Management of Scientific Documents and Visualization of Citation Relationships using Weighted Key Scientific Terms

Hui Wei, Youbing Zhao, Enjie Liu, Shaopeng Wu, Zhikun Deng, Farzad Parvinzamir, Feng Dong and Gordon Clapworthy
CCGV, University of Bedfordshire, Luton, U.K.

Keywords: Data Management, Term Weighting, Ontology, Text Mining, TF-IDF, Visualization.

Abstract: Effective management and visualization of scientific and research documents can greatly assist researchers by improving understanding of relationships (e.g. citations) between the documents. This paper presents work on the management and visualization of large corpuses of scientific papers in order to help researchers explore their citation relationships. Term selection and weighting are used for mining citation relationships by identifying the most relevant. To this end, we present a variation of the TF-IDF scheme, which uses external domain resources as references to calculate the term weighting in a particular domain; document weighting is taken into account in the calculation of term weighting from a group of citations. A simple hierarchical word weighting method is also presented. The work is supported by an underlying architecture for document management using NoSQL databases and employs a simple visualization interface.

1 INTRODUCTION

Adequate management and visualization of scientific and research documents can offer valuable assistance to researchers by improving their understanding of relationships (e.g. citations) between the documents. This has attracted much attention in research communities in natural language processing, information retrieval and information visualization.

Effective management of scientific and research documents involves a wide spectrum of techniques, including document indexing for the creation of numeric representations of the documents; ranking of key scientific terms; and weighted representations of the documents, etc. Term selection and weighting are used to identify the most relevant terms and assign a numeric value to each term to indicate the contribution of the term to its document.

This paper presents our work on the management and visualization of large corpuses of scientific papers in order to help researchers explore their citation relationships. We have processed 13 years of publication in the ACM SIGGRAPH conferences in Computer Graphics (CG), where it is generally recognised that SIGGRAPH publication represents the latest advances of CG technologies. Citation relationships captured in time can also indicate the evolution of research topics over years.

At the pre-processing stage, text mining is used to extract citation relations (namely the reference list) and metadata obtained from raw PDF format. The metadata describe a document in terms of its title, year, authors, etc. The information is stored in the document repository. We use terms to represent document content as most of the existing approaches (the Vector Space Model (VSM), or known as *bag of words*.) Standard terms from a document are collected with their occurrence after lemmatization and Stop Words removal.

The data management is implemented by following a NoSQL scheme in order to address scalability. We have studied characters of different types of NoSQL data repositories which are employed for retrieving information. CouchDB was selected because of its on-the-fly document transformation. A semantic repository, the Sesame RDF, was used to describe key scientific terms and their synonyms in the CG field. We use an external resource MAS keyword API (MAS API) as the input data to create the ontologies.

The citation relationships between the documents in the repository are analysed and stored using a graph repository, enabling quick citation path retrieval.

Wei, H., Zhao, Y., Wu, S., Deng, Z., Parvinzamir, F., Dong, F., Liu, E. and Clapworthy, G.
Management of Scientific Documents and Visualization of Citation Relationships using Weighted Key Scientific Terms.
DOI: 10.5220/0005981501350143
In *Proceedings of the 5th International Conference on Data Management Technologies and Applications (DATA 2016)*, pages 135-143
ISBN: 978-989-758-193-9
Copyright © 2016 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

From a pair or a group of related citations, we define a term-weighting scheme, which selects important terms according to their relevance to the cited documents, taking into account the popularity of the scientific terms in the relevant year, as well as their occurrence in the entire SIGGRAPH corpus. Terms appearing in higher ranked documents should be given higher weights.

The data are finally visualized using a directed graph controlled by a user-specified path length. The graph shows all paths that satisfy the restriction imposed by the path length. The weighted terms are shown in the graph in descending order.

In summary, our contributions are as follows:

- an approach for the management of large scale corpuses of scientific documents that work seamlessly with the underlying text mining framework to support efficient data retrieval
- a term weighting scheme allowing for the ranking of key scientific terms over years at both document and corpus levels
- a visualization method to display citation relationships between the scientific documents together with weighted scientific terms.

