Ren XL, Ren KJ, Xu ZC *et al.* Improving ocean data services with semantics and quick index. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 36(5): 963–984 Sept. 2021. DOI 10.1007/s11390-021-1374-0

Improving Ocean Data Services with Semantics and Quick Index

Xiao-Li Ren[1,2], *Member*, *CCF*, Kai-Jun Ren[1,2,*], *Member*, *CCF*
Zi-Chen Xu[3,*], *Senior Member*, *CCF*, *Member*, *ACM*, *IEEE*, Xiao-Yong Li[2], *Senior Member*, *CCF*
Ao-Long Zhou[1,2], Jun-Qiang Song[1,2], and Ke-Feng Deng[2], *Member*, *CCF*

[1] *College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China*
[2] *College of Meteorology and Oceanography, National University of Defense Technology, Changsha 410073, China*
[3] *College of Computer Science and Technology, Nanchang University, Nanchang 330031, China*

E-mail: {renxiaoli18, renkaijun}@nudt.edu.cn; xuz@ncu.edu.cn
{sayingxmu, zhouaolong10, junqiang, dengkefeng}@nudt.edu.cn

Received February 11, 2021; accepted August 20, 2021.

Abstract Massive ocean data acquired by various observing platforms and sensors poses new challenges to data management and utilization. Typically, it is difficult to find the desired data from the large amount of datasets efficiently and effectively. Most of existing methods for data discovery are based on the keyword retrieval or direct semantic reasoning, and they are either limited in data access rate or do not take the time cost into account. In this paper, we creatively design and implement a novel system to alleviate the problem by introducing semantics with ontologies, which is referred to as Data Ontology and List-Based Publishing (DOLP). Specifically, we mainly improve the ocean data services in the following three aspects. First, we propose a unified semantic model called OEDO (Ocean Environmental Data Ontology) to represent heterogeneous ocean data by metadata and to be published as data services. Second, we propose an optimized quick service query list (QSQL) data structure for storing the pre-inferred semantically related services, and reducing the service querying time. Third, we propose two algorithms for optimizing QSQL hierarchically and horizontally, respectively, which aim to extend the semantics relationships of the data service and improve the data access rate. Experimental results prove that DOLP outperforms the benchmark methods. First, our QSQL-based data discovery methods obtain a higher recall rate than the keyword-based method, and are faster than the traditional semantic method based on direct reasoning. Second, DOLP can handle more complex semantic relationships than the existing methods.

Keywords data service, ocean data, ontology, semantic representation

1 Introduction

Ocean science is currently entering into big data era with the exponential growth of information technology and advances in ocean observing. On the one hand, various sensors and autonomous observing platforms have acquired Petabytes of observation data [1]. On the other hand, the large-scale applications such as numerical prediction models generate about Terabytes of simulation results each day. The explosive growth of the volume and diversity of ocean data poses new challenges for data management [2–4].

To overcome the problem, scientists have proposed guidelines for the publication of digital resources like datasets, codes, workflows, and research objects, in a manner that makes them findable, accessible, interoperable, and reusable (FAIR) [5]. Researchers outlined how FAIR principles apply to ocean data [6]; however, there is still no unified or standard data service model to make ocean data be FAIR so far. Typically, we list

Regular Paper
Special Section of APPT 2021
A preliminary version of the paper was published in the Proceedings of SCC 2020.
This work was partially supported by the National Key Research and Development Program of China under Grant No. 2018YFB0203801, and the National Natural Science Foundation of China under Grant Nos. 61702529 and 61802424.
*Corresponding Author (Kai-Jun Ren helps with the original idea and theoric design. Zi-Chen Xu significantly helps with the implementation and writing. They both guided the work of the paper and contribute equally.)

the main challenges for ocean data services as follows.

Data Access. The lack of standard and rich oceanographic metadata makes it impossible to identify and access datasets uniformly. As far as we know, there are several ocean data inventories, such as Marine Environmental Data Inventory (MEDI) [7], Ocean Data Acquisition System (ODAS) metadata①, European Directory of the Initial Ocean-observing System (EDIOS)②, and Argo floats metadata③, which describe observing systems or observation data from their own perspectives with various standards.

Data Interoperation. The ocean data types and formats from different platforms and observing systems are diverse and heterogeneous, and the terminologies may be ambiguous, which bring great challenges to data interoperation.

Data Finding. It is difficult to find valuable information from the large amount of data without an effective index. According to the statistics, by the ocean observing platforms deployed in the past 10 years, the amount of observation data transmitted in one year is equivalent to that acquired in the past century [6].

The proliferation of heterogeneous data sources (unstructured, semi-structured and structured) brings a new urgent need for tools which allow to query heterogeneous data in a flexible way [8]. One of the most challenging problems is how to improve data services to support knowledge discovery better [6, 9]. Fortunately, information technologies such as service computing can be used for describing various datasets with multiple heterogeneous formats, and improve data services in the ocean domain.

Service-oriented computing (SOC) as an efficient computing model for distributed computing, cross-organizational digital data sources sharing, and application integration [10, 11], has been developed rapidly in the past two decades. Ontology-based semantic web as one of the key technologies in service computing, can not only construct the unified description of heterogeneous resources, but also improve interoperability among services [12]. Particularly, one of the most powerful features of ontology is that it provides a way to express explicit knowledge of a conceptual domain, from which the implicit new knowledge can be inferred through logical reasoners [13], e.g., it is widely used to describe sensors and observations [14, 15]. It is worthwhile to note that in recent years ontology has been widely applied to describe heterogeneous resources in high-performance computing (HPC) [16, 17], cloud service environments [18–20], and Internet of things (IoT) [21, 22]. Therefore, it is natural to utilize ontology to represent the heterogeneous ocean data and improve data discovery by service computing technologies.

Service discovery, especially the Web service discovery, has been widely studied in recent years. The basic service discovery solutions range from the initial proposals that rely on the syntactic description of services (syntactic-based methods) [23], to more generic solutions that combine user-provided semantic descriptions (semantic-based methods) [24–26]. As a vast number of various types of services are available, service discovery solutions are desired to handle much complex queries effectively and efficiently. As a consequence, the expanded hybrid solutions have attracted much attention [27, 28]. However, existing service discovery methods still have several limitations [27, 29, 30].

First, most of the existing Web services are described using WSDL (Web Services Description Language) and not associated with semantics, and a majority of the syntactic-based methods adopt keyword-matching techniques to find the published services. However, they fail to discover the services whose functionalities are semantically equivalent or similar to the query, and returns only a few services that exactly match the request, even though information retrieve (IR) techniques are usually leveraged to enhance the performance [23, 31].

Second, the semantic-based methods are traditionally proposed to overcome the issues of keyword-matching methods, by using ontology-based semantic Web service description languages to annotate the services. However, for data services, on the one hand, it is impractical for service publishers to tag all the services, especially for the ocean data that continues to grow rapidly, and lack of recognized data ontologies. On the other hand, the traditional service discovery methods based on direct reasoning perform semantic reasoning during service querying. From the perspective of service requesters, they can return some of the services that are semantically related to their requests, but it is time-consuming and inefficient [32].

Third, most of the hybrid solutions recommend candidate services for users to select and determine their final queries, and focus on how to enhance the queries

①https://www.jcomm.info/index.php?option=com, Mar. 2020.

②https://www.seadatanet.org/Metadata/, Aug. 2021.

③http://www.argodatamgt.org/Documentation, Aug. 2021.

of users [27]. It requires multiple interactions with the users, and may obtain the exact services they needed ultimately. However, the requirement for multiple attempts and interactions increases the burden on users.

Here, we design and implement a novel system called DOLP (Data Ontology and List-Based Publishing), which combines the semantics with IR techniques to improve the performance of data service discovery. As shown in Fig.1, in DOLP we mainly improve the ocean data services in the following three aspects. First, we propose a unified semantic model called OEDO (Ocean Environmental Data Ontology). Second, we propose an optimized quick service query list (QSQL) data structure for enhancing services querying. Third, we propose two algorithms for optimizing QSQL hierarchically and horizontally.

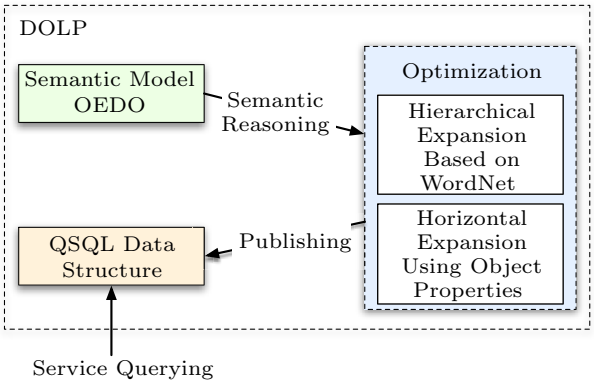


Fig.1. Overview of DOLP.

Specifically, we introduce semantics into ocean data by constructing the OEDO model, which represents multiple heterogeneous ocean data by metadata. We publish the concepts of the OEDO model as input and output parameters of the data service interfaces. The QSQL data structure was proposed in our previous studies based on IR techniques for retrieving services quickly [16,32]. It stores the semantically related services which are inferred in service publishing stage. We optimize QSQL further in this paper. Based on the OEDO model and the basic QSQL, we extend QSQL hierarchically and horizontally respectively to generate the optimized QSQL. Firstly, we perform hierarchical expansion based on WordNet [33], which extends the semantics of the domain concepts. Then, based on the RDF and OWL semantics, horizontal expansion using object properties is implemented, which aims to extend the semantic relationships of the data services and improve the data access rate.

The contributions of this paper are summarized as follows.

• We construct the OEDO model to provide a unified semantic representation for heterogeneous ocean data resources.

• In order to expand the semantics of the data services, we optimize the QSQL data structure [16,32], by extending the domain concepts hierarchically based on WordNet, and extending the relationships between concepts horizontally using multiple object properties.

• Based on the OEDO model and the optimized QSQL, we propose the hybrid data service publishing and discovery system DOLP to improve data services.

• We conduct extensive experiments to validate the effectiveness and efficiency of our proposals in data service discovery and data access.

This paper extends our earlier published conference paper [34] in several aspects. First, we focus on publishing data as services to improve the discovery of heterogeneous ocean data, and survey more state-of-the-art studies related to service discovery. Second, we represent more details of the unified OEDO model by RDF and OWL semantics, and extend the model concepts, such as the Sensor class. Third, a new algorithm based on multiple object properties is proposed, to further extend the QSQL data structure and improve the efficiency of data discovery and access.

The rest of this paper is organized as follows. In Section 2, we analyze and discuss related work. Section 3 introduces the semantic OEDO model. In Section 4, we describe the hierarchical and horizontal QSQL optimization algorithms in detail. Section 5 presents and analyzes our experimental results. In Section 6, we summarize this paper.

2 Related Work

2.1 Ontologies for Heterogeneous Data

In view of the advantages in solving semantic interoperability, ontology has been widely used in the representation of multiple heterogeneous resources.

Typically, Janowicz and Compton developed a generic ontology design pattern for observation-based data on the semantic web [14], which also forms the top-level of the semantic sensor network (SSN) ontology developed by the W3C semantic sensor network incubator group (SSN-XG). The SSN ontology can be seen from four main perspectives, that is, a sensor perspective, an observation perspective, a system perspective, and a feature and property perspective [15]. Bermudez

et al.[35] proposed a marine platform ontology, which is fostered by the Marine Metadata Interoperability (MMI) Project[36] and jointly developed by a team of data system developers from the marine science and ontological communities. Lowry and Leadbetter[37] summarized how data discovery, markup and aggregation are semantically supported in the European Marine Observation and Data Network (EMODnet)[37]. In order to provide the solution of a certain class of applied problems of climate data, the first version of the primitive Web Ontology Language (OWL)-ontology of collections of climate and meteorological data is presented[38], which is a component of expert and decision-making support systems intended for quick search for climate and meteorological data.

However, the aforementioned ontologies are just constructed from various perspectives with their own metadata, and have not been practically used to realize data service discovery yet.

2.2 Service Discovery

As described in Section 1, the existing service discovery methods can be classified into three categories from the perspective of the technology used, i.e., the syntactic-based methods, the semantic-based methods, and the hybrid methods.

Syntactic-Based Methods. Most of the traditional syntactic-based methods query services via matching keywords between user queries and services, and usually incorporate IR techniques to enhance the performance, such as hierarchical clustering algorithm[23,31]. Dong *et al.*[31] used a clustering algorithm to calculate the similarity between user queries and service operations. Plebani and Pernici[39] proposed a Web service retrieval method based on similarity evaluation, which measures the degree of similarity between multiple service interfaces by analyzing the structures of WSDL documents and the terms they contain. However, they still fail to get services whose functionalities are semantically similar to the desired one, and limited in the precision and recall rate[40].

