Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 91 (2016) 460 – 469

Procedia Computer Science

Information Technology and Quantitative Management (ITQM 2016)

Creating NoSQL Biological Databases with Ontologies for Query Relaxation

Naresh Kumar Gundla[a], Zhengxin Chen[a,*]

[a]Department of Computer Science,University of Nebraska at Omaha,Omaha and NE 68182-0500, USA

Abstract

The complexity of building biological databases is well-known and ontologies play an extremely important role in biological databases. However, much of the emphasis on the role of ontologies in biological databases has been on the construction of databases. In this paper, we explore a somewhat overlooked aspect regarding ontologies in biological databases, namely, how ontologies can be used to assist better database retrieval. In particular, we show how ontologies can be used to revise user submitted queries for query relaxation. In addition, since our research is conducted at today's "big data" era, our investigation is centered on NoSQL databases which serve as a kind of "representatives" of big data. This paper contains two major parts: First we describe our methodology of building two NoSQL application databases (MongoDB and AllegroGraph) using GO ontology, and then discuss how to achieve query relaxation through GO ontology. We report our experiments and show sample queries and results. Our research on query relaxation on NoSQL databases is complementary to existing work in big data and in biological databases and deserves further exploration.

Keywords: NoSQL databases; Query Relaxation; Ontology; MongoDB; AllgroGraph

1. Introduction

The complexity of building biological databases is well-known and ontologies play an extremely important role in biological databases. However, much of the emphasis on the role of ontologies in biological databases has been on the construction of databases. In this paper, we explore a somewhat overlooked aspect regarding ontologies in biological databases, namely, how ontologies can be used to assist better database retrieval. In particular, we show how ontologies can be used to rewrite user submitted queries for query relaxation. In addition, since our research is conducted at today's "big data" era, our investigation is centered on NoSQL databases which serve as a kind of "representatives" of big data.

* Corresponding author. Tel.: +1-402-554-3625'.
E-mail address: zchen@unomaha.edu.

Traditionally query relaxation has been used to rewrite relational queries to deal with failed queries. Query relaxation, as a widely used technique in the database field, is intended to "broaden" a query to return more answers, in the event that no (or few) answers are returned from an initial search query. Query relaxation is important because it can better serve users' information needs. In this research, we take the first step to explore this issue and present our experimental work in query relaxation for big data. We have identified pros and cons of storing ontology in NoSQL Databases and demonstrate the usage of ontologies for searching in another database of same domain in NoSQL database. To show this, we have employed the Gene Ontology and transformed the relational schema to MongoDB and AllegroGraph.

1.1. NoSQL for bioinformatics databases

1.1.1. Basics of NoSQL

As an answer to handling challenging features not found in traditional data (data characterized by volume, velocity, variety, veracity, etc., or big data), non-traditional databases (NoSQL) emerge to offer alternative, flexible and more scalable data stores. .NoSQL databases provide a mechanism for storage and retrieval of data that employs less constrained consistency models than traditional relational databases. Motivations for this approach include simplicity of design, horizontal scaling and finer control over availability. Most NoSQL databases can also be called schema-free databases. The key advantage of the schema-free design is that it enables applications to quickly upgrade the structure of data without table rewrites. It also allows for greater flexibility in storing heterogeneously structured data. Although traditional databases have strictly imposed ACID properties (atomicity, consistency, isolation and durability), a NoSQL database may not fully support these properties. Instead, they are required to satisfy the BA SE properties (Basically available, Soft state and eventually consistent).

NoSQL databases are actually quite diverse. There are several different prototypes (or categories) of NoSQL databases, including Column-based, document-based (with MongoDB as a particular example), key value-based, as well as graph-based (with AllegroGraph as an example), etc.

1.1.2. NoSQL for biological data: Sample NoSQL databases used in bioinformatics

Due to the extreme complexity and various challenges faced by biological data [1], researchers have turned to NoSQL databases for solutions other than traditional ones. For example [2] quantitatively compared the latencies of different data stores on storing and querying proteomics datasets (mass spectrometry or MS) data over NoSQL databases such as MongoDB and HBase. In another study, [3] reported an experimental comparison on PostgreSQL (a traditional relational database product) and Neo4j (a graph database) using data imported from STRING v9.1: protein-protein interaction networks containing 20,140 proteins and 2.2 million interactions, showing that speedup in Neo4j could be hundred or thousand times higher. They concluded that graph databases are ready for bioinformatics and can offer great speedups over relational databases on certain problems.

Reference [4] studied high dimensional biological data retrieval optimization with NoSQL technology using a key-value model. In [5], authors conducted a scientific experiment as a computational workflow. It is a study on provenance data storage for bioinformatics in a document-oriented NoSQL database system, MongoDB. In addition, CouchDB has been used to build three new bioinformatics resources [6].

All of these sample applications has indicated that time of NoSQL for biological data has arrived. Continuing this trend of research, in this study, we present our methodology of building NoSQL biological databases with ontologies.

2. Importance of ontologies in bioinformatics

2.1. Ontologies in bioinformatics

Ontologies are specifications of a relational vocabulary. An ontology is defined as a formal, explicit specification of a shared conceptualization [7]. An ontology can be considered as a system of categories accounting for a particular vision of the world [8]. Since ontologies provide a concise and unambiguous description of principle relevant entities with their potential, valid relations to each other [9], they facilitate the sharing of knowledge between human and human, human and computer, as well as computer and computer.

