Ontology based heterogeneous materials database integration and semantic query

Shuai Zhao, and Quan Qian

Citation: AIP Advances **7**, 105325 (2017);
View online: https://doi.org/10.1063/1.4999209
View Table of Contents: http://aip.scitation.org/toc/adv/7/10
Published by the American Institute of Physics

Ontology based heterogeneous materials database integration and semantic query

Shuai Zhao and Quan Qian[a]
School of Computer Engineering & Science, Shanghai University, Shanghai 200444, China and Materials Genome Institute, Shanghai University, Shanghai 200444, China

(Received 6 August 2017; accepted 23 October 2017; published online 31 October 2017)

Materials digital data, high throughput experiments and high throughput computations are regarded as three key pillars of materials genome initiatives. With the fast growth of materials data, the integration and sharing of data is very urgent, that has gradually become a hot topic of materials informatics. Due to the lack of semantic description, it is difficult to integrate data deeply in semantic level when adopting the conventional heterogeneous database integration approaches such as federal database or data warehouse. In this paper, a semantic integration method is proposed to create the semantic ontology by extracting the database schema semi-automatically. Other heterogeneous databases are integrated to the ontology by means of relational algebra and the rooted graph. Based on integrated ontology, semantic query can be done using SPARQL. During the experiments, two world famous First Principle Computational databases, OQMD and Materials Project are used as the integration targets, which show the availability and effectiveness of our method.
https://doi.org/10.1063/1.4999209

I. INTRODUCTION

Traditionally, the materials science heavily relies on costly experiments and simulation based methods to understand the intrinsic mechanisms of the relationships among processing-structure-property-performance (PSPP). Currently, the big data generated by high throughput experiments and computations has provided a great opportunities for data-driven based techniques, which is one of the three pillars of materials genome initiatives (MGI). Data-driven materials techniques are playing a big role in revealing PSPP relationships in materials science, which not only can be used for both property prediction based forward models, but also materials discovery based inverse models. So, the data-driven featured materials science is regarded as the essential content of materials informatics, which provides the foundations for fourth paradigm of materials discovery.[1,2]

With the data generating techniques are getting easier than before, the requirement of data sharing and data integration gradually become urgent.[3] However, this problem faces many challenges. Firstly, the data types are quite different from discrete to continuous, from simple text to complex images, videos, etc. Secondly, in different data sources the data schema or format are quite different which makes it difficult to understand with each other. Thirdly, the data quality in different sources are also different, which makes data evaluation and selection difficult. Finally, even we have found the data, recognizing what the rows and columns represent can be another challenge, because many of the datasets have machine-readable descriptions, but often these are in very large data dictionary files full of terminology that is often designed primarily for the experts in a given field.

[a]Corresponding Author Email: qqian@shu.edu.cn

Ontology, which is used to capture knowledge about some domain of interest, is widely used in knowledge engineering, information retrieval, information integration, etc. An ontology usually describes not only the concepts in a domain, but also the relationships that hold among those concepts. This paper presents a methodology that integrating heterogeneous relational databases by transforming one database into ontology and mapping others into it, and then do semantic query on the integrated ontology based system.

The rest of the paper is organized as follows: Section II and III discuss the related work and the framework of the whole system. Mapping from the relational database to ontology and other heterogeneous database integration are described in Section IV and V. The experimental integrations of two famous materials databases are shown in Section VI. Section VII provides some final conclusions and directions for the future work.

II. RELATED WORK

The integration of distributed heterogeneous database, sometimes called data integration, is an active area of research. Concerning about the heterogeneous database integration, it can be divided into two categories according to the query method. One is the data warehouse which means the whole data is integrated and stored in one data source. The other is on-demand retrieval that only when the end users send the query to the system, the query execution engine extracts data from the different data sources.

Query expansion is an important issue in the field of information retrieval. Chokri et al put forward Ontology-based Query Expansion[4] which can expend a single SQL (structured query language) query into several queries. It utilize the synonym and parent concept in ontology to fulfil the expansion which is only suitable when the attribute in database equals to the concept in ontology.

Bonatti et al put forward an ontology extended relation (OER), which contains an ordinary relation as well as an associated ontology conveying semantic meaning about the terms being used.[5] They extended the relational algebra to query OERs. And the advantage of their method is that OER model can not only be directly built on top commercial relational databases, but also can be scaled to handle large data sets.

Ranganathan and Liu[6] proposed a system to bridge the semantic gap between the user given queries and the queries can be answered by the database. They use domain knowledge contained in ontologies, that extends relational databases with the ability to answer semantic queries expressed in SPARQL.[7] Based on a semantic model of data, end users express their queries in SPARQL, and they get back semantically relevant results. The experimental results show a good performance on sample relational database, using a combination of standard and custom ontologies.

Ontologies are becoming increasingly commonplace for semantically representing knowledge in a formal manner that facilitates sharing and integrating rich information for materials informatics. Moreover, ontologies can support logic reasoning by rule engines that enhance knowledge acquisition automatically. Generally speaking, the purpose of materials informatics ontologies can be defined as three concrete objectives:[8] (1) Translate data and information into knowledge that is useful, not only to materials scientists, but also to application engineers, regulators, and other users. (2) Curate the knowledge base to align with the emerging materials scientific research and industrial application development. (3) Present the knowledge in a flexible architecture that is understandable to each kind of user group. So far, quite a lot materials ontologies have been defined, and the representative ones are as follows:

Plinius ontology[9] is the earliest materials ontology which is developed for ceramic materials that covers the conceptualisation of the chemical composition of materials. The Plinius ontology is given as a conceptual construction kit, involving several sets of atomic concepts and construction rules for making complex concepts. Plinius ontology does not depend on specific language so that it can be implemented in several languages or tools, such as Prolog, Ontolingua and LOOM.

Ashino et al[10] developed an information platform for data exchange between heterogeneous materials data resources, in which there are two components, materials data portal service and ontology based materials data exchange. The materials data portal service mainly aggregates materials

databases' information through RDF site summary (RSS) technology, which includes materials type, properties or other items to identify a material database. Ashino ontology covers quite a few fields in materials science especially on thermal properties. The Ashino ontology contains more than 600 classes implemented in OWL (Web ontology language).

Cheung et al[11] developed a semantic web application, named MatSeek, that aims to integrate heterogeneous databases associated with materials science. MatSeek relies on a machine-processable OWL ontology (MatOnto[12]) to correlate processing parameters with nano-structure, physical and chemical properties to help scientists discover potential new materials for specific and high-priority applications. MatSeek provides a federated search interface over several critical materials science databases, such as the Inorganic Crystal Structure Database (ICSD), NIST Phase Equilibria Diagrams Database (PED), etc.

ONTORULE steel ontology[13] is developed by European Union which is focus on coils, defects, phenomena etc. It is developed for the steel industry which aims to build the conceptual model with the steel use case. The ONTORULE ontology is implemented in OWL.

FreeClassOWL[14] is developed for European construction and building materials market, which allows for the fine-grained descriptions and search for products, suppliers, and warehouses for any building-related sourcing needs. Based on FreeClassOWL, Eurobau Utility ontology and the BauDataWeb RDF dataset, BauDataWeb[15] has become one of the largest and richest public datasets for a well-defined vertical sector that is available on the Semantic Web.

Premkumar et al[16] developed a novel Semantic Laminated Composites Knowledge Management System (SLACKS) that reuses part of the structure of Ashino ontology and MatOnto ontology. SLACKS ontology is developed for the engineering of laminated composites structures which integrates relevant domains of the product life cycle, such as design, analysis, manufacturing and materials selection through the engineering case study of a wind turbine blade. Using SLACKS ontology, it reveals a usable product life cycle knowledge tool that can facilitate efficient knowledge creation, retrieval and reuse from product design to manufacturing.