Semantic-Based Methods. This kind of methods is proposed to overcome the limitations of the syntactic-based methods. The basic idea of the initial logic-based methods is to represent services by ontology-based semantic web service description languages[27], e.g., SAWSDL, OWL-S and WSMO[24], to leverage the powerful features of ontology. For example, Rodríguez-Mier *et al.* proposed a logic-based fine-grained I/O web service discovery method[25]. Chen *et al.*[26] proposed a semantic similarity measure that combines functional similarity and process similarity. It has been proved that the efficiency of these methods is better than that of syntactic-based methods, but the complexity of semantic reasoning and similarity computation is high.

In addition to web services, semantic-based methods have been successfully applied to new service models, such as cloud services[41,42] and HPC services[16]. Rekik *et al.*[41] proposed an ontology for the cloud service description, which mainly covers three layers, namely IaaS, PaaS and SaaS. Parhi *et al.*[42] designed an ontology-based cloud infrastructure service discovery and selection system by defining functional and non-functional concepts, as well as attributes and relations of infrastructure services. Zhou *et al.*[16] constructed an ontology model to represent the cross-regional HPC resources, and proposed a service discovery method based on the model to improve the efficiency of resources discovery. However, the resources of both cloud and HPC are limited and the types of resources are clear and definite. It is difficult to extend these methods to data services, which are required to cope with the growing and diverse types of data.

It is worth noting that the ontology-based semantic models also play an important role in heterogeneous data access. Ontology-based data access (OBDA) has been extensively studied by researchers[43–48]. The OBDA paradigm[49] has emerged as a proposal to provide convenient and user-friendly access to data repositories, which are focused on relational data resources in the past decade. However, they cannot handle semi-structured and unstructured data.

Besides, many schemes have been proposed to discover valuable data from heterogeneous data sources. For example, to integrate the proliferate heterogeneous data sources, Buron *et al.*[8] proposed an ontology-based RDF integration mechanism of heterogeneous data. The mechanism enables joint querying of the data and the ontology. Query answering in RDF knowledge bases through reformulation has been studied to query data and schema triples together[50]. Peng *et al.*[51] proposed techniques for processing SPARQL queries over a large RDF graph in a distributed environment. Quamar *et al.*[52] proposed an ontology-based conversation system for domain-specific (healthcare) knowledge bases from various data sources. However, it is hard to access these data in the form of data services, which is much easier for users to find the valuable data from the large amount of heterogeneous data sources.

Hybrid Methods. In recent years, many hybrid methods have been proposed to improve the performance of more complex services discovery, including the syntactic-semantic hybrid methods [28] and query expansion methods [27, 29]. For instance, Garriga *et al.* [28] presented a structural-semantic hybrid approach to help developers in retrieval and selection of services from a service registry. Paliwal *et al.* [29] utilized clustering to classify the services accurately based on service functionality, and enhance service requests by adding relevant ontology concepts. Zhang *et al.* [27] proposed a service discovery approach by utilizing similar service goals extracted from textual service descriptions, which are then recommended to users to select as an expanded query [53]. It helps service requesters obtain similar services accurately with a simple keyword query. The query expansion methods improve the quality of user queries a lot. However, users need to select the recommended service manually, and it may require multiple interactions, which increases the burden of users who are lack of relevant domain knowledge. To achieve more realistic service composition, Ren *et al.* [32] proposed a service discovery method using WordNet and multiple heterogeneous ontologies. However, the semantic relationships from WordNet are limited [28, 32], and some concepts from the domain ontologies may be irrelevant [27]. What is more, most of the query expansion methods do not take the time cost into account, which is important to service requesters and cannot be neglected when a large number of services exist.

To sum up, the existing syntactic-based methods, semantic-based methods, and hybrid methods still face challenges in efficient and high-quality data service discovery. Specifically, syntactic-based methods are limited in the data access rate due to the lack of semantic information [40]. Semantic-based methods need to annotate services with rich domain concepts, which is difficult for service publishers and users with little domain knowledge, and the semantic reasoning is time-consuming [32]. Most of hybrid methods require multiple interaction with users and do not consider the response time of the querying [27, 53]. However, from the perspective of data users, efficient and high-quality data service discovery should be able to obtain the required services quickly and accurately, which can be measured by the response time, precision and recall rate of the querying.

In this paper, we propose a novel system, DOLP, for improving ocean data services. It consists of three components: the ontology model OEDO that is constructed using ocean metadata and is published as the input/output parameters of data services, the QSQL data structure for storing the published and pre-inferred concepts and relationships in OEDO model, and the optimization algorithms for extending the semantics of the OEDO model. The DOLP system focuses on reducing the time of data service discovery and improving data access by optimizing the QSQL data structure.

3 Construction of the OEDO Model

As there are few open ocean data ontologies that can be used to publish data services, we first construct the core of the OEDO model manually using ocean metadata, which aims to provide a unified semantic representation for heterogeneous ocean data. The ODEO model is open sourced through Github④.

3.1 Semantic Representation

The semantic web data model RDF represents data as a collection of triples in the form of (s, p, o), which describes the relationship between the two entities subject s and object o by property (or predicate) p. Naturally, it can be represented as a graph where subjects and objects are vertices, and the properties are edge labels. Based on RDF and RDF schema (RDFS), W3C released OWL to enhance the semantic expression. OWL uses description logic as the theoretical basis, and it has rich knowledge representation and reasoning ability, which is the designated ontology description language of W3C. In this paper, we extend OWL to represent the heterogeneous ocean data resources with a hierarchical structure, as summarized in Table 1.

Table 1. Semantic Representation

Semantics	Notation
$Class(c_i)$	(c_i, τ, C_i)
$equalClassOf(C_i, C_j)$	$(C_i, =, C_j)$
$subClassOf(C_i, C_j)$	$(C_i, \prec_{sc}, C_j)$
$grandcldClassOf(C_i, C_j)$	$(C_i, \prec_{sc}, C_k)$, $(C_k, \prec_{sc}, C_j)$
$siblingClassOf(C_i, C_j)$	$(C_i, \prec_{sc}, C_k)$, $(C_j, \prec_{sc}, C_k)$, $(C_i, \neq, C_j)$
$Domain(p_i)$	$(p_i, \hookleftarrow_d, C_i)$
$Range(p_i)$	$(p_i, \hookrightarrow_r, C_i)$
$subPropertyOf(p_i, p_j)$	$(p_i, \prec_{sp}, p_j)$

④https://github.com/sayingxmu/DOLP, Sept. 2021.

3.1.1 Hierarchical Structure of Concepts

A domain ontology can be represented by a set of triples like (s, p, o), where subjects s and objects o are the entities, describing either ontology concepts or blank nodes.

Definition 1 (Class). *Given $\forall i$, C_i represents the i-th concept of ontology O, and any instance of C_i is denoted as (c_i, τ, C_i), i.e., c_i is an instance of C_i.*

Definition 2 (Equal Class). *Given the semantics of C_i and C_j are equivalent, we define that C_i is the equal class of C_j, denoted as triple $(C_i, =, C_j)$.*

Definition 3 (Sub Class). *Given the concepts C_i and C_j, C_i is defined as the sub class of C_j (or C_j is the super class of C_i), denoted as the triple $(C_i, \prec_{sc}, C_j)$, if the semantics of the concepts satisfies $C_i \subseteq C_j$.*

Definition 4 (Grandchild Class). *Given the concepts C_i, C_j, and C_k, C_i is defined as the grandchild class of C_j (or C_j is the grandparent class of C_i), if there exist $(C_i, \prec_{sc}, C_k)$ and $(C_k, \prec_{sc}, C_j)$.*

Definition 5 (Sibling Class). *Given the concepts C_i, C_j, and C_k, C_i and C_j are defined as the sibling class for each other, if there exist $(C_i, \prec_{sc}, C_k)$, $(C_j, \prec_{sc}, C_k)$, and $(C_i, \neq, C_j)$.*

3.1.2 Properties

In OWL, the properties of ontology represent the relationship between two entities. Generally, the object properties ("owl: ObjectProperty") and datatype properties ("owl: DatatypeProperty") are user-defined according to the domain knowledge. Besides, there are inherent properties. Similar to the concepts, the properties also can be described with a hierarchical structure.

Definition 6 (Sub Property). *Given the properties p_i and p_j, p_i is defined as the sub-property of p_j, if the semantics of p_i is the subset of p_j (i.e., $p_i \subseteq p_j$), which is denoted as the triple $(p_i, \prec_{sp}, p_j)$.*

For each property, we define two operators $\hookleftarrow_d$ and $\hookrightarrow_r$ to represent the domain and the range of the property, respectively, e.g., triples $(p_i, \hookleftarrow_d, C_i)$ and $(p_i, \hookrightarrow_r, C_j)$ show that C_i is the domain and C_j is the range of property p_i. To differentiate the latter description of user-defined property, we define the properties $\hookleftarrow_d, \hookrightarrow_r, \prec_{sp}, \prec_{sc}, =$, and $\neq$ as inherent properties, which are used to describe the hierarchical structure of an ontology with RDF.

According to Definitions 2–6, the property p in triple (s, p, o) represents the object properties or datatype properties; thus $p \notin \{\tau, \hookleftarrow_d, \hookrightarrow_r, \prec_{sp}, \prec_{sc}, =, \neq\}$.

3.2 Top Level Class

Generally, ocean data consists of observations, numerical simulation results, and manually recorded data by human beings. Because numerical simulation results are usually released in a regular structure, and the amount of manually recorded data is relatively small, the OEDO model in this paper mainly focuses on the large-scale, multi-source, and heterogeneous ocean observation data. Three elements including the observations, the sensors for observations, and the observing platforms are the core of data-intensive science, especially in the ocean area. To make the OEDO model as comprehensive as possible, on the one hand, we have investigated many typical ocean observation projects and systems to integrate domain concepts, including the MMI project [36], the European SeaDataNet⑤, the EMODnet⑥, the Integrated Ocean Observing System of NOAA (IOOS)⑦, the Global Ocean Observing System (GOOS)⑧, and the sub-projects of them. On the other hand, we have reviewed the aforementioned metadata inventories and the related literature [14, 15, 35, 38, 54] to obtain ontology classes, relations, and the corresponding object properties.

The top level of the proposed OEDO is depicted in Fig.2, which consists of the key concepts including Observation, Sensor, System and Platform. In addition, the relations between these concepts are represented by the object properties. Specifically, Observation is observed by (isObservedBy) some Sensors, which is deployed on (isDeployedOn) observing Platforms, and the observing systems include (hasPlatform) various Platforms.

As shown in Fig.2, there are many reverse object properties in OEDO, e.g., the object property isObservedBy is the reverse of ("owl: inverseOf") hasObservation. In this way, we can obtain the observation of a certain sensor, and at the same time if we retrieve some observations, we can obtain the sensor that senses the observation data, as well as the platform where the sensor is deployed, and the related information. This

⑤https://www.seadatanet.org, Aug. 2021.
⑥http://eurogoos.eu/emodnet, Aug. 2021.
⑦https://ioos.noaa.gov, Aug. 2021.
⑧http://www.goosocean.org, Aug. 2021.

is designed to enhance the data linking capability and the data discovery efficiency.

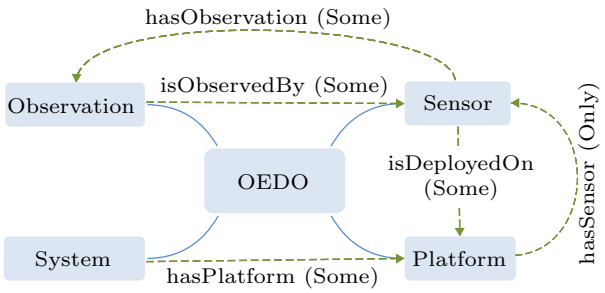


Fig.2. Key concepts and relations of OEDO [34].

Note that, in this paper we mainly focus on the concepts of Platform, Sensor and Observation, which are closely related to data collections and services. Thus, in the following we describe the representations in the OEDO model in detail.

3.3 Ocean Observing Platform

As described in our earlier work [34], platforms are the infrastructure of ocean observing activities. However, there are many kinds of platforms produced by various manufacturers, and deployed and managed by different organizations. Thus, the multiple standards from different organizations lead to unclear description of the datasets produced by them, and even the datasets cannot be reused. In this subsection, we describe the representation of these platforms by hierarchical classification and property description in detail.

3.3.1 Hierarchical Classification

In order to describe the observing platforms in a hierarchical way, we first categorize them into four subclasses, which are all annotated by the is-a relation and it is transitive ("owl:TransitiveProperty"). Moreover, as shown in Fig.3, we further subdivide platforms into three layers. Specifically, the second layer of the ocean observing platforms consists of four types, i.e., 1) GroudBasedPlatform, which is located on the surface of the Earth; 2) OceanBasedPlatform, including the platforms that are both on the ocean surface and underwater; 3) SpaceBasedPlatform, which mainly refers to satellites; 4) AirBasedPlatform, including those above the surface of the Earth and below the middle atmosphere.