Data stored in biological databases are extremely complex [1]. Among other things, the same biological object may have different names, and the same name may mean different things. Take one of the most fundamental concepts "gene" as an example. Both definitions shown below appear in the same special database issue of Nucleic Acids Res in 1998 [9].

- Definition A: A gene is a DNA fragment that can be transcribed and translated into a protein [10].
- Definition B: A gene is a DNA region of biological interest with a name and that carries a genetic trait or phenotype which includes non-structural coding DNA regions like intron, promoter and enhancer. – GenBank and GSDB [11].

Because of this known challenge, it is almost mandatory to incorporate ontologies in biological database development. Advantages of incorporating ontologies in biological databases are obvious, because they offer a kind of community reference (neutral authoring), can be used to define database schema or define a common vocabulary for database annotation (ontology as specification), can help understanding database annotation and technical literature, can guide and interpret analyses and hypothesis generation, as well as provide common access to information, because we can have ontology-based search by forming queries over databases using ontologies [12]. Ontologies also play important roles in data mining and text mining (e.g., [13]). In this paper, we explore the aspect of ontology-based search; in particular, we focus on how ontologies can benefit query relaxation, an important issue has been overlooked by biological database community.

The particular ontology to be used in our study is Gene Ontology (GO) [14]. The GO project is a collaborative effort to address the need for consistent descriptions of gene products in different databases. The use of GO terms by several collaborating databases facilitates uniform queries across them. The controlled vocabularies are structured so that one can query them at different levels. GO terms are organized in structures called directed acyclic graphs (DAGs), which differ from hierarchies in that a child, or more specialized, term can have many parents, or less specialized, terms.

2.2. Using ontologies for query relaxation

As stated earlier, query relaxation is intended to "broaden" a query to return more answers, in the event that no (or few) answers are returned from an initial search query. Query relaxation is important because it can better serve users' information needs. Query relaxation in traditional databases has been extensively studied [15], and the use of using ontologies for query relaxation has received much of attention. Ontologies have been incorporated for expanded database keyword search [16] and have gone beyond structured databases to guide XML query relaxation [17]. Exploring the use of ontologies for query relaxation at big data era becomes an even more important issue, because of the big challenges associated with the main features of big data (volume, velocity, variety, veracity, etc.) [18] Recently, query relaxations of an input query that can be rewritten and evaluated in an environment of collaborating autonomous and heterogeneous data sources [19], but more research on query relaxation related to this particular aspect of big data, along with other aspects of big data, are yet to be studied.

Although biological database development has been long associated with ontologies, the issue of using ontologies to rewrite queries for query relaxation is not widely studied and thus deserves much more attention. This is particularly true when we talk about today’s big data.

3. Building NoSQL databases with GO: MongoDB and AllegroGraph

Motivated from observations made above, recently we have built two NoSQL databases incorporating GO ontology.

3.1. MongoDB

MongoDB [20] is an open source, document database designed for ease of developing and scaling. A record in Mongo DB is a document, which is a data structure composed of fields and values pairs. MongoDB documents are similar to JSON objects. The value of a field may include other documents, arrays, and arrays of document. Advantages of using documents are, they correspond to native data types in many programming languages, embedded documents and arrays reduce the need for expensive joins and dynamic schema supports fluent polymorphism.

Gene Ontology in MongoDB: The data from the relational database [21] is converted into JSON objects for each Gene Ontology term and these JSON objects are imported into the mongo DB. To make hierarchy search faster, we are calculating the child nodes up to level 3 recursively and storing them as an embedded JSON into their corresponding term.

An application database can be a database which store information from different domains for example NCBI, Fly Base, gene terms etc. To demonstrate our search methodology we have created a new database which contains GO terms randomly selected from the gene ontology and converted to JSON objects. The created database contains the information about the gene term for example (name, accession, description).

3.2. AllegroGraph

A graph database uses graph structures with nodes, edges, and properties to represent and store data, provides index-free adjacency, meaning every element contains a direct pointer to its adjacent elements and no index lookups are necessary. As a well-known graph database, AllegroGraph is developed by Franz, Inc. which also develops Allegro Common Lisp, an implementation of Common Lisp. AllegroGraph supports SPARQL, RDFS++, and Prolog reasoning from numerous client applications. Although it is a graph database (a prototype of NoSQL), unlike many other NoSQL databases, it is claimed to fully support ACID properties. AllegroGraph uses efficient memory utilization in combination with disk-based storage, enabling it to scale to billions of quads while maintaining superior performance. [22] Therefore, AllegroGraph has a very rich “cultural heritage.” Logical structure of AllegroGraph is shown in Fig 1.

