WorkflowHunt: combining keyword and semantic search in scientific workflow repositories

Juan S. B. Diaz
Institute of Computing
University of Campinas
Campinas, Brazil
Email: juan.diaz@students.ic.unicamp.br

Claudia Bauzer Medeiros
Institute of Computing
University of Campinas
Campinas, Brazil
Email: cmbm@ic.unicamp.br

***Abstract*—Scientific datasets, and the experiments that analyze them are growing in size and complexity, and scientists are facing difficulties to share such resources. Some initiatives have emerged to try to solve this problem. One of them involves the use of scientific workflows to represent and enact experiment execution. There is an increasing number of workflows that are potentially relevant for more than one scientific domain. However, it is hard to find workflows suitable for reuse given an experiment. Creating a workflow takes time and resources, and their reuse helps scientists to build new workflows faster and in a more reliable way. Search mechanisms in workflow repositories should provide different options for workflow discovery, but it is difficult for generic repositories to provide multiple mechanisms. This paper presents WorkflowHunt, a hybrid architecture for workflow search and discovery for generic repositories, which combines keyword and semantic search to allow finding relevant workflows using different search methods. We validated our architecture creating a prototype that uses real workflows and metadata from myExperiment, and compare search results via WorkflowHunt and via myExperiment's search interface.**

I. Introduction

The reproducibility crisis in science has become visible. Datasets and data analyses are becoming increasingly more complex and many scientists do not make public these resources; There is a lack of tools to allow reproducibility and communication of data analyses [1]. Nevertheless, some efforts have emerged to help solving this problem, such as The Open Science movement [2]. One particular approach involves the use of scientific workflows.

A scientific workflow is a step by step description of a scientific process for achieving a scientific objective, normally expressed in terms of inputs, outputs, and tasks [3]. Usually, scientists use workflow repositories to publish, store and share workflows with their peers in academia. An increasing number of scientific workflows are potentially relevant for one or more scientific domains [4]. The access to those workflows is usually open to anyone interested in their reuse, re-purpose or experiment replication. Given such availability of workflows, a research question still remains; that is, how can a scientist find and select the most appropriate workflow(s) for reuse by a given experiment?

The problem handled in this paper is the search and discovery of relevant workflows for one or more scientific domains. This problem is important because creating new workflows can be expensive in terms of time and resources. Reuse of relevant workflows helps researchers to build new workflows in a faster and more reliable way.

There are mainly three approaches for search in scientific workflow repositories: keyword-based search, structure-based search, and semantic-based search [4], [5], [6], [7]. Keyword-based search uses the workflow metadata to return workflows associated with the exact words used by scientists in their queries. Structure-based search uses the workflow topology to find workflows with similar structure. Finally, semantic-based search or ontology-based search uses semantic annotations to find workflows where annotations are similar in meaning to the terms in scientists' query. Each approach has advantages and disadvantages. Ideally, search mechanisms should combine these approaches, but this is seldom supported due to the variety of domains involved. For example, keyword-based search is the preferred method offered by workflow repositories because it is easy to use and users are accustomed to using keyword-based search in search engines like Google. Nevertheless, this approach has limitations because of the heterogeneous nature of the terms that are used to refer to the same scientific concept. This causes problems when a scientist wants to search workflows related to a scientific concept, but s/he knows just a subset of the terminology used to refer to that concept and gets partially relevant results. Structure-based search is computationally complex, and hard to perform with heterogeneous sources. While semantic-based search presents more advantages, it usually requires knowing languages with complex syntax like SPARQL [8].

Our approach is a first step towards solving this search problem. This paper presents the design and implementation of a retrieval system for scientific workflow repositories, which allows searching using a keyword-based interface powered by semantic knowledge extracted from domain ontologies. This work was validated with data from myExperiment [9], which is one of the largest scientific workflow repositories at the moment.

Our work expands the search options in scientific workflow repositories. Moreover, our platform improves the interoperability among different scientific domains by standardizing workflow annotations via ontologies.

The rest of this paper is organized as follows: Section II

presents the theoretical foundations and related work; Section III presents the system architecture; Section IV presents details about the implementation; Section V presents a case study and finally, Section VI presents the conclusions and future work.

II. Theoretical Foundation and Related Work

A. Scientific Workflows and Repositories

In science, workflows are "executables" of scientific processes, where these processes mainly refer to computational science simulations and data analyses [4]. Workflows are often represented as Directed Acyclic Graphs (DAG). Many Scientific Workflow Management Systems (SWMS) exploit the DAG representation to offer a user-friendly interface, where the user can model workflows in a high-level manner.

Workflow repositories emerged as a particular solution for the reproducibility problem since they allow scientists to publish and share their workflows. Bioinformatics is recognized for having a long tradition in exchange and publication of scientific resources; for this reason, many scientific workflow repositories contain a large number of bioinformatics workflows. However, other scientific fields are increasing their participation in workflow sharing via repositories – e.g., chemistry [7] or geosciences.

Usually, workflow authors upload their creations to repositories through a process that includes filling a form with some metadata that describes the workflow. Such metadata is often used for search and discovery of relevant workflows. In this paper, we define *workflow metadata* as a dictionary, where the keys are the field names and the values are free text.

We present some representative scientific workflow repositories.

1) MyExperiment [9]: This is a collaborative platform to publish and share scientific workflows. Most users of this platform are scientists in the life sciences; however, other communities (e.g., chemistry, social statistics, and music information retrieval) are expanding their participation in myExperiment. It stores workflows, metadata, data inputs, provenance information, versions, among other assets [9]. Moreover, it supports workflows from different SWMS such as Taverna, RapidMiner, Galaxy, KNIME, Kepler, among others. It allows keyword-based search and filtering by some workflow metadata such as title, description, tags, user, license, etc. Furthermore, it provides structure-based search for workflow discovery using workflows that share the same services as similarity measure.

The myExperiment project hosts 392 scientific groups, 10.501 registered users and more than 2.800 scientific workflows in its database (in May, 2017). Since its creation, it has expanded the scientific domains that use it and has been used in several research projects [6], [5], [10].

2) CrowdLabs [11]: This is a collaborative environment for scientists inspired by social websites. This platform works with workflows in VisTrails and VisMashup [11]. VisTrails is an open-source SWMS with a particular focus on data analysis and visualization, and VisMashup is a VisTrails extension to allow user interaction with workflows using a web-based solution [11].