MatML is an extensible markup language (XML) developed especially to facilitate the materials information exchange, which can uniformly represent materials property data to resolve syntactic and structural heterogeneity.[17,18] Although, MatML is simple, flexible and understandable, Ashino and Oka[19] have shown that MatML is not adequate for data exchange between heterogeneous materials database and proposed a ontology framework to define the structure of domain concepts. Zhang et al[20] proposed an approach to transform MatML-based materials data into an OWL ontology (named MatOWL). Using MatOWL, materials data can then be explored in a more semantic way. Furthermore, MatOWL can be mapped to other ontologies with logic rules to provide more semantic context for domain experts. Using MatOWL more materials knowledge can be obtained by reasoning on the OWL ontology.

III. THE FRAMEWORK OF THE WHOLE SYSTEM

Supposing we have two heterogeneous databases, the idea of our method is to convert one basic materials database to a materials ontology and then integrate the other. After that the data in each database will be integrated into the ontology. End users can do semantic query on the integrated ontology. Figure 1 shows the whole procedure.

Although there exists several materials ontologies as mentioned in related work, most of ontologies tend to represent one sort of material, or special fields and applications. Recently, ontologies are often built manually, sometimes it is complicated and time consuming that needs domain experts to participate in. So, we adopt a semi-automatic method to build a material ontology according to the structure of a comprehensive materials database. First of all, we extract the relational model from database by DBC API. Then we generate a material ontology according to the relational model and some conversion rules. The tuples from database can also be converted to the ontology individuals according to the conversion rules. And then we build an algebraic model for the materials ontology and the other materials database. After that we get the relation between them, which can be used to convert the data from the database into the individuals of ontology. Through these steps, the heterogeneous databases can be integrated together. Moreover, when using relation algebra to integrate the

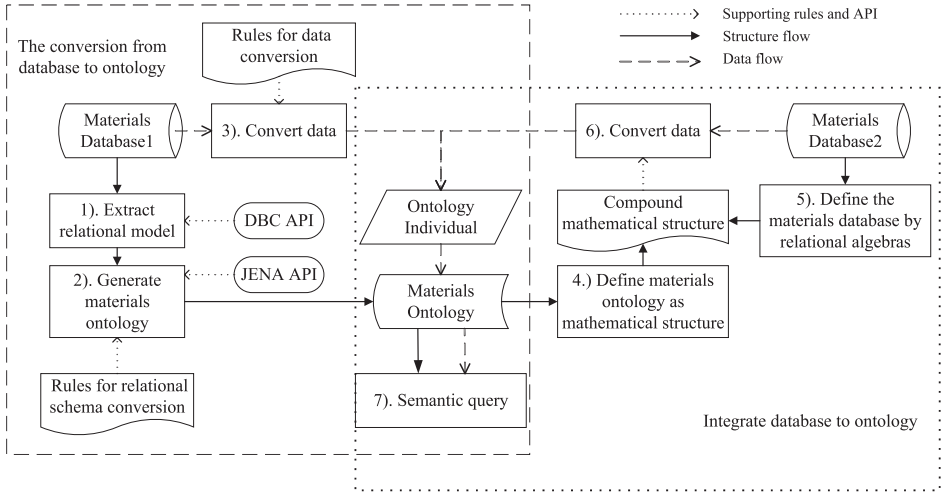

FIG. 1. System architecture of the ontology based heterogeneous databases integration.

ontology and the database, ontology individuals are used as the data carrier which is more suitable to SPARQL. And we adopt ontology rules to make the query results more accurate.

IV. MAPPING FROM RELATIONAL DATABASE TO ONTOLOGY

Currently, building ontology according to the relational database can be divided into three categories, which are manual, semi-automatic and automatic. Manual, for instance[21] is usually for a particular field. During the manual ontology building, some hidden mapping relations could be found, but it is time-consuming and difficult for normal researchers to build the ontology manually. Semi-automatic, such as[22] is usually realized by interacting with the domain experts. During the procedure of building ontology, end users can participate in the verification and modification of the mapping results. Methods for automatically building ontology is rarely used because of the low accuracy. Therefore, in this paper, we also use the semi-automatic approach to build the ontology.

About the ontology representation, Web Ontology Language (OWL) is the latest standard recommended by W3C.[23] It is a vocabulary extension of Resource Description Framework (RDF). And OWL facilitate greater machine interpretability of web content than that of XML, RDF and RDFS. So in this paper we choose OWL as the ontology description language.

A. Materials science tetrahedron for root concepts

In materials science, we mainly focus on the study of the structure, performance, processing and properties, which is called materials science tetrahedron[24] as shown in Figure 2.

In Figure 2, the structure of materials include bonding structure, crystal structure and organization structure. The bonding structure includes chemical bonds (ionic bonds, covalent bonds, metal bonds) and physical bonds (hydrogen bonds, molecular bonds). The crystal structure of the material includes crystal, non-crystal and quasi crystal. The organization structure refers to the characteristics which represented by the different components of materials. The properties of materials are the response

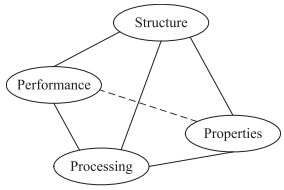

FIG. 2. Materials science tetrahedron.

of a material to electrical, magnetic, optical, thermal and mechanical loads, that include mechanical properties, physical properties, chemical properties, etc. Processing means all unit operations, milling, blending, tableting and relevant processing parameters. Performance is a kind of characterization parameters of a material under certain conditions, in order to describe the act or result of the material. It usually includes manufacturability, content uniformity, etc.

Therefore, for materials ontology building, we create five owl:Class for *material, structure, properties, processing* and *performance*. And most of the owl:Class converted from database will be the subclass of them.

B. Conversion from relational database to ontology

Our approach is to classify the relation, and then formalize the corresponding conversion rules. Using database commander (DBC), we can utilize database relational model and convert it according to the predefined conversion rules. And then output OWL based ontology. Figure 3 shows the whole conversion process.

Definition 1: **Relational Database**. The relational database is a 6-tuple model: $R^d = \{U, D, DOM, F, PK, FK\}$, where R^d is the name of relation; U is the set of attribute names that come from the relation; D is the domain that the attributes in U come from; DOM is the mapping set from attributes to domains, in which we use DOM(U_i) to specify the type, range, length, etc. of U_i; F is the set of the data dependencies among attributes; PK(R^d) is the set of the primary keys of R^d and FK(R^d) is the set of the foreign keys of R^d.

Definition 2: **Ontology**. We can describe an ontology by a number of sets of concepts, relations, lexical entries, and links between these entities. The definition of the ontology is 5-tuple model:[25] $O = \{C, H^c, R, rel, A^o\}$, where O is the ontology name. C is the set of concepts. H^c is a taxonomy of concepts with multiple inheritance. For example, $H^c(C_1, C_2)$ notes that C_1 is the subconcept of C_2. R is a set of non-taxonomic relations described by their domain and rang restrictions. *rel(R)* describes a heterarchy of relations. For example, $rel(R) = (C_1, C_2)$ specifies that there is a relation R between C_1 and C_2. A^o is the set of axioms.

Definition 3: **Keywords Set**. K is the set of keywords in materials science, including different kinds of materials name such as performance, properties, structures and processing, for example, tensile strength, density, cold forming, porous, etc. The keywords set can be added as needed.

Next, we will divide different types of relations and discuss how to define different rules for mapping the relational database to ontology automatically.

Type (a): For the database relation, it has primary keys but does not have foreign keys. That is $|PK(R_i^d)| \geq 1$ and $|FK(R_i^d)| = 0$. This kind of relation is the basic entity. We have 4 rules for this kind of mapping.