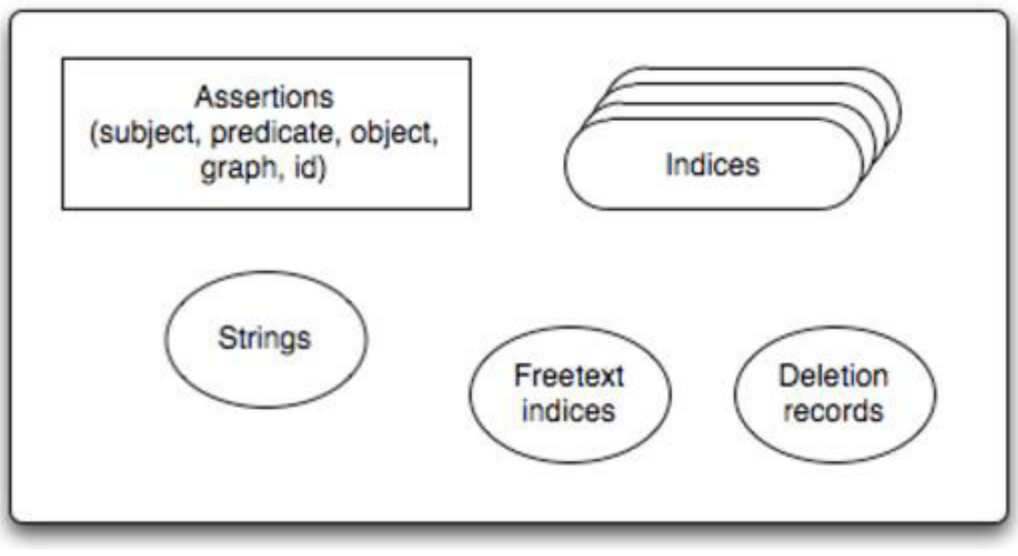


Fig. 1. Allegro Graph Structure

As a database and application framework for building Semantic Web applications, AllegroGraph can store different types of data (Turtle, TriX, TriG, RDF/XML) and meta-data as triples; query these triples through various query APIs like SPARQL and Prolog.

Some specifics of our application is discussed below.

Gene Ontology in Allegro Graph: Allegro graph supports RDF/XML file, so the gene ontology RDF/XML file is imported into the database and graph DB supports the hierarchal queries to retrieve the descendant's for a given term. To make the search efficient, we have created the indices for Go term "name" and "is-a" predicate.

Calculation of Descendants for a given GO Term: We are calculating the child's level by level using the SPARQL queries and combining the results of each level. For example, first we will get the level1 child's, next level2 child's and level3 child's. After that we will combine the three level results into single list (which will be the descendant's list for a given GO Term id up to level3):

- Level1 Query: "PREFIX go:<http://www.geneontology.org/dtds/go.dtd#> select ?term ?name where{ ?term go:name ?name . ?term go:is_a <http://www.geneontology.org/go#"+ id + "> . }";
- Level2 Query: "PREFIX go:<http://www.geneontology.org/dtds/go.dtd#> select ?term ?name where{ ?term go:name ?name . ?term go:is_a ?term_1 . ?term_1 go:name ?name_1 . ?term_1 go:is_a <http://www.geneontology.org/go#"+ id + "> . }";
- Level3 Query: "PREFIX go:<http://www.geneontology.org/dtds/go.dtd#> select ?term ?name where{ ?term go:name ?name . ?term go:is_a ?term_2 . ?term_2 go:is_a ?term_1 . ?term_1 go:is_a <http://www.geneontology.org/go#"+ id + "> . }";

Application Database in AllegroGraph: The application database contains the same terms as in the mongo application database and these terms are converted to the RDF/XML format to store in allegro graph.

4. Using ontologies for query relaxation in MongoDB and Allegro Graph

4.1. Methodology for Searching in MongoDB

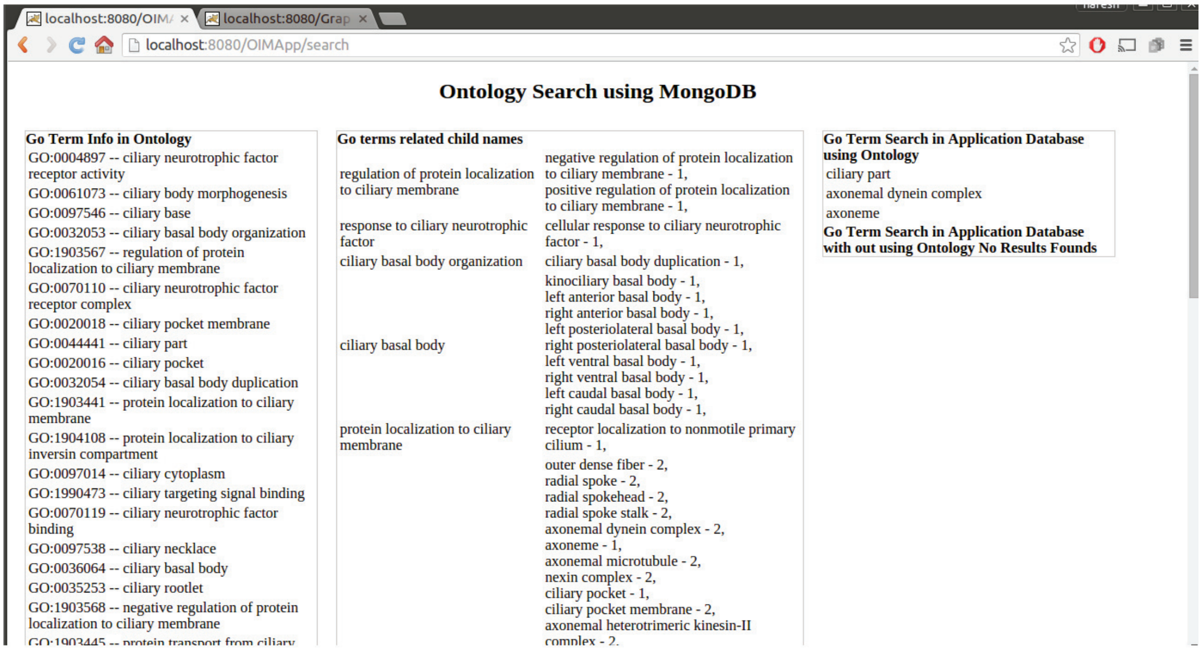
For MongoDB, below are the steps followed for searching a term in application database using ontology and calculated descendants.