To clearly illustrate the representation of hierarchical classification in the OEDO model, we take OceanSurfacePlatform as an example. Generally, platforms deployed on ocean surface include fixed ones like moored buoys, and the mobile ones like boat, ship, research vessel, tow fish, and drifting buoys, which are usually used for obtaining sea surface data, such as temperature, salinity and wave height. These items are the commonly used keywords for users to find the datasets they need, and thus we add them as instances of the ontologies. In this way, we try to improve the data discovery services with the semantic extension of OEDO.

3.3.2 Property Description

With the help of the ODAS metadata and the Global Change Master Directory (GCMD) keywords[9], we represent various platforms in an interoperable form. Totally, there are 10 basic classes for describing a platform, i.e., ID, URL, Type, Feature, Location, Sensor, ValidTime, TransmitTime, DataFormat, and Organization in the OEDO model. The details of the corresponding object properties are summarized in Table 2.

Taking the concept Sensor as an example, it can be represented by the triple (Platform, hasSensor, $sensor_i$), where hasSensor is one of the object properties of the class Platform, and $sensor_i$ is a specific

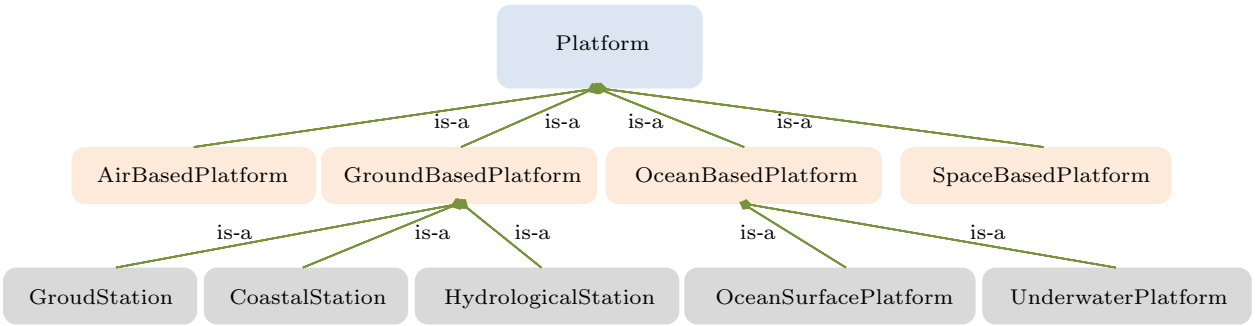


Fig.3. Classification of platforms [34].

[9] https://forum.earthdata.nasa.gov/app.php/tag/GCMD%20Keywords, Aug. 2021.

Table 2. Object Properties of Platforms [34]

Object Property	Range	Description
hasID	ID	Unique identification number
hasURL	URL	Web address to obtain its additional information
hasType	Type	Type of the platform, such as moored buoy type
hasFeature	Feature	Shape and size (i.e., length, width or diameter)
hasLocation	Location	Where the platform is deployed (i.e., longitudes, latitudes and vertical height)
hasSensor	Sensor	Sensors deployed on the platform
hasValidTime	ValidTime	The initiation date and the operational end-date
hasTransTime	TransmitTime	The data transmitted in real time, near real time or delay model, and the time interval of transmitting
hasDataFormat	DataFormat	Data format of produced data
hasOrganization	Organization	Who deploys and manages the platform

sensor deployed on this platform. The domain and the range of hasSensor are represented by triples (hasSensor, $\hookleftarrow_d$, Platform) and (hasSensor, $\hookrightarrow_r$, Sensor), respectively. These triples are the basis of the horizontal extending of the QSQL optimization, since they facilitate the semantic reasoning based on the RDF entailment rules, and the details are shown in Subsection 4.3.

3.4 Sensor

In ocean domain, sensors refer to the physical objects that detect external stimuli (i.e., changes in the physical world) [14, 55, 56]. Various sensors are deployed on the specific observing platform to perform sensing and transform an incoming stimulus into another, often with digital representation, which is represented as Observation in the OEDO in Subsection 3.5. The representation of sensors in our model is shown in Fig.4. We extract the general and core concepts and relationships of sensors in the SSN ontology [15], and extend the properties of sensors for ocean data acquisition.

Generally, each sensor has its own external survival condition (SurvialCondition), and only under this condition can it be able to (hasCapability) measure the features of interest (Property), such as Wind and Pressure, which are described as the subclasses of MeasurementCapability.

But what needs to be noted is that the performance of a specific sensor may be affected by the prevailing conditions. For example, the Accuracy of a sensing device may vary greatly under different external environmental conditions. Therefore, we relate class MeasurementCapability with Performance through the object property hasPerformance in our OEDO model.

3.5 Observation

In order to be consistent with the SSO design pattern [14, 15], observations in OEDO are modeled as the sensor output and events in the physical world. The

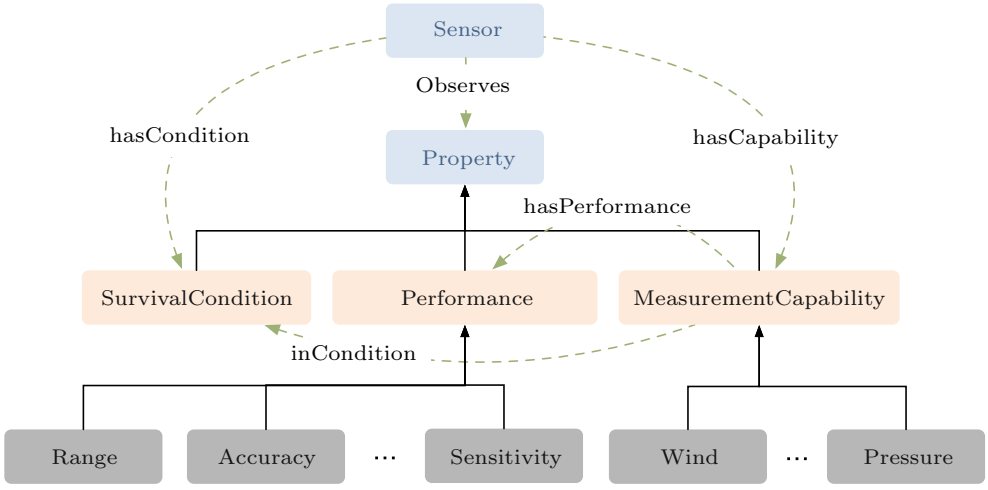


Fig.4. Representation of sensors.

details are described in our previous study [34].

3.5.1 Hierarchical Classification

As the development of ocean monitoring systems, there are observations from multiple disciplines that include various physical parameters. In this subsection, we represent the hierarchical classification of observations in OEDO, which is designed to facilitate data access for data users from different disciplines.

As shown in Fig.5, there are seven subclasses of ocean observations. Taking MarineMeteorology as an example, the sensors deployed on the buoy platform sense stimuli 10 meters above sea surface, such as wind (speed and direction), air pressure, humidity, and visibility. These stimuli are interpreted as physical parameters in datasets, which will be described in Subsection 3.5.2 in detail.

3.5.2 Metadata Description of Dataset

Generally, observation is an ensemble of datasets received by the sensors with different spatial resolutions and temporal grids, which can be described with its own metadata. In our OEDO model, we represent the main items of metadata which focus on improving the FAIR services of ocean data, and model them as datatype properties and object properties of the unified ontology to make various datasets interoperable. The simplified representation of datasets in the OEDO model is shown in Fig.6, in which the blue lines refer to datatype property, while green lines refer to object property.

In our model, any dataset must be identified by a name (hasName) and a unique ID (hasID), which are both represented by datatype properties. It is worth pointing out that the name is crucial for data discovery,

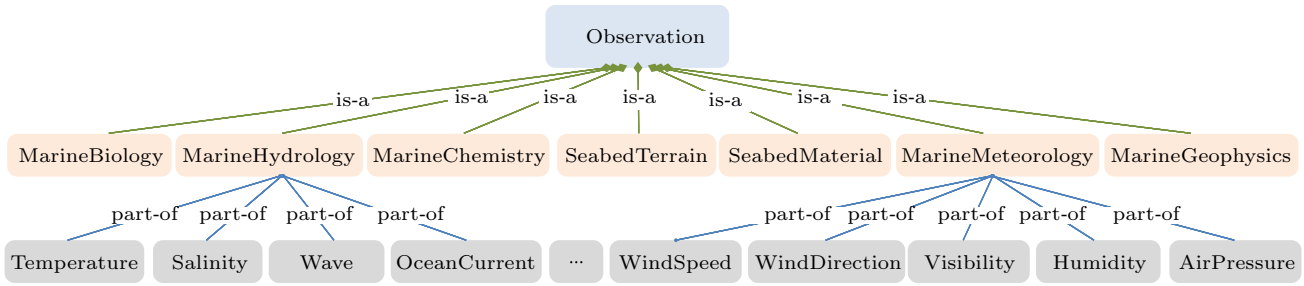


Fig.5. Classification of observations [34].

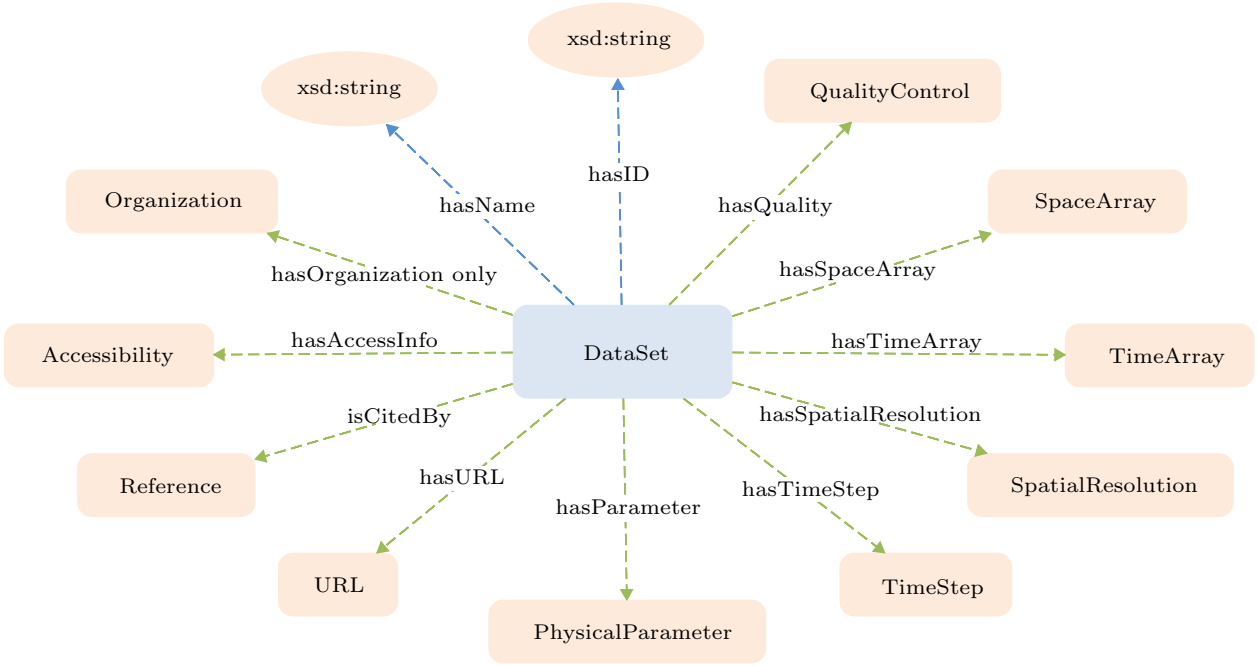


Fig.6. Simplified representation of datasets [34].

that is, making the dataset findable. Data users will learn the accessibility information by the base classes of Accessibility, URL and the corresponding object properties, which satisfy the accessible requirement of FAIR principle. In addition, users can cite the dataset by references (Reference), which will encourage data producers and organizations to share their data, and facilitate data access further. The stimuli sensed at a certain resolution (SpatialResolution) and time step (TimeStep) are interpreting as physical field (PhysicalParameter), which can be represented by a four-dimensional variable including space location (SpaceArray), and time sequences (TimeArray), i.e., (longitude, latitude, vertical depth, time). With the information of dataset publication organization (Organization), as well as the detailed quality control and quality assurance (QualityControl), the dataset can be reused without much additional endeavor for tracing.

4 QSQL Optimization

The proposed OEDO model has represented the heterogeneous ocean data in a unified form using metadata and taxonomy. In this section, we publish the OEDO model as data services to improve data discovery and access. QSQL is essentially a data structure and is exclusively designed to overcome the inefficiency problem of the traditional semantic service discovery methods based on direct reasoning, which implement the computationally complex and time-consuming semantic reasoning during querying. QSQL has been proved to be an effective data structure for service discovery [10, 16, 32]. Here, we first introduce the basic structure of QSQL, and then describe the details of the two optimization methods, named as hierarchical extending and multi-property reasoning.