3) Galaxy [12]: This is a genomic research platform to handle computational biology experiments. It supports computation analysis, the capture of provenance data from in silico experiments, and workflow annotation. Furthermore, this platform allows interaction with the research components (e.g., datasets, analysis, workflows, and annotations) using a web-based format [12]. This repository provides keyword search based on workflow metadata (name, owner, annotation, and tags). As downside, it just supports workflows from its SWMS.

4) CloudFlows [13]: This is a social platform for creation, execution, and sharing of data mining workflows. Users use web browsers to develop their workflows without installing specific software to execute them [13]. It does not provide search but allows workflow discovery by listing all the workflows in the platform, and just supports workflows from its SWMS.

5) PBase: This repository allows storing provenance data using ProvONE, which is a standard for modeling, representing and sharing provenance information [14]. PBase can store workflows (from VisTrails, Taverna, and Kepler), prospective provenance, and retrospective provenance [14]. Moreover, it provides keyword-based search and structure-based search using neo4j graph DBMS and Cypher as the query language.

6) OPMW Workflow Repository [15]: Unlike other repositories, it stores both executable workflows and abstract workflows, which allows independence from a particular execution environment [15]. Abstract workflows describe the workflow components and are human readable to make workflows more understandable; therefore, more reusable [15]. Anyone can use the workflows in this repository without problems; however, just users with knowledge in Wings can contribute to it [15]. It provides semantic search via an SPARQL endpoint using workflow metadata such as descriptions of the inputs, processes, and outputs and workflow execution provenance traces such as intermediate results, outputs, software codes.

B. Ontologies and Semantic Annotations

An ontology is "an explicit specification of a conceptualization" [16], where a conceptualization is an abstract perspective about something (e.g., objects, concepts, and relationships). Explicit specification means the definition of a formalism to map the abstract meaning of conceptualization in something concrete [16]. Ontologies also include the relationships among the concepts they contain [17]. An *ontology concept* is unique in an ontology and is often known as ontology class. An *ontology term* is a string that identifies an ontology concept – e.g., synonyms and labels. Consequently, an ontology concept is often linked to several ontology terms [18].

Superclasses represent high-level concepts and subclasses represent more detailed concepts [17]. Such relationships can be represented using a graph (see Fig. 1) or a hierarchical structure (see Fig. 2). Both show a part of the EDAM ontology. EDAM is an ontology of bioinformatics information, including operations, types of data, topics, and formats [19].

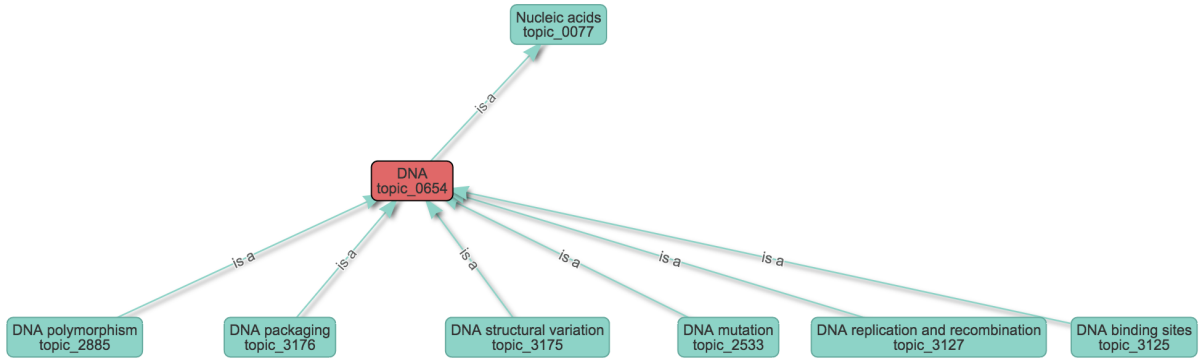

Fig. 1. Partial graph representation of EDAM Ontology via Ontology Lookup Service

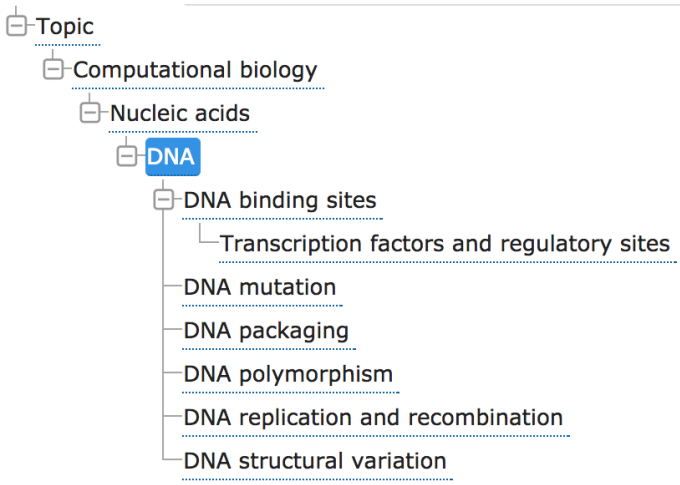

Fig. 2. Partial hierarchy representation of EDAM Ontology via Ontology Lookup Service

An annotation is a particular kind of data that describe other data [20]. For Macario et al. [20] semantic annotations are annotations that use ontologies to eliminate ambiguities and promote a common understanding of concepts. In this paper, we will borrow the formal definition of semantic annotation from Oren et al. [21]: a semantic annotation is a tuple (s, p, o, c), where s is the subject of annotation (in our case, a term in a domain ontology), o is the object of annotation (in our case, a workflow), p is the predicate (the relationship between s and o), and c is the context (provenance information) [21]. For us, $p =$ "$occurs - in$".

C. Workflow Retrieval

As mentioned in Section I, the retrieval process for workflow repositories can be performed via a wide range of methods; some of these methods are keyword-based search, topology-based search (also known as structure-based search), and semantic-based search.

In more detail, keyword-based retrieval systems use a free text query to match against subsets of metadata (e.g., the title, descriptions, and tags) for each workflow in the repository. Keyword-based methods are widely implemented on scientific workflow repositories – e.g., myExperiment [22]. Another example is presented in [23], which implements a keyword-based search engine for workflows collected from Kepler, Taverna, and myGrid. This search method has good precision, retrieving relevant workflows that match the scientist's query. The downside is that in some cases it has a low recall because some relevant results are hidden. Scientists may use different words to refer to the same concepts, and annotate a workflow according to distinct criteria. Hence, in this case, this method just returns a subset of all the relevant results for the user's query.