- Step1: Search for the term [uses Mongo Full-text Search] in the Gene Ontology database. The results give the list of GO terms.
- Step2: For each term in the list obtained in step 1, get the descendants up to level 3 and add to the list (descendants).[ontology based keyword search] [22][7].
- Step3: Now search the application database for all the terms in the list (descendants).

4.2. Example Query and Results for Searching in MongoDB: Keyword: "ciliary"

Query Processing: In Fig 2. The leftmost section shows the step 1 results, the middle section shows the list of descendants I.e. step 2 and the right most section shows the step3 results.

In the right most section the heading "Go Term Search in Application Database without using ontology" will show the search results of the term in the application database (only exact string search).



localhost:8080/OIMApp/search

Ontology Search using MongoDB

Go Term Info in Ontology

GO:0004897 -- ciliary neurotrophic factor receptor activity
GO:0061073 -- ciliary body morphogenesis
GO:0097546 -- ciliary base
GO:0032053 -- ciliary basal body organization
GO:1903567 -- regulation of protein localization to ciliary membrane
GO:0070110 -- ciliary neurotrophic factor receptor complex
GO:0020018 -- ciliary pocket membrane
GO:0044441 -- ciliary part
GO:0020016 -- ciliary pocket
GO:0032054 -- ciliary basal body duplication
GO:1903441 -- protein localization to ciliary membrane
GO:1904108 -- protein localization to ciliary inversin compartment
GO:0097014 -- ciliary cytoplasm
GO:1990473 -- ciliary targeting signal binding
GO:0070119 -- ciliary neurotrophic factor binding
GO:0097538 -- ciliary necklace
GO:0036064 -- ciliary basal body
GO:0035253 -- ciliary rootlet
GO:1903568 -- negative regulation of protein localization to ciliary membrane
GO:1903445 -- protein transport from ciliary

Go terms related child names

regulation of protein localization to ciliary membrane	negative regulation of protein localization to ciliary membrane - 1, positive regulation of protein localization to ciliary membrane - 1,
response to ciliary neurotrophic factor	cellular response to ciliary neurotrophic factor - 1,
ciliary basal body organization	ciliary basal body duplication - 1,
ciliary basal body	kinociliary basal body - 1, left anterior basal body - 1, right anterior basal body - 1, left posteriolateral basal body - 1, right posteriolateral basal body - 1, left ventral basal body - 1, right ventral basal body - 1, left caudal basal body - 1, right caudal basal body - 1,
protein localization to ciliary membrane	receptor localization to nonmotile primary cilium - 1,
	outer dense fiber - 2, radial spoke - 2, radial spokehead - 2, radial spoke stalk - 2, axonemal dynein complex - 2, axoneme - 1, axonemal microtubule - 2, nexin complex - 2, ciliary pocket - 1, ciliary pocket membrane - 2, axonemal heterotrimeric kinesin-II complex - 2.

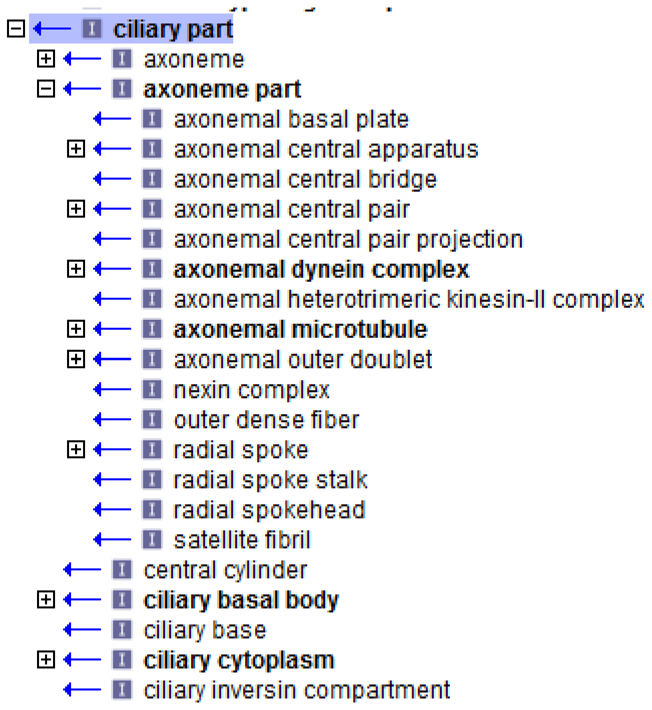
Go Term Search in Application Database using Ontology

ciliary part
axonemal dynein complex
axoneme

Go Term Search in Application Database with out using Ontology No Results Founds

Fig. 2. search results for the GO Term "ciliary"

As shown in Fig 3 the terms "axoneme" and "axonemal dynein complex" are subtypes of the term "ciliary part". The GO term with name "ciliary" is not present in the application database but there exist a term with name "ciliary part". When we search the application database without using ontology it was unable to retrieve the term "ciliary part" and the total results retrieved are 0. So by using ontology we got the terms which are related to the search term as shown in Fig 2.