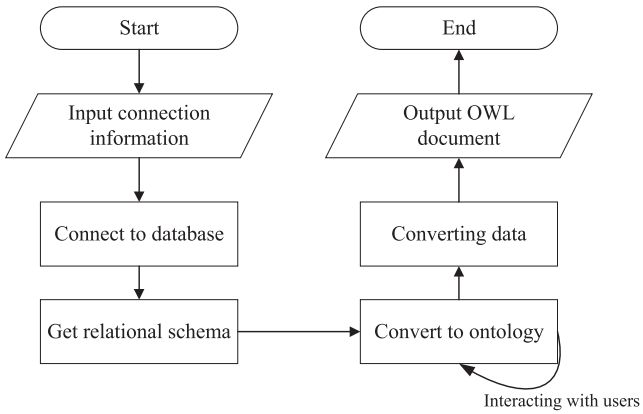

FIG. 3. Conversion flow chart from relational database to OWL based ontology.

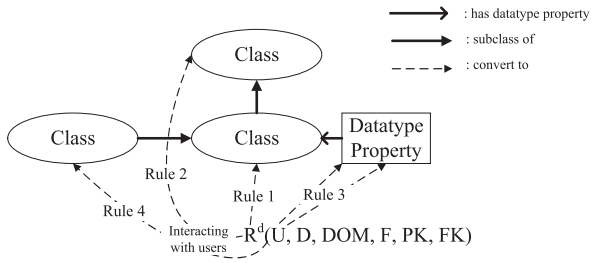

FIG. 4. Conversion rules for type (a).

Rule 1: Convert each database relation name to a concept. That is $R_i^d \rightarrow C_i$. During the actual conversion process, C_i is a owl:Class.

Rule 2: If a database relation name is a materials keyword, then we can set the corresponding concept taxonomy. That is to say, if $R_j^d \in K$, then we can get $H^c(C_i, C_{material})$ or $H^c(C_i, C_{properties})$ or $H^c(C_i, C_{performance})$ or $H^c(C_i, C_{structure})$ or $H^c(C_i, C_{processing})$. In the actual conversion process, we set C_i owl:subclass-of $C_{property}$ or $C_{material}$ etc. For example, if the database relation name is yield strength(a kind of mechanical performance), we set C_i owl:subclass-of $C_{performance}$ to enrich the ontology.

Rule 3: Convert each attribute to a concept and set the corresponding non-taxonomy concepts relation. That is, for each $U_j \in U$, we have $U_j \rightarrow C_j$ and $rel(R_j) = (C_i, C_j)$. During the actual conversion, we create a owl:DatatypeProperty for U_j and set its rdfs:domain = C_i, $DOM(U_j)$ →rdfs:range.

Rule 4: If an attribute name is a materials keyword, we convert it to a concept and set the corresponding concept taxonomy. That is, if $U_j \in K$ we have $U_j \rightarrow C_j$ and $H^c(C_j, C_i)$. During the actual conversion, C_j is a owl:Class. We set C_j owl:subclass-of C_i and C_j owl:subclass-of $C_{propertie}$ or $C_{material}$, etc. according to the keyword at the same time.

Figure 4 is the visual conversion rules for type (a). During the processing of *Rule 2* and *Rule 4*, setting subclass should be supervised by the domain experts. Figure 5 is an example for type (a). Metallic is the database relation name and it's a materials keyword, so that we convert it to an owl:class and set it to a subclass of Materials. Tensile Strength, Grade, Formula, Mass and Name are the attributes of Metallic materials. The Tensile Strength is a materials keyword, we convert it to owl:class and set it to a subclass of Metallic materials. For the rest of attributes, each of them will be converted to an owl:DatatypeProperty and set its domain to Metallic materials and set its range according its DOM.

Type (b): For the database relation which only has one primary key and one foreign key, moreover the primary key is the same to the foreign key. That is, $|PK(R_i^d)| = 1$, $|FK(R_i^d)| = 1$ and

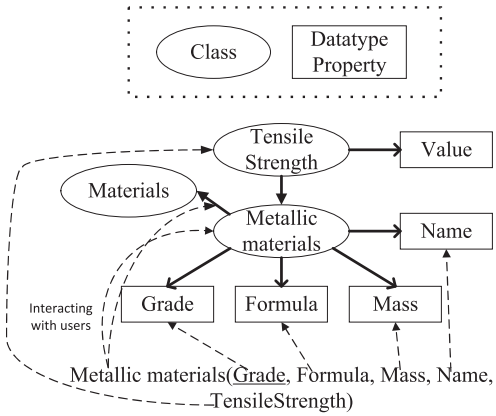

FIG. 5. A conversion example for type (a).

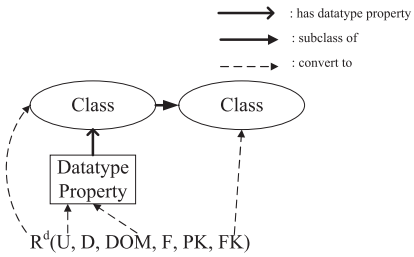

FIG. 6. Conversion rules for type (b).

$PK(R_i^d) = FK(R_i^d)$. This kind of relation is usually caused by the inheritance among entities. In this condition, we use the following rule for mapping.

Rule 5: Convert the database relation to a concept and set the corresponding concept taxonomy according to its foreign key. That is, $R_i^d \rightarrow C_i$ and set $H^c(C_i, C_j)$. During the actual conversion process, C_i is an owl:Class and we set C_i owl:subclass-of C_j.

Figure 6 shows the conversion rules for type (b) and Figure 7 is a mapping example for type (b). The relation has one foreign key. High Strength Steel is the database relation name and Grade is the foreign key which related to Metallic materials. We convert the High Strength Steel to an owl:class and set it as a subclass of Metallic materials.

Type (c): For the database relation which has two attributes, two primary keys and two foreign keys. And each of the foreign key is a primary key for another database. That is, $|PK(R_i^d)| = |U| = 2$ and for $FK(R_i^d) = \{fk_1, fk_2\}$ we have $\{fk_1\} = PK(R_j^d)$ and $\{fk_2\} = PK(R_k^d)$. This kind of relation is usually caused by the many-to-many relationship among entities. In this case, we use the following rules for mapping.

Rule 6: According to the concepts corresponding to the foreign keys, create two relations to specify the many-to-many relationship between two concepts. That is $FK(R_i^d) \rightarrow \{rel(R_1) = (C_j, C_k)$ and $rel(R_2) = (C_k, C_j)\}$. During the actual conversion process, we convert R_1 and R_2 to owl:ObjectProperty. And R_1's rdfs:domain = C_j, R_1's rdfs:range = C_k and R_2's rdfs:domain = C_k, R_2's rdfs:range = C_j. And we set R_1 owl:inverse-of R_2.

Figure 8 is the conversion rules for type (c) and Figure 9 is a mapping example of type (c). Structures_id is the foreign key related to Structures and Element_id is the foreign key related to Element. We need to build the relation between Structures and Element so that we convert two attributes to two owl:ObjectProperties. Set the owl:ObecjtProperty Structures-Element's domain to Structures and range to Elements. Similarly, Element-Structures should be handled in the same way.

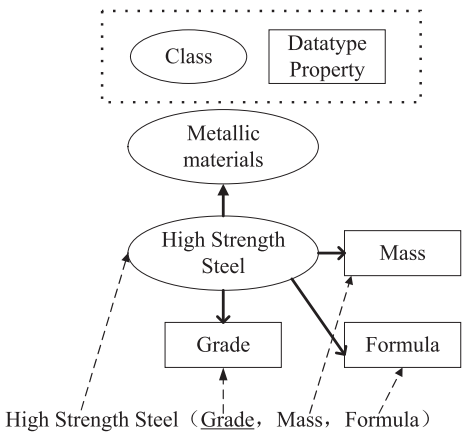

FIG. 7. A conversion example for type (b).