4.1 Basic Structure of QSQL

As described in Section 3, the semantic information in the ontology model can be represented by directed graphs. The vertex represents each basic ontology class, while the arc is notated as the relationship between the corresponding concepts. QSQL is created at the service publishing stage, and it stores the semantic network graph in adjacency lists. The basic structure of QSQL is shown in Fig.7.

Note that each QSQL element represents an ontology concept, which consists of a link domain (Fig.7(a)) and a data domain (Fig.7(b)). The link domain stores the inferred relationships from the service model, including the links to its Equal, Super, Sibling, Sub, Grandparent and Grandchild classes, which will speed up service queries by avoiding repeated reasoning, while the data domain stores services that use this concept as their input or output under different matching degrees.

4.2 Hierarchical Expansion of QSQL

QSQL is designed to reduce the query time of the published services, which simplifies the service model by the assumption that QSQL elements are the concrete ontology concepts published by the service model. As a result, service publishers are implicitly required to annotate their service interfaces by domain concepts. Accordingly, service users are also required to query services by concrete semantic concepts. However, there is usually a lack of recognized and interoperable domain ontology. Furthermore, most cloud service publishers and application users do not have much domain knowledge, especially for the field of data-intensive ocean science, which will also affect the capability of service discovery and access [34].

In this context, to improve the hierarchical reasoning performance of our model, we extend the concepts in OEDO by adding the corresponding relations of synonym, hypernym and hyponym in WordNet, which enables fuzzy matching.

4.2.1 *Extending Rules*

The extending rules are described in detail in the following. The symbols related to the rules are defined in Table 3.

Rule 1. *Since it is likely that there is no concrete concept in the model that completely matches the input/output parameters of a data service, we extend equal classes by the WordNet relation synonym, which can be written as*:

$$\forall C_i \in C,\ E_i = E_i \cup Syn_w(C_i).$$

Rule 2. *Extending parent classes of the concepts associated with is-a relation by their hypernym*:

$$\begin{aligned} &\forall C_i \in C, \\ &Grdp_i = Grdp_i \cup Hype_w(Sup_i), \\ &Sup_i = Sup_i \cup Hype_w(C_i). \end{aligned}$$

Rule 3. *Extending subclasses of the concepts associated with part-of relation by their hyponym*:

$$\forall C_i \in C,$$

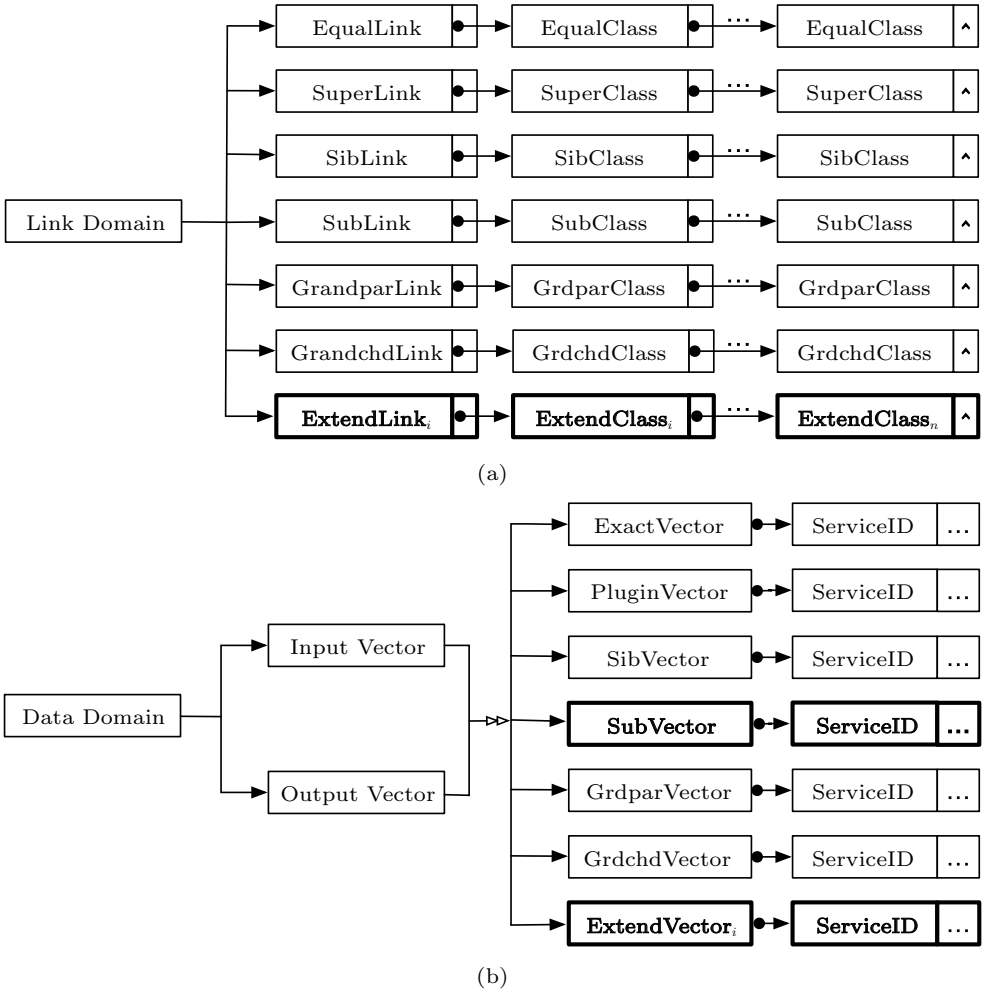


Fig.7. Basic structure of QSQL. The extended nodes are marked in bold. (a) Link domain of QSQL. (b) Data domain of QSQL.

$$Grdc_i = Grdc_i \cup Hypo_w(Sub_i),$$
$$Sub_i = Sub_i \cup Hypo_w(C_i).$$

Table 3. Symbol Definitions of Rules to Extend Semantics in OEDO

Symbol	Semantics in OEDO
E_i	Equal classes of C_i
Sup_i	Super classes of C_i
Sub_i	Sub classes of C_i
$Grdp_i$	Grandparent classes of C_i
$Grdc_i$	Grandchild classes of C_i
Ext_i	Extended classes of C_i
$Syn_w(C_i)$	Synonyms of C_i
$Hype_w(C_i)$	Hypernyms of C_i
$Hypo_w(C_i)$	Hyponyms of C_i

It is worth mentioning that in addition to extending the basic class in Fig.7, we add a new subclass in this paper to represent part-of relations in OEDO, which is achieved by Rule 3. The motivation for adding this type is that ocean data have special characteristics. For instances, ocean data usually have spatiotemporal properties, and the time span and the spatial coverage are unlikely completely consistent with the requirement of users, but the existing data within the temporal or the spatial range can be recommended or provided to users. Similarly, an application user may require a dataset with multiple ocean physical fields, such as ocean temperature and salinity. If there is no complete dataset containing all the required fields, the request may be satisfied by providing multiple subsets, each of which contains one or more required fields.

4.2.2 *Hierarchical Extending*

Specifically, we model the concepts in OEDO as data services with input and output parameters, and then publish the services to the optimized QSQL with the extending rules. Then, we generate a data service

index list to facilitate data access and improve data discovery. Algorithm 1 shows the procedure of publishing the output parameters of a data service to the optimized QSQL. The main processing steps of hierarchical extending are summarized as follows:

1) getting the concrete concepts from the model and the synonym from WordNet corresponding to each parameter, and extending the equal classes by Rule 1, which is described by lines 2–4;

2) for each element of equal classes, finding if the element has been added to QSQL; if not, building the concept's node, appending the data service ID to its ExactVector, and then building the EqualLink of the node (lines 6–10);

3) inferring the super classes of each element in equal classes (line 11) by reasoner, extending it by retrieving the hypernym from WordNet according to Rule 2 (lines 12 and 13), and setting the PluginVector and SuperLink of the node (lines 19 and 20);

4) applying Rule 2 and Rule 3 to extend grandparent and grandchild classes respectively (line 22);

5) returning an optimized QSQL with data service model published in it at the end (line 24).

Algorithm 1. hierarchical_extending

Input: basic QSQL Q_s, output parameter P of service S_i
Output: optimized data resource list Q_s

1: **for** parameter $P_i \in P$ **do**
2: $C_i = get\_modelconcept(P_i)$
3: $Syn_w = get\_Synonym\_WordNet(P_i)$
4: $E_i = C_i \cup Syn_w$; /* Rule 1 */
5: **for** each E_i **do**
6: **if** $!search\_servicelist(Q_s, E_i)$ **then**
7: $build\_conceptnodes(Q_s, E_i)$
8: **end if**
9: $Q_s.E_i.OUTPUT.Exact\_Vec.add(S_i)$
10: $Q_s.E_i.EqualLink = build\_link(P_i, E_i)$
11: $Sup_i = get\_superconcept(E_i)$
12: $Hype_w = get\_hypernym\_wordnet(E_i)$
13: $Sup_i = Sup_i \cup Hype_w$; /* Rule 2*/
14: **end for**
15: **for** each Sup_i **do**
16: **if** $!search\_servicelist(Q_s, Sup_i)$ **then**
17: $build\_conceptnodes(Q_s, Sup_i)$
18: **end if**
19: $Q_s.Sup_i.Output.Plugin\_Vec.add(S_i)$
20: $Q_s.Sup_i.SuperLink = build\_link(P_i, Sup_i)$
21: **end for**
22: extending grandparent and grandchild classes using Rule 2 and Rule 3 respectively
23: **end for**
24: **return** Q_s

4.3 Horizontal Expansion of QSQL

The hierarchical extending based on WordNet (Algorithm 1) supports fuzzy matching in case that there is no exact concept in the model. In addition, when the exact concept corresponding to a user's query does exist in the model, users may desire to find the concepts directly or indirectly related to the concrete concept, i.e., the neighbor concepts that are related to the queried one through one- or multi-hop relationships.

In this case, we propose a multi-property reasoning algorithm (Algorithm 2), which makes full use of user-defined object properties to improve the performance of data discovery and access further. Based on the definitions in Subsection 3.1, we conduct semantic reasoning from two perspectives, i.e., RDFS entailment rules and OWL semantics.

Algorithm 2. multiproperty_reasoning

Input: basic QSQL Q_s, output parameter P of service S_i
Output: optimized data resource list Q_s

1: **for** parameter $P_i \in P$ **do**
2: $C_i = get\_modelconcept(P_i)$
3: **if** $C_i ==$ NULL **then**
4: Continue
5: **end if**
6: $toExtend = C_i$
7: **while** $toExtend \neq \emptyset$ **do**
8: **for** $C_i' \in toExtend$ **do**
9: $p_i = get\_objectproperty(C_i')$ /* RDF rules */
10: **if** $p_i ==$ NULL **then**
11: Continue
12: **end if**
13: **for** $p_i' \in p_i$ **do**
14: $Ext_i = get\_objectvalue(p_i')$
15: $dist = cal\_distance(C_i, Ext_i)$
16: **if** $dist > H$ **then**
17: Continue
18: **end if**
19: $toExtend = toExtend \cup Ext_i$
20: **if** $!search\_servicelist(Q_s, Ext_i)$ **then**
21: $build\_conceptnodes(Q_s, Ext_i)$
22: **end if**
23: $Q_s.Ext_i.Output.Ext\_Vec.add(S_i)$
24: $Q_s.Ext_i.ExtendLink = build\_link(P_i, Ext_i)$
25: **end for**
26: **end for**
27: $toExtend = toExtend - C_i'$
28: **end while**
29: **end for**
30: **return** Q_s;

4.3.1 Reasoning Based on RDFS Entailment Rules

According to the entailment patterns given in OWL 2 RDF-Based Semantics[10], we formalize the reasoning rule as $tail(r) \Rightarrow head(r)$, if the implicit triple $head(r)$ can be inferred from the explicit triples $tail(r)$ [8], which is extended from RDFS entailment rules. We list the inference rules closely related to the object properties in Table 4.

Table 4. RDF Entailment Rules [8]

Rule Name	Entailment Rule
rdfs2	$(p_i, \hookleftarrow_d, C_i), (C_j, p_i, C_k) \Rightarrow (C_j, \tau, C_i)$
rdfs3	$(p_i, \hookrightarrow_r, C_i), (C_j, p_i, C_k) \Rightarrow (C_k, \tau, C_i)$
rdfs5	$(p_i, \prec_{sp}, p_j), (p_j, \prec_{sp}, p_k) \Rightarrow (p_i, \prec_{sp}, p_k)$
rdfs7	$(p_i, \prec_{sp}, p_j), (C_i, p_i, C_j) \Rightarrow (C_i, p_j, C_j)$
ext1	$(p_i, \hookleftarrow_d, C_i), (C_i, \prec_{sc}, C_j) \Rightarrow (p_i, \hookleftarrow_d, C_j)$
ext2	$(p_i, \hookrightarrow_r, C_i), (C_i, \prec_{sc}, C_j) \Rightarrow (p_i, \hookrightarrow_r, C_j)$
ext3	$(p_i, \prec_{sp}, p_j), (p_j, \hookleftarrow_d, C_i) \Rightarrow (p_i, \hookleftarrow_d, C_i)$
ext4	$(p_i, \prec_{sp}, p_j), (p_j, \hookrightarrow_r, C_i) \Rightarrow (p_i, \hookleftarrow_r, C_i)$

4.3.2 Reasoning Based on OWL Semantics

The characteristics of object properties also imply reasoning functions, such as "owl: inverseOf" and "owl: TransitiveProperty", which help to improve the access and recall rate of the concepts effectively.