- ciliary part
  - axoneme
  - axoneme part
    - axonemal basal plate
    - axonemal central apparatus
    - axonemal central bridge
    - axonemal central pair
    - axonemal central pair projection
    - axonemal dynein complex
    - axonemal heterotrimeric kinesin-II complex
    - axonemal microtubule
    - axonemal outer doublet
    - nexin complex
    - outer dense fiber
    - radial spoke
    - radial spoke stalk
    - radial spokehead
    - satellite fibril
  - central cylinder
  - ciliary basal body
  - ciliary base
  - ciliary cytoplasm
  - ciliary inversin compartment

Fig. 3. ciliary part hierarchy

4.3. Methodology for Searching in AllegroGraph Database

Searching in AllegroGraph Database is conducted in following steps:

- Step1. Search for the term [Regular expression search (no Stemming compared to mongo)] in the Gene ontology Repository, the result gives the list of GO terms.
- Step2. For each term find the descendants for level1, level2, level3 and combine to one list and add the parent term also. Repeat the step for all the Go Terms in step1.
- Step3. Now search the application database for all the term in the list (descendants)

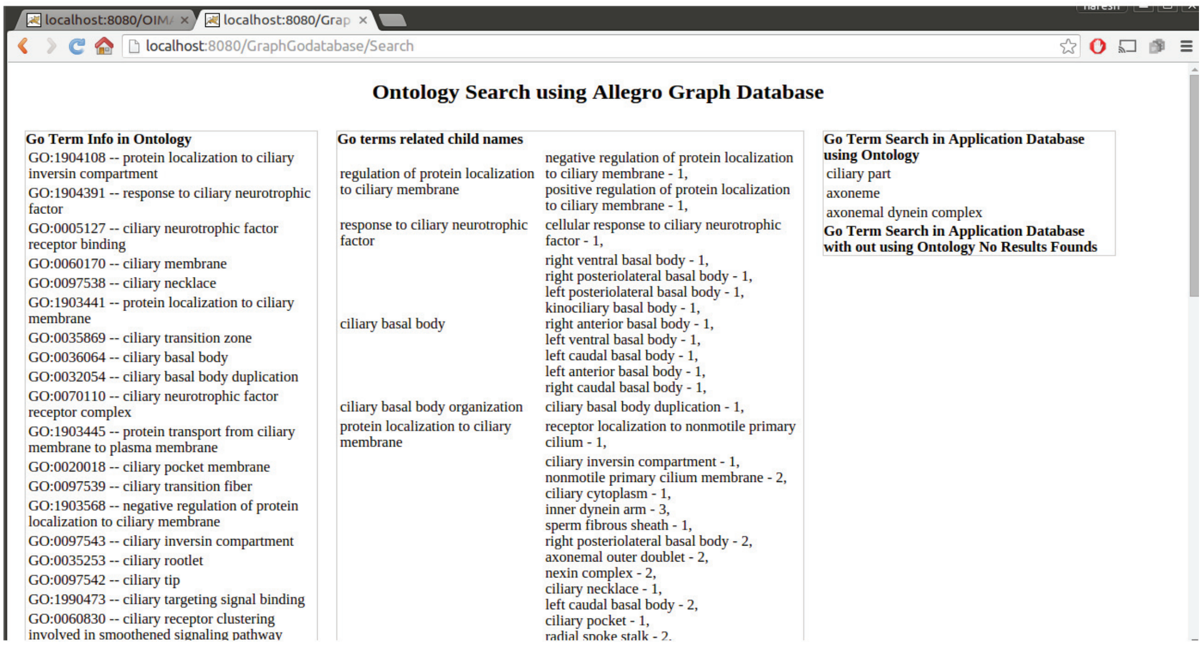
4.4. Example Query and Results for Searching in AllegroGraph Database: Keyword: "ciliary"

In this example, we are searching for the keyword "ciliary" and we found 0 results, whereas with the query rewriting using ontology we can get more results.

Query processing is shown in Fig. 4, where the leftmost section shows the step 1 results, the middle section shows the list of descendants (i.e. step 2) and the right most section shows the step3 results.

In the right most section the heading "Go Term Search in Application Database without using ontology" will show the search results of the term in application database (only exact string search)

As shown in Fig 3, the terms "axoneme" and "axonemal dynein complex" are a subtype of the term "ciliary part". The GO term with name "ciliary" is not present in the application database but there exist a term with name "ciliary part". When we search the application database without using ontology it was unable to retrieve the "ciliary part" term and the total results retrieved are 0. So by using ontology we can get the terms which are related to the search term as shown in Fig 4.



