BioBroker: Knowledge Discovery Framework for Heterogeneous Biomedical Ontologies and Data

Feichen Shen, Yuyung Lee
University of Missouri, Kansas City, MO, USA
Email: fsm89@mail.umkc.edu

Abstract
A large number of ontologies have been introduced by the biomedical community in recent years. Knowledge discovery for entity identification from ontology has become an important research area, and it is always interesting to discover how associations are established to connect concepts in a single ontology or across multiple ontologies. However, due to the exponential growth of biomedical big data and their complicated associations, it becomes very challenging to detect key associations among entities in an inefficient dynamic manner. Therefore, there exists a gap between the increasing needs for association detection and large volume of biomedical ontologies. In this paper, to bridge this gap, we presented a knowledge discovery framework, the BioBroker, for grouping entities to facilitate the process of biomedical knowledge discovery in an intelligent way. Specifically, we developed an innovative knowledge discovery algorithm that combines a graph clustering method and an indexing technique to discovery knowledge patterns over a set of interlinked data sources in an efficient way. We have demonstrated capabilities of the BioBroker for query execution with a use case study on a subset of the Bio2RDF life science linked data.

Keywords
Knowledge Discovery, Ontology, Linked Data

1. Introduction
With a large number of ontologies have been introduced by the biomedical community in recent years, one of the issues researchers are facing in healthcare and biomedical research is the challenging in analytics associated with large,
complex, and dynamic healthcare data (e.g., electronic health records (EHRs), biomedical ontologies). Since there lacks appropriate tools and computational infrastructure that can be fully understood and utilized by involved personnel, very few capacities can be found to carry out analyses of these datasets [1]. As the demand for the integration and analysis of data has been growing steadily, the first effort toward connecting scattered biomedical data materialized as a data movement by the biomedical community (i.e., the Linked Data) [2].

Increasingly, we are also seeing the emergence of biomedical and scientific collaboration. The Semantic Web Health Care and Life Sciences Interest Group (HCLSIG) [3] was formed to “improve collaboration, research and development, and innovation in the information ecosystem of the health care and life science domains using Semantic Web technologies”. In this drive, the large amounts of biomedical data have been specified and shared via machine-readable formats, such as the Resource Description Framework (RDF) [4] and the Ontology Web Language (OWL) [5]. Ontologies are developed to easily extend the work of others and share across different domains. These semantic web technologies make it easier and more practical to integrate, query, and analyze the full scale of relevant biomedical and healthcare data for constructing cost effective health care systems [6]. From then on, knowledge discovery for entity identification from ontologies and various datasets [7] [8] [9] has become an important research area.

Although semantic web provides a solution for biomedical information exchange, there still exist significant difficulties on semantic seamless interoperability and interchange [10] [11] [12]. What is more, existing semantic approaches for linking are promising, but due to the exponential growth of biomedical big data and their complicated associations, it needs expensive computational capabilities to find key associations among entities in an inefficient dynamic manner [13] [14] [15]. The investigation on detecting associations among entities in a single ontology or across multiple ontologies is always an interesting topic [16] [17] [18] and there exists a gap between the increasing needs for association detection and large volume of biomedical ontologies.

Many efforts have been made to perform knowledge discovery with semantic web techniques. For example, in general settings, vSparQL was introduced to enable application ontologies to be derived from these large, fragmented sources such as the FMA [19]. The SMARTSPACE proposed a distributed platform for semantic knowledge discovery from services using multi-agent approach [20]. As a knowledge discovery task combined knowledge and clinical data, clinical ontology has been incorporated into collaborative filtering algorithm in our previous work to predict rare disease diagnosis [21] [22]. The PEMAR introduced a smart phone middleware for activity recognition discovery based on semantic models [23]. The GLEEN project aims to develop a service to simplify views for complex ontologies [24]. A mobile-cloud computing framework was established to discover infrastructure condition based on a back-end semantic knowledge discovery engine [25]. In our previous work, we have built a situation aware
mobile applications framework [26] [27] to discovery users’ activities in a dynamic way based on the semantic web rule language (SWRL) [28]. In biomedical domains, Tao et al., have investigated the usage of semantic web technologies to discovery patient group based on advanced phenotyping algorithms [29] [30]. Based on the pharmacogenomics knowledge base (pharmgkb) [31], Zhu et al., have leveraged web ontology language (OWL) and cheminformatics approaches to assist drug repositioning in breast cancer [32]. However, these studies didn’t investigate the knowledge discovery on heterogeneous ontologies.

In this study, we presented a knowledge discovery framework BioBroker, which equipped with innovative algorithms that combine graph clustering method and an indexing technique. The aim of this framework is to generate cohesive query statements out of heterogeneous ontologies and execute these queries for the purpose of knowledge acquisition and discovery.