Assuming that the property p of triple (s, p, o) is inverse of property p', when querying s, o can be accessed through p. At the same time, by the inverse property p', s and its associated concepts can be accessed when o is given. In other words, the two-way query of concepts can be realized through "owl: inverseOf" semantics.

Example 1. Extreme weather, such as typhoons, usually causes serious disasters. Forecasting of typhoon path and intensity is an important means of disaster prevention and mitigation. The first task of forecasting is to find the relevant datasets of the sea area. Let us assume a certain location of the sea area (SpaceArray) is given. As shown in Fig.8, if the properties between DataSet and SpaceArray are inverse, then the corresponding DataSet can be found through the given sea area (SpaceArray), as well as the associated addresses (URL), and the ocean physical elements (PhysicalParameter) contained in the datasets, such as sea surface temperature (SST), salinity, and density [57].

The inherent properties $\prec_{sp}$, $\prec_{sc}$, $=$, and $\neq$ are obviously transitive ("owl: TransitiveProperty"), e.g., grandcldClassOf(C_i, C_j) in Table 1 is defined by the transitive property $\prec_{sc}$. Apart from these, some user-defined object properties also specify the transitive characteristics. Through "owl: TransitiveProperty", the hierarchical structure of ontology can be extended easily.

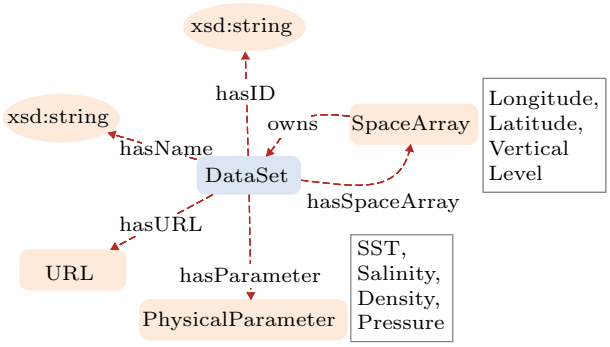


Fig.8. Example of reasoning based on OWL semantics.

4.3.3 Multi-Property Extending

Based on the RDF entailment rules and the OWL semantics, we propose Algorithm 2 to extend the semantics of the OEDO model by user-defined object properties horizontally, and store the extended neighbor concepts in QSQL. The details of the multi-property reasoning can be summarized as follows:

1) determining whether the model contains the exact concept corresponding to each data service parameter P_i, if so, add it to set *toExtend* as a concept to be extended (lines 2–6);

2) for any parameter P_i, if the set *toExtend* is non-empty (line 7), then for each concept C_i' in the *toExtend*, get its object properties set p_i (lines 8–12);

3) for each property p_i', get the corresponding concept Ext_i, and calculate the distance (hops) between the related concepts Ext_i and the exact concept C_i; if the distance is less than a specified value H, add Ext_i to set *toExtend* for further extending (lines 13–19);

4) if the related concept Ext_i has not been built yet, then set the *ExtendVector* and *ExtendLink* of the node (lines 23 and 24);

5) after reasoning on all the properties of the concept C_i', delete it from set *toExtend* (line 27).

Note that threshold H can be set dynamically according to the precision and recall rate of service discovery. By default, we set $H = 3$, and we will discuss the value in Subsection 5.3.2.

[10] https://www.w3.org/TR/rdf11-mt/#rdfs-entailment, Aug. 2021.

5 Experiment and Evaluation

In this section, we evaluate the effectiveness and efficiency of the proposed schemes in terms of data discovery and data access. The experiments are conducted on a PC with 1.2 GHz Intel Core m5 and 8 GB RAM. The system is implemented by Java language. We use *protégé* 5.5 to construct the OEDO model, which is a free, open-source ontology editor and framework for building knowledge-based solutions in many scientific and commercial areas. The package of *protégé* can be easily imported in Java for parsing the concepts and relationships of the ontology model. Racer 2.0 is used as the basic semantic reasoning tool, and Mysql 8.0.18 is used to store the published data services.

Considering the lack of real data services of ocean data, we generate 500 abstract data services by using the OEDO ontology as the output of the services, and generate request models to simulate service requests of data users. It should be noted that in all the experiments, the threshold in the algorithm is set to the default value, unless otherwise specified, e.g., the hops $H = 3$ in Algorithm 2 for default. The maximum number of queries presented in this paper is 35. It is enough for comparing the time performance with other methods, and the quality of data service discovery is not directly affected by the number of queries [32]. We run each experiment three times and report the average result.

Before presenting the experiments and analysis, we give the following naming conventions to simplify the frequently-used description firstly.

• *Evaluation Metrics.* Symbols T, Tr, M, S, P, R, and $F1$ represent for the evaluation metrics response time, time trend, memory usage, semantic capabilities, precision, recall, and $F1$-score respectively.

• *Evaluation Methods.* DR and KW are the abbreviations of the traditional semantic service discovery methods based on direct reasoning and keyword respectively [16, 32]. QB, QH and QM are the abbreviations of the three methods based on the basic QSQL, QSQL optimized by algorithm *hierarchical_extending*, and QSQL optimized by algorithm *multiproperty_reasoning*, respectively. In addition, Q represents all the three methods (QB, QH and QM) based on QSQL structures.

• *Experiment Analysis.* The abbreviation letters of evaluation methods are used as the subscripts of evaluation metrics, e.g., T_{DR} represents the response time of the traditional semantic method based on direct reasoning.

5.1 Feasibility Analysis

5.1.1 Time Performance Analysis

Generally, service query methods include keyword query [58] and semantic query. Due to the semantic reasoning functionality, the performance of semantic query method is better than that of keyword query in service discovery. However, the traditional semantic services discovery method based on direct reasoning performs semantic inference during querying, resulting in the low efficiency of service discovery. In view of this, QSQL data structure is designed to infer and store semantic relationships in the service publishing stage, and thus to avoid reasoning during service querying.

In order to evaluate the feasibility of using extended QSQL to improve the efficiency of semantic service query, we analyze the ratio of semantic reasoning and matching time to total query time. As shown in Fig.9, we take several typical concepts of the OEDO model as examples, and use the traditional semantic query method to query data services. We can see that the loading and reasoning time accounts for the largest proportion, with an average of 75.4%, and the matching time only accounts for 1.7%.

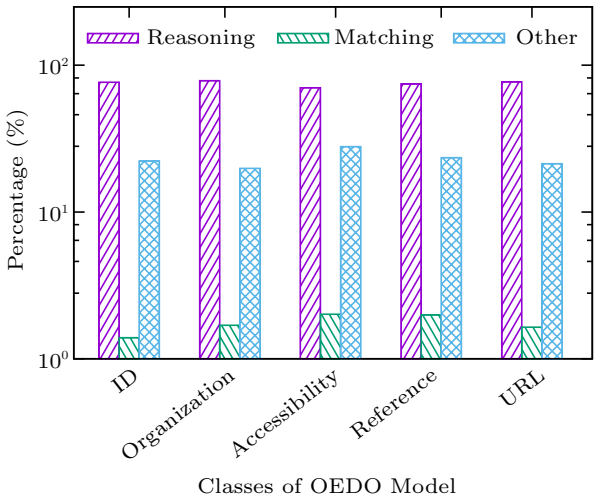


Fig.9. Time distribution of semantic services query.

Therefore, it can be expected that advancing the semantic reasoning to the service publishing stage will effectively shorten the service query time. The improvement of data discovery performance is verified in Subsection 5.2.

5.1.2 Space Performance Analysis

Semantic reasoning is not only a time-consuming task, but also a memory-intensive task. Frequent infer-

ences of semantic service querying may cause memory jitter. In this subsection, we evaluate the memory usage (M) in semantic reasoning by publishing different numbers of data services.

We report the average results of running each method three times. Particularly, in order to eliminate the impact of service comparisons in publishing and repeated data writing, we implement one of the three runs of each method with the empty database table for storing the inference results.

The comparison between the direct reasoning method and the QSQL-based methods is shown in Fig.10. We conclude and analyze the results as follows.

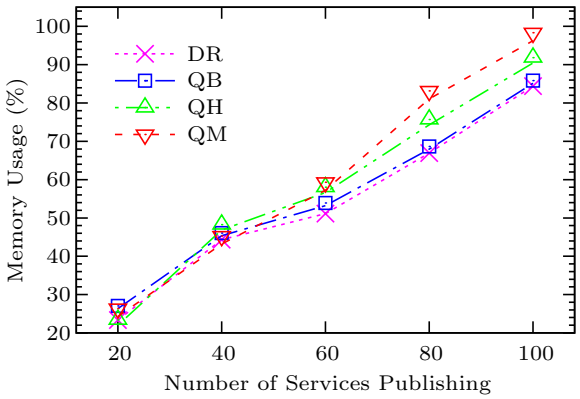


Fig.10. Memory usage of semantic reasoning.

1) $M_{\mathrm{DR}} \leqslant M_{\mathrm{Q}}$. The reason is that QSQL-based methods need to dynamically construct the QSQL data structures during reasoning. What is more, the QH method extends the hierarchical semantics by reasoning based on WordNet, and the QM method extends the semantic by reasoning based on multiple object properties.

2) The more the publishing services N_p, the larger the memory usage M. This illustrates that the memory utilization of semantic reasoning will continue to increase as the number of publishing services increases.

3) Given a certain N_p, e.g., when $N_p = 100$, we can get that $M_{\mathrm{QB}} < M_{\mathrm{QH}} < M_{\mathrm{QM}}$. Moreover, the more the services N_p, the larger the difference in M between the four methods. It shows that the memory usage M is positively related to the complexity of semantic relations S, i.e., $M \propto S$. This is consistent with the fact of $S_{\mathrm{QB}} \subset S_{\mathrm{QH}} \subset S_{\mathrm{QM}}$, which is analyzed in detail in Subsection 5.3.1.

The traditional DR method performs semantic reasoning during service querying, which may cause system instability when processing large amount of service querying. Therefore, in order to support the reasoning of more complex semantic relationships, it is essential to avoid reasoning during service querying.

5.2 Performance of Data Service Discovery

To evaluate the efficiency of our schemes, we test and analyze the response time (T) and time trend (Tr) of data service discovery, by conducting the following two groups of comparative experiments respectively.

- Comparison between different kinds of query methods, i.e., the traditional DR method, the KW method and the QB method, where QB is the baseline of the series of semantic querying methods based on QSQL structures.
- Comparison between the series of the semantic query methods based on QSQL. This is designed to evaluate the performance of the two optimizing algorithms, i.e., QH and QM methods. Again, the benchmark is the QB method.

Note that the comparison between DR, KW and QB has been reported in our previously published conference paper [34]. We present it here again to compare the response time of DR, KW and QB with that of QH and QM, as the OEDO model is extended in this paper.

5.2.1 *Response Time*

The response time of processing different numbers of queries by different methods is shown in Fig.11.

Firstly, the response time of the three types of methods is shown in Fig.11(a) where we can see the followings.

1) T_{DR} increases as the number of query services N_q increases, and T_{DR} is the longest when N_q is determined.

2) $T_{\mathrm{KW}} > T_{\mathrm{QB}}$ and $T_{\mathrm{KW}} < T_{\mathrm{DR}}$, but the KW method does not support semantic reasoning, which is analyzed in Subsection 5.3.1.

3) T_{QB} is the shortest one when N_q is determined, $T_{\mathrm{QB}} \ll T_{\mathrm{DR}}$ and $T_{\mathrm{QB}} \ll T_{\mathrm{KW}}$. The reason is that the QSQL data structure stores the semantic relationships of the OEDO model and generates a service index list, and it only needs to get the related services from the list when processing a query. However, the DR method needs to perform semantic reasoning during querying, and the reasoning time accounts for a large proportion of the response time, as analyzed in Subsection 5.1.1.