Ontology Search using Allegro Graph Database

Go Term Info in Ontology

GO:1904108 -- protein localization to ciliary inversin compartment
GO:1904391 -- response to ciliary neurotrophic factor
GO:0005127 -- ciliary neurotrophic factor receptor binding
GO:0060170 -- ciliary membrane
GO:0097538 -- ciliary necklace
GO:1903441 -- protein localization to ciliary membrane
GO:0035869 -- ciliary transition zone
GO:0036064 -- ciliary basal body
GO:0032054 -- ciliary basal body duplication
GO:0070110 -- ciliary neurotrophic factor receptor complex
GO:1903445 -- protein transport from ciliary membrane to plasma membrane
GO:0020018 -- ciliary pocket membrane
GO:0097539 -- ciliary transition fiber
GO:1903568 -- negative regulation of protein localization to ciliary membrane
GO:0097543 -- ciliary inversin compartment
GO:0035253 -- ciliary rootlet
GO:0097542 -- ciliary tip
GO:1990473 -- ciliary targeting signal binding
GO:0060830 -- ciliary receptor clustering involved in smoothened signaling pathway

Go terms related child names

regulation of protein localization to ciliary membrane	negative regulation of protein localization to ciliary membrane - 1, positive regulation of protein localization to ciliary membrane - 1,
response to ciliary neurotrophic factor	cellular response to ciliary neurotrophic factor - 1,
ciliary basal body	right ventral basal body - 1, right posteriolateral basal body - 1, left posteriolateral basal body - 1, kinociliary basal body - 1, right anterior basal body - 1, left ventral basal body - 1, left caudal basal body - 1, left anterior basal body - 1, right caudal basal body - 1,
ciliary basal body organization	ciliary basal body duplication - 1,
protein localization to ciliary membrane	receptor localization to nonmotile primary cilium - 1,
	ciliary inversin compartment - 1, nonmotile primary cilium membrane - 2, ciliary cytoplasm - 1, inner dynein arm - 3, sperm fibrous sheath - 1, right posteriolateral basal body - 2, axonemal outer doublet - 2, nexin complex - 2, ciliary necklace - 1, left caudal basal body - 2, ciliary pocket - 1, radial spoke stalk - 2,

Go Term Search in Application Database using Ontology

ciliary part
axoneme
axonemal dynein complex

Go Term Search in Application Database with out using Ontology No Results Founds

Fig. 4. Search results for the GO Term “ciliary”

4.5. Comparison and recommendation

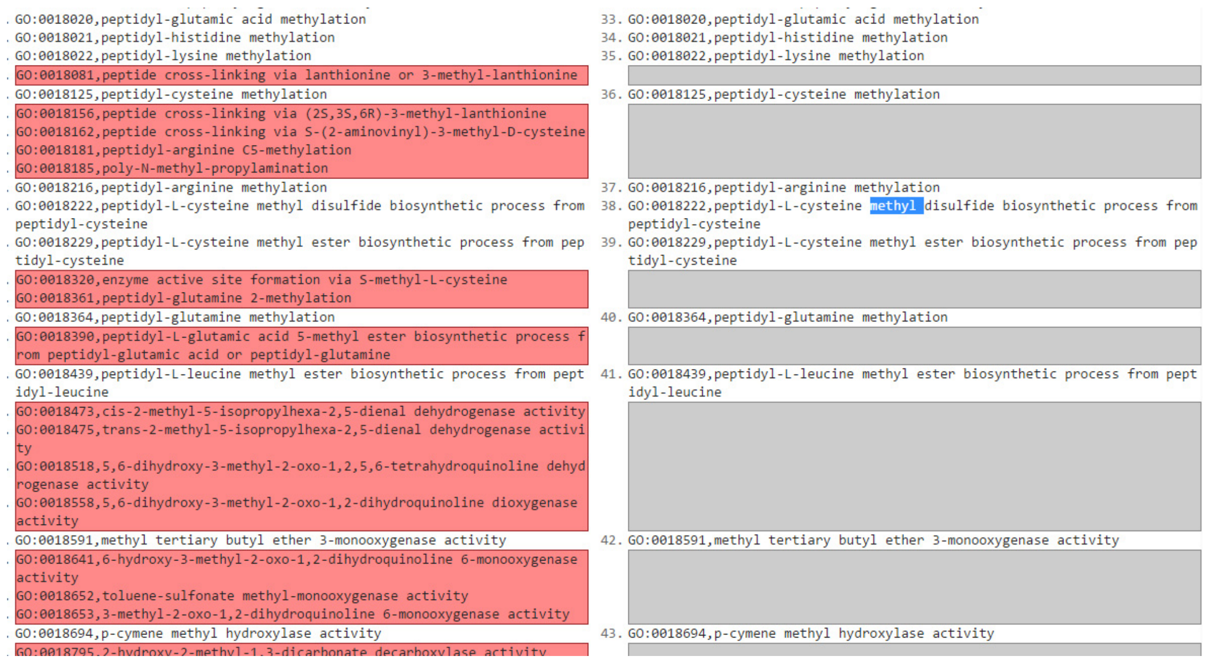
We have compared searching on MongoDB and AllgroGraph for different queries. Table 1 shows the number of terms retrieved in each system for three different queries.