In the following, we first introduce materials used in this study. Next, we describe the methods and evaluation approaches used to build and test the framework. We then present the results followed by discussion. Lastly, we conclude and discuss potential future directions.

2. Materials

**The Resource Description Framework (RDF)**

The RDF is a standard model for data interchange and information exchange on the web. It extends the linking structure of the web to use URIs to name the relationship between things as well as the two ends of the link, which are usually referred to as a triplet <subject, predicate, object> [33]. Ontologies are built upon the RDF with restrictions and axioms.

**Bio2RDF**

Bio2RDF is a collection of biological knowledge bases which leverages semantic web technologies to provide interlinked life science data [34]. In this study, we used Bio2RDF release 2 and picked three widely used biomedical ontologies as a group of heterogeneous datasets for evaluation. They are the DrugBank [35] ontology, the HUGO Gene Nomenclature Committee (HGNC) [36], and the Mouse Genome Informatics (MGI) database [37].

**Cytoscape**

The Cytoscape is an open source software used to visualize bioinformatics information and network [38]. In this study, we used the Cytoscape version 3.0.2 to develop the BioBroker knowledge discovery plugin.

**OpenLink Virtuoso**

The OpenLink Virtuoso is a triple store database for managing linked data from existing data silos [39]. In this study, we installed the Virtuoso version 6.1 to store the heterogeneous ontologies.

3. Methods

The objective of this research is to find predicate patterns with a high degree of
connectivity and identify a relatively small number of hops via highly connected nodes to traverse the RDF graphs. We are presenting how to define and discover such patterns of those significant nodes and use them for scalable query processing. We present our predicate-centric model in terms of definition of predicate patterns, discovery of patterns, and usage of patterns during query processing. Figure 1 summarizes the proposed framework and the following paragraphs illustrate each process and methodology respectively.

**Predicate Patterns**

A predicate $P$ is representing a binary relation between two concepts ($c_1$ and $c_2$) in ontology. In RDF/OWL, $P$ is represented as a property to express any kind of relationship (e.g., SubClassOf, Type) between domain (subject) and range (object) [5]. The domain and range may be either from the same ontology or from different ontologies. In our study, relationships are defined by the empirical analysis of ontology data. We are particularly interested in predicates (relationships) that are different from existing approaches like PSPARQL [40] and SPARQLer [41]. Apart from being similar, predicates may share other aspects, e.g., sharing the same subjects or the same objects as well as the connectivity between predicates. This focuses on not only concepts among graphs but links and other structural aspects of the concepts. In this study, the two types of predicate patterns are defined as follows.

**Share Patterns**: As shown in Table 1, this type of pattern describes the comprehension of the relationships between interacting nodes such as shared subjects and shared objects through the given predicate. Assume that two predicates are given as follows: $P_1 <S_i, O_i>$ and $P_2 <S_j, O_j>$ where $S_i, S_j$ are a set of subjects

![Figure 1. The BioBroker Framework.](image)

<table>
<thead>
<tr>
<th>Patterns</th>
<th>Semantic and Pragmatic Knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Exact</td>
</tr>
<tr>
<td>Subject-Object Share</td>
<td>$S_i == S_j$ &amp; $O_i == O_j$</td>
</tr>
<tr>
<td>Subject Share</td>
<td>$S_i == S_j$</td>
</tr>
<tr>
<td>Object Share</td>
<td>$O_i == O_j$</td>
</tr>
</tbody>
</table>
Table 2. Predicate connectivity patterns.

<table>
<thead>
<tr>
<th>Patterns</th>
<th>Semantic and Pragmatic Knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path Connectivity</td>
<td>Si → P1 → O1 → P2 → Oj ( P1 \neq P2 &amp;&amp; O1 = Sj )</td>
</tr>
<tr>
<td>Cycle Connectivity</td>
<td>Si → P1 → O1 → P2 → Oj ( P1 \neq P2 &amp;&amp; O1 = Sj &amp;&amp; Si = Oj )</td>
</tr>
</tbody>
</table>

and \( Oi, Oj \) are a set of objects in given ontologies.

Connection Patterns: According to Table 2, the connection pattern is a frequently recurring pattern with predicates observed during ontology analysis and query processing as the basis for joining one query pattern to another. This pattern is based mainly on the connectivity of concept(s) through the respective predicates. This type of pattern describes the comprehension of the connectivity relationships between interacting predicates. Assume that two predicates are given as follows: \( P1 <Si, Oi> \) and \( P2 <Sj, Oj> \) where \( P1 \) is directly connected to \( P2 \) through \( Oi \) in the given ontologies.