The rest of the paper is organized as follows. Section 2 provides an overview of related works, Section 3 describes data management in this task, and Section 4 describes our term weighting method. Section 5 presents our approach to visualization, while Section 6 summarizes our work.

2 RELATED WORK

We review related works in three areas: NoSQL data management, term weighting, text visualization.

2.1 Data Management

Katarina (2013) mentioned that graph databases are very efficient in traversing relationships. Kivikangas and Ishizuka (2012) introduced a semantic representation format Concept Description Language (CDL). They store semantic data presented by CDL in Neo4j graph database and utilize semantic relationships to improve queries.

Most applications use one or two data repositories in their data layer support, we use 4 NoSQL repositories to support indexing and querying.

2.2 Term Weighting

Term selection and term weighting (TW) are important processing phases for text categorization which have been investigated in recent years.

A term-weighting scheme can affect not only text classification, but also other text mining tasks, such as sentiment analysis, cross-domain classification and novelty mining (Tang et al., 2009; Zhang et al., 2009; Tsai et al., 2011). A classic term-weighting scheme introduced in Debole and Sebastiani (2003) is based on 3 assumptions: 1. the multiple occurrence of a term in a document is related to the content of the document itself; 2. terms uncommon throughout a collection better discriminate the content of the document; 3. long documents are not more important than short ones, so normalize the length of documents.

By using sorted term-weighting at a document level terms that are important stand out from repeated or redundant terms, so the user can benefit from quickly extracting useful information (Zhang and Tsai, 2009).

Debole and Sebastiani also introduced *supervised term weighting*, designed for IR applications that involve supervised leaning, such as text filtering and text categorization. They proposed a number of "supervised variants" of TF-IDF weighting.

Domeniconi et al. (2015) proposed a supervised variant of the TF-IDF scheme, based on computing the usual IDF factor without taking documents of the category to be recognized into account. The idea is to avoid decreasing the weight of terms included in documents of the same category, so that words appearing in several documents of the same category are not undercounted. Another variant they proposed is based on relevance frequency, considering occurrences of words within the category itself.

Li et al. (2012) proposed a cross-domain method extracting sentiment and topic lexicons without counting labelled data in the interested domain but counting labelled data in another related domain.

Another cross-domain approach (Domeniconi et al., 2014) creates explicit representations of topic categories, which can be used for comparison of document similarity. The category representations are iteratively refined by selecting the most similar target documents. Further, Tsai and Kwee (2011) compared and discussed the impact of TW on the evaluation measures, and recommended the best TW function for both document and sentence-level novelty mining.

None of these works uses citing relations as a factor in TW.

2.3 Text Visualization

Xinyi et al. (2015) designed 5 views for representing topics. They set different font sizes on words of a

topic based on the occurrence probability in a "word cloud" view. Spatial information of topics is presented in "scatterplot" view in which similar topics are placed close to each other. The evolution of topics over 10 years is represented by a Sankey diagram. They use Treemap to represent their three tree-structure topic results as a hierarchical structure of topics, and they represent the trends of a topic by a Stream diagram.

Mane et al. (2004) presents a way to generate co-word association maps of major topics based on highly frequent words and words with a sudden increase in usage. They use a Fruchterman-Reingold layout to draw co-occurrence relations in 2D, but the data source for a citation is only collected from the title and keywords.

Chen (2004) visualizes salient nodes in a co-citation study, with a focus on three types of node: landmark, hub and pivot nodes. They apply time slicing, thresholding, modelling, pruning, merging and mapping methods to prune a dense network.

We have not found an existing visualization method that uses citing paths.

3 DATA MANAGEMENT

We define 4 logical data entities: Citation, Corpus, Reference and Keyword. A *Citation* is a published paper that is managed in our system in full text and PDF. A set of Citations published in the same year is a *Corpus*. A *Reference* is a cited paper in the reference list from a Citation. A *Keyword* of a citation is a CG keyword that appears at least once in one citation.

We used as benchmarks 1228 publications from 13 years of ACM SIGGRAPH conferences (2002-2014). Corpuses are organised by year, which introduces a time factor as it is strongly related to topics, and we use this natural corpus as our logic corpus.