Table 1. Results Comparison between MongoDB and Allegro Graph Database

Search Term	Mongo DB	Allegro Graph Database
intracellular organelle	Step1 number of terms retrieved:170	Step1 number of terms retrieved:171
	Step2 number of terms retrieved:4066	Step2 number of terms retrieved:4052
	Step3 number of terms retrieved:6	Step3 number of terms retrieved:6
	Step4 number of terms retrieved: 0	Step4 number of terms retrieved: 0
methylation	Step1 number of terms retrieved:145	Step1 number of terms retrieved:235
	Step2 number of terms retrieved:299	Step2 number of terms retrieved:381
	Step3 number of terms retrieved:5	Step3 number of terms retrieved:5
	Step4 number of terms retrieved:1	Step4 number of terms retrieved:1
synapse	Step1 number of terms retrieved:42	Step1 number of terms retrieved:42
	Step2 number of terms retrieved:152	Step2 number of terms retrieved:152
	Step3 number of terms retrieved:2	Step3 number of terms retrieved:2
	Step4 number of terms retrieved: 0	Step4 number of terms retrieved: 0

From Table 1, we observed that the MongoDB retrieved more terms in step 1 when compared to Allegro Graph Database for the two query search terms. The difference in result is due to their implementation of full text search in their databases. Fig 5, shows the difference in search results for the search keyword 'methylation' the left side of the image shows the results of mongodb and right side shows the results of allegrograph. Allegro graph unable to retrieve some results that contains "-" symbol.

More comparative studies are still needed. Comparing the execution of query relaxation in two NoSQL databases (MongoDB and AllegroGraph), we have the following preliminary recommendation: When the child nodes in the hierarchy are static (i.e., the graph is static) then we can use the MongoDB because the child items will be calculated in advance and embedded into the corresponding term as JSON, so time needed to retrieve the children will be very fast compared to Allegro Graph database. In other domain where the graph is dynamic then it is better to use AllegroGraph when compared to MongoDB. However, we should keep in mind that in order to use AllegroGraph, we have to know the SPARQL query language to write efficient queries.



```
. GO:0018020,peptidyl-glutamic acid methylation
. GO:0018021,peptidyl-histidine methylation
. GO:0018022,peptidyl-lysine methylation
. GO:0018081,peptide cross-linking via lanthionine or 3-methyl-lanthionine
. GO:0018125,peptidyl-cysteine methylation
. GO:0018156,peptide cross-linking via (2S,3S,6R)-3-methyl-lanthionine
. GO:0018162,peptide cross-linking via S-(2-aminovinyl)-3-methyl-D-cysteine
. GO:0018181,peptidyl-arginine C5-methylation
. GO:0018185,poly-N-methyl-propylamination
. GO:0018216,peptidyl-arginine methylation
. GO:0018222,peptidyl-L-cysteine methyl disulfide biosynthetic process from
  peptidyl-cysteine
. GO:0018229,peptidyl-L-cysteine methyl ester biosynthetic process from pep
  tidyl-cysteine
. GO:0018320,enzyme active site formation via S-methyl-L-cysteine
. GO:0018361,peptidyl-glutamine 2-methylation
. GO:0018364,peptidyl-glutamine methylation
. GO:0018390,peptidyl-L-glutamic acid 5-methyl ester biosynthetic process f
  rom peptidyl-glutamic acid or peptidyl-glutamine
. GO:0018439,peptidyl-L-leucine methyl ester biosynthetic process from pept
  idyl-leucine
. GO:0018473,cis-2-methyl-5-isopropylhexa-2,5-dienal dehydrogenase activity
. GO:0018475,trans-2-methyl-5-isopropylhexa-2,5-dienal dehydrogenase activi
  ty
. GO:0018518,5,6-dihydroxy-3-methyl-2-oxo-1,2,5,6-tetrahydroquinoline dehyd
  rogenase activity
. GO:0018558,5,6-dihydroxy-3-methyl-2-oxo-1,2-dihydroquinoline dioxygenase
  activity
. GO:0018591,methyl tertiary butyl ether 3-monooxygenase activity
. GO:0018641,6-hydroxy-3-methyl-2-oxo-1,2-dihydroquinoline 6-monooxygenase
  activity
. GO:0018652,toluene-sulfonate methyl-monooxygenase activity
. GO:0018653,3-methyl-2-oxo-1,2-dihydroquinoline 6-monooxygenase activity
. GO:0018694,p-cymene methyl hydroxylase activity
. GO:0018795,2-hydroxy-2-methyl-1,3-dicarbonate decarboxylase activity
```

```
33. GO:0018020,peptidyl-glutamic acid methylation
34. GO:0018021,peptidyl-histidine methylation
35. GO:0018022,peptidyl-lysine methylation

36. GO:0018125,peptidyl-cysteine methylation

37. GO:0018216,peptidyl-arginine methylation
38. GO:0018222,peptidyl-L-cysteine methyl disulfide biosynthetic process from
    peptidyl-cysteine
39. GO:0018229,peptidyl-L-cysteine methyl ester biosynthetic process from pep
    tidyl-cysteine

40. GO:0018364,peptidyl-glutamine methylation

41. GO:0018439,peptidyl-L-leucine methyl ester biosynthetic process from pept
    idyl-leucine

42. GO:0018591,methyl tertiary butyl ether 3-monooxygenase activity

43. GO:0018694,p-cymene methyl hydroxylase activity
```

Fig. 5. Search results comparision for the GO Term “methylation”

5. Conclusion

In this paper, started with a brief review of some recent developments in NoSQL biological databases, we addressed the important issues of building NoSQL biological databases with ontologies, as well as how to achieve query relaxation using ontologies. We discussed our methodology, described experiments and presented our results. Our research is complementary to existing studies in NoSQL biological databases. Since this is a new direction of research, a number of important issues are yet to be investigated; for example:

-- Explore how to take advantage of unique features of various prototypes of NoSQL databases (such as document-based or graph-based databases) for query relaxation;

-- Conduct in-depth comparative studies among various prototypes of NoSQL databases;

-- Explore and compare the use of different ontologies;

-- Explore different types of query relaxation, and study the exact role of ontologies in each type;

-- Evaluate the effectiveness of query relaxation (because query relaxation may not always be beneficial or always needed, so there is a need to study under which conditions query relaxation is most useful and effective).