Ontology Clustering with Predicates

Based on the defined two predicate patterns, we found out that predicates play an important role as hubs to share information and connect entities among heterogeneous data. Therefore, we gave a hypothesis that graphs can be fuzzy clustered based on predicate sharing and distance measurement, and data in the same clustered group have a closer relationship than when in different ones.

Predicate Neighboring Level Determination: First, we need to define the boundary of domains in terms of sets of concepts and relations over the datasets. For this purpose, we proposed a predicate neighboring algorithm to determine the closeness of each of the two different predicates. Different shapes of edges denote different relationships between predicates \( p_i \) and \( p_j \) through concepts \( C \). Level 1 has four different combinations that are based on a predicate sharing pattern as well as a connection pattern. Levels 2 and 3 have two various paths, respectively, that are based only on a predicate connection pattern. The formal definition is shown in Definition 1. It is obvious to find that the closeness of the relationship decreases as the level increases. Here we set the upper limit to three because we assume any relationship between predicates and beyond three levels is sparse.

**Definition 1**: Given a directed graph \( G(V,E) \), Vertices \( V_s, V_p, V_o \) denote subject, predicate, and object nodes in the RDF graph, respectively. Let \( d(p_i, p_j) \) represent the shortest distance between \( p_i \) and \( p_j \), \( r(p_i, p_j) \) determine the reachability between \( p_i \) and \( p_j \), \( n(p_i, p_j) \) indicates the neighbors’ closest level between \( p_i \) and \( p_j \):

\[
n(p_i, p_j) = \begin{cases} 
1, & \text{if } d(p_i, p_j) = 1 \\
2, & \text{if } d(p_i, p_j) = 2 \&\& r(p_i, p_j) = \text{true} \\
3, & \text{if } d(p_i, p_j) = 3 \&\& r(p_i, p_j) = \text{true}
\end{cases}
\]
Predicate Similarity Measurement Calculation

We utilized clustering approach to discover predicate association patterns from ontologies. The similarity based confusion measurement for the clustering algorithm varies based on different neighboring levels for each pair of predicates. Basically, we give higher weightage to closer predicates and lower weightage to further predicates. We give Definitions 2, 3, and 4 based on three levels respectively. The formula to generate a confusion matrix for a clustering algorithm is given by Definition 5.

**Definition 2:** Denote \( p_i \) and \( p_j \) as predicates in an RDF graph. A set of sets \( \{ \{ S_1 \}, \{ S_{1j} \} \} \) contain all the predicates such that \( \forall m \in \{ S_1 \} \rightarrow n(m, p_j) = 1 \) and \( \forall n \in \{ S_{1j} \} \rightarrow n(n, p_j) = 1 \). Let \( P(x) \) represent the number of entities that directly connect to predicate set \( x \) and \( E(e) \) represents the number of entities for a given entity set \( e \). Given entity set \( \{ C_1 \} \) so that \( \forall e_i \in \{ C_1 \} \rightarrow e_i \in \{ S_{1i} \} \) and \( e_i \in \{ S_{1j} \} \). The probability-based similarity \( PS_{ij} = \frac{E(C_i) \cdot E(C_j)}{P(S_{1i}) \cdot P(S_{1j})} \).

**Definition 3:** Denote \( p_i \) and \( p_j \) as predicates in an RDF graph. A set of sets \( \{ \{ S_1 \}, \{ S_{1j} \}, \{ S_{2i} \}, \{ S_{2j} \} \} \) contain all the predicates such that \( \forall m \in \{ S_1 \} \rightarrow n(m, p_j) = 1 \), \( \forall n \in \{ S_{1j} \} \rightarrow n(n, p_j) = 1 \), \( \forall x \in \{ S_{2i} \} \rightarrow n(x, p_i) = 2 \) and \( \forall y \in \{ S_{2j} \} \rightarrow n(y, p_j) = 2 \). Let \( P(x) \) represent the number of entities that directly connect to predicate set \( x \) and \( E(e) \) represent the number of entities for a given entity set \( e \). Given entity set \( \{ C_1 \} \) such that \( \forall e_i \in \{ C_1 \} \rightarrow e_i \in \{ S_{1i} \} \) and \( e_i \in \{ S_{1j} \} \). The probability-based similarity \( PS_{ij} = \frac{E(C_i) \cdot E(C_j)}{P(S_{1i}) \cdot P(S_{1j})} \) and \( PS_{ij}^2 = \frac{E(C_i) \cdot E(C_j)}{P(S_{1i}) \cdot P(S_{1j})} \).