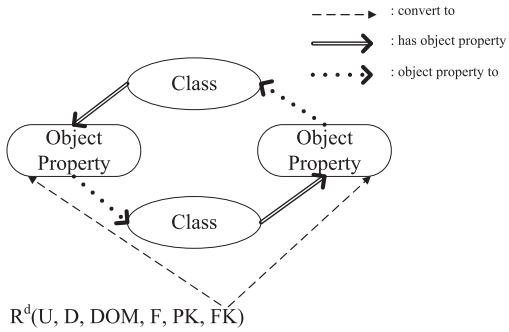

FIG. 8. Conversion rules for type (c).

Type (d): For the database relation which primary key is not empty and only has one foreign key. That is, $|PK(R_i^d)| \neq 0 \cap |FK(R_i^d)| = 1$. This kind of relation is usually caused by the one-to-one or one-to-many relationship between two entities.

Type (e): For the database relation which primary key is not empty and has more than two foreign keys. That is, $|PK(R_i^d)| \neq 0 \cap |FK(R_i^d)| \geqslant 2$. This kind of relation is usually caused by the multiple relations among entities.

The database relation of type (d) and (e) should be convert to a owl:Class. So that can use Rules 1~4 of type (a). Besides Rule 7 can be used to specify the foreign key relation, cardinality restrictions in OWL can specify the one-to-one and one-to-many relationships. Therefore we use the following 2 rules:

Rule 7: According to each concept corresponding to the foreign keys, create a relation to specify the relationship between two concepts. That is, for each $fk_m \in FK(R_i^d)$ there exists $\{fk_m\} = PK(R_j^d)$ and we have $fk_m \rightarrow rel(R_m) = (C_i, C_j)$. During the actual conversion process, we convert R_m to a owl:ObjectProperty. R_m's rdfs:domain = C_i, R_m's rdfs:range = C_j. If fk_m can not be empty we set owl:minCardinality = 1, otherwise set owl:minCardinality = 0.

Figure 10 is the conversion rules for type (d) and (e). Figure 11 is an example for type (d) and (e), where Materials has two foreign keys Performance_id and Structures_id. We convert both of them to owl:ObjectProperty whose domain are Materials, and range are Performance and Structures.

And there are cardinality restrictions for foreign keys on the basis of whether they are empty or not. We can use Rule 8 and Rule 9 for conversion.

Rule 8: If the attribute can not be empty we should set its cardinality restriction. That is if $U_j \in U \neq null$ we have $rel(R_j) = (C_i, C_j)$. During the actual conversion process, we convert R_j to a owl:DatatypeProperty and set its owl:Cardinality = 1.

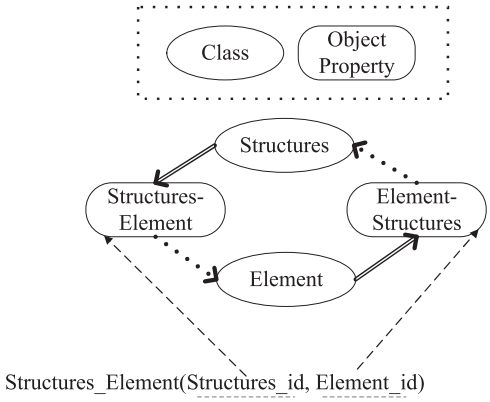

FIG. 9. A conversion example for type (c).

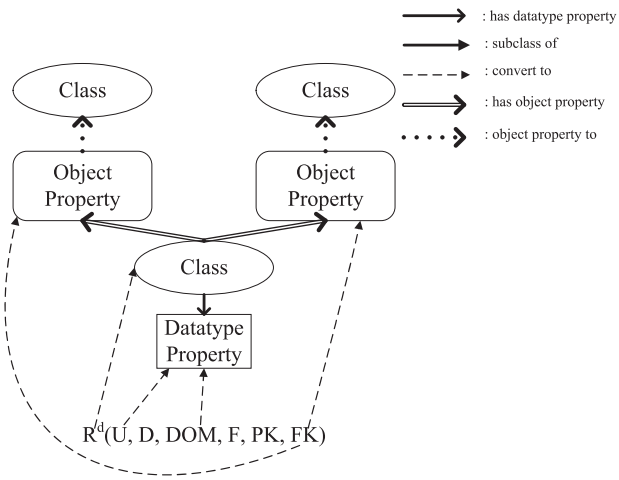

FIG. 10. Conversion rules for type (d) and (e).

Rule 9: If the attribute can be empty we should set its cardinality restriction. That is if $U_j \in U$ we have $rel(R_j) = (C_i, C_j)$. During the actual conversion process, we convert R_j to a owl:DatatypeProperty and set its owl:maxCardinality = 1.

C. The data conversion

In order to integrate heterogeneous databases, we should convert the data from database to the ontology's individuals. After the conversion according to the rules as mentioned above, we can convert the data easily. Suppose that $I(C_i)$ is the individual of C_i and $t(R_i^d)$ is the data tuples of R_i^d.

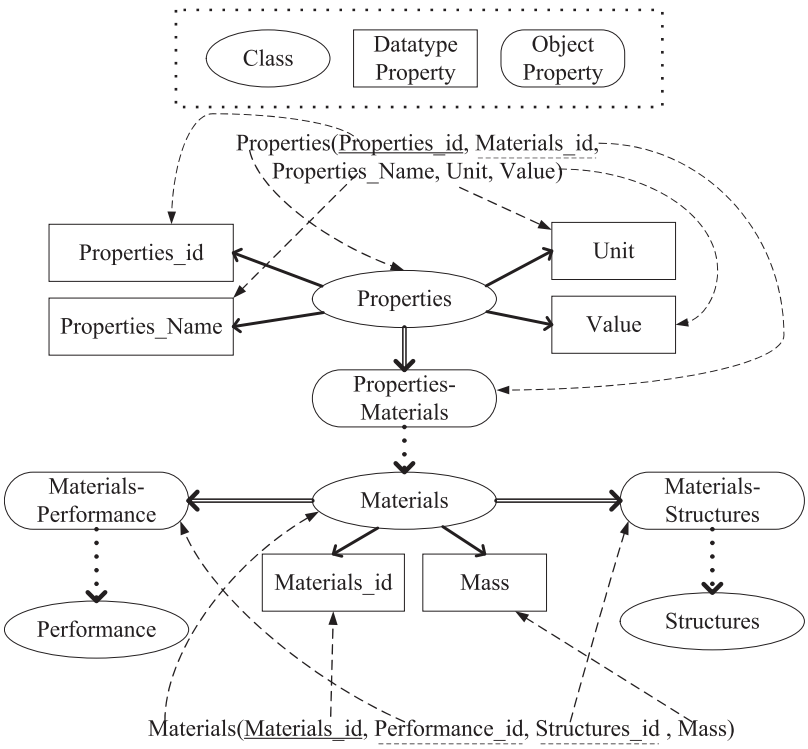

FIG. 11. A conversion example for type (d) and (e).

Rule 10: Convert each tuple in database to a individual and give a unique identifier. That is, for each $t_j \in t(R_i^d) \rightarrow I_j(C_i)$. During the actual conversion process, we use the database relation name and the primary key name to be the unique identifier. And we convert the data in tuple to the statement and make it connect to the corresponding owl:DataProperty. Then we convert the foreign keys to owl:ObjectProperty.

So far, using the above conversion rules, we have obtained a materials ontology where all the data of one materials database have been stored. Next, we should consider how to integrate other database to the created ontology easily.

V. INTEGRATE OTHER DATABASES TO ONTOLOGY

In order to integrate other databases to ontology, we build a mathematical structure for the materials ontology and the other materials database.

A. Define the ontology as a mathematical structure

In ontology, "C" is the set of the concepts. "$\times$" is a binary operation on C that represents the combination of two concepts. It is obvious that the order of the combination of two concepts doesn't matter. So the concept $c = c_1 \times c_2$ and $c' = c_2 \times c_1$ are the same. That means "$\times$" is commutative. In addition, "$\times$" should be idempotent.