Topology-based retrieval systems use the workflow structure to search for workflows with similar topology in the repository. Workflows regularly use three elements in the DAG representation: the inputs, the outputs, and the processes. The elements are used individually or collectively to perform a search based on the workflows that contain the same workflow elements. For example, the work in [5] presents a method that takes into account the order of each element; the method uses a topological sort, a topological comparison, and a normalization of the results to obtain a better accuracy. This search method is a good choice for workflow discovery. Nevertheless, it needs a workflow or a subset of the workflow components as input to return relevant results. Topology alone is not a good choice in searching, and has to be accompanied by e.g., keyword-based search. Moreover, it requires expensive preprocessing of workflows in a repository.

Semantic-based retrieval systems use ontologies to match queries against the semantic annotations in the workflow repositories. This method has the higher recall and precision among the search methods. Nevertheless, it usually needs languages with complex syntax (like SPARQL) that represent a barrier for the user [8]. Some works try to combine this method with others; for example, the work in [4] uses a hybrid approach between structure-based search and semantic-based search. It presents a method for similarity search in a few semantic workflows using a graph representation and an A* search algorithm to perform the similarity search.

Our approach differs from the mentioned above by propos-

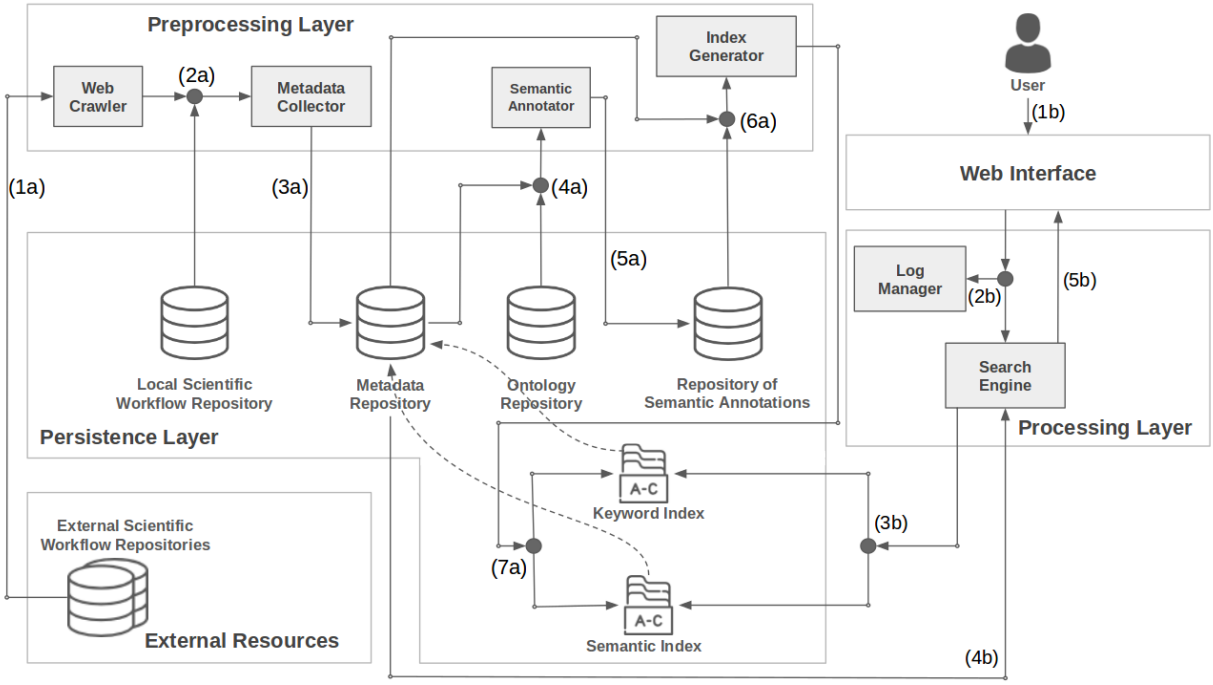

Fig. 3. Retrieval System Architecture for semantic search.

ing a retrieval system architecture that uses a hybrid search method combining keyword-based and semantic-based search. Our architecture focuses on expanding the search options, improving workflow interoperability among different scientific domains by using ontologies to semantically annotate workflows, and keep a standard terminology for different scientific fields. Our solution allows workflow reuse and re-purpose from different scientific fields.

III. System Architecture

The architecture of our retrieval system is inspired by the work in [8] and [24]. Fig. 3 shows this architecture, which comprises three layers. The Persistence Layer is composed by four repositories: Local Workflow Repository, Metadata Repository, Ontology Repository, and Repository of Semantic Annotations. It also includes the keyword index and the semantic index. The Preprocessing Layer prepares data for subsequent search, which is performed by the Processing Layer. The latter provides all the services needed to look for workflows using Keyword search or Semantic search. These services can be accessed through the Web Interface (scientists who are looking for workflows of interest). Finally, solid arrows represent the data flow in the system, and dashed arrows represent data connections among elements.

Our system requires preparing the repositories, so users can perform a search. This preparation is performed offline. It starts when the Metadata Collector extracts metadata from external (1a) and internal (2a) repositories. External workflow data requires a Web Crawler; scientists can also take advantage of workflows created by their own research team, stored in the Local Workflow Repository. Metadata is stored in the Metadata Repository (3a). Next, the Semantic Annotator semantically annotates the metadata using the ontologies provided by the Ontology Repository (4a). These semantic annotations are stored in the Repository of Semantic Annotations (5a). The Index Generator then creates indices for metadata (keyword index) and semantic annotations (6a-7a). We use an inverted index for keyword-based search and a modified version of the same index data structure for semantic search (see Subsection III-B). Both indices refer to the metadata, which contains the workflow identifier (see Fig. 7). Once annotations and metadata repositories are created and indices are constructed, the system is ready to handle queries using keyword and semantic search. Each new workflow processed needs to go through this process. If new ontologies are added, steps (4a) onwards are executed.

A typical retrieval scenario starts when a user poses a query through the Web Interface (1b). It then redirects the query to the Search Engine (2b); the Log Manager stores the query for performance management (2b). The Search Engine executes the query using the keyword and semantic indices (3b). These indices are linked to the workflow metadata in the Metadata Repository. The result is a list of links pointing at the relevant workflows, presented to the user at the interface (4b-5b).