Given entity set \( \{ C_2 \} \) such that \( \forall e_i \in \{ C_2 \} \rightarrow e_i \in \{ S_{2i} \} \) and \( e_i \in \{ S_{2j} \} \). The probability-based similarity \( PS_{ij}^3 = \frac{E(C_i) \cdot E(C_j)}{P(S_{2i}) \cdot P(S_{2j})} \). Thus,

\[
PS_{ij} = \text{Max}(PS_{ij}^1, PS_{ij}^2, PS_{ij}^3) \]

**Definition 4:** Denote \( p_i \) and \( p_j \) as predicates in an RDF graph. A set of sets \( \{ \{ S_1 \}, \{ S_{1j} \}, \{ S_{2i} \}, \{ S_{2j} \}, \{ S_{3i} \}, \{ S_{3j} \} \} \) contain all the predicates such that \( \forall m \in \{ S_1 \} \rightarrow n(m, p_j) = 1 \), \( \forall n \in \{ S_{1j} \} \rightarrow n(n, p_j) = 1 \), \( \forall x \in \{ S_{2i} \} \rightarrow n(x, p_i) = 2 \) and \( \forall y \in \{ S_{2j} \} \rightarrow n(y, p_j) = 2 \), \( \forall t \in \{ S_{3i} \} \rightarrow n(t, p_j) = 3 \) and \( \forall k \in \{ S_{3j} \} \rightarrow n(k, p_j) = 3 \). Let \( P(x) \) represent the number of entities that directly connect to predicate set \( x \) and \( E(e) \) represents the number of entities for a given entity set \( e \). Given set \( \{ C_1 \} \) such that \( \forall e_i \in \{ C_1 \} \rightarrow e_i \in \{ S_{1i} \} \) and \( e_i \in \{ S_{1j} \} \) or \( \forall e_i \in \{ C_1 \} \rightarrow e_i \in \{ S_{1j} \} \) and \( e_i \in \{ S_{1j} \} \). The probability-based similarity \( PS_{ij}^4 = \frac{E(C_i) \cdot E(C_j)}{P(S_{1i}) \cdot P(S_{1j})} \) and \( PS_{ij}^5 = \frac{E(C_i) \cdot E(C_j)}{P(S_{1i}) \cdot P(S_{1j})} \).
Given set \( \{ C_2 \} \) such that \( \forall e_1 \in \{ C_2 \} \rightarrow e_2 \in \{ S_{i2} \} \) and \( e_2 \in \{ S_{j2} \} \), or \( \forall e_1 \in \{ C_2 \} \rightarrow e_2 \in \{ S_{j2} \} \) and \( e_2 \in \{ S_{i2} \} \).

The probability-based similarity \( PS_{ij}^3 = \frac{E(C_2)}{P(S_{i2})} * \frac{E(C_2)}{P(S_{j2})} \) and \( PS_{ij}^4 = \frac{E(C_2)}{P(S_{i2})} * \frac{E(C_2)}{P(S_{j2})} \).

Given set \( \{ C_3 \} \) such that \( \forall e_1 \in \{ C_3 \} \rightarrow e_2 \in \{ S_{i3} \} \) and \( e_2 \in \{ S_{j3} \} \). The probability-based similarity \( PS_{ij}^5 = \frac{E(C_3)}{P(S_{i3})} * \frac{E(C_3)}{P(S_{j3})} \) thus.

\( PS_{ij} = \max(PS_{ij}^3, PS_{ij}^4, PS_{ij}^5, PS_{ij}^6) \).

**Definition 5:** Given confusion matrix CM and total number of predicate \( n \).

Denote \( PS_{ij} \) as the probability-based similarity score between predicates \( p_i \) and \( p_j \), based on different levels, so that:

\[
CM[p_i, p_j] = \begin{cases} 
PS_{ij}, & \text{if } p_i \neq p_j, 0 \leq i \leq n, 0 \leq j \leq n \\
1, & \text{if } p_i = p_j, 0 \leq i \leq n, 0 \leq j \leq n 
\end{cases}
\]

We posit that predicate clustering is a required step for efficient query processing involving the alignment and integration of ontologies. Here we clarify our approach to efficient query processing and query generation within the above theoretical framework. A query processing consists of a collection of several relationships between multiple properties. Given that properties are more closely related to some properties more than others, property clustering and partition can be utilized for efficient query processing—the task of classifying a collection of properties into clusters. The guiding principle is to minimize inter-cluster similarity and maximize intra-cluster similarity, based on the notion of semantic distance.