So, let $< C, \times, e_c >$ be a commutative idempotent monoid of concepts. e_c is a kind of pseudo-concept which is neutral to the concepts. That is to say, for $\forall C_i \in C$ we have $C_i \times e_c = e_c \times C_i = C_i$.

It is obvious that the combination operator "$\times$" satisfies the associative law. However, it is worth mentioning that, in some cases, there are some no real meaning concepts coming from the combination of concepts in C. And those concepts just satisfy the closure property of the monoid.

Definition 4: **Part-of-relation**. For every $c_1, c_2 \in C$, if c_1 is part-of c_2, we denote $c_1 \sqsubseteq c_2 \Leftrightarrow \exists (c|c \in C : c_1 \times c = c_2)$.

The part-of-relation is a partial order. It satisfies the three axioms of posets. Since the mereological relationships is a simplification of a partial order,[26] we can use the poset properties to build the structure of the ontology and link it to the material database.

Starting from the concept of the main domain, we define L is a subset of C which contains the main concept of C and all its parts until the atoms. The part-of-relation forms a boolean lattice of concepts,[27] that is $\mathscr{L} = (L, \sqsubseteq)$. The pseudo-concept concept e_c is also included in L. So that two concepts do have one concept e_c even if they are structurally unrelated.

There may be some concepts in L associated with other concepts in C by one or more relationships. The concept in L we call it the ancestor element of the relationship in that case. We consider that only subsets of concepts can be connected to the lattice. The relationships can be represented as rooted graphs and the root of the rooted graph is always part of the lattice.

$\mathscr{G} = \{G_i\}_{i \in L}$ is a family of rooted graphs. Each G_i we associate a relation R_i which should be connected to the top element. For $a \in C$, R_a defined as $\{(x, x)|x = a\}$.

Definition 5: **Rooted Graph**. For $C_i \in C$ and R_i is a relation on C_i. Rooted graph $G_i = (C_i, R_i)$, *iff* $\exists!(t_i|t_i \in C_i : R_{t_i} \subset R_i^*)$, where "*" is the power operation. As,[28] we denote a rooted graph as $G_i = (C_i, R_i, t_i)$, and t_i is the root of G_i.

Thus, we can define the ontology as a mathematical structure:

Definition 6: **Mathematical Structure of Ontology**. Supposing C is the set of concepts. $\mathscr{L} = (L, \sqsubseteq)$ is a Boolean lattice. $\mathscr{G} = \{G_i\}_{i \in L}$ is a family of rooted graphs. So, we have the mathematical structure of the ontology $\mathscr{O} = (C, \mathscr{L}, \mathscr{G})$.

The definition specifies that each relation has a ancestor element which is the root of the rooted graph. For each root in $\mathscr{G}$ ends up in $\mathscr{L}$. The definition also ensures that all the concepts in relation can connect to the lattice.

Algorithm 1. The integration procedure of two heterogeneous databases.

1: Take all main concepts and its parts and subparts until the atoms from C to build the boolean lattice with part-of-relation.
2: Supposing there exists a set of concepts $V \subseteq (C - L)$, where each concept $V_i \in V$ has $rel(R_i) = (L_j, V_i)$ and $L_j \in L$. For every $L_j \in L$, use L_j and the corresponding V_i to build the rooted graph.
3: Use relational algebra $\mathscr{A} = (A, +, \cdot, -)$ to demonstrate the structure of materials database.
4: For each entity $E_1, E_2, \cdots, E_i \in E$ (E is also an entity), we have $\tau'(E_1) = C_1, \tau'(E_2) = C_2, \cdots, \tau'(E_i) = C_i$ to associate each entity to the concept it instantiates.
5: The type of each entity E_i, $\tau(E_i) = \rho(\tau'(E_i)) = \rho(C_i)$. Then can get the type of entity E, $\tau(E) = \tau(E_1 \cdot E_2 \cdot \cdots \cdot E_i) = \tau(E_1) \times \tau(E_2) \times \cdots \times \tau(E_i)$.
6: Convert the data from the database to the individuals according to the type of it.

B. Relational database structure

To model the materials database, we can use the relational algebra.

Definition 7: **Relational Algebra**. The simplified relational algebra $\mathscr{A}$ is:

$$\mathscr{A} = (A, +, \cdot, -)$$

Where, A is the set of relations. "+" and "·" are the binary operations on A which means the intersection and the union operations for the relations. "−" is the unary operations on A means the complement of the relation.

Considering that $\mathscr{U}$ is the set of attributes $\mathscr{U}_1, \mathscr{U}_2, \cdots, \mathscr{U}_n$. $J \subseteq \mathscr{U}$ is a set of attributes called type. For each $\mathscr{U}_i \in \mathscr{U}$, there is a attribute domain $D(A_i)$ for A_i which can not be empty. We call a relation of type J is a set of tuples A and the element $A_i \in A$ is called a tuple. For a tuple $A_i \in A$, we have $\tau(A_i) = \tau(A) = J$. τ is a operator can get the type of A. In relational database, J is represented as a table with its columns representing each attribute in J. And the tuples represent as rows in a table.

Herein, we can use the relational algebra $\mathscr{A} = (A, +, \cdot, -)$ to describe the other materials database. And then we can connect two structures together. In order to do so, we define the operator τ': $A \to C$ to connect the entities to the corresponding concept in the rooted graph. And for rooted graph $G_i = (C_i, R_i, t_i)$, we have a operator $\rho_i : C_i \to L$, $\rho_i(c) = t_i$, where $c \in C_i$. Finally we define the type operator $\tau : A \to L$ as $\tau = \rho \circ \tau'$.

Definition 8: **Type of Entity Combination**. The type of the entities combination is the combination of the type of each entity, that is $\tau(a \cdot b) = \tau(a) \times \tau(b)$.

Definition 9: **Mathematical Structure of Heterogeneous Materials Databases**. Supposing $\mathscr{O} = (C, \mathscr{L}, \mathscr{G})$ is the structure of ontology, $\mathscr{A} = (A, +, \cdot, -)$ is the relational algebra and $\tau = \rho \circ \tau'$ is the type operator. And then the mathematical structure of heterogeneous materials databases is $\mathscr{S} = (\mathscr{O}, \mathscr{A}, \tau)$.

The integration procedure of two heterogeneous databases is as Algorithm 1.

VI. EXPERIMENTS

We use OQMD[29,30] and Materials Project[31] as the experimental databases which are two famous First Principle Computational Databases. The E-R models of the two databases are as shown in Figures 12 and 13. Using mapping rules mentioned in section IV, we convert the OQMD database to an ontology. In order to avoid the identifier duplication, we use *TableName-ColumnName* to describe the owl:DatatypeProperty. For example, for the table Elements, which is a table belongs to type (e). We build an owl:class Elements. And all the attributes of table Elements are converted to the owl:DatatypeProperty which domain are Elements. Then we covert all the foreign keys to the owl:ObjectProperties which domain are Elements, and range are Atoms, Compositions and Structures. After tuning we get the converted ontology as shown in Figure 14.