Storing and indexing semantic annotations is a requirement for query processing in our system. Usually, these tasks tend to be expensive in time and computational resources, but they improve search quality. Moreover, they are calibrated/executed with less frequency than queries. Subsection III-A presents algorithms to create semantic annotations based on detection of ontology concepts in the values of the metadata fields of workflows. Subsection III-B present details about the semantic

inverted index . Finally, Subsection III-C presents the semantic search.

A. Semantic Annotations

Our system uses the Semantic Annotator to create the semantic annotations by combining the workflow metadata in the Metadata Repository and the ontologies available in the Ontology Repository. The Semantic annotator uses the SEMANTIC_ANNOTATION algorithm (see Fig. 4), which is inspired by the Open Biomedical Annotator [18]. Our algorithm receives as input the metadata of workflows to be annotated, a list of ontologies (including ontology terms, ontology concepts, and relationships), and a list of options for semantic expansion. It stores the resulting semantic annotations in the Repository of Semantic Annotations. We create semantic annotations associating many ontology concepts with a workflow through the $occurs-in$ relationship. C represents the context or provenance information such as creation date of the annotation.

The algorithm starts by creating a dictionary of ontology terms, which includes labels and synonyms (lines 4-12). Next, the algorithm iterates over the list of workflow metadata and extracts the text that belongs to each metadata field. This text is used as input for the TEXT_ANNOTATION algorithm (see Fig. 5), which identifies the ontology concepts that are in the text and creates semantic annotations (including annotations obtained by semantic expansion - lines 13-19). Finally, the semantic annotations are stored in the Repository of Semantic Annotations.

In more detail, the TEXT_ANNOTATION algorithm (see Fig. 5) receives as input a text, a dictionary of ontology terms (including labels and synonyms), and a list of options for semantic expansion. Its output is a list of semantic annotations with the ontology concepts detected in the input text and their corresponding semantically expanded annotations according to the input options. The algorithm starts by sorting the dictionary of ontology terms by the length of the ontology terms in descending order (line 3). We give priority to larger strings because they probably have more semantic content than shorter strings. Moreover, we should avoid overlapping semantic annotations in the text as suggested in [25]. For example, consider a string "a nucleic acid sequence" that belongs to some metadata value of a workflow. If our system annotates that string with the ontology term "nucleic acid sequence" (EDAM:data_2977), then it should not create an annotation with the ontology term "sequence" (EDAM:data_2044) on the same string because this causes an overlap. Overlap can be avoided by replacing reoccurring terms by blanks. A given substring cannot be annotated by more than one ontology term, thus avoiding overlapping (lines 4-13). Finally, the algorithm returns semantic annotations with the corresponding semantic expansion (line 14).

The SEM_EXPANSION algorithm (see Fig. 6) expands an initial semantic annotation using the super/sub class structure of the ontologies. It receives as input a semantic annotation and a list of options for semantic expansion and returns a

Require: W is the metadata of workflows to be annotated, O is a list of Ontologies, and Opt is a list of options for semantic expansion

```
1:  function SEMANTIC_ANNOTATION(W, O, Opt)
2:      D ← ∅
3:      SA ← ∅
4:      for each ontology ∈ O do
5:          for each term ∈ ontology do
6:              label ← term.label
7:              D.add(label)
8:              for each synonym ∈ term do
9:                  D.add(synonym)
10:             end for
11:         end for
12:     end for
13:     for each wm ∈ W do
14:         for each field ∈ wm.fields() do
15:             t ← wm[field].getValue()
16:             sa ← TEXT_ANNOTATION(t, D, Opt)
17:             SA.add(sa)
18:         end for
19:     end for
20:     storeAnnotations(SA)
21: end function
```

Fig. 4. Semantic Annotation Algorithm

Require: T is the free text, D is a list of ontology terms, and Opt is a list of options for semantic expansion

```
1:  function TEXT_ANNOTATION(T, D, Opt)
2:      SA ← ∅
3:      D ← sortByLength(D)
4:      for each d ∈ D do
5:          if d ∈ T then
6:              sa ← (d, occurs − in, T, C)
7:              SA.add(sa)
8:              se ← SEM_EXPANSION(sa, Opt)
9:              SA.add(se)
10:             w ← createWildcard(d, WHITE_SPACE)
11:             replace(T, d, w)
12:         end if
13:     end for
14:     return SA
15: end function
```

Fig. 5. Semantic Annotations from Free Text Algorithm

Require: sa is a semantic annotation and Opt is a list of options for semantic expansion

1: **function** SEM_EXPANSION(sa, Opt)
2: $SE \leftarrow \emptyset$
3: **if** $Opt.has\_semantic\_generalization$ **then**
4: $g \leftarrow generalization(sa, Opt.g\_size, C)$
5: $SE.addList(g)$
6: **end if**
7: **if** $Opt.has\_semantic\_specialization$ **then**
8: $s \leftarrow specialization(sa, Opt.s\_size, C)$
9: $SE.addList(s)$
10: **end if**
11: **if** $Opt.has\_sdist\_expansion$ **then**
12: $sde \leftarrow sdist\_expansion(sa, Opt.sd\_size, C)$
13: $SE.addList(sde)$
14: **end if**
15: **return** SE
16: **end function**

Fig. 6. Semantic Expansion Algorithm

list of semantic annotations expanded from the original one. Options are generalization (lines 3-6), specialization (lines 7-10), and semantic distance expansion (lines 11-14). Generalization (resp. specialization) creates new semantic annotations with the same structure of the initial semantic annotation but replacing the ontology concept associated with its superclasses (resp. subclasses) in the ontology. The parameter g_size represents the number of superclasses that will be included in the semantic expansion (resp. s_size for number of subclasses). Semantic distance expansion creates new semantic annotations with the same structure of the initial semantic annotation but replacing the ontology concept associated with its neighbors in the ontology. The parameter sd_size represents the number of edges that separate the original ontology concept from the ontology concepts that will be included in the expansion.

For example, in the case of generalization, consider a metadata string "...gene ids for that chromosome" that belongs to the description of a workflow with title "chicken_ensembl_gene_id"[1]. Our system detects the ontology term "chromosome" that belongs to the ontology concept EDAM:topic_0654 and creates a semantic annotation $(EDAM : topic\_0654, occurs-in, myExperiment : 902, \{date : \text{“}2017-05-04\text{”}\})$. Next, the system finds that EDAM:topic_0077 is the superclass of EDAM:topic_0654 and creates another semantic annotation $(EDAM : topic\_0077, occurs-in, myExperiment : 902, \{date : \text{“}2017-05-04\text{”}\})$, which is result of replacing the ontology concept in the initial semantic annotation by one of its superclasses in the ontology (in this case, the immediate superclass).