**Hierarchical Fuzzy C-Means Clustering Algorithm**

To discover the correlation between predicates, we used an innovative Hierarchical Fuzzy C-Means (HFCM) clustering algorithm. We created the HFCM algorithm and made a functional extension based on a Fuzzy C-Means clustering algorithm [42] [43]. In general, we set a machine capacity threshold to denote a certain number of triplets that each machine can hold. In addition, we kept applying the HFCM algorithm on each cluster until the number of triplets for each cluster was less than or equal to the threshold or no further change of numbers of elements for each cluster could be made. When compared to traditional Fuzzy C-Means algorithm, the HFCM is able to provide clustering topics in a hierarchical manner and provide flexibility to select clusters by levels. The algorithm of the HFCM is given in **Algorithm 1**.

**Indexing for Ontology and Data**

Based on the variety of large biomedical data spreading in different clusters, a new indexing technique was developed for representing predicate patterns of ontologies from the clusters. Specifically, a two-level encoding approach has
Algorithm 1. Hierarchical Fuzzy C-Means Clustering

Input: Initialize list of data $X = \{X_1, \ldots, X_n\}$ with size $n$, threshold $t$, number of cluster $c$

Output: List of $c$ cluster centers $C = \{C_1, \ldots, C_c\}$

1. for $c = 1$ to $n$

2. List candidateList = $\arg \min \sum \sum |x_i - c_j|$

3. Apply Silhouette Width on candidateList, give value $q$

4. end for

5. Choose optimal $q$. List finalList = candidateList

6. for each cluster set $s$ in finalList

7. int clusterSize = $s$.size()

8. if(clusterSize $>$ $t$)

9. $n = c$

10. do 1-18

11. end if

12. else if All clusterSize $\leq t$

13. return $C$

14. end else if

15. else if All clusterSize doesn’t change

16. return $C$

17. end else if

18. end for

been developed to index the RDF schema, instance, and triple. For the cluster spaces, the two-level hierarchical indexing technique provides efficient representation of complex relations between nodes and predicate association patterns. We used binary encoding to index OWL/RDF schema and make binary with bitmap encoding together to index the OWL/RDF instance. For schema level, our assumption is that the size of schema for each medical and healthcare knowledge base should be a constant. The total size of schema encoding can be controlled even if binary encoding increases drastically. We used the binary index from binary 10 and started encoding with predicate to make sure all the predicate encoding was less than the entities encoding. For instance level, we assigned a unique bitmap index to each instance under its schema encoding. Our design philosophy is that instances with different schemas can share the same encoding but instances under the same schema must be assigned a unique indexing. Therefore, with the huge amount of instances, bitmap indexing could be used in a scalable way and the combination of both binary and bitmap indexing uniquely determined an instance. For triple level, we applied logic or operation on schema encoding of the RDF subject, predicate, and object to generate the result. If a triple did not have a cycle, then we set the object schema encoding to be larger than the subject encoding. If a triple had a cycle, we used the right most bit as the indication of cycle bit and set the subject encoding as larger than the object encoding. In such a design, we can easily differentiate a cycle triple with a non-cycle one. Definition 6 illustrates this encoding approach in specific.
Definition 6: Given the \( i \)-th node \((i \geq 0)\) of Schema set \( \{S\} \), \( j \)-th node \((j \geq 0)\) of Instance set \( \{I\} \), predecessor set \( \{m\} \) and \( \{n\} \) contain all the father nodes of \( i \) and \( j \), respectively. Denote each RDF triplet \( t \) as \( \{s, p, o\} \). Let \( S(i) \) represent schema encoding set, \( I(j) \) represent instance encoding set, \( TS(t) \) represent triple schema encoding set, \( TI(t) \) represent triple instance encoding set and integer number \( R \) represent the magnitude of the data:

\[
S(i) = \begin{cases} 
\{S(i-1) \lor 2^i\}, & \text{if } \{m\} \neq \emptyset \text{ and } \forall i \in \{m\} \\
\{2^i\}, & \text{if } \{m\} = \emptyset
\end{cases}
\]

\[
I(j) = \begin{cases} 
\left\{S(i) + \frac{I(j-1)+1}{R}\right\}, & \text{if } \{n\} \neq \emptyset \text{ and } \forall j \in \{n\} \\
\left\{S(i) + \frac{1}{R}\right\}, & \text{if } \{n\} = \emptyset
\end{cases}
\]

\[
TS(t) = \begin{cases} 
\{S(s) \lor S(p) \lor S(o) | S(o) > S(s) > S(p) > 1\}, & \text{if } \{s, p, o\} \text{ does not form a cycle} \\
\{S(s) \lor S(p) \lor S(o) \lor 1 | S(s) > S(o) > S(p) > 1\}, & \text{if } \{s, p, o\} \text{ forms a cycle}
\end{cases}
\]

\[
TI(t) = \left\{TS(t) + \frac{I(s)}{R} + \frac{I(o)}{R}\right\}
\]

Query Processing using Predicate Patterns based Clustering and Indexing

An intuitive query system was implemented based on clustering and indexing based on predicate patterns for imported medical data. Due to this innovated approach, the users’ query could be answered with high accuracy and performance. A structured representation of semantic relations between concepts can be intuitively extended to query systems. Some features of our prototype Biobroker framework are listed below.