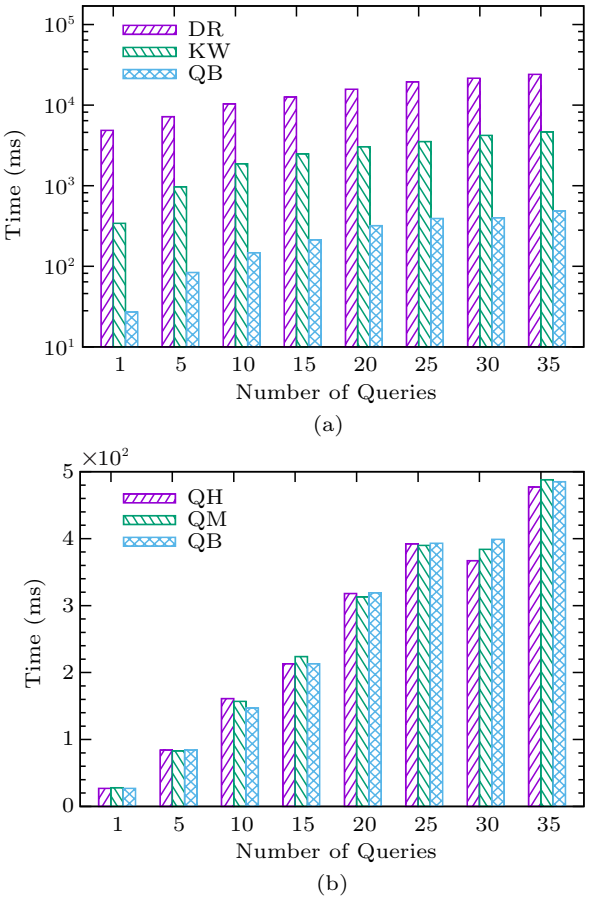


Fig.11. Response time of querying by different methods. (a) Different kinds of methods. (b) QSQL-based methods.

Secondly, Fig.11(b) illustrates the response time of querying services published by the series of QSQL-based methods. In general, $T_{\mathrm{QB}} \approx T_{\mathrm{QH}} \approx T_{\mathrm{QM}}$ when N_q is determined. QH and QM are extended from QB hierarchically and horizontally, and the extending is completed at the service publishing stage. Therefore, the response time of querying based on them does not increase, which is consistent with the original intention of the QSQL design.

5.2.2 Time Trend

The trends of response time for each query are shown in Fig.12, which illustrate the stability of each method. In Fig.12(a), we can see that Tr_{DR} changes drastically as the number of queries changes. In comparison, Tr_{KW} and Tr_{QB} are relatively stable, regardless of how many queries are processed.

The trends comparison of QSQL-based methods is shown in Fig.12(b). Except for querying only one service (the overhead accounts for a lot), the average time difference of multiple service queries is less than 3 ms. It shows that the three QSQL-based methods are stable and reliable, and the hierarchical and the horizontal extending of the structure still guarantee the stability.

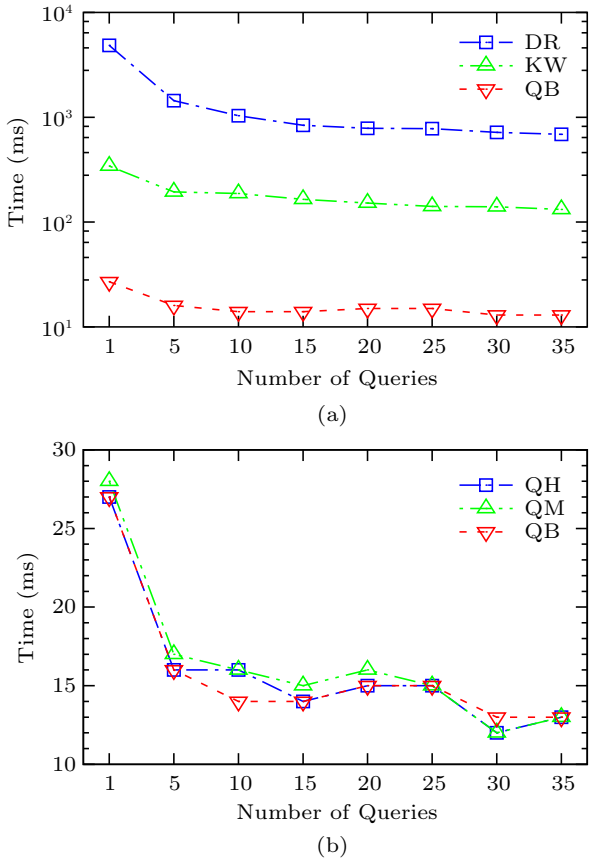


Fig.12. Trends of average response time of different methods. (a) Different kinds of methods. (b) QSQL-based methods.

We conclude the average query time of various methods in Table 5, where we can see that the performance of the three QSQL-based methods is significantly better than that of DR and KW methods. Compared with the basic QSQL method QB, the extending of the QSQL structure (QH and QM) neither improves nor affects the performance of data service querying shown in bold in Table 5, because the semantic reasoning is completed in the service publishing stage. But they improve the efficiency of data access, which is analyzed in Subsection 5.3.

Table 5. Comparison of Average Query Time of Methods DR, KW, QB, QH and QM

Services Discovery Method	Average Query Time (ms)
DR	1 423
KW	194
QB	16
QH	**16**
QM	**16**

5.3 Quality of Data Service Discovery

5.3.1 Semantic Capabilities Analysis

The KW and DR methods are traditional service query methods. The advantage of semantic query is that it supports reasoning according to the semantic relations. In addition to semantic reasoning, the QH method extends the semantics of services hierarchically by WordNet, while the QM method enhances the ability of semantic association horizontally by multiple object properties of the unified OEDO model.

The semantic capabilities of the five methods are concluded in Table 6, which shows as follows.

Table 6. Comparison of Semantic Capabilities Supported by Methods KW, DR, QB, QH and QM

Relationship	KW	DR	QB	QH	QM
Equal	Y	Y	Y	Y	Y
Sibling	N	Y	Y	Y	Y
Parent	N	Y	Y	Y	Y
Grandparent	N	Y	Y	Y	Y
Grandchild	N	Y	Y	Y	Y
Sub	N	N	N	**Y**	**Y**
Property-related	N	N	N	**N**	**Y**

1) The KW method only supports querying the services that exactly match the users' requirement. If the keywords inputted by a data user do not exactly match the published data services, the user will get nothing. The DR method can not only return the services that exactly match the requested one, but also recommend related services according to the semantic relationship between the requested and the published ones, i.e., $S_{\mathrm{DR}} \supset S_{\mathrm{KW}}$.

2) Both DR and QB methods support the top five semantic relationships; thus $S_{\mathrm{DR}} = S_{\mathrm{QB}}$.

3) The optimized QH method lists the sub services, and the QM method recommends the property-related services of the requested one as shown in bold in Table 6, which are corresponding to the data domain of the optimized QSQL structures, namely $S_{\mathrm{QB}} \subset S_{\mathrm{QH}} \subset S_{\mathrm{QM}}$.

In a nutshell, the semantic capabilities satisfy $S_{\mathrm{KW}} \subset S_{\mathrm{DR}} = S_{\mathrm{QB}} \subset S_{\mathrm{QH}} \subset S_{\mathrm{QM}}$. The QM method proposed in our paper supports processing more complex semantic relations.

5.3.2 Quality Evaluation

In this subsection, we evaluate the effectiveness of our proposed method in terms of precision (P), recall (R), and $F1$-score ($F1$) of data services, which are defined as follows:

$$P = \frac{TP}{TP + FP},$$
$$R = \frac{TP}{TP + FN},$$
$$F1 = 2 \times \frac{P \times R}{P + R},$$

where TP, FP, and FN represent for the true positive, false positive and false negative services, respectively.

As mentioned in Subsection 4.3.3, the hops H between the concepts and their neighbors in Algorithm 2 can be set dynamically. Thus, we evaluate the effects of H on the precision and recall of data services discovery by the QM method first, and then compare it with the other methods.

Effect of H on the QM Method. In our OEDO model, we describe the relationships between concepts by object properties in the form of $C_1 \xrightarrow{p_1} C_2 \xrightarrow{p_2} C_3 \xrightarrow{\cdots}$. For instance, in Platform $\xrightarrow{\text{hasSensor}}$ Sensor $\xrightarrow{\text{observes}}$ Observation $\xrightarrow{\text{hasDataset}}$ Dataset, and the hops H between classes Platform and Dataset is 3. We set the hops between the queried concepts and their neighbors as $H = 1$, 2, 3, 4, and 5 respectively to test the precision and recall of the QM method, and calculate the $F1$-score, as shown in Fig.13.

It can be seen in Fig.13(a) that when $H = 1$, P_{QM} gets its maximum value. However, R_{QM} does not increase with the increase of H, and R_{QM} is the optimal when $H = 3$ (Fig.13(b)), as the number of hops between most classes in our OEDO model is 3. We can conclude that the services discovery quality of QM is the best when $H = 3$, i.e., when $F1_{\mathrm{QM}}$ gets the maximum value (Fig.13(c)). The advantage of the QM method is its high recall, as shown in Fig.13(d), and $R_{\mathrm{QM}} > P_{\mathrm{QM}}$ except for $H = 1$, when the extended services are the immediate neighborhood of the queried services.

Comparison Between Different Methods. To evaluate the services discovery quality of our methods, we compare them with the traditional DR method, the KW method and our baseline method QB. The experimental results are shown in Fig.14.

As shown in Fig.14(a), P_{DR} and P_{QB} are the highest (92.1%). Because methods DR and QB implement the same semantic reasoning procedure, they support the same semantic relationships (as shown in Subsection 5.3.1). In comparison, P_{QH} and P_{QM} are slightly lower than P_{DR} and P_{QB}, which is determined by the semantic accuracy of the extended concepts. Nevertheless, $P_{\mathrm{Q}} > P_{\mathrm{KW}}$, i.e., the precision of our QSQL-based

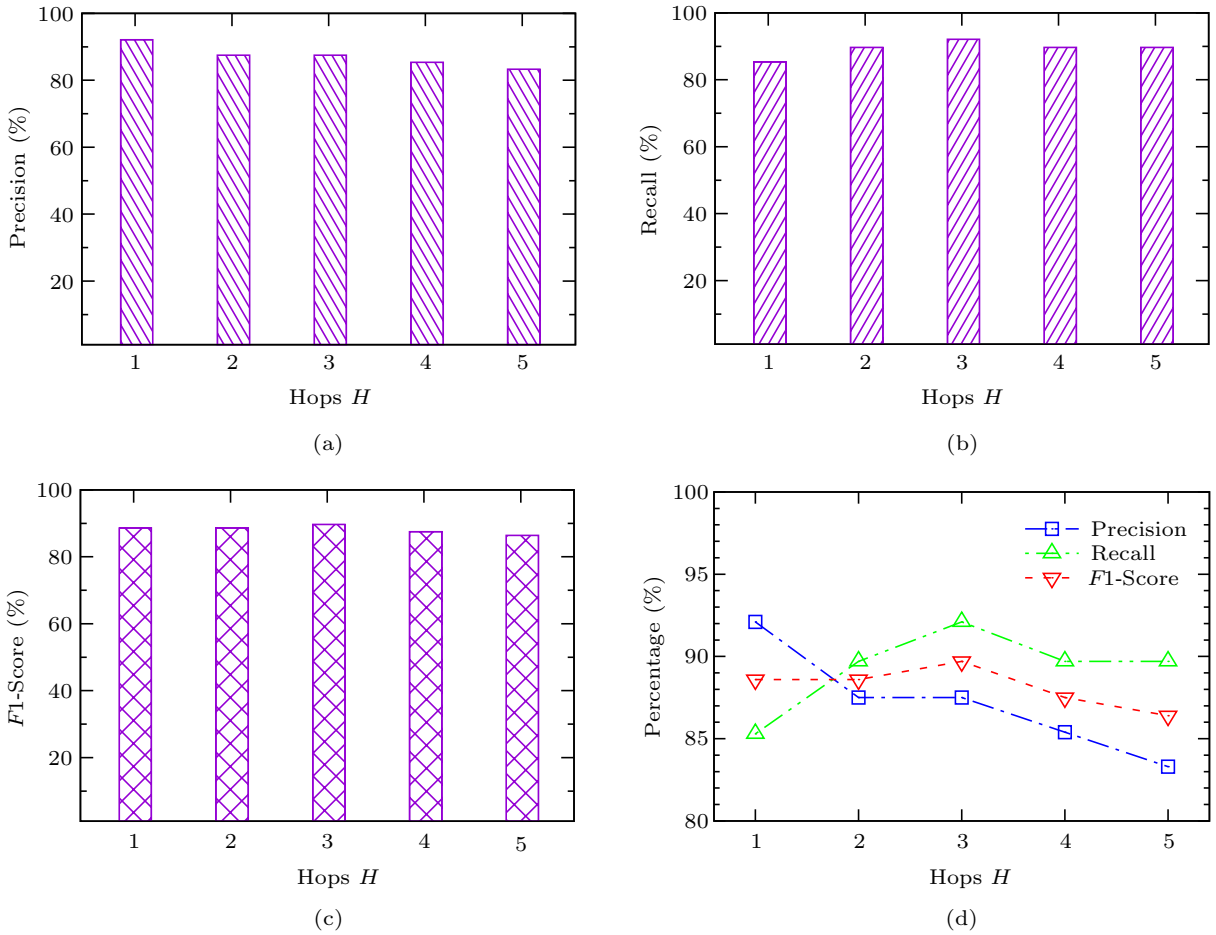


Fig.13. Effect of H on the services discovery quality of the QM method. (a) Precision. (b) Recall. (c) $F1$-score. (d) Comparison of the metrics.

methods is higher than that of the KW method, which does not support fuzzy matching.