[1]http://www.myexperiment.org/workflows/902

B. Semantic Indexing

This subsection provides details about the construction of semantic inverted indexes. Our algorithm is a modification of the general algorithm for creating an inverted index proposed by Manning et al [26] and works as follows:

- Collect the workflow metadata.
- Remove the strings that represent an ontology concept in the workflow metadata and include a list of ontology concept identifiers extracted from the semantic annotations.
- Tokenize the text in each value of the workflow metadata fields. This step is done by splitting the text in words and considering each word as a token. In this step, ontology concept identifiers are considered as words.
- Perform NLP preprocessing. This step is necessary to decrease the number of tokens to be processed, e.g., stemming, removing punctuation, removing stop words, converting strings to lowercase.
- Index the workflow metadata associating each term to a list of workflow identifiers. If a workflow identifier is in the list, it means the term is contained in the workflow metadata.

Though similar to the algorithm proposed by Manning et al [26], we introduced a step before tokenizing the text because we need to include semantics in the inverted index. The inverted index is considered to be one of the most efficient structures for supporting text search [26].

In addition to the advantages offered by the semantic search, replacing strings by their ontology concept identifiers reduces the size of the dictionary used in the inverted index because many words can be mapped into one ontology concept. Moreover, some ontology terms contain more than one word. The next subsection explains our semantic search and its interaction with the semantic inverted index.

C. Semantic Search

The semantic search algorithm uses a two-step approach:

Step 1. The algorithm receives the same dictionary of ontology terms used to semantically annotate the workflow metadata. After that, the algorithm uses the dictionary to find ontology concepts in the user query. If some ontology concept is detected, then it is replaced by the ontology concept identifier.

Step 2. The algorithm tokenizes the query and uses the semantic inverted index to find the workflows associated with the tokens in the query. Moreover, it ranks the results using TF-IDF.

This search approach allows to find relevant workflows, even when the user uses words that are not present in the workflow metadata.

IV. Implementation

We developed the WorkflowHunt prototype (available at http://www.workflowhunt.com) with MySQL as the underlying DBMS, and myExperiment as the External Repository. We use EDAM and CHEMINF ontologies to semantically annotate myExperiment workflow metadata. EDAM is an ontology

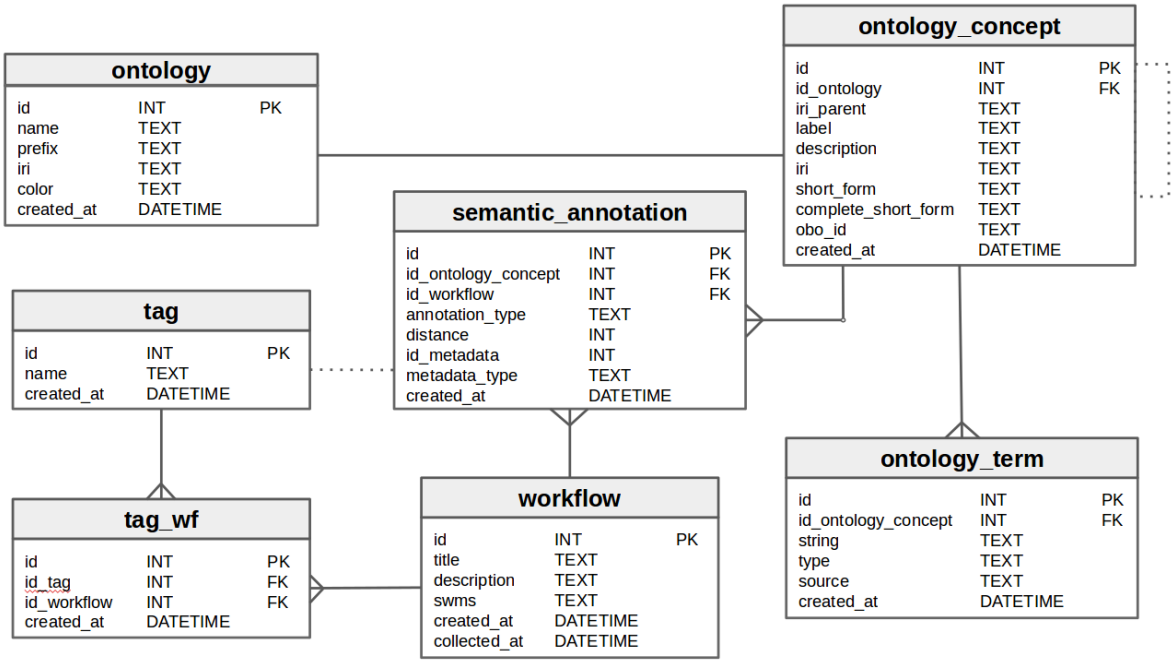

Fig. 7. Database Schema – indices omitted.

for bioinformatics information, including operations, types of data, topics, and formats [19]. CHEMINF is an ontology for chemical information, including terms, and algorithms used in chemistry [27]. Fig. 7 shows the repositories' schemas. Tables ("ontology", "ontology_concept", and "ontology_term") belong to the Ontology Repository. Tables ("tag", "tag_wf", and "workflow") belong to the Metadata Repository.

The ontology concepts and terms (see Fig. 7) were collected using the Ontology Lookup Service [28]. This service allows querying, browsing, and navigating over a database that integrates several biomedical ontologies and related vocabulary. We used WordNet [29] as an additional source of synonyms to complement the ontology terms collected via Ontology Lookup Service. Ontology concepts are associated to one ontology and include the Internationalized Resource Identifier (IRI) and the IRI of its superclass for eventual generalization of semantic annotations. Ontology terms are stored in table "ontology_term" and are associated with one ontology concept. Such terms include a text, the term type (e.g., synonym, or labels), and the source (e.g., Ontology Lookup Service, or WordNet).

MyExperiment provides an API[2] to collect the workflow metadata (title, description, tags, etc.) from its repository in XML format. The Web Crawler (see Fig. 3) collects such information and the Metadata Collector transforms the raw metadata to store a subset of the available metadata (id, title, description, tags, and SWMS) in the table "workflow" (see Fig. 7). However, tags are stored in the "tag" table and they are linked to the "workflow" table via the "tag_wf" table.