The raw resource of a Citation is a PDF file. These citations are semi-structured, and they follow a certain template – in this case, the ACM format. We use text mining to extract META data for each citation by identifying basic information.

For a Reference in the reference list of a Citation, we extract the title, year and authors as its identity. There are two possibilities: this reference is either a citation that already exists in the system, or it is not a SIGGRAPH publication. At this stage, we assume SIGGRAPH represents a history of topics in CG. Based on this assumption, and in order to simplify the problem, only references that can be matched to citations in our system are considered. The other references are stored, but not processed.

Although the keyword list section in a paper represents the author's point of view, it cannot reflect important information in most cases. Authors may use different phrases to represent the same concept, such as "3D"/"three dimensional", "level of detail"/ "LOD", and so on. To resolve this problem, an ontology is introduced. An ontology is a formal, explicit specification of a shared conceptualization (Gruber 1993; Borst 1997).

Due to the complexity of data, we employed four type of data store (a semantic repository, an index and search repository, a document repository, and a graph repository) for efficient data management and information retrieval. We take full advantage of their features and strengths. Utilizing these repositories in combination can effectively store and index data with reliability and efficiency to supply meaningful information in support of scientific research.

3.1 Semantic Repository

The standard keyword list we used as shared concept is fetched from the MAS API, It supplies a keyword function representing keyword objects in many fields. For the "computer" area, it covers "computer graphics", "computer vision", "machine learning", "artificial intelligence" etc.- 24 fields in total. We target our research in the "computer graphics" field, where we collected 13670 keywords.

Each CG keyword in CG field was described as an ontology graph model with nodes and edges. A keyword is an RDF (Resource Description Framework) with "rdf:type" of CG. It has synonyms described by the "owl:sameAs" predicate. The outcome of this work is that each keyword in a citation can be mapped to a node with type of CG in the semantic repository. We chose Sesame (Fensel etc, 2005) as our RDF repository as it supplies API for creating, parsing, storing, inferencing and querying. It can also be connected to the Semantic annotation tool GATE which we used for extracting the META data. From the "GATE ontology, Gazzetter producer" output, we can calculate the frequency of each keyword.

3.2 Document Repository

The document repository (CouchDB) is designed for web application, and files can be treated as attachments of a document. By passing a document id, attachments of a document can be accessed easily. Since CouchDB treats each record as a document without considering its properties, a database can

contain a large number of documents. A property, *docType* is used to distinguish document types from corpus, citation, keyword frequency and doc references. Each of the documents in the database was set a *docType* value. It plays the role of a table in relational databases that holds a structured format with collection of related data. For documents, a virtual table of data structures is created for this schema-less repository.

Some benefits from CouchDB are:

- A design document "View". As in any relational database, documents can be sorted by the key of a view. Furthermore, values emitted from the view can not only be fetched from data stored in the database directly but can also be calculated from functions written in Javascript.
- A type of function in design documents with the property name *validate_doc_update*. Each document has to be valid through all this kind of functions defined in a database when creating or updating. Consequently, data structure of documents in this schema-less database will meet our expectation.
- The Reduce function reduces the list to a single value. It is useful for data aggregation to create a summary of a data group.

3.3 Graph Repository

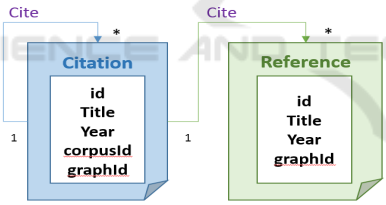

Figure 1: Relations of citing and cited citations.

As mentioned earlier, in the reference list of a citation, if a reference is not managed as a citation by our system, it will stay unchanged in the document repository. But if a reference is already managed in the document repository, the two objects are merged into one object in the graph repository. As we have full information in document repository for both citing and cited document, we can build relations between them. Along with the year of each citation, this relation can reflect the evolution of interest from year to year in the context of CG. In Figure 1, the data model in the graph repository is simplified to show only the citation and its cited citations (blue entities/relations). Within the 1228 citations from SIGGRAPH, this relationship is complex. As a directed graph, the longest citing chain we found has a length of 8 in one direction– this chain covers the whole 13 year span.