-- Conduct experimental studies to performance issues related to query relaxation.

References

[1]. Elmasri RA, Navathe S. Fundamentals of Database Systems(7/e): Pearson; 2016.
[2]. Shao B, Conrad T. Are NoSQL Data Stores Useful for Bioinformatics Researchers? A comparative study of storing and querying strategies for proteomics mass-spectrometry data. International Journal on Recent and Innovation Trends in Computing and Communication (IJRITCC). 2015; 3(3): p. 1704-1708.
[3]. Have C, Jensen L. Are graph databases ready for bioinformatics? Bioinformatics. 2013; 29(24): p. 3107–3108.
[4]. Wang S, Pandis I, Wu C, He S, Johnson D, Emam I, et al. High dimensional biological data retrieval optimization with NoSQL technology. BMC Genomics. 2014.
[5]. Guimarães V, Hondo F, Almeida R, Vera H, Holanda M, Araujo A, et al. A study of genomic data provenance in NoSQL document-oriented database systems. Bioinformatics and Biomedicine (BIBM)-IEEE. 2015: p. 1525-1531.
[6]. Manyam G, Payton M, Roth J, Abruzzo L, Coombes K. Relax with CouchDB — Into the non-relational DBMS era of bioinformatics. Genomics. 2012; 100(1): p. 1-7.
[7]. TR G. A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition. 1993; 5(2): p. 199-220.
[8]. Guarino N, Oberle D, Steffen S. "What is an ontology?". In Handbook on ontologies.: Springer Berlin Heidelberg, p. 1-17.
[9]. Schulze-Kremer S. Ontologies for molecular biology. Pac. Symp. Biocomput.. 1998; 3: p. 695-706.
[10].Letovsky SI, Cottingham RW, Porter CJ, Li PW. GDB: the Human Genome Database. Nucleic Acids Res. 1998; 26(1): p. 94-99.
[11].Harger C, Skupski M, Bingham J, Farmer A, Hoisie S, P H, et al. The Genome Sequence DataBase (GSDB): improving data quality and data access. Nucleic Acids Res. 1998; 26(1): p. 21-6.
[12].Andreasen T, Henrik B, Rasmus K. On ontology-based querying.
[13].Spasic I, Ananiadou S, McNaught J, Kumar A. Text mining and ontologies in biomedicine: Making sense of raw text. BRIEFINGS IN BIOINFORMATICS. 2005; 6(3): p. 239-251.
[14].Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, et al. Gene ontology: tool for the unification of biology. Nature Genetics. 2000; 25(1): p. 25-29.
[15].Cuppens F, R. Demolombe. Cooperative answering: A methodology to provide intelligent access to databases. Proceedings of the 2nd International Conference on Expert Database Systems. 1988;: p. 621-643.
[16].B. Balla , Z. Chen. Expanding database keyword search for database exploration. Procedia Computer Science. 2013; 17: p. 198-205.
[17].Hill J, Torson J, Guo B, Chen Z. Toward Ontology-guided Knowledge-driven XML Query Relaxation. In Proc. IEEE CIMsim; 2010.
[18].Dong X, Srivastava D. Big Data Integration. VLDB. 2013.
[19].Kantere V, Orfanoudakis G, Kementsietsidis A, T S. Query Relaxation across Heterogeneous Data Sources. Proc. CIKM. 2015;: p. 473-482.
[20].MongoDB. [Online]. Available from: HYPERLINK "www.mongodb.com/" www.mongodb.com/ .
[21].The Gene Ontology Consortium. Gene Ontology Consortium: going forward. Nucl Acids Res. 2015; 43(D1): p. D1049-1056.
[22].Kementsietsidis A, Lim L, Wang. M. Supporting ontology-based keyword search over medical databases. In AMIA 2009 Symposium Proceedings; 2008; NY.
[23].AllegroGraph-Wikipedia. [Online]. Available from: HYPERLINK "https://en.wikipedia.org/wiki/AllegroGraph" https://en.wikipedia.org/wiki/AllegroGraph .
[24]. Stevens R. Ontologies in Bioinformatics. [Online]. Available from: http://www.pdg.cnb.uam.es/BioLink/SpecialInterestTextMining/PRESENTATIONS/Ontologies_in_Bioinformatics.ppt.
[25]. Karp PD. Principles and pitfalls of Ontologies (Definitions, Components, Subtypes). [Online]. Available from: http://www.biopax.org/Docs/2003-02-20_OntologyTutorial.ppt.
[26]. Valeria G, Fernanda H, Rodrigo A, Harley V, Maristela H, Aleteia A, et al. A study of genomic data provenance in NoSQL document-oriented database systems. IEEE BIBM. 2015.
[27]. Ontologies in Bioinformatics: A Survey. [Online]. Available from: http://lsdis.cs.uga.edu/~cthomas/resources/bioont.ppt.