[2]http://wiki.myexperiment.org/index.php/Developer:API

The Semantic Annotator uses the SEMANTIC_ANNOTATION algorithm, which takes input data from "ontology_term", "workflow", "tag_wf", and "tag" tables to populate the "semantic_annotation" table. The "semantic_annotation" table stores the metadata type of the field (e.g., title, description, or tag), the annotation type – e.g., direct annotation or semantic expansion (generalization, specialization, or distance expansion). Additionally, the system stores the distance between the original ontology concept detected and the ontology concept used in the annotation. Distance equals zero for direct annotations.

We used ElasticSearch to create and store our inverted index and semantic inverted index. ElasticSearch [30] is a project based on Lucene and provides distributed and scalable search using inverted indexes, and full-text search or term-based search. ElasticSearch manages our Index Generator, and Search Engine. The indices link to the information in the "workflow" table.

The Search Interface was built with PHP (framework CodeIgniter) and Bootstrap to allow testing our system in a user-friendly manner. Finally, logs are stored in a MySQL database, recording the interactions between the user and our system for future improvements.

Fig. 8 shows a semantic search of the query "chromosomes" in WorkflowHunt, which retrieves 77 workflows. Tabs "Keyword" and "Semantics", respectively allow the scientist to execute keyword or semantic search. For each workflow retrieved, WorkflowHunt shows the semantic annotations by clicking on "READ MORE". Moreover, scientists can compare the results by clicking on the "Compare results" link. This feature was used to facilitate the analysis of results in the case study.

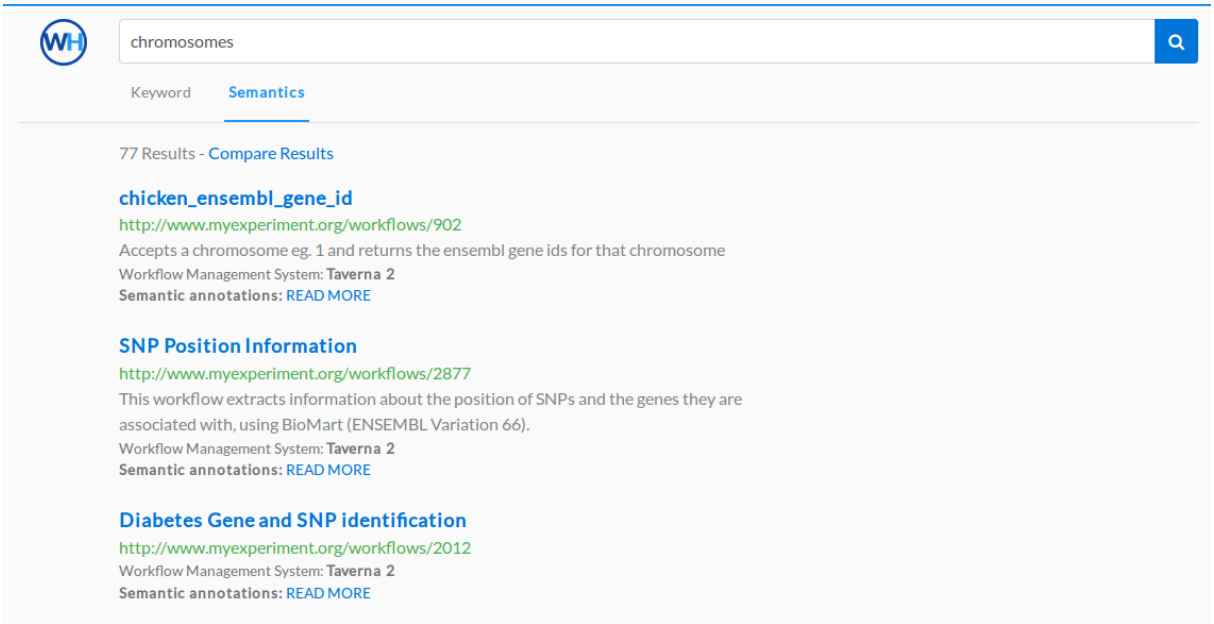

Fig. 8. Semantic-based search results.

V. Case Study

The case study is based on real data collected from the myExperiment repository through the process explained in Section IV. We implemented the system using two ontologies (CHEMINF and EDAM). Those ontologies contain 3,045 ontology concepts linked to a dictionary of 5,057 ontology terms. After the annotation process, we got 27,697 semantic annotations (51.5% by direct annotation and 48.5% by semantic expansion - generalization). Such annotations were extracted from 15,054 metadata fields (title, description, and tags) that belong to 2,873 workflows.

We compared WorkflowHunt (keyword search and semantic search) with the retrieval system provided by myExperiment (see Table I and Table II) to analyze the cases for which it is convenient to use each system. We show a small subset of the comparison tables for space reasons; however, such tables keep the gist of our findings. For an extended version of the comparison tables, please visit the GitHub of the project [3].

Table I shows a general comparison about the number of workflows returned by each system given a query. Table II presents a fine-grained comparison to know the number of workflows that are returned by one system and others not. Both tables allow us to analyze different cases where a system outperforms the others.

Let q be a scientist's query, and A, B, and C sets of workflows defined as follows:

A = set of workflows retrieved using the keyword search provided by myExperiment given the query q

B = set of workflows retrieved using the keyword search provided by WorkflowHunt given the query q

C = set of workflows retrieved using the semantic search provided by WorkflowHunt given the query q

The search comparisons allow us to analyze six cases:

Case 1. $(A - B) \neq \emptyset$: This case occurs when given a query, a search in myExperiment returns workflows that the keyword search in WorkflowHunt does not (queries 2, 3 and 4, see table II). The main reason is that myExperiment indexes more metadata than WorkflowHunt[4]. For example, consider query 2: "Ecology", which produces 5 workflows that belong to $(A - B)$. After an analysis, we found that in most of them like the "Age specific analysis"[5] workflow contained the string "... Journal of Ecology" in the descriptions of some workflow inputs. At the moment, WorkflowHunt just indexes title, description, and tags.

Case 2. $(B - C) \neq \emptyset$: This case occurs when a keyword search in WorkflowHunt returns workflows that the semantic search in WorkflowHunt does not (queries 2 and 3, see table II). The main reason is that the semantic search in WorkflowHunt replaces substrings in the query that are detected as ontology concepts. Such substrings can have more than one word, which means that a semantic search query can have fewer tokens than a keyword search in WorkflowHunt. Moreover, keyword search in WorkflowHunt does not consider the position of the words in the query, and semantic search in WorkflowHunt does. For example, consider the query "Pathway simulation", which produces 140 workflows that belong to $(B - C)$. We found that there are no semantic annotations in WorkflowHunt associated to "Pathway simulation" (ontology concept EDAM:operation_3562). Nevertheless, keyword search in WorkflowHunt returned 140 workflows because the

[3]https://github.com/jbeleno/workflowhunt

[4]https://goo.gl/STIQWR

[5]http://www.myexperiment.org/workflows/3286.html

TABLE I
COMPARISON AMONG DIFFERENT SEARCH APPROACHES

Query Identifier	Query	Search Methods		
		$\|\|A\|\|$ = ‖myExp. search results‖	$\|\|B\|\|$ = ‖WH keyword search results‖	$\|\|C\|\|$ = ‖WH semantic search results‖
1	Chromosomes	1	1	77
2	Ecology	24	19	42
3	Pathway simulation	19	140	0
4	Enzymes	16	8	11

TABLE II
DESCRIPTION OF RESULTS AMONG DIFFERENT SEARCH APPROACHES

	Query Identifier			
Set description	**1**	**2**	**3**	**4**
$\|\|A \cap B\|\|$	1	19	17	8
$\|\|B \cap C\|\|$	1	18	0	8
$\|\|C \cap A\|\|$	1	21	0	9
$\|\|A \cap B \cap C\|\|$	1	18	0	8
$\|\|A - B\|\|$	0	5	2	8
$\|\|A - C\|\|$	0	3	19	7
$\|\|B - C\|\|$	0	1	140	0
$\|\|C - B\|\|$	76	24	0	3
$\|\|C - A\|\|$	76	21	0	2
$\|\|B - A\|\|$	0	0	123	0
$\|\|A - (B \cup C)\|\|$	0	2	2	7
$\|\|B - (C \cup A)\|\|$	0	0	123	0
$\|\|C - (A \cup B)\|\|$	76	21	0	2

metadata of those workflows contains the words "Pathway" and "simulation", but such words could appear in different order, in different metadata fields of the same workflow or could be separated by several words.

Case 3. $(A - C) \neq \emptyset$: This case occurs when a keyword search in myExperiment returns workflows that the semantic search in WorkflowHunt does not. This case summarizes the reasons given in Case 1 and 2: myExperiment indexes more metadata fields, semantic search in WorkflowHunt uses fewer tokens than keyword search, and keyword search in myExperiment has lax rules about the position of the words given in the query to match against the workflow metadata. This case occurs e.g., with queries "Pathway simulation" and "Enzymes".

Case 4. $(B - A) \neq \emptyset$: This case occurs when a keyword search in WorkflowHunt returns workflows that the keyword search in myExperiment does not. The main reason is that WorkflowHunt has a lower threshold in the number of words that should be matched against the workflow metadata than the myExperiment keyword search. This case occurs with the query "Pathway Simulation", where 123 workflows belong to $(B-A)$. For example, the workflow "Rank Phenotype Terms"[6] is in the set because it has the word "pathway" in the workflow metadata. However, it does not have the word "simulation" in the metadata.

Case 5. $(C - A) \neq \emptyset$: This case occurs when a semantic search in WorkflowHunt returns workflows that the keyword search in WorkflowHunt does not. The main reason is that semantic search in WorkflowHunt allows finding concepts with similar meaning and not just keywords. Therefore, the results are semantically similar to what is searched. For example, consider the query "chromosomes", where keyword search in myExperiment returns 1 workflow. The string "chromosomes" represents the ontology concept EDAM:topic_0654 that is linked to many ontology terms: "DNA", "Ancient DNA", "DNA analysis", "Chromosomes", "deoxyribonucleic acid", "desoxyribonucleic acid", and "chromosome". Hence, semantic search in WorkflowHunt finds workflows that match with such ontology terms in the workflow metadata, returning 77 workflows.

Case 6. $(C - B) \neq \emptyset$: This case occurs when a semantic search in WorkflowHunt returns workflows that the keyword search in WorkflowHunt does not. The reason and example are the same than the presented in Case 5.

The cases where the difference between sets is empty ($(A-B) = \emptyset$, $(A-C) = \emptyset$, $(B-C) = \emptyset$, $(C-B) = \emptyset$, $(C-A) = \emptyset$, and $(B - A) = \emptyset$,) are not interesting because the systems return the same set of workflows for a query.

Some of the conclusions given about set A are based on reverse engineering applied in myExperiment and a study of the source code of that website[7].

VI. CONCLUSIONS AND FUTURE WORK

This paper presented the design and implementation of the WorkflowHunt retrieval system for scientific workflow repositories, which allows searching using a keyword-based interface powered by semantic knowledge extracted from domain ontologies. This system contributes to helping scientists in workflow reuse and re-purpose, and experiment replication, expanding the search options in scientific workflow repositories.Our implementation is validated on the 2,873 myExperiment workflows.

[6]http://www.myexperiment.org/workflows/854.html
[7]https://github.com/myExperiment/myExperiment

There remains the following question – which is best, WorkflowHunt or dedicated repository interfaces? First, WorkflowHunt is generic, and thus can be used for other repositories that publish their metadata (e.g., via an API) – and thus supports hunt across a set of repositories. Second, our case study section just comments on the number of retrieved workflows, but not in their suitability to the query – for this, we need the experts that submitted the query to answer that question. Third, the flexibility provided by semantic search allows tailoring queries to distinct domains – even to the point of letting scientists provide their own customized domain ontology.

WorkflowHunt is online and there is space for improvement. For example, the inclusion of a knowledge base per scientific domain or the insertion of instances in ontologies allows creation of more complex indexes and queries like those implemented in Broccoli [31], [32]. For semantic annotations, part of our future work involves interacting with domain experts to better choose the appropriate ontology terms to use in those annotations. Also, we can use NLP and machine learning to automatically detect the scientific domain of a workflow and use the correct ontology to annotate it. Finally, we can use the approach in [10] to take advantage of workflow structure and create automatic annotations with similar structure as presented in [4].

We also need to validate system use with domain experts.

ACKNOWLEDGMENT

Work partially financed by CNPq (305110/2016-0), FAPESP CCES (2013/08293-7), CNPq/INCT in Web Science (557128/2009-9) and CAPES (07/2016 - 1629106).

REFERENCES

[1] R. Peng, "The reproducibility crisis in science: A statistical counterattack," *Significance*, vol. 12, no. 3, pp. 30–32, 2015.