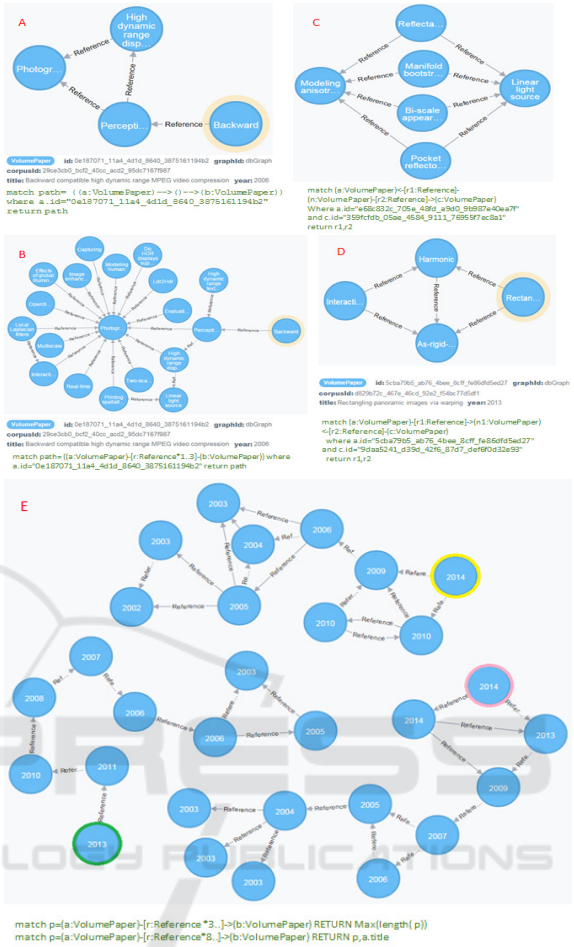

Figure 2: Cite relations in Graph Repository.

In the graph repository Neo4j (Hongcheng et al., 2013), each entity is represented as a node which is identified by a LABEL. We define "citation" as a node entity and "cite" as a relationship that connects nodes in a directed way. The following are some use cases to demonstrate cite-cited relations in Figure 2:

A. *Navigate and Retrieve Cited Citations from a Given Citation.*
The given citation is a citing paper. The result from query A gives direct cited and indirect cited papers.

B. *Citing/Cited Papers from a Given Paper.*
The given citation is a citing or cited paper. The result from query B gives relations from the given citation in 1 to 3 layers.

C. *Detect Similar Citing Citations.*
Two given citations are cited citations. The result from query C shows similar citing citations from the two given citations.

D. *Detect Similar Cited Citations.*
Two given citations are citing citations. The result from query D presents similar cited citations from the two given citations.

E. *Detect the Longest Path*.
This query finds the length of the longest path in the repository and uses it to search paths at the given length. In the current 13-year corpus, the longest path is 8, with 9 nodes. Finally it finds all the paths with length 8 – in our work we found 2 paths of this length. The starting nodes are marked in yellow, pink and green contour in figure 2 E.

3.4 Search Repository

Views in a document repository are the primary tool used for querying the CouchDB documents. A *View* function accepts parameters and gives emit [key, value] pairs as a result. Whether a paper contains user defined keywords or not is a main querying method to help users to search related papers. If we use user-defined keywords as the parameter to query a view, this view needs to emit a key that contains the user defined keywords. From a predefined virtual table structure, it is difficult to predict which property should be used as a key for searches from the user's side. For this purpose, we employ a search server called Elasticsearch which provides a document-oriented, full-text search engine with a RESTful API.

The repository contains only brief description that includes the corpus information, title, author, year and the full text part of a paper as an attachment. This function is supplied by: Mapper Attachments Type for the Elasticsearch plugin. With the brief description, the searched papers from the search engine contain all the information needed for a list presentation and need no further information retrieval from the document repository.

4 IMPORTANT TERMS

As mentioned earlier we have extracted a standard keyword list for each citation. Most citations in CG field use less than 100 standard keywords out of 13,670. Each citation related keyword is calculated by their occurrence. This frequent appearance indicates importance from the author's view. TW can help text mining tasks in terms of text classification, topics extraction, and sentence analysis and further help reader to grasp the main idea of each citation in a large corpus.

