spatial data of the charging points and the transaction data of the charging behaviors are stored in the geography nodes and the application nodes, respectively. They are fused through the logical nodes for information recommendation. The interface of the prototype developed to test the usability of the multi-layer IoT database schema is illustrated in Figure 5.

In Figure 5, the left-hand side interface shows the map of the charging points. Customers could input the types of their vehicles for recommendation. The middle interface in Figure 5 shows the searching process. It is actually the process of traversal and calculation through the nodes organized according to the multi-layer IoT database schema. The right-hand side interface in Figure 5 shows the result the system recommends to the customers.

We have also designed an experiment to verify the scalability of our method with randomly created charging point locations in Shanghai area. In the experiment, the number of the charging points ranges from 2000 to 194,000. The increment of the logical nodes, the height of the tree of the IoT data, and the average search time are shown in Figures 6(a)–(c), respectively.

As shown in Figure 6(a), the number of logical nodes grows linearly with the number of charging points. In

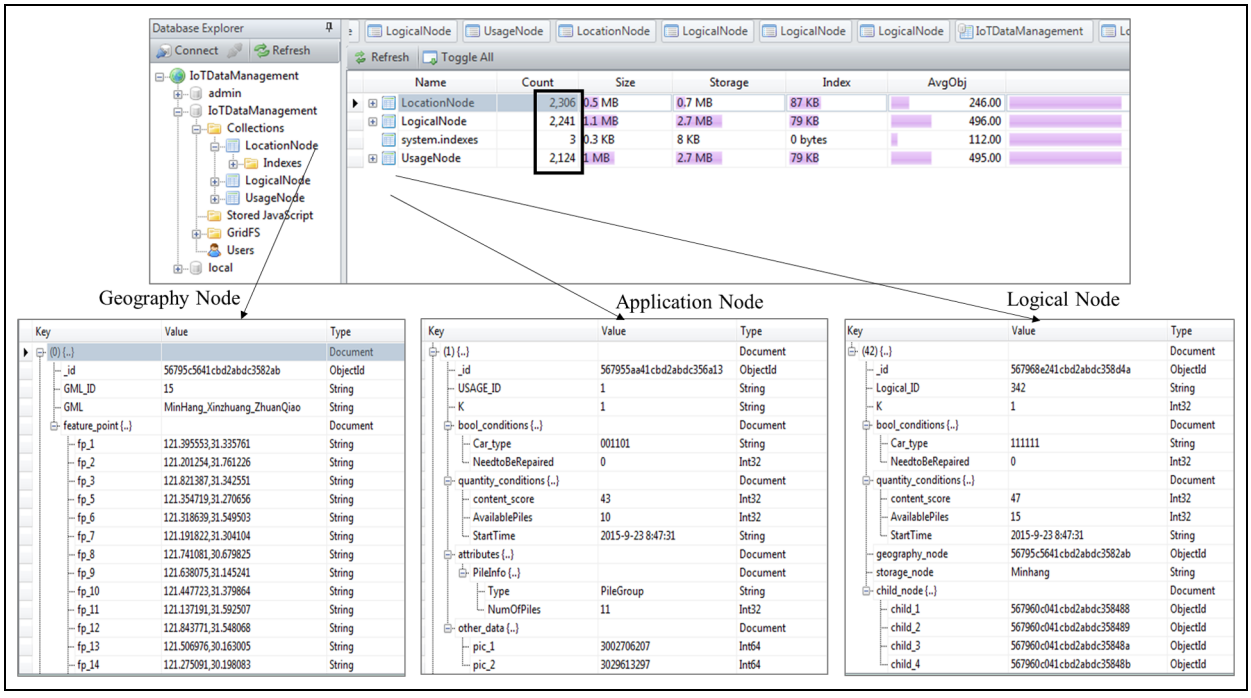

Figure 4. The IoT data stored in the charging point recommendation system.