Then we build the mathematical model for the ontology and the other materials database Materials Project, as mentioned in Section V. Figure 15 shows the main part of the integrated model. It shows

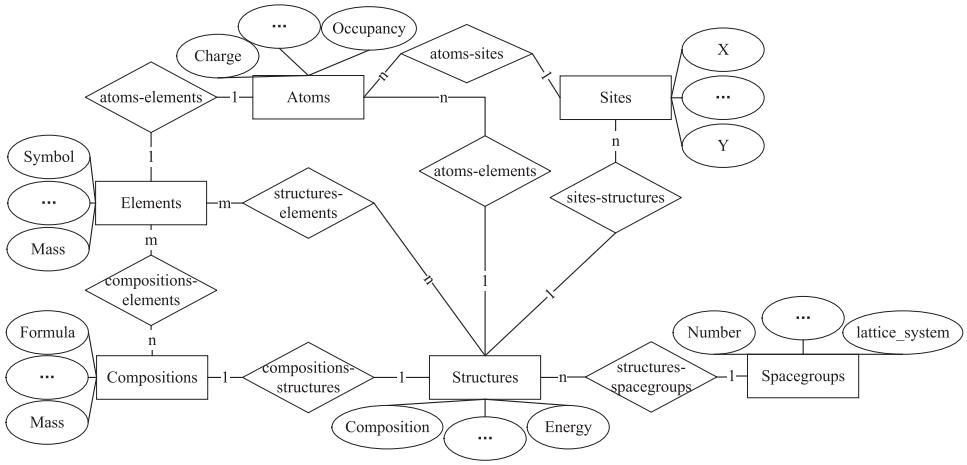

FIG. 12. E-R model of OQMD database.

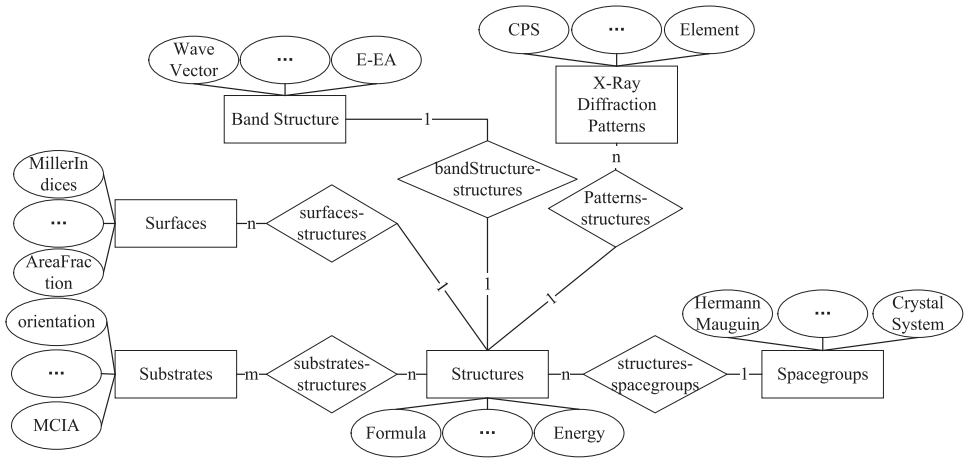

FIG. 13. E-R model of Materials Project.

that the concepts, such as *materials*, *structure*, *elements*, *spaceGroup*, e_c, etc. form the lattice of concepts. Concepts *hall, latticesystem, symbol, point_group*, etc. are part of the rooted graph. And *(-P 4 2 3), cubic*, $P6_3/mmc$, etc. are entities.

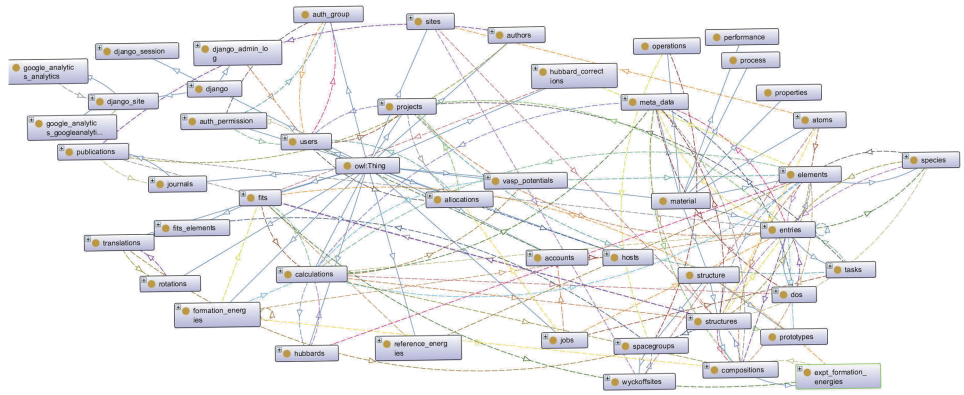

FIG. 14. The ontology structure of OQMD database.

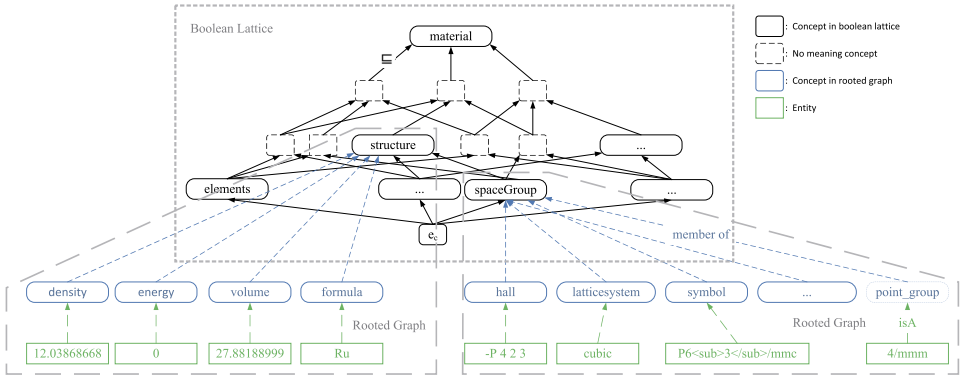

FIG. 15. A part view of the integrated model.

Through the matching of synonyms, homoionym and some manual correction we could map the concepts from ontology and the data from database. However, in the actual processing, one database is often difficult to cover another one. So new concepts should be created if we can not find the corresponding concepts for data.

For example, a tuple a(12.03868668, 0, 27.88188999, Ru, -P 4 2 3, cubic, $P6_{3/mmc}$, 4/mmm), we get the type for each attribute, such as $\tau'(-P\ 4\ 2\ 3)$ = hall. And in the rooted graph, we have $\rho(hall)$ = spacegroup. Thus we can get the type of $(-\ P\ 4\ 2\ 3)$ as Equation (1), and all the types of the rest attributes can be obtained in the similar way.

$$\tau(-P\ 4\ 2\ 3) = \tau'(-P\ 4\ 2\ 3) \circ \rho(hall) = spacegroup \tag{1}$$

Through definition 8, we get $\tau(a) = \tau(12.03868668) \times \tau(27.88188999) \times \cdots \times \tau(4/mmm)$. And we get $\tau(a) = structure$. Then we can convert the data from the database to the individuals according to the type of tuple a. Thus the data of heterogeneous materials database are integrated together.

Once the integrated ontology created, semantic query is allowed on the ontology which is integrated from two materials databases, OQMD and Materials Project. We can construct SPARQL to extract the information from the integrated ontology. For example, if we want to query all the *structures* that *latticesystem* equals *cubic*, construct a semantic query with SPARQL as SPARQL Query Example 1.

SPARQL Query Example 1

```
1: PREFIX this: <http://shu.edu.cn/material/ontology#>
2:      SELECT ?structure ?volume ?spacegroups WHERE{
3:           ?structure this:structures-spacegroups ?spacegroups.
4:                ?spacegroups this:spacegroups-lattice_system ?lattice.
5:                     ?structure this:structures-volume ?volume.
6:                          Filter regex(?lattice,'Cubic','i')
7:      }
```

We show part of the results in Table I. During implementation, JENA API[32] is used to execute the SPARQL query. All the results return with *ResultSet* format which can be operated easily. We can see that both of the data in two databases can be retrieved together that do not need to construct different SQL statement for each relational database. It is also worth mentioning that the query is executed on the ontology not the relational database, so the complicated and time-consuming union or join operations are avoided when the query involves multiple tables.

TABLE I. Part of semantic query results from integrated ontology.