The keyword part in MAS API supplies the keywords' name along with two other important properties: publication count, which indicates the number of publications of each keyword, and citation count, which gives the total number of citations of all the publications using this keyword.

Table 1: Top 10 keywords sorted by citation.

Keywords	Publication	Citation
Computer Graphic	4729	99608
Real Time	4208	68950
Three Dimensional	2131	46419
Texture Mapping	1010	34038
Geometric Model	1028	30263
Volume Rendering	1418	29171
Ray Tracing	1195	29061
Virtual Environment	1904	28844
Virtual Reality	2342	27932
Level of Detail	1146	27846

Table 1 shows the top 10 keywords in CG, sorted by citations. There are 13670 keywords in this field in total. In the top 10 keyword list of other fields, for example in Computer Vision (12839 keywords), the same keywords such as "real time" appear again, as some domains have similar research topics to others (Xinyi 2015). The Inverse Document Frequency (IDF) reflects the importance of a word to a document in a collection or corpus.

We use 4 different methods to highlight characteristic terms: field level, citation level in CG, year level in CG, and hierarchical topic names.

4.1 Field Term Weighting

The field TW highlights characteristic terms in each field. We fetch keywords from MAS API in 24 fields of the computer domain and treat these keyword fields as 24 documents. In the keyword corpus of the domain, $D=\{d_1, d_2, \ldots d_{|D|}\}$, each file contains multiple keywords with occurrence of publication count or citation count (Table1). In CG, the document contains 13670 keywords. A document d_j is represented as a word vector $w^j=\{w_{1j}, w_{2j}, \ldots w_{nj}\}$ in an n-dimensional vector space. Each word should be mapped to a weight factor W_i in the document.

We assume the data fetched from MAS API is counted by a large corpus of related field citations. Hence the citation count property of each keyword in a field document is the occurrence value of this keyword in a field of this corpus.

Using TF-IDF, the weight factor of keywords in each field document is found – terms appearing frequently in the corpus are expected to have less importance. This filters out the more common terms.

We use the raw frequency and inverse document frequency smooth (IDF) method to calculate each TW in a field: N is the number of the total fields (here 24), and n_t is the occurrence of a keyword in other fields. Tf is the citation count of each keyword in Table 1.

$$F_{Wi}(dj, w^j) = Tf * \log(1+\frac{N}{n_t})$$

The outcome of this is that in CG, each keyword is assigned a weight indicating its importance in CG compared to other fields. This result is used as a global weighting result and mapped to a local weighting result such as Citation TW and Year TW.

4.2 Citation Term Weighting

In citations of a field, each citation emphasizes different topics even if they have similar frequent terms. Occurrence of a term is related to the content. Field TW in Section 4.1 also indicates the relevance of a term to the field compared with other fields. In this section, we try to identify citation terms that are different from other citations in the same corpus. For each citation keyword, we calculate local IDF L_{idf} with the following equation:

$$L_{idf} = \log(1+\frac{N}{n_t})$$

where N is the citation number of our corpus (1228) and n_t is occurrence of this keyword in other citations.

By using the MapReduce function provided by the document repository, it is easy to obtain a keyword summary of the occurrence of a keyword in citations, since each keyword is related to a citation id. In one citation, a keyword only has 1 record with frequency value in this citation. If we map this value to 1, the reduced result is the occurrence n_t in all citations.

$$C_{wi}(dj,wi) = Tf * F_{Wi}(dj, w^j) * L_{idf}$$

The Tf is the citation related term frequency in the document repository. This identifies important keywords for this citation in CG.

4.3 Year Term Weighting

In the CG corpus, citations in each year contribute to its keywords in terms of weighting. If one citation has a higher weighting than others, the keywords used in this citation should be weighted higher than those used in citations with lower weighting. In other words, document weighting contributes to TW when calculating TW in a group of citations.