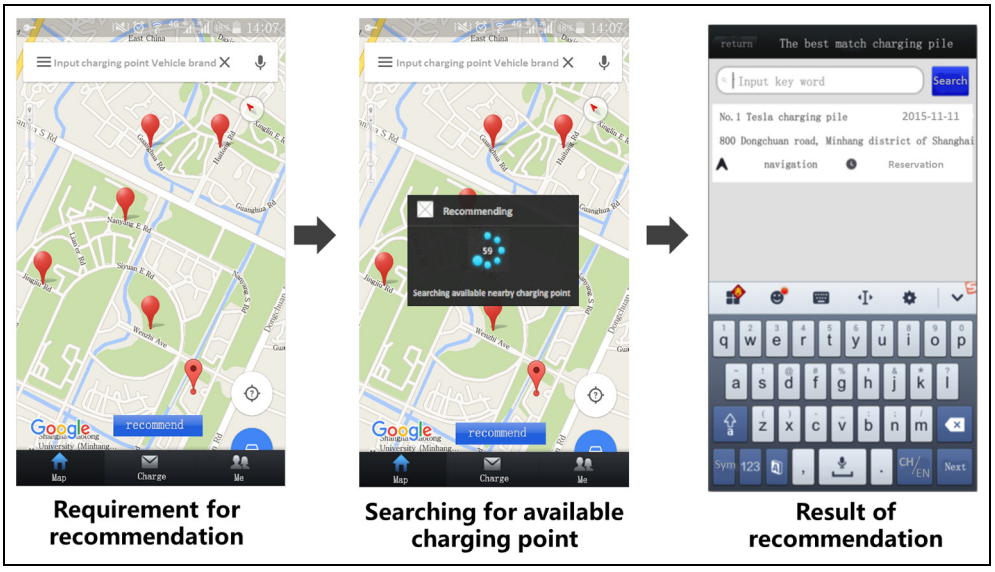

Figure 5. Interface of charging points search and recommendation.

our method, a logical node splits into more logical nodes when the number of the included charging points reaches a threshold, thus the number of the logical nodes is proportional to the number of charging points. While the included charging points in a leaf node have not reached the threshold, the number of logical nodes will not increase.

Figure 6(b) shows that the number of layers, which is equal to the height of the tree storing the IoT data, grows logarithmically with the number of the charging points. This is because that in our solution, the IoT data are organized in a tree structure. When the number of charging points increases, a new leaf node will be added. When the leaf nodes are full, the height of the tree will increase by 1.

According to the pruned searching algorithm, the traveling process starts at the root of the tree and ends at a leaf node of the tree, and this leads to logarithmic

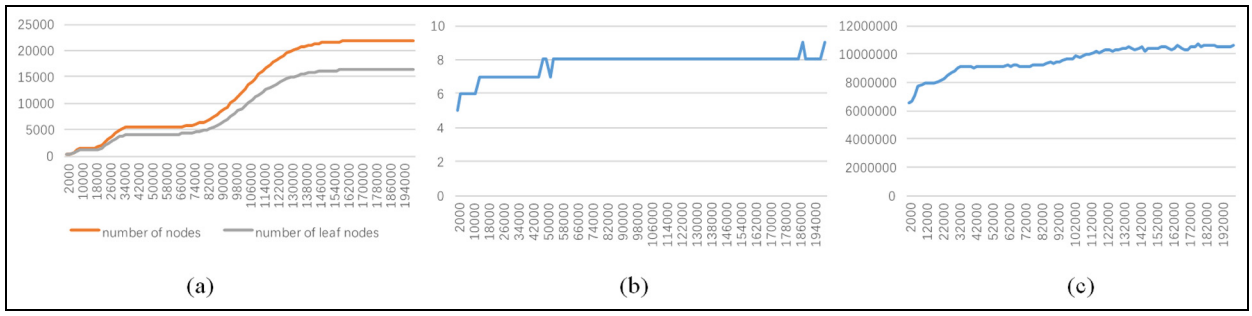

Figure 6. Experimental results of the scalability of our method (a) Logical node numbers increase linearly with charging point numbers, (b) Tree height increases logarithmically with charging point numbers and (c) Search time increases logarithmically with charging point numbers.

Table 4. Method comparison.

Feature		Methods		
		I3	SDS scheme	Multi-layer IoT database schema
Data representation method	Basic unit	Cell	Zone	Node
	Shape of the units	Rectangular	Almost rectangular	Arbitrary
	Space organization	Quadtree	SDS zones	Logical node
	Storage method	Disk	Data-centric storage	Storage node
	Partition space by	Keyword quantity	Data similarity	Value of K
Data manipulation method	Temporal data processing Method	Not mentioned	Spatial–temporal dimension	Timestamp
	Query by	Keyword	Keyword Similar query	Application requirement
	Support similarity search	No	Yes	No
	Big data processing	Good	Good	Good
	Application environment	Location-based service	WSN	IoT application
System comprehensive feature	Flexibility	General	Good	Good
	Practicability	General	General	Good
	Coupling	High	General	Low

SDS: Similarity Data Storage; IoT: Internet of things.

time complexity of the searching algorithm. This means that most searching time is spent on shifting from a logical node of higher layer to its sub-logical node on lower layer. As shown in Figure 6(c), the average search time increases logarithmically with the number of charging points. For the big data environment of O2O applications, we think the logarithmic time complexity of the searching algorithm is feasible.