Integrated OWL/RDF Schema Clustering: Different OWL/RDF medical sets can be imported to the BioBroker. Our system is able to parse the schema based on data and apply the HFCM algorithm on schema based on predicate similarity. Clustering graphs are also generated accordingly and triple with the same predicate among different schema sets can be linked. Figure 2 shows predicate-based clustering graphs with 3 Bio2RDF data schema after supplying the BioBroker a predicate similarity feature vector by clicking the H-Fuzzy C-means Clustering button. Detailed predicate clustering information was also listed in the clustering panel on the left. Because we used hierarchical approach in addition to the Fuzzy C-means clustering, our system provided options to display different levels of data. Figure 2 shows an example with level 3.

Query Boundary: Query processing can be optimized based on the proposed concept of query boundary. The boundary can be determined by predicate association and clustering sets. A query boundary characterizes a particular dynamic reasoning and query capability of the proposed model that is specifically tailored.
for query semantics. More specifically, it can be proved that for a specific kind of user’s query, there exist a fixed set of abstract patterns that are involved in the query processing process. This fixed set is called the query boundary for the specific type of users’ query. A query was described into query boundary within clusters and the BioBroker used a different green color to indicate such boundary specifically. As an example shown in Figure 3, we included three non-built-in predicates which are `drugbank:enzyme`, `drugbank:action` and `drugbank:drug`.

**Interactive Query Design with Suggestion:** The predicates extracted from a user’s SPARQL query [44] are mapped to the predicate neighboring and clustering results. If matched, then a set of relevant queries for different medical data sets can be composed with the significant properties (relationships) between concepts that are identified by the proposed formula in this paper. The BioBroker provides a customized query design and suggestion feature to make it convenient for users to design benchmark queries. An extended function is developed to add query based on predicate association. Users can choose available triple from the drop down list and they can also assign variables for subjects and
objects. As shown in Figure 3, a query schema suggestion (Drug -> enzyme -> Resource and Drug -> target -> Resource) was given to user. Meanwhile, all the related instances were read from database and listed for users to choose. Here we gave DB00072 as subject drug name and set ?R1 and ?R2 as objects. In this example, The BioBroker was able to find target names for drugbank:enzyme and drugbank:target based on given drug instance. A query boundary with integrated graph is also showed in this example.

Query Indexing to Optimize Benchmark Query Performance: The BioBroker translates each SPARQL query to query indexing format based on medical Ontology and data indexing. Therefore, executing the SPARQL query is actually performing logical operations on schema binary indexing and mathematical operations on instance bitmap indexing. In Figure 3, a SPARQL query was given and its corresponding query graph was shown. The BioBoker translated the SPARQL query into binary format and generated results for user.

Evaluation

The BioBroker prototype system was implemented using Java on Eclipse Juno Integrated Development Environment [45]. Apache Jena API was used to parse OWL/RDF datasets and retrieve triple information. We used R computing environment [46] to implement algorithms and generate predicate clusters. We designed a plugin to generate query and schema graphs by programming with Cytoscape 3.0.238. We embedded an encoding query engine in the plugin and provided suggested query option based on the clustering results. To report the similarity measurements of the predicates in these datasets on to excel files, we used Java Excel API [47].

The evaluation of the BioBroker system is conducted in terms of the valid of clustering result and justified query benchmark generation. We used three ontologies from Bio2RDF release 2 to evaluate our system. Detailed information for each ontology is given in Table 3. In addition to that, we eliminated some RDF built-in predicates and types for getting the best clustering result.

To select the optimal clustering algorithm for knowledge discovery, we first compared performances yielded by the Hierarchical Fuzzy C-Means (HFCM), the Partition Around Medoids (PAM) algorithm [48], the Clustering Large Application (CLARA) algorithm [49], the K-Means clustering algorithm [50] and the Hierarchical Clustering (HC) algorithm [51]. To get the optimal number of clusters, we used Silhouette Width (SW) [52] to evaluate different results and chose the one with the biggest score. In addition, we used the Sum of Squares for Error (SSE) metric [53] to double check the optimal number of clusters for the

<table>
<thead>
<tr>
<th>Table 3. Bio2RDF ontology information.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ontology</strong></td>
</tr>
<tr>
<td>----------------------------------------</td>
</tr>
<tr>
<td>Drugbank</td>
</tr>
<tr>
<td>HUGO Gene Nomenclature Committee (HGNC)</td>
</tr>
<tr>
<td>Mouse Genome Informatics (MGI)</td>
</tr>
</tbody>
</table>
selected optimal clustering algorithm.