Structure	Volume	Spacegroups	Datasource
structures-pri-243	25.77007415	spacegroups-pri-243	Material Project
structures-pri-267	265.8010647	spacegroups-pri-267	Material Project
structures-34112	10.8604	spacegroups-229	OQMD
structures-34224	20.8806	spacegroups-216	OQMD
structures-pri-496	367.4211175	spacegroups-pri-496	Material Project
structures-34611	77.4177	spacegroups-221	OQMD
structures-pri-167	69.53995864	spacegroups-pri-167	Material Project
structures-34448	33.1271	spacegroups-229	OQMD

Furthermore, we can create some additional rules for the materials ontology to refine query results. For example, when a relational database has some common data and the other does not, then we can create some rules to add a external relation for that part of unlinked data. For example, if we want to extract all the individuals with *structures* that has a relation with element "C". We may construct the SPARQL query as SPARQL Query Example 2.

SPARQL Query Example 2

```
1: PREFIX this: <http://shu.edu.cn/material/ontology#>
2: PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
3:      SELECT ?structures ?volume ?spacegroups WHERE{
4:           ?structure this:structures-elements ?element.
5:                ?element this:elements-symbol ?symbol.
6:                     ?structure this:structures-volume ?volume.
7:                          ?structure this:structures-spacegroups ?spacegroups.
8:                               Filter regex(?symbol, '∧C$')
9:      }
```

Although the individuals can be queried from two materials databases. However, as shown in Figure 16, the relationship between *structures* and *elements* only exist in OQMD database. So, the relationship between *structure* data and *element* data in OQMD database is well organized. But the structure data in Materials Project have no connections with the element data. When execute the query 3, the results can only be responded from the OQMD database. Table II shows the results before adding rules. We can see that only those data from OQMD database can be retrieved. To get the better result we can add a external rule as Ontology rule example 1.

Ontology rule example 1

```
1: [rule:(?structure this:structures-composition ?composition)
2:      (?element this:elements-symbol ?symbol)regex(?composition, ?symbol)
3:           - >(?structure this:structures-elements ?element)].
```

The rule means that if the *composition* of *structures* contains a certain element we connect the *structures* and the corresponding *element* together by owl:ObjectProperty. Thus, when a end user searches *structures* containing some *elements* in Materials Project can also be responded which can not be done earlier. Table III shows the searching results after adding rules. From Table III, we can see that two results come from OQMD and four from Materials Project.

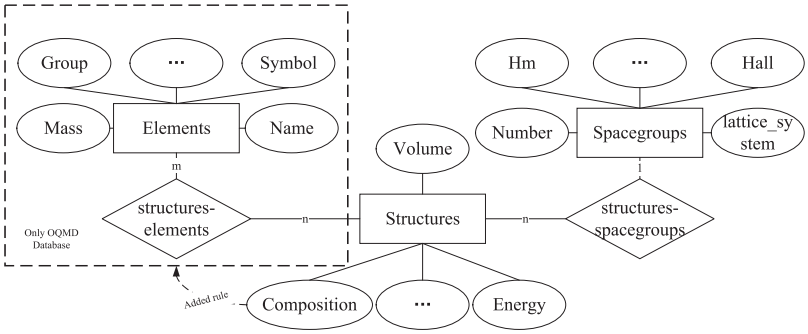

FIG. 16. A partial view of the E-R model.

TABLE II. Query results before adding rules.

Structure	Volume	Spacegroups	Datasource
structures-34224	20.8806	spacegroups-216	OQMD
structures-34170	16.624	spacegroups-225	OQMD

TABLE III. Query results after adding rules.

Structure	Volume	Spacegroups	Datasource
structures-34224	20.8806	spacegroups-216	OQMD
structures-34170	16.624	spacegroups-225	OQMD
structures-pri-42	41.13742744	spacegroups-pri-42	Material Project
structures-pri-21	44.91792373	spacegroups-pri-21	Material Project
structures-pri-55	11.41878254	spacegroups-pri-55	Material Project
structures-pri-149	21.21334856	spacegroups-pri-149	Material Project
structures-pri-41	22.87020916	spacegroups-pri-41	Material Project

Ontologies creation is milliseconds which can be ignored. Most of the time spent in the method is the conversion of individuals and the relationship between individuals. Figure 17 and Figure 18 show when the amount of data are millions, the conversion of individuals and the relationship costs several minutes. And the single query costs half seconds.

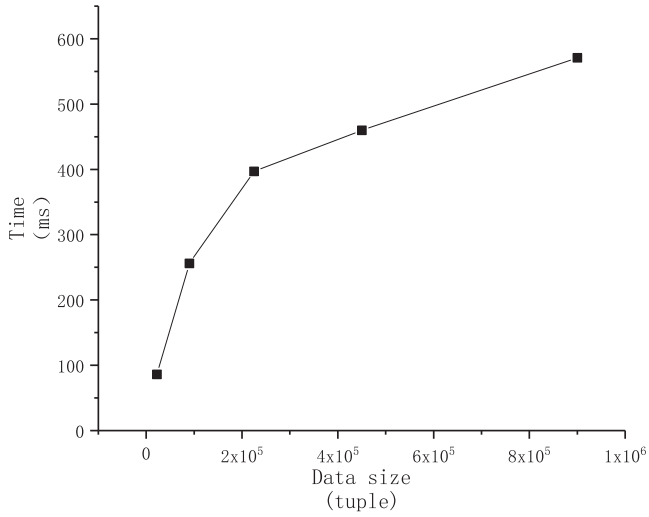

FIG. 17. Query efficiency.

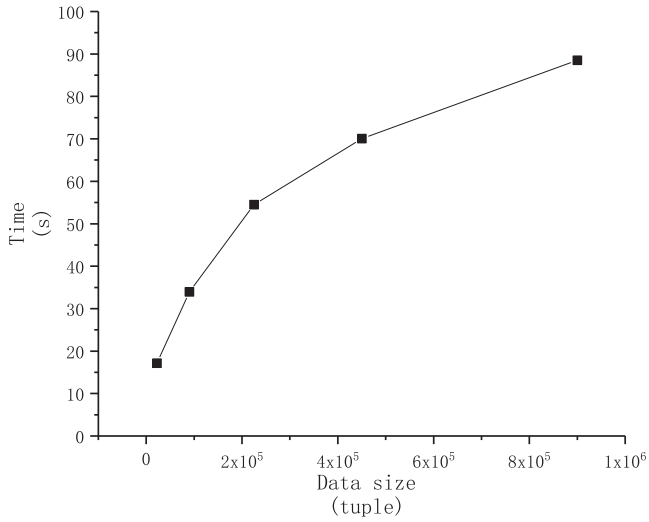

FIG. 18. Ontology individuals conversion efficiency.

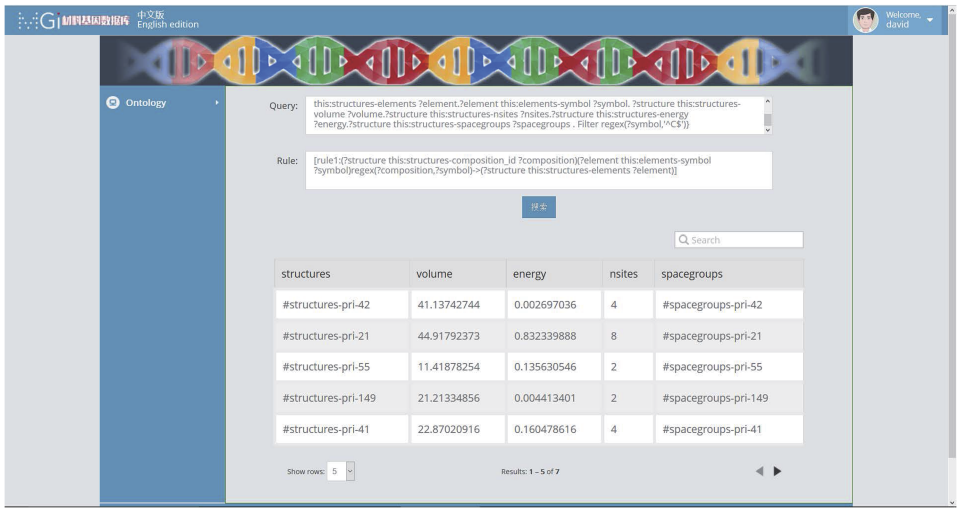

FIG. 19. A prototype system of ontology based data integration and semantic query.

Currently, we have implemented a prototype system deployed in https://matdata.shu.edu.cn. And the interface for defining rules and semantic query looks like Figure 19.

VII. CONCLUSIONS AND FUTURE WORK

With the fast development of materials science, materials big data, especially come from high throughput experiments and computations, increase rapidly. However, different databases has their own schemas and structures which bring great challenges for data sharing and integration. The main work of the paper are as follows:

- Presents a set of conversion rules to transform the relational materials database to ontology, that is general can be used in other areas.
- Builds up a mathematical model for the materials ontology and the heterogeneous materials database, which allows to map the data in database to the individuals of ontology. So as to integrate the heterogeneous databases.

Considering the future work, several directions can be done further. Firstly, during ontology creation, reduce the manual interventions as little as possible without affecting accuracy. Secondly, separate the data and ontology physically to improve the query performance further. Finally, visualize the SPARQL construction, makes it easier for normal non-professional users to use friendly and conveniently.

ACKNOWLEDGMENTS

This work is partially sponsored by National Key Research and Development Program of China(2016YFB0700504, 2017YFB0701601), Shanghai Municipal Science and Technology Commission(15DZ2260301), Natural Science Foundation of Shanghai(16ZR1411200). The authors gratefully appreciate the anonymous reviewers for their valuable comments.

[1] K. Rajan, "Materials informatics: The materials 'gene' and big data," Annual Review of Materials Research **45**(1), 153–169 (2015).
[2] A. Agrawal and A. Choudhary, "Perspective: Materials informatics and big data: Realization of the 'fourth paradigm' of science in materials science," APL Materials **4**(5), 1–17 (2016).
[3] X. Li, D. Zhang, Z. Liu, Z. Li, C. Du, and C. Dong, "Materials science: Share corrosion data," Nature **527**(7579), 441–442 (2015).
[4] A. Ranganathan and Z. Liu, "Information retrieval from relational databases using semantic queries," in *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, Arlington, Virginia, USA, November 2006, pp. 820–821.
[5] P. Bonatti, Y. Deng, and V. S. Subrahmanian, "An ontology-extended relational algebra," in *2003 IEEE International Conference on Information Reuse and Integration (IRI2003)*, Las Vegas, NV, USA, October 2003, pp. 192–199.
[6] A. Ranganathan and Z. Liu, "Information retrieval from relational databases using semantic queries," in *Proceedings of the 15th ACM International Conference on Information and Knowledge Management*, Arlington, Virginia, USA, November 2006, pp. 820–821.
[7] L. Feigenbaum and E. Prud'hommeaux, "SPARQL by example: A tutorial," Cambridge Semantics, 2013, http://www.cambridgesemantics.com/semantic-university/sparql-by-example.
[8] K. Rajan, Ed., *Informatics for Materials Science and Engineering: Data-Driven Discovery for Accelerated Experimentation and Application*. Elsevier Inc., 2013.
[9] P. E. van der Vet, P.-H. Speel, and N. J. Mars, "The plinius ontology of ceramic materials," in *Proceedings of 11th European Conference on Artificial Intelligence(ECAI'94)*, Amsterdan, Netherlands, August 1994, pp. 187–205.
[10] T. Ashino, "Materials ontology: An infrastructure for exchanging materials information and knowledge," Data Science Journal **9**, 54–61 (2010).
[11] K. Cheung, J. Hunter, and J. Drennan, "MatSeek: An ontology-based federated search interface for materials scientists," IEEE Intelligent Systems **24**(1), 47–56 (2009).
[12] "MatOnto user manual(version 1.7)," Website, 2017, http://docs.matonto.org/.
[13] E. Rubiera and C. de Sainte Marie, "ONTORULE: From business knowledge to ontology- and rules-based applications," in *Proceedings of the 2011 Conference of the Center for Advanced Studies on Collaborative Research*, Toronto, Ontario, Canada, November 2011, pp. 327–329.
[14] A. Radinger, B. Rodriguez-Castro, A. Stolz, and M. Hepp, "BauDataWeb: The Austrian building and construction materials market as linked data," in *Proceedings of the 9th International Conference on Semantic Systems*, Graz, Austria, September 2013, pp. 25–32.
[15] A. Radinger, M. Hepp, and O. Handle, "BauDataWeb: The European building and construction materials database for the semantic web," Website, 2013, http://semantic.eurobau.com/.
[16] V. Premkumar, S. Krishnamurty, J. C. Wileden, and I. R. Grosse, "A semantic knowledge management system for laminated composites," Advanced Engineering Informatics **28**(1), 91–101 (2014).
[17] J. Kaufman and E. Begley, "MatML: A data interchange markup language," Advanced Materials & Processes **161**(11), 35–36 (2003).
[18] "MatML version 3.1 schema," Website, 2004, http:www.matml.org/schema.htm.
[19] T. Ashino and N. Oka, "Development of information platform for data exchange between heterogeneous material data resources," in *Proceedings from the Materials Science & Technology Conference, Detroit, Michigan*, Detroit, Michigan, September 2007, pp. 1851–1861.
[20] X. Zhang, C. Hu, and H. Li, "Semantic query on materials data based on mapping MatML to an OWL ontology," Data Science Journal **8**, 1–17 (2009).
[21] H. Chen, Z. Wu, H. Wang, and Y. Mao, "Rdf/rdfs-based relational database integration," in *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*. IEEE, 2006, pp. 94–94.
[22] D. Dou, P. LePendu, S. Kim, and P. Qi, "Integrating databases into the semantic web through an ontology-based framework," in *Data Engineering Workshops, 2006. Proceedings. 22nd International Conference on*. IEEE, 2006, pp. 54–54.
[23] D. L. McGuinness, F. Van Harmelen *et al.*, "OWL web ontology language overview," W3C recommendation **10**(10), 2004 (2004).
[24] C. C. Sun, "Materials science tetrahedron—A useful tool for pharmaceutical research and development," Journal of Pharmaceutical Sciences **98**(5), 1671–1687 (2009).
[25] A. Maedche and S. Staab, "Ontology learning for the semantic web," IEEE Intelligent systems **16**(2), 72–79 (2001).
[26] A. C. Varzi, "Mereology," 2014.

[27] B. A. Davey and H. A. Priestley, *Introduction to lattices and order*. Cambridge university press, 2002.
[28] F. Harary, "The number of linear, directed, rooted, and connected graphs," Transactions of the American Mathematical Society **78**(2), 445–463 (1955).
[29] J. E. Saal, S. Kirklin, M. Aykol, B. Meredig, and C. Wolverton, "Materials design and discovery with high-throughput density functional theory: The open quantum materials database (OQMD)," JOM **65**(11), 1501–1509 (2013).
[30] S. Kirklin, J. E. Saal, B. Meredig, A. Thompson, J. W. Doak, M. Aykol, S. Rühl, and C. Wolverton, "The open quantum materials database (OQMD): Assessing the accuracy of DFT formation energies," npj Computational Materials **15010**, 1–14 (2015).
[31] A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, and K. A. Persson, "Commentary: The materials project: A materials genome approach to accelerating materials innovation," APL Materials (011002), 1–11 (2013).
[32] J. J. Carroll, I. Dickinson, C. Dollin, D. Reynolds, A. Seaborne, and K. Wilkinson, "Jena: implementing the semantic web recommendations," in *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, New York, NY, USA, May 2004, pp. 74–83.