The recall rates of the five methods are shown in Fig.14(b). The QH query method is extended hierarchically by the lexical WordNet, and the synonyms, hypernyms, and hyponyms of the requested data services can be found easily. Thus, the recall $R_{\mathrm{QH}} > R_{\mathrm{QB}}$, $R_{\mathrm{QH}} > R_{\mathrm{DR}}$, and $R_{\mathrm{QB}} = R_{\mathrm{DR}}$. R_{QM} is the highest recall rate, as the QM method horizontally extends the semantic relations by the multiple object properties of the OEDO model (with the default hops $H = 3$).

As shown in Fig.14(c), $F1_{\mathrm{QM}}$ is the largest among the five methods, followed by $F1_{\mathrm{QH}}$. Although there exist $P_{\mathrm{QB}} > P_{\mathrm{QH}}$ and $P_{\mathrm{QB}} > P_{\mathrm{QM}}$, we can get that $R_{\mathrm{QB}} < R_{\mathrm{QH}}$ and $R_{\mathrm{QB}} < R_{\mathrm{QM}}$ lead to $F1_{\mathrm{QB}} < F1_{\mathrm{QH}} < F1_{\mathrm{QM}}$ (Fig.14(d)). Therefore, we can conclude that the overall services query quality of QM and QB proposed in this paper are higher than that of other methods.

6 Conclusions

In this paper, we designed and implemented a novel system called DOLP to improve ocean data services by introducing semantics, which consists of three components: the OEDO ontology model which is constructed by ocean metadata and published as the input/output parameters of data services, the QSQL data structure for storing the pre-inferred concepts and semantic relationships, and the optimization algorithms for semantic extending.

Compared with the keyword-based method, the semantic-based DOLP with the optimization algorithms supports querying data services semantically related to the users requests. Therefore, a higher data recall rate is obtained. The querying performance of DOLP outperforms the traditional direct-reasoning semantic method, which is beneficial from the optimized QSQL data structures.

Although the OEDO model is ocean-domain spe-

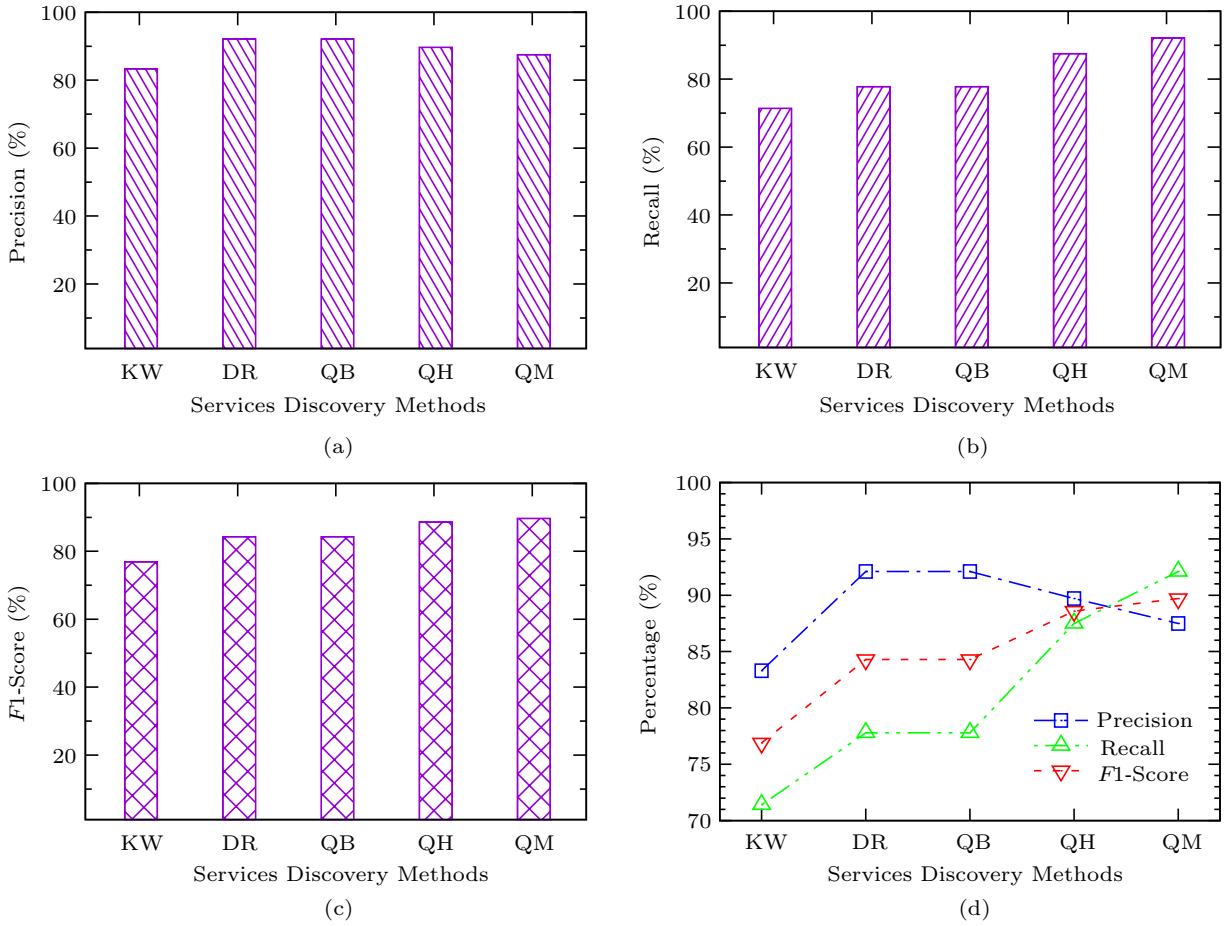


Fig.14. Services discovery quality of different methods. (a) Precision. (b) Recall. (c) $F1$-score. (d) Comparison of the metrics.

cific, it can be easily replaced by other ontologies and extended to the corresponding applications. QSQL is a general data structure, which is designed for storing the pre-inferred semantic information, thus to reduce the service query delay. The optimization algorithms are domain-independent. Furthermore, the three components of our system are loosely coupled. In other words, the system is scalable.

While we believe DOLP to be an effective system for ocean data services, there are still limitations. We attempt to construct the OEDO model using as comprehensive ocean metadata as possible. However, the type and amount of ocean data is growing rapidly, and correspondingly, the metadata is increasing. Therefore, the OEDO model needs to be updated regularly. It is expected that the ODEO model can be updated automatically with the latest metadata. The hierarchical optimization of QSQL is based on WordNet, which is limited in the domain knowledge. Integrating special domain knowledge bases will bring better performance of data access and become more friendly to domain users, which is a key challenge faced by the ocean data services.

As future work, to improve the performance of our system further, we plan to extend the OEDO model by considering the latest metadata of observations, the numerical prediction results, as well as the ocean environmental data recorded by human beings, and integrating ocean domain knowledge. In addition to storing the semantic relationships, QSQL will be further optimized by constructing the temporal and spatial indices.

Acknowledgement(s) We would like to thank the anonymous reviewers for their valuable and constructive comments. We thank Dr. Xiang Wang from National University of Defense Technology for the discussion and useful commentary on various drafts of this paper.

References

[1] Agapiou A. Remote sensing heritage in a petabyte-scale: Satellite data and heritage earth engine© appli-

cations. *Int. J. Digit. Earth*, 2017, 10(1): 85-102. DOI: 10.1080/17538947.2016.1250829.

[2] Alotaibi R, Bursztyn D, Deutsch A, Manolescu I, Zampetakis S. Towards scalable hybrid stores: Constraint-based rewriting to the rescue. In *Proc. the 2019 International Conference on Management of Data*, Jun. 2019, pp.1660-1677. DOI: 0.1145/3299869.3319895.

[3] Mattson T, Rogers J, Elmore A J. The BigDAWG polystore system. In *Making Databases Work: The Pragmatic Wisdom of Michael Stonebraker*, Brodie M L (ed.), Association for Computing Machinery and Morgan & Claypool, 2019, pp.279-289. DOI: 10.1145/3226595.3226620.

[4] Elmore A J, Duggan J, Stonebraker M *et al.* A demonstration of the BigDAWG polystore system. *Proc. VLDB Endow.*, 2015, 8(12): 1908-1911. DOI: 10.14778/2824032.2824098.

[5] Wilkinson M D, Sansone S A, Schultes E, Doorn P, Da Silva Santos L O B, Dumontier M. A design framework and exemplar metrics for FAIRness. *Scientific Data*, 2018, 5: Article No. 180118. DOI: 10.1038/sdata.2018.118.

[6] Tanhua T, Pouliquen S, Hausman J *et al.* Ocean FAIR data services. *Front. Mar. Sci.*, 2019, 6: Article No. 440. DOI: 10.3389/fmars.2019.00440.

[7] Reed G. Project report: Marine environmental data inventory (MEDI). In *Proc. the 19th Session of the IOC Committee on International Oceanographic Data and Information Exchange*, March 2007.

[8] Buron M, Goasdoué F, Manolescu I, Mugnier M. Ontology-based RDF integration of heterogeneous data. In *Proc. the 23rd International Conference on Extending Database Technology*, March 30-April 2, 2020, pp.299-310. DOI: 10.5441/002/edbt.2020.27.

[9] Wilkinson M D, Dumontier M, Aalbersberg I J *et al.* The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, 2016, 3: Article No. 160018. DOI: 10.1038/sdata.2016.18.

[10] Ren K, Liu X, Chen J, Xiao N, Song J, Zhang W. A QSQL-based efficient planning algorithm for fully-automated service composition in dynamic service environments. In *Proc. the 2008 IEEE International Conference on Services Computing*, Jul. 2008, pp.301-308. DOI: 10.1109/SCC.2008.26.

[11] Crasso M, Mateos C, Zunino A, Campo M. Easysoc: Making web service outsourcing easier. *Inf. Sci.*, 2014, 259: 452-473. DOI: 10.1016/j.ins.2010.01.013.

[12] Brabra H, Mtibaa A, Sliman L, Gaaloul W, Gargouri F. Semantic web technologies in cloud computing: A systematic literature review. In *Proc. the 2016 IEEE International Conference on Services Computing*, Jun. 27-Jul. 2, 2016, pp.744-751. DOI: 10.1109/SCC.2016.102.

[13] Imam F T. Application of ontologies in cloud computing: The state-of-the-art. arXiv:1610.02333, 2016. http://arxiv.org/abs/1610.02333, Jan. 2021.

[14] Janowicz K, Compton M. The stimulus-sensor-observation ontology design pattern and its integration into the semantic sensor network ontology. In *Proc. the 3rd International Workshop on Semantic Sensor Networks*, Nov. 2010, pp.64-78.

[15] Compton M, Barnaghi P M, Bermudez L *et al.* The SSN ontology of the W3C semantic sensor network incubator group. *J. Web Semant.*, 2012, 17: 25-32. DOI: 10.1016/j.websem.2012.05.003.

[16] Zhou A, Ren K, Li X, Zhang W, Ren X. Building quick resource index list using WordNet and high-performance computing resource ontology towards efficient resource discovery. In *Proc. the 21st IEEE International Conference on High Performance Computing and Communications, the 17th IEEE International Conference on Smart City and the 5th IEEE International Conference on Data Science and Systems*, Aug. 2019, pp.885-892. DOI: 10.1109/HPCC/SmartCity/DSS.2019.00129.

[17] Castañé G G, Xiong H, Dong D, Morrison J P. An ontology for heterogeneous resources management interoperability and HPC in the cloud. *Future Gener. Comput. Syst.*, 2018, 88: 373-384. DOI: 10.1016/j.future.2018.05.086.

[18] Sun L, Ma J, Wang H, Zhang Y, Yong J. Cloud service description model: An extension of USDL for cloud services. *IEEE Trans. Serv. Comput.*, 2018, 11(2): 354-368. DOI: 10.1109/TSC.2015.2474386.

[19] Challita S, Paraiso F, Merle P. Towards formal-based semantic interoperability in multi-clouds: The FCLOUDS framework. In *Proc. the 10th IEEE International Conference on Cloud Computing*, Jun. 2017, pp.710-713. DOI: 10.1109/CLOUD.2017.98.

[20] Yongsiriwit K, Sellami M, Gaaloul W. A semantic framework supporting cloud resource descriptions interoperability. In *Proc. the 9th IEEE International Conference on Cloud Computing*, Jun. 27-Jul. 2, 2016, pp.585-592. DOI: 10.1109/CLOUD.2016.0083.

[21] Bermudez-Edo M, Elsaleh T, Barnaghi P M, Taylor K. IoT-Lite: A lightweight semantic model for the internet of things and its use with dynamic semantics. *Pers. Ubiquitous Comput.*, 2017, 21(3): 475-487. DOI: 10.1007/s00779-017-1010-8.