For query evaluation, we selected eight query benchmarks [54] [55] and used BioBroker and Virtuoso to execute each of them for query outputs validation and query execution performance test. The machine we used to execute queries has an Intel Pentium G3220 3.00 GHz CPU. The memory size is 12 GB and the storage size is 1 TB.

4. Results

Evaluation for HFCM Algorithm

As shown in Figure 4, according to SW score, all algorithms produced the optimal performances at the point when number of clusters became 2. We found that K-Means yielded the highest SW score as 0.9, HFCM produced the suboptimal performance as 0.88, and the other three algorithms contributed to a same SW score as 0.76. Although SW for K-Means was higher than the one for HFCM, there is no statistical significant difference between them. Therefore, we selected HFCM as the optimal algorithm since it is able to provide additional soft partition capabilities, which was useful for distributed query processing. As a result, the HFCM produced 7 clusters in total as final outputs based purely on non-built-in RDF predicates. We then used the SSE metric to confirm the optimal number of clusters for the HFCM. As shown in Figure 5, the first concavity point for the SSE plot proved that the optimal number of clusters is 2.

![Figure 4. Evaluations for clustering algorithms.](image_url)
**Figure 5.** The sum of squares for error (SSE) plot for the HFCM.

**Table 4.** Hierarchical fuzzy c-means clustering result.

<table>
<thead>
<tr>
<th>ID</th>
<th>Size</th>
<th>Predicate Name</th>
<th>Cluster 1</th>
<th>Cluster 2</th>
<th>Cluster 3</th>
<th>Cluster 4</th>
<th>Cluster 5</th>
<th>Cluster 6</th>
<th>Cluster 7</th>
</tr>
</thead>
</table>

Detailed clustering information by the HFCM is given in **Table 4.** We found that cluster 1 mainly focused on the homogeneous ontology MGI and provided knowledge about associations for phenotype, gene marker, and protein and so on. Cluster 2 discovered cross-domain knowledge between the HGNC and the DrugBank, indicating their associations with other ontologies, such as the OMIM [56] and the UniProt [57]. Cluster 3 depicted biomedical information in HGNC, including gene symbol, ensemble, and outgoing linkage to other knowledge...
bases. Cluster 4, 5, 6, and 7 are all about the DrugBank with different focuses on ingredient, target, interaction, and enzyme respectively.

Evaluation for Query Performance

Query benchmark was established and detailed information of queries can be found in Figure 6. Specifically, Query 2 and 5 were designed based on the online benchmarks with some modifications due to the data version compatible issue, and the rest were designed from the BioBroker suggestions by choosing predicates from single cluster or multiple clusters. In these query graphs, we used color black, blue, pink, red and green color to present entities from the HGNC, the MGI, the DrugBank, built-in predicates/entities, and query boundary respectively. Queries 1 to 4 were designed mainly based on homogeneous DrugBank and the rest queries were designed based on heterogeneous ontologies. Query 1 was about finding interactions between drug and enzyme. Query 2 aimed to detect interactions among drugs. The objective of query 3 is to find ingredient of all mixtures. Query 4 targeted on mining food interactions with drugs. Query 5 was composed of knowledge from the HGNC and the MGI, describing associations among gene symbols, markers, and proteins. Query 6 was composed of information extracted from the HGNC and the DrugBank, illustrating the common protein for pairs of gene symbols and drug target. Query 7 was also made up of information from the HGNC and the DrugBank, introducing the relationship between drug targets and gene symbols. Query 8 is a mixed query with all three ontologies, which aimed to find all gene symbols, drug-targets and gene markers with a common ensemble genome.

We executed all queries on the Virtuotoso Database and retrieved relevant

Figure 6. Homogeneous and heterogeneous query graphs.
Table 5. Query execution results.