A straightforward way to assign a score to a citation is to find the citing number. In our graph repository we have stored citing relationships A→B for each citation. To find the citing number of B, just query the number of A. Let's name this score as "Score(d_j)". For a keyword weighting from a citation

$$Rank(d_j,w_j) = Score(d_j) * C_{wi}(d_j,w_i)$$

Assuming year contained citation number is Yn, then rank over a year can be calculated as:

$$Rank(w_i, year) = \sum_{n=1}^{Yn} Rank(dj, wi).$$

4.4 Hierarchical Word Weighting

Many keywords are composites of several individual words, and the occurrences of some words are meaningful to the group of keywords contain them. One such example in CG is *rendering*, as in image based rendering, real time rendering, non-photorealistic rendering, etc.; we call these "hierarchical words". To find the importance of a hierarchical word in its field, we describe here a simple alternative to the TF-IDF method.

We treat the keyword as individual tokens. Each token contributes to its own keyword equally with the score of the citation count of that keyword. In cases where a word is contained in multi keywords, the score of this word is the sum value of all the citation values of those keywords. From this step, in 24 fields' keywords of MAS API, we calculated hierarchical words for each field with its score. In a single field, this score implies the meaning of term frequency in one document; we give it the name TF.

The second step is to count the occurrence of the hierarchical word in the total of the 24 field documents. By accumulating the TF value in all 24 fields, it is defined as TotalTF.

We are now able to calculate the importance score of each hierarchical word. The main idea that is different from TF-IDF is to improve term frequency importance rather than document importance. The reason is that words that occur once in a field should be treated as being less important in this field than those that occur multiple times. This method leads to a higher accuracy in this context.

$$Rwi = \sqrt{TF} * idf^2 ,$$

where $idf = \log(1+ \frac{TF}{a+(TotalTF-TF)})$

This expression enlarges the global factor and narrows the local factor. The constant "a" is used to avoid cases in which TotalTF–TF is zero. That means the term occurs only in CG, not other fields, and is therefore important in CG.

4.5 Citation Distance

Citing relation is a strong relationship between two citations. From the diagram of citing path, the width of a relation can represent how strong the link is. We mentioned earlier that each citation is related to a

keywords list, with the frequency numbers in the document repository. Percentile Weighting of each keyword in a citation indicates how important a keyword is relative to a citation. One citation with more common keywords in higher percentile value to another citation would also have a higher cosine similarity value as citation distance.

5 DATA VISUALIZATION

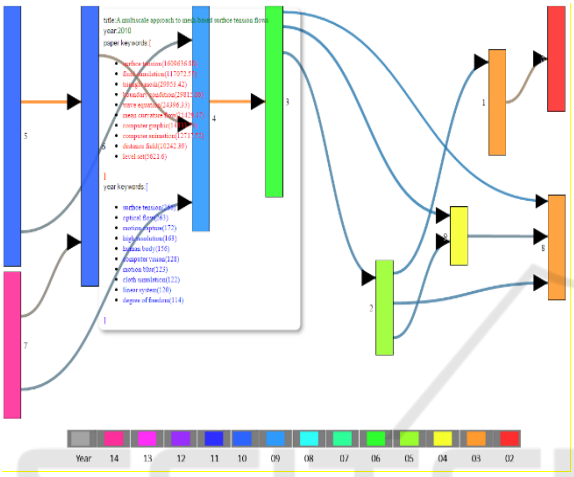

Figure 3: Citing relations when depth is set to 8.

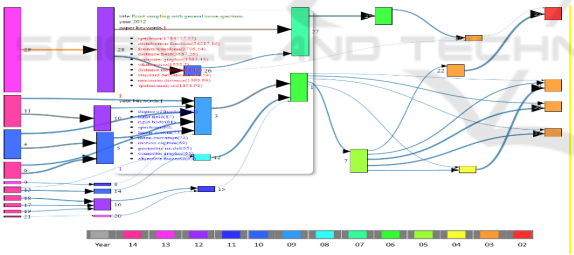

Figure 4: Citing relations when depth is set to 7.

We design an interactive tool to represent citation relations as a direction graph. A node represents one citation, and a link represents the relationship between two citations. An in-direction link of a node means it is cited by another citation; an out-direction link means it is citing other citation.