[22] Elsaleh T, Enshaeifar S, Rezvani R, Acton S T, Janeiko V, Bermudez-Edo M. IoT-Stream: A lightweight ontology for Internet of Things data streams and its use with data analytics and event detection services. *Sensors*, 2020, 20(4): Article No. 953. DOI: 10.3390/s20040953.

[23] Cong Z, Fernández A, Billhardt H, Lujak M. Service discovery acceleration with hierarchical clustering. *Inf. Syst. Frontiers*, 2015, 17(4): 799-808. DOI: 10.1007/s10796-014-9525-2.

[24] Roman D, Kopecký J, Vitvar T, Domingue J, Fensel D. WSMO-Lite and hRESTS: Lightweight semantic annotations for Web services and RESTful APIs. *J. Web Semant.*, 2015, 31: 39-58. DOI: 10.1016/j.websem.2014.11.006.

[25] Rodríguez-Mier P, Pedrinaci C, Lama M, Mucientes M. An integrated semantic web service discovery and composition framework. *IEEE Trans. Serv. Comput.*, 2016, 9(4): 537-550. DOI: 10.1109/TSC.2015.2402679.

[26] Chen F, Li M, Wu H, Xie L. Web service discovery among large service pools utilising semantic similarity and clustering. *Enterp. Inf. Syst.*, 2017, 11(3): 452-469. DOI: 10.1080/17517575.2015.1081987.

[27] Zhang N, Wang J, Ma Y, He K, Li Z, Liu X F. Web service discovery based on goal-oriented query expansion. *J. Syst. Softw.*, 2018, 142: 73-91. DOI: 10.1016/j.jss.2018.04.046.

[28] Garriga M, Renzis A D, Lizarralde I, Flores A, Mateos C, Cechich A, Zunino A. A structural-semantic web service selection approach to improve retrievability of web services. *Inf. Syst. Frontiers*, 2018, 20(6): 1319-1344. DOI: 10.1007/s10796-016-9731-1.

[29] Paliwal A V, Shafiq B, Vaidya J, Xiong H, Adam N R. Semantics-based automated service discovery. *IEEE Trans. Serv. Comput.*, 2012, 5(2): 260-275. DOI: 10.1109/TSC.2011.19.

[30] Ma S P, Chen Y J, Syu Y, Lin H J, FanJiang Y Y. TEST-Oriented RESTful service discovery with semantic interface compatibility. *IEEE Trans. Serv. Comput.*. DOI: 10.1109/TSC.2018.2871133.

[31] Dong X, Madhavan J, Halevy A Y. Mining structures for semantics. *ACM SIGKDD Explorations Newsletter*, 2004, 6(2): 53-60. DOI: 10.1145/1046456.1046463.

[32] Ren K, Xiao N, Chen J. Building quick service query list using WordNet and multiple heterogeneous ontologies toward more realistic service composition. *IEEE Trans. Serv. Comput.*, 2011, 4(3): 216-229. DOI: 10.1109/TSC.2010.24.

[33] Miller G A. WordNet: A lexical database for English. *Commun. ACM*, 1995, 38(11): 39-41. DOI: 10.1145/219717.219748.

[34] Ren X, Li X, Deng K, Ren K, Zhou A, Song J. Bringing semantics to support ocean FAIR data services with ontologies. In *Proc. the 2020 IEEE International Conference on Services Computing*, Nov. 2020, pp.30-37. DOI: 10.1109/SCC49832.2020.00011.

[35] Bermudez L, Graybeal J, Arko R. A marine platforms ontology: Experiences and lessons. In *Proc. the ISWC Workshop on Semantic Sensor Networks*, November 2006.

[36] Graybeal J, Bermudez L, Bogden P, Miller S, Watson S. Marine metadata interoperability project: Leading to collaboration. In *Proc. the IEEE International Symposium on Mass Storage Systems and Technology*, Jun. 2005, pp.14-18. DOI: 10.1109/LGDI.2005.1612458.

[37] Lowry R, Leadbetter A. Semantically supporting data discovery, markup and aggregation in the European marine observation and data network (EMODnet). In *Proc. the European Geosciences Union General Assembly*, April 27-May 2, 2014.

[38] Bart A A, Churuksaeva V V, Fazliev A Z, Privezentsev A I, Gordov E P, Okladnikov I G, Titov A G. Ontological description of meteorological and climate data collections. In *Proc. the 19th International Conference on Data Analytics and Management in Data Intensive Domains*, Oct. 2017, pp.266-272.

[39] Plebani P, Pernici B. URBE: Web service retrieval based on similarity evaluation. *IEEE Trans. Knowl. Data Eng.*, 2009, 21(11): 1629-1642. DOI: 10.1109/TKDE.2009.35.

[40] Wang Y, Lin X, Wu L, Zhang W. Effective multi-query expansions: Collaborative deep networks for robust landmark retrieval. *IEEE Trans. Image Process.*, 2017, 26(3): 1393-1404. DOI: 10.1109/TIP.2017.2655449.

[41] Rekik M, Boukadi K, Ben-Abdallah H. Cloud description ontology for service discovery and selection. In *Proc. the 10th International Conference on Software Engineering and Applications*, Jul. 2015, pp.26-36. DOI: 10.5220/0005556400260036.

[42] Parhi M, Pattanayak B K, Patra M R. An ontology-based cloud infrastructure service discovery and selection system. *Int. J. Grid Util. Comput.*, 2018, 9(2): 108-119. DOI: 10.1504/IJGUC.2018.10012792.

[43] Calvanese D, Giacomo G D, Lembo D, Lenzerini M, Poggi A, Rodriguez-Muro M, Rosati R, Ruzzi M, Savo D F. The MASTRO system for ontology-based data access. *Semantic Web*, 2011, 2(1): 43-53. DOI: 10.3233/SW-2011-0029.

[44] Rodríguez-Muro M, Kontchakov R, Zakharyaschev M. Ontology-based data access: Ontop of databases. In *Proc. the 12th International Semantic Web Conference*, Oct. 2013, pp.558-573. DOI: 10.1007/978-3-642-41335-3_35.

[45] Pinto F D, Lembo D, Lenzerini M, Mancini R, Poggi A, Rosati R, Ruzzi M, Savo D F. Optimizing query rewriting in ontology-based data access. In *Proc. the 16th International Conference on Extending Database Technology*, Mar. 2013, pp.561-572. DOI: 10.1145/2452376.2452441.

[46] Hovland D, Kontchakov R, Skjæveland M G, Waaler A, Zakharyaschev M. Ontology-based data access to Slegge. In *Proc. the 16th International Semantic Web Conference*, Oct. 2017, pp.120-129. DOI: 10.1007/978-3-319-68204-4_12.

[47] Lanti D, Xiao G, Calvanese D. Cost-driven ontology-based data access. In *Proc. the 16th International Semantic Web Conference*, Oct. 2017, pp.452-470. DOI: 10.1007/978-3-319-68288-4_27.

[48] Botoeva E, Calvanese D, Cogrel B, Corman J, Xiao G. A generalized framework for ontology-based data access. In *Proc. the 2018 International Conference of the Italian Association for Artificial Intelligence*, Nov. 2018, pp.166-180. DOI: 10.1007/978-3-030-03840-3_13.

[49] Xiao G, Calvanese D, Kontchakov R, Lembo D, Poggi A, Rosati R, Zakharyaschev M. Ontology-based data access: A survey. In *Proc. the 27th International Joint Conference on Artificial Intelligence*, Jul. 2018, pp.5511-5519. DOI: 10.24963/ijcai.2018/777.

[50] Buron M, Goasdoué F, Manolescu I, Mugnier M. Reformulation-based query answering for RDF graphs with RDFS ontologies. In *Proc. the 16th International Conference*, Jun. 2019, pp.19-35. DOI: 10.1007/978-3-030-21348-0_2.

[51] Peng P, Zou L, Özsu M T, Chen L, Zhao D. Processing SPARQL queries over distributed RDF graphs. *The VLDB J.*, 2016, 25(2): 243-268. DOI: 10.1007/s00778-015-0415-0.

[52] Quamar A, Lei C, Miller D, Özcan F, Kreulen J, Moore R J, Efthymiou V. An ontology-based conversation system for knowledge bases. In *Proc. the 2020 International Conference on Management of Data*, Jun. 2020, pp.361-376. DOI: 10.1145/3318464.3386139.

[53] Zhang N, Wang J, Ma Y. Mining domain knowledge on service goals from textual service descriptions. *IEEE Trans. Serv. Comput.*, 2020, 13(3): 488-502. DOI: 10.1109/TSC.2017.2693147.

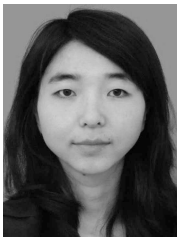
[54] Dividino R, Soares A, Matwin S, Isenor A W, Webb S, Brousseau M. Semantic integration of real-time heterogeneous data streams for ocean-related decision making. In *Proc. the Specialists' Meeting on Big Data and Artificial Intelligence for Military Decision Making*, May 2018.

[55] Wilson W J, Yueh SH, Dinardo S J, Chazanoff S L, Kitiyakara A, Li F K, Rahmat-Samii Y. Passive active L- and S-band (PALS) microwave sensor for ocean salinity and soil moisture measurements. *IEEE Trans. Geosci. Remote Sens.*, 2001, 39(5): 1039-1048. DOI: 10.1109/36.921422.

[56] Loni Z M, Espinosa H G, Thiel D V. Floating monopole antenna on a tethered subsurface sensor at 433 MHz for ocean monitoring applications. *IEEE Journal of Oceanic Engineering*, 2017, 42(4): 818-825. DOI: 10.1109/JOE.2016.2639111.

[57] Liu S S, Sun L, Wu Q, Yang Y J. The responses of cyclonic and anticyclonic eddies to typhoon forcing: The vertical temperature-salinity structure changes associated with the horizontal convergence/divergence. *Journal of Geophysical Research: Oceans*, 2017, 122(6): 4974-4989. DOI: 10.1002/2017JC012814.

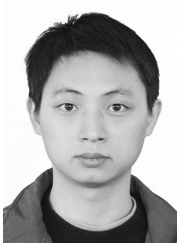
[58] Smits G, Pivert O, Jaudoin H, Paulus F. AGGREGO SEARCH: Interactive keyword query construction. In *Proc. the 17th International Conference on Extending Database Technology*, Mar. 2014, pp.636-639. DOI: 10.5441/002/edbt.2014.62.



Xiao-Li Ren received her B.S. degree in computer science and technology from Central South University, Changsha, in 2008, and her M.S. degree in computer application technology from Dalian University of Technology, Dalian, in 2011. Currently she is a Ph.D. candidate in the College of Computer Science and Technology at National University of Defense Technology, Changsha. Her research interests include service-oriented computing and ocean big data. She is a member of CCF.

Kai-Jun Ren received his Ph.D. degree in computer science and technology from the National University of Defense Technology (NUDT), Changsha, in 2008. He is currently a professor at NUDT, Changsha. His main research interests include service-oriented computing, high-performance computing, scientific workflow and AI. He is a member of CCF.

Zi-Chen Xu received his Ph.D. degree in electronic and computer engineering from the Ohio State University, Columbus, in 2016. He is a full professor with Nanchang University, Nanchang. His research spans in the area of data-conscious complex system, including profile analysis, system optimization, and storage system design/implementation. He is a senior member of CCF and a member of ACM and IEEE.



Xiao-Yong Li received his Ph.D. degree in computer science and technology from College of Computer Science, National University of Defense Technology (NUDT), Changsha, in 2013. Currently he is an associate professor in the College of Meteorology and Oceanography at NUDT, Changsha. His current research interests include data stream management, parallel computing, uncertain queries, cloud computing and service-oriented computing. He is a senior member of CCF.

Ao-Long Zhou received his B.S. and M.S. degrees in computer science and technology from National University of Defense Technology (NUDT), Changsha, in 2014 and 2020, respectively. Currently he is a Ph.D. candidate in the College of Computer Science and Technology at NUDT, Changsha. His research interests include service oriented computing, cloud computing and machine learning.

Jun-Qiang Song received his B.S. and M.S. degrees in applied mechanics from National University of Defense Technology (NUDT), Changsha, in 1983 and 1986, respectively. He is currently a full professor of College of Meteorology and Oceanography at NUDT, Changsha. His research interests include numerical weather prediction and high-performance computing. Prof. Song is an academician of Chinese Academy of Engineering.

Ke-Feng Deng received his B.S., M.S., and Ph.D. degrees in computer science from National University of Defense Technology (NUDT), Changsha, in 2007, 2010, and 2015, respectively. He is currently an associate professor in the College of Meteorology and Oceanography at NUDT, Changsha. His research interests include parallel and distributed systems, cloud computing, scientific workflow, ocean big data and machine learning. He is a member of CCF.