<table>
<thead>
<tr>
<th>Query Number</th>
<th>Answers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>( r = \text{DrugBank:DB00157}_711, s = \text{DrugBank:DB00157}, e = \text{DrugBank:12} )</td>
</tr>
<tr>
<td>Q2</td>
<td>( \text{ddi} = \text{DrugBank:DB00001_DB01381}, d1 = \text{DrugBank:DB00001}, d2 = \text{DrugBank:DB00001}, ) ( \text{patent} = \text{uspatent:5180668}, \text{data2} = 2010-01-19 )</td>
</tr>
<tr>
<td>Q3</td>
<td>( m = \text{DrugBank:Cauterex}, i = \text{domase alfa + fibrinolysin + gentamicin sulfate}, d = \text{DrugBank:DB00003} )</td>
</tr>
<tr>
<td>Q4</td>
<td>( d = \text{DrugBank:DB00006}, f = \text{Dan Shen, dong quai, evening primrose oil, gingko, policosanol, willow bark} )</td>
</tr>
<tr>
<td>Q5</td>
<td>( \text{hgn} = \text{HGNC:26946}, \text{marker} = \text{MGI:1913367}, \text{mgi} = \text{MGI:1913367}, \text{uni} = \text{Uniprot:Q9CR13}, \text{ens} = \text{Ensembl: ENSMUSG00000019689} )</td>
</tr>
<tr>
<td>Q6</td>
<td>( \text{Hgn} = \text{HGNC:7863}, \text{target} = \text{DrugBank:11}, u1 = \text{UNIPROT:Q13423}, u2 = \text{Uniprot:Q13423} )</td>
</tr>
<tr>
<td>Q7</td>
<td>( \text{Target} = \text{DrugBank:9}, \text{hgn} = \text{HGNC:5211} )</td>
</tr>
<tr>
<td>Q8</td>
<td>( \text{Target} = \text{DrugBank:6601}, \text{hgn} = \text{HGNC:24427}, \text{mgi} = \text{MGI:88574}, \text{e1} = \text{Ensembl: ENSMUSG00000015340}, \text{e2} = \text{ENSMUSG000000197953} )</td>
</tr>
</tbody>
</table>

Table 6. Query performance comparison (in milliseconds).

<table>
<thead>
<tr>
<th></th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
<th>Q4</th>
<th>Q5</th>
<th>Q6</th>
<th>Q7</th>
<th>Q8</th>
</tr>
</thead>
<tbody>
<tr>
<td>BioBroker</td>
<td>8</td>
<td>17</td>
<td>10</td>
<td>5</td>
<td>8</td>
<td>4</td>
<td>4</td>
<td>25</td>
</tr>
<tr>
<td>Virtuoso</td>
<td>525</td>
<td>954</td>
<td>590</td>
<td>125</td>
<td>403</td>
<td>219</td>
<td>168</td>
<td>1037</td>
</tr>
</tbody>
</table>

results as shown in Table 5. Here we only demonstrated one output for each query.

We also tested query execution performances on Bio2RDF DrugBank, HGNC and MGI dataset with query 1 - 8. We compared our indexed query performance with Virtuoso based SPARQL query performance. The small scale data we used has 3,651,750 triples and 105 properties. The performance comparison results are showed in Table 6. We observed that the BioBroker has a significant faster execution performance than Virtuoso in milliseconds, which indicated that the use of distributed index technique is able to accelerate the query process.

5. Discussion

There are several studies for extension of the SPARQL query with some extended patterns such as path SPARQL [40] and semantic Association discovery [41]. Protein-protein interaction was analyzed with SPARQL based RDF decomposition [3]. However, these are all graph based pattern matching approaches that may not be appropriate for a huge volume of evolving data and subsequently, not suitable for discovering assertions from such data. Therefore, we used a pattern-based approach for analyzing ontologies whose concepts were either subjects or objects in the discovered predicate patterns and used them for query processing. The clustering enhanced the query designing and query processing by providing an ultimately better comprehension of the relationships be-
between interacting nodes on the data. The dynamic clustering allowed us to execute highly specific queries and dynamically expand or shrink knowledge and data space as well as share new data with other clouds making it possible to achieve scalable reasoning.

### 6. Conclusions and Future Work

This paper presents a predicate pattern based model equipped with index technique for query suggestion, visualization, scalable query and reasoning with large biomedical ontology schema and data. The proposed model transforms conjunction SPARQL queries into efficient pattern based queries over a set of interlinked medical data sources. The benefits of predicate-based query processing were shown with discovery of predicate patterns. The proposed model was evaluated with the Bio2RDF datasets and the experimental results of the query designing and results showed the superiority of the proposed predicate-centric model compared to existing query models.

In the future, we will combine graph network analysis approaches [58] [59] with clustering algorithm to provide network motif [60] analysis and LDA-based topic modelling [61]. Furthermore, parallel and distributed algorithms, using the indexing technique, will be developed.

Human Phenotype Ontology (HPO) [62] has been developed as a controlled vocabulary for phenotypes by mining and integrating phenotype knowledge from medical literature and ontologies. HPO also provides associations with other biomedical resources such as the Gene Ontology [63]. We have developed an annotation pipeline leveraging HPO for phenotypic characterization on clinical data [64] [65]. In the future, we will combine knowledge-driven and data-driven approaches to investigate knowledge discovery from clinical domains to facilitate translational research.

### References


161-165.