In figure 2 E, we introduced a method to find the longest path from the graph database and to obtain nodes and links. Returning a graph from the Rest API will result in a large data package unless skilled querying is used. Since each relation between nodes A to B are described as URIs of start, end nodes and relationship as in Figure 5. To obtain further properties from the URIs, more queries are needed. Our technical method is to obtain the starting node.

ID list first, then use each starting node to fetch its own path. As the longest path is 8 in this case, the starting node id, all the relationships can be presented in a direction graph: A←B. Traversal of a path can then be converted to a list of node pair IDs as we know the default direction is "←".


```
"data" : [ [ {
    "directions" : [ "->" ],
    "start" : "http://localhost:7474/db/data/node/226",
    "nodes" : [ "http://localhost:7474/db/data/node/226", "http://localhost:7474/db/data/node/117" ],
    "length" : 1,
    "relationships" : [ "http://localhost:7474/db/data/relationship/223" ],
    "end" : "http://localhost:7474/db/data/node/117"
  } ],
...
]
```


Figure 5: Graph of a relation (node A to node B).

```
MATCH p= (a: Citation) <-[r: Refer * 8 ..] - (b: Citation)
    where a.id='b7ca85bb_0e57_4f38_b800_772215579bd2'
    WITH [x in NODES (p) | ID (x)] AS vp1, [y in NODES (p) | ID (y)] AS vp2
    MATCH (n1: Citation) <-[r1]-(n2: Citation)
    WHERE ID (n1) in vp1 and ID (n2) in vp2
    RETURN distinct n1.id, n2.id
```

Figure 6: Convert a path to list in cypher.

In Figure 6, the cypher query language supplied many functions such as multi match, filter, inner functions, etc., that help us to query data with low cost. In figure 3 with two paths of length 8, 14 pairs of id list returned from graph DB, which contributes to links directly, without any data conversion.

We use the idea of the Sankey diagram to describe this relationship. These are a type of flow diagram in which the widths of the arrows are proportional to the flow quantity. In case one node has multiple out branches to other nodes. The length of the node equals the sum value of the width of each branch. The weight of each branch decides its width. In citing/cited relation, a natural characteristic is that most citing nodes are published later than cited nodes. Another characteristic is that one node of a year can be cited by other nodes of different years. Because of these characteristics, if the node's width is equal to the sum value of it's out branches width, there will be lots of overlap of nodes and links.

To avoid this, we use relatively narrow links (see Figure 3,4). The node length is proportional to the sum of its out-links' weights. A simple property that can easily identify a node on the diagram is year. Nodes with same colour are citations that have been published in the same year. The weights of the links are presented by width and colour, stronger links have more weight than thinner links, and links with similar colour presents similar weight values.

When a mouse hovers on a node, an opacity layer appears showing details of the citation. Only the citation ID, year and title of META data are stored in graph DB to reduce overlap data. When we are querying a path, only the citation ID that is mapped to the citation ID in Document DB is returned. Further calculations such as the year's keywords list, citation keywords list and root keywords list are all performed from our Document DB.

From figure 3, we can see that paper 8 (in orange, title:"Keyframe control of smoke simulation", year: 2003) is an important citation, which affected other citations in this field from year 2004 (paper 9 in yellow) to year 2014 (paper 7 in pink).

From figure 3, we selected one path and the newest node, displaying the top rank terms used in the citation in figure 7. It shows, these terms are very relevant to the content.

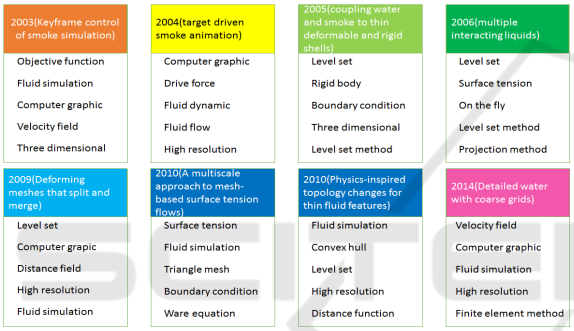

Figure 7: Top 5 ranking keywords with the path length 8.