Discussion

From the viewpoints of data representation method, data manipulation method, and system comprehensive features, we compared the multi-layer IoT database schema with I3[14] and the Similarity Data Storage (SDS) scheme.[18] The comparison is shown in Table 4. I3 divides the data space by quadtree and uses a concept of "keyword cell" to represent keywords with spatial information. An efficient query algorithm of top-k keyword searching is proposed in I3. SDS is a distributed spatial–temporal scheme for WSN. It uses a two-dimensional space to manage the spatial and temporal attributes at the same time. Temporal attributes have the same importance as the spatial attributes in IoT applications. SDS performs better in processing the temporal information. The multi-layer IoT database schema is more flexible because it decouples the layers that represent different aspects of O2O data. Furthermore, it is more user-friendly because users can define their own space partition method and design personalized query form.

From the comparisons in Table 4, it could be concluded that the proposed method achieves more flexible data queries through decoupling geography data from application data.

The multi-layer IoT database schema classifies and distributes the information involved in O2O applications into different types of data nodes, such as

location nodes, storage nodes, and application nodes, and according to the properties of the data, it can help to decouple the O2O data completely. The logical nodes connect and coordinate the other three kinds of data nodes fulfilling the data query in O2O applications. This brings more flexibility to develop diverse systems to meet user requirements.

Conclusion

The main contributions of this article are summarized as follows:

1. We proposed a multi-layer IoT database schema to describe the heterogeneous information of O2O systems in a decoupled hierarchical way. The multi-layer IoT database schema includes four kinds of data nodes, which are geography nodes, application nodes, storage nodes, and logical nodes. The logical nodes act as coordinators during data query. All other data nodes are connected to the logical nodes to form unified support for diverse O2O applications.
2. We have designed a searching algorithm based on the multi-layer IoT database schema to realize O2O data query considering both location and content information. The algorithm creates spatial indexes on the location attributes of the logical nodes to quickly target the geography zones. Pruning strategy is also used to reduce the complexity of the algorithm.
3. We have proposed an architecture using the multi-layer IoT database schema to implement O2O systems. In the architecture, IoT data are collected from the sensor networks and stored in MongoDB and HDFS. The main aim of the multi-layer database schema is to facilitate the O2O-oriented data query considering the spatial data transferred from the IoT sensors, such as GPRSs, as well as considering the transaction data transferred from the O2O systems, such as smart cities.

Although the multi-layer IoT database schema is designed to handle both structured and unstructured data, the prototype in this article did not realize the combination of the processing of structured data, such as location data and transaction data, with unstructured data, such as videos and images. The future research will focus on the searching method involving videos and images for O2O systems.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This research was supported by the National Natural Science Foundation of China under nos 61373030 and 71171132.

References

1. Sun S, Cegielski CG and Li Z. Amassing and analyzing customer data in the age of big data: a case study of Haier's Online-to-Offline (O2O) business model. *J Inform Technol Case Appl Res* 2015; 17(3–4): 156–165.
2. Jiang L, Xu LD, Cai H, et al. An IoT-oriented data storage framework in cloud computing platform. *IEEE T Ind Inform* 2014; 10(2): 1443–1451.
3. Li S, Xu LD and Wang X. Compressed sensing signal and data acquisition in wireless sensor networks and Internet of Things. *IEEE T Ind Inform* 2013; 9(4): 2177–2186.
4. Fazel F, Fazel M and Stojanovic M. Random access compressed sensing for energy-efficient underwater sensor networks. *IEEE J Sel Area Comm* 2011; 29(8): 1660–1670.
5. Wang L, Xu LD, Bi Z, et al. Data cleaning for RFID and WSN integration. *IEEE T Ind Inform* 2014; 10(1): 408–418.
6. Wang W and Guo D. Towards unified heterogeneous event processing for the Internet of Things. In: *IEEE the 3rd international conference on the Internet of Things (IOT)*, Wuxi, China, 24–26 October 2012, pp.84–91. New York: IEEE.
7. Somov A, Dupont C and Giaffreda R. Supporting smart-city mobility with cognitive Internet of Things. In: *IEEE future network and mobile summit (FutureNetwork-Summit)*, Lisboa, 3–5 July 2013, pp.1–10. New York: IEEE.
8. Liu H, Liu Y, Wu Q, et al. A heterogeneous data integration model. In: Bian F, Xie Y, Cui X, et al. (eds) *Geo-informatics in resource management and sustainable ecosystem*. Berlin and Heidelberg: Springer, 2013, pp.298–312.
9. Brock AL. The Physical Markup Language. MIT Auto-ID Center, February 2001.
10. Botts M and Robin A. *OpenGIS sensor model language (SensorML) implementation specification*. Technical report OGC 07-000, 17 July 2007. Open Geospatial Consortium Inc.
11. Mohamed N and Al-Jaroodi J. A survey on service-oriented middleware for wireless sensor networks. *Serv Oriented Comput Appl* 2011; 5(2): 71–85.
12. Li Z, Lee KCK, Zheng B, et al. Ir-tree: an efficient index for geographic document search. *IEEE T Knowl Data En* 23: 585–599.
13. Rocha-Junior JB, Gkorgkas O and Jonassen S. Efficient processing of top-k spatial keyword queries. In: Pfoser D, Tao Y, Mouratidis K, et al. (eds) *Advances in spatial and temporal databases*. Berlin and Heidelberg: Springer, 2011, pp.205–222.
14. Zhang A, Tan KL and Tung AKH. Scalable top-k spatial keyword search. In: *Proceedings of the 16th*

international conference on extending database technology, Genoa, 18–22 March 2013, pp.359–370. New York: ACM Press.
15. Zhang C and Li Y. Research on data storage system of geographical spatial information based on grid. In: *2015 international conference on intelligent systems research and mechatronics engineering*, Zhengzhou, China, 11–13 April 2015. Amsterdam: Atlantis Press.
16. Liao ZX and Peng WC. Clustering spatial data with a geographic constraint: exploring local search. *Knowl Inform Syst* 2011; 31: 153–170.
17. Lu J, Lu Y and Cong G. Reverse spatial and textual k nearest neighbor search. In: *Proceedings of the ACM SIGMOD international conference on management of data*, Athens, 12–16 June 2011, pp.349–360. New York: ACM Press.
18. Shen H, Zhao L and Li Z. A distributed spatial-temporal similarity data storage scheme in wireless sensor networks. *IEEE T Mobile Comput* 2011; 10(7): 982–996.
19. Yaish H, Goyal M and Feuerlicht G. Multi-tenant elastic extension tables data management. *Procedia Comp Sci* 2014; 29: 2168–2181.
20. Su X and Swart G. Oracle in-database Hadoop: when MapReduce meets RDBMS. In: *Proceedings of the 2012 ACM SIGMOD international conference on management of data*, Scottsdale, AZ, 20–24 May 2012, pp.779–790. New York: IEEE.
21. Curé O, Kerdjoudj F, Le Duc C, et al. On the potential integration of an ontology-based data access approach in NoSQL stores. In: *IEEE 2012 third international conference on emerging intelligent data and web technologies (EIDWT)*, Bucharest, 19–21 September 2012, pp.166–173. New York: IEEE.
22. Yu B, Cuzzocrea A, Jeong D, et al. On managing very large sensor-network data using bigtable. In: *2012 12th IEEE/ACM international symposium on cluster, cloud and grid computing (CCGrid)*, Ottawa, ON, Canada, 13–16 May 2012, pp.918–922. New York: IEEE.
23. Cai X, Li W, He F, et al. Customized encryption of computer aided design models for collaboration in cloud manufacturing environment. *J Manuf Sci E-T ASME* 2015; 137: 040905.
24. Cai XT, He FZ, Li WD, et al. Encryption based partial sharing of CAD Models. *Integr Comput-Aid E* 2015; 22: 243–260.
25. Velumadhava Rao R and Selvamani K. Data security challenges and its solutions in cloud computing. *Procedia Comp Sci* 2015; 48: 204–209.
26. Bocchi L and Melgratti H. On the behaviour of general purpose applications on cloud storages. *Serv Oriented Comput Appl* 2015; 9(3): 213–227.
27. Dede E, Govindaraju M, Gunter D, et al. Performance evaluation of a MongoDB and hadoop platform for scientific data analysis. In: *Proceedings of the 4th ACM workshop on Scientific cloud computing*, New York, 17–21 June 2013, pp.13–20. New York: ACM Press.
28. Geography Markup Language (GML 2.0). http://xml.coverpages.org/gml01-029-20010228.pdf