[2] T. Hey and M. C. Payne, "Open science decoded," *Nature Physics*, vol. 11, no. 5, pp. 367–369, 2015.

[3] B. Ludäscher, S. Bowers, and T. McPhillips, "Scientific workflows," *Encyclopedia of Database Systems*, pp. 2507–2511, 2009.

[4] R. Bergmann and Y. Gil, "Similarity assessment and efficient retrieval of semantic workflows," *Information Systems*, vol. 40, pp. 115–127, 2014.

[5] J. Starlinger, S. Cohen-Boulakia, S. Khanna, S. B. Davidson, and U. Leser, "Effective and efficient similarity search in scientific workflow repositories," *Future Generation Computer Systems*, vol. 56, pp. 584–594, 2016.

[6] J. Starlinger, B. Brancotte, S. Cohen-Boulakia, and U. Leser, "Similarity search for scientific workflows," *Proceedings of the VLDB Endowment*, vol. 7, no. 12, pp. 1143–1154, 2014.

[7] L. A. Carvalho, K. Belhajjame, and C. B. Medeiros, "Converting scripts into reproducible workflow research objects," in *e-Science (e-Science), 2016 IEEE 12th International Conference on*. IEEE, 2016, pp. 71–80.

[8] S. Kara, Ö. Alan, O. Sabuncu, S. Akpınar, N. K. Cicekli, and F. N. Alpaslan, "An ontology-based retrieval system using semantic indexing," *Information Systems*, vol. 37, no. 4, pp. 294–305, 2012.

[9] D. De Roure, C. Goble, and R. Stevens, "The design and realisation of the virtual research environment for social sharing of workflows," *Future Generation Computer Systems*, vol. 25, no. 5, pp. 561–567, 2009.

[10] B. García-Jiménez and M. D. Wilkinson, "Automatic annotation of bioinformatics workflows with biomedical ontologies," in *International Symposium On Leveraging Applications of Formal Methods, Verification and Validation*. Springer, 2014, pp. 464–478.

[11] P. Mates, E. Santos, J. Freire, and C. T. Silva, "Crowdlabs: Social analysis and visualization for the sciences," in *International Conference on Scientific and Statistical Database Management*. Springer, 2011, pp. 555–564.

[12] J. Goecks, A. Nekrutenko, and J. Taylor, "Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences," *Genome biology*, vol. 11, no. 8, p. R86, 2010.

[13] J. Kranjc, V. Podpečan, and N. Lavrač, "Clowdflows: a cloud based scientific workflow platform," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2012, pp. 816–819.

[14] V. Cuevas-Vicenttín, P. Kianmajd, B. Ludäscher, P. Missier, F. Chirigati, Y. Wei, D. Koop, and S. Dey, "The pbase scientific workflow provenance repository," *International Journal of Digital Curation*, vol. 9, no. 2, pp. 28–38, 2014.

[15] D. Garijo and Y. Gil, "A new approach for publishing workflows: abstractions, standards, and linked data," in *Proceedings of the 6th workshop on Workflows in support of large-scale science*. ACM, 2011, pp. 47–56.

[16] T. R. Gruber *et al.*, "A translation approach to portable ontology specifications," *Knowledge acquisition*, vol. 5, no. 2, pp. 199–220, 1993.

[17] L. Yu, *A developers guide to the semantic Web*. Springer Science & Business Media, 2011.

[18] C. Jonquet, N. Shah, and M. Musen, "The open biomedical annotator," in *AMIA summit on translational bioinformatics*, 2009, pp. 56–60.

[19] J. Ison, M. Kalaš, I. Jonassen, D. Bolser, M. Uludag, H. McWilliam, J. Malone, R. Lopez, S. Pettifer, and P. Rice, "Edam: an ontology of bioinformatics operations, types of data and identifiers, topics and formats," *Bioinformatics*, vol. 29, no. 10, pp. 1325–1332, 2013.

[20] C. G. N. Macário, S. R. de Sousa, and C. B. Medeiros, "Annotating geospatial data based on its semantics," in *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2009, pp. 81–90.

[21] E. Oren, K. Möller, S. Scerri, S. Handschuh, and M. Sintek, "What are semantic annotations," *Relatório técnico. DERI Galway*, vol. 9, p. 62, 2006.

[22] C. A. Goble, J. Bhagat, S. Aleksejevs, D. Cruickshank, D. Michaelides, D. Newman, M. Borkum, S. Bechhofer, M. Roos, P. Li *et al.*, "myexperiment: a repository and social network for the sharing of bioinformatics workflows," *Nucleic acids research*, vol. 38, no. suppl 2, pp. W677–W682, 2010.

[23] Q. Shao, P. Sun, and Y. Chen, "Wise: A workflow information search engine," in *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*. Ieee, 2009, pp. 1491–1494.

[24] M. Fernandez, V. Lopez, M. Sabou, V. Uren, D. Vallet, E. Motta, and P. Castells, "Semantic search meets the web," in *Semantic Computing, 2008 IEEE International Conference on*. IEEE, 2008, pp. 253–260.

[25] H. Bast, B. Buchhold, E. Haussmann *et al.*, "Semantic search on text and knowledge bases," *Foundations and Trends® in Information Retrieval*, vol. 10, no. 2-3, pp. 119–271, 2016.

[26] C. D. Manning, P. Raghavan, H. Schütze *et al.*, *Introduction to information retrieval*. Cambridge university press Cambridge, 2008, vol. 1, no. 1.

[27] J. Hastings, L. Chepelev, E. Willighagen, N. Adams, C. Steinbeck, and M. Dumontier, "The chemical information ontology: provenance and disambiguation for chemical data on the biological semantic web," *PloS one*, vol. 6, no. 10, p. e25513, 2011.

[28] R. G. Côté, P. Jones, R. Apweiler, and H. Hermjakob, "The ontology lookup service, a lightweight cross-platform tool for controlled vocabulary queries," *BMC bioinformatics*, vol. 7, no. 1, p. 97, 2006.

[29] C. Fellbaum, *WordNet*. Wiley Online Library, 1998.

[30] C. Gormley and Z. Tong, *Elasticsearch: The Definitive Guide*. " O'Reilly Media, Inc.", 2015.

[31] H. Bast and B. Buchhold, "An index for efficient semantic full-text search," in *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. ACM, 2013, pp. 369–378.

[32] H. Bast, F. Bäurle, B. Buchhold, and E. Haußmann, "Semantic full-text search with broccoli," in *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 2014, pp. 1265–1266.