6 CONCLUSIONS

With the keywords function of MAS API, an ontology is created to extract the field standard keywords frequency. The API is also used to collect keywords in 24 fields of computer domain. We use a series of TW methods to highlight characteristics of terms. The citing relations we stored in the graph database is a dense network. We take full advantage of the 4 repositories that effectively store and index the citation data and hence supply meaningful information. The interactive visual view can present citing relations, similarity and indicate salient citations.

ACKNOWLEDGEMENTS

The research is supported by the FP7 Programme of the European Commission within projects Dr Inventor [611383] and CARRE [611140] .

REFERENCES

Borst W, 1997. Construction of Engineering Ontologies. PhD thesis, Institute for Telematica and Information Technology, University of Twente, Netherlands.

Chen C, 2004. Searching for intellectual turning points: Progressive knowledge domain visualization, *Proc Natl Acad Sci 101 (suppl 1):5303–5310.*

Debole F, Sebastiani F, 2003. Supervised term weighting for automated text categorization. *In Proc 2003 ACM Symp Applied Computing*, pp 784–788. ACM Press.

Domeniconi G, Moro G, Pasolini R, Sartori C, (2015). A Study on Term Weighting for Text Categorization: A Novel Supervised Variant of tf.idf. In *Proc. 4th Intl Conf Data Management Technologies and Applications*, pp. 26-37.

Domeniconi G, Moro G, Pasolini R, Sartori C, 2014. Cross-domain Text Classification through Iterative Refining of Target Categories Representations. *In: Proc 6th Int Conf on Knowledge Discovery and Information Retrieval (KDIR).*

Elasticsearch attachment plugin: https://github.com/elastic./elasticsearch-mapper-attachments.

Fensel D, Hendler J, Lieberman H, Wahlster W, Berners-Lee T, 2005. Sesame: An Architecture for Storing and Querying RDF Data and Schema Information, MIT Press eBook Chapters: Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential, pp 197-222.

Grolinger K, Higashino WA, Tiwari A, Capretz MAM, 2013. Data management in cloud environments: NoSQL and NewSQL data stores, Journal of Cloud Computing: Advances, Systems and Applications 2013, 2:22 doi:10.1186/2192-113X-2-22.

Gruber TR, 1993. A Translation Approach to Portable Ontologies. Knowledge Acquisition, 5(2):199–220.

Huang H, Dong Z, 2013. Research on architecture and query performance based on distributed graph database Neo4j, *Proc 3rd Int Conf Consumer Electronics, Communications and Networks (CECNet),* pp 533-536.

Jiang X, Zhang J, 2016, A text visualization method for cross-domain research topic mining, Journal of Visualization (online).

Kivikangas P, Ishizuka M, 2012. Improving Semantic Queries by Utilizing UNL Ontology and a Graph Database, *Proc 6th IEEE Int Conf Semantic Computing*, pp 83-86.

Li F, Pan S J, Jin O, Yang Q, Zhu X., 2012. Cross-domain co-extraction of sentiment and topic lexicons. In *Proc 50th Annual Mtg Assoc for Computational Linguistics: Long Papers - Volume 1* (ACL12), pp. 410-419.

MAS API: http://academic.research.microsoft.com/about/Microsoft/Academic/Search/API/User/Manual.pdf.

Mane KK, Börner K, 2004. Mapping topics and topic bursts in PNAS, *Proc Natl Acad Sci 101 (suppl 1):5287–5290.*

Tang W, Kwee AT, Tsai FS, 2009. Accessing contextual information for interactive novelty detection. In: *Proc. European Conf Inf'n Retrieval (ECIR) Contextual Information Access, Seeking and Retrieval Evaluation*, pp. 1–4.

Tsai FS, Kwee AT, 2011. Experiments in term weighting for novelty mining. *Expert Systems with Applications*, 38(11):14094–14101.

Zhang Y, Tsai FS, 2009. Combining named entities and tags for novel sentence detection. In: *Proc WSDM Wkshp on Exploiting Semantic Annotations in Inf'n Retrieval (ESAIR 2009)*, pp. 30–34.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS