Scholars Journal of Engineering and Technology (SJET)
Sch. J. Eng. Tech., 2015; 3(2A):117-123
©Scholars Academic and Scientific Publisher
(An International Publisher for Academic and Scientific Resources)
www.saspublisher.com

ISSN 2321-435X (Online)
ISSN 2347-9523 (Print)

Research Article

Sniper: A search engine for domain semantic knowledge

Zhiqiang Wang
College of information and technology, Heilongjiang Bayi Agricultural University, Daqing 163319, China.

***Corresponding author**
Zhiqiang Wang
Email: byndliufang@163.com

Abstract: This paper presents Sniper, a knowledge-based computer field search engine in Semantic Web. Sniper takes WordNet as background ontology and integrates the entities in the semantic documents by mapping them to the synsets of WordNet. Sniper returns the most related knowledge in computer field as result according to user's query. The search results of Sniper are displayed in the form of list of entities, including concepts and instances, which are extracted from heterogeneous ontology, so the result is close to user’s intention. Comparing with the traditional search engines and the current semantic search engine, Sniper can show the relevant information that users want more accurately. The index structure of sniper is an inverted index structure of path that based on domain ontology is proposed in this paper. The all of distinct entities in domain ontology are indexed; and for each distinct entity an inverted list, storing the entity and the identifiers of the path containing the entity, and experimental results show that the index structure is more suitable for the query of multiple keywords and it can eliminate the ambiguity and achieve the semantic expansion of query keywords.
Keywords: Semantic search engine, semantic documents, ontology mapping, inverted index; paths of ontology

INTRODUCTION

In 2000 Tim Berners-Lee made a formal presentation on the concept of Semantic Web and its architecture at the XML2000 international conference. The Semantic Web is an extension of the current Web. The core idea of Semantic Web is knowledge-sharing, between computers and computers, people and computers. By this means the resource in the Semantic Web is machine understandable and rich of semantic. Unlike the traditional search engines which based on the World Wide Web (WWW) (such as Google[1], Yahoo!, Baidu etc.), semantic search engines based on the Semantic Web and apply the semantic technology to improve the performance. How to build the architecture of semantic search engine to make good use of semantic information of Semantic Web is a new challenge.

With the development of Semantic Web, semantic search engine become a hot research issue. There are different types of semantic search engine at present according to different object and purpose for searching. We divide the semantic search engine into three types in this paper. First, the search engines that based on Web, apply natural language processing to search, aim to improve the way we find information by unlocking the meaning encoded in ordinary human language, such as, PowerSet[2], Hakia[3], etc. Second, the search engine that fetched data from both the traditional and the Semantic Web. Andreas Harth proposed a pipelined architecture[4] that fetched large amounts of semi-structured data from the Web and transformed them into RDF. It is a architecture to crawl and index data from both the traditional and the Semantic Web. Third, the search engines that searched for semantic documents, such as, Swoogle[5] , Watson[6] and Falcons[7] , etc. They have similar style with traditional search engine interface, use keyword search, supply the services of search for RDF, RDFS, OWL, etc, and display the search results for each semantic document. But it is difficult to understand the search results for users who lack of knowledge of Semantic Web.

In this paper, we propose a knowledge-based semantic search engine-Sniper on computer field .The search result of Sniper is different from the semantic search engines mentioned above which aim at semantic documents. Sniper does not only list the search results to the user, but give the best meet with user's query demand. It analyses the intent of user's query, correspond concepts or instances with query keywords, and integrates all related structured information of entities on base of ontology mapping, recommend the most relevant resources for users and show all of these information in a whole page.

The remainder of this paper is organized as following: In Section 2 we detail the structure of Sniper which contains Focused crawler, Ontology mapping and Semantic Index. Section 3 details Sniper semantic search engine, analyses and explains search results In

Section 4 concludes the paper and propose possible research work in future.

Sniper structure

The structure of Sniper is composed of 4 main modules, as illustrated in Figure 1 which is: Focused Semantic Crawler, Ontology Mapping, Parser, and Indexer.

Focused Semantic Crawler Under the guidance of the Domain Ontology, Focused Semantic Crawler fetches the semantic documents for computer field from Web. Semantic documents that fetched from Web are divided into two categories, ontology and other semantic document.

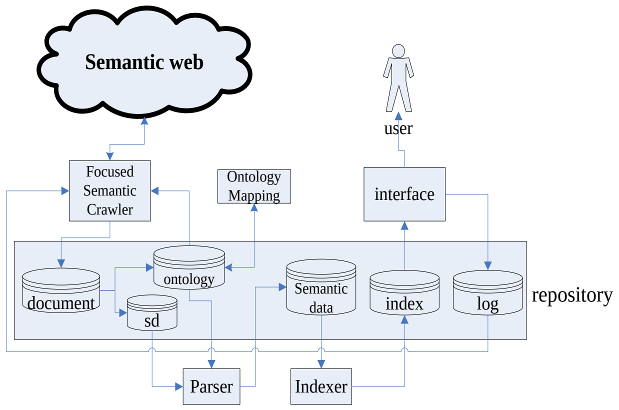


Fig. 1 Framework of Sniper

Ontology Mapping The Ontology Mapping is the process of determining correspondences between concepts from different ontology. It can enrich and expand the Domain Ontology, then the new Domain Ontology indirect Focused Semantic Crawler to fetch semantic documents.

Parser The function of Parser is to parse all the semantic documents including Domain Ontology.

Indexer The Indexer is to build index for the semantic data that is parsed by parser. During the process of index, we also build index for the data of Ontology Mapping and the result of Semantic Clustering based on concepts and instances and store the data into repository. So it is convenient for user to query.

1.1 Focused Semantic Crawler

It is important to the semantic search engines, because the Focused Semantic Crawler provides the materials for showing and analysing. The main difference between Semantic Web crawler and Web crawler is that they analyse different type of documents. The Semantic Web crawler deals with semantic documents. The semantic documents model obtains topic information by parsing the syntax of RDF or OWL. So it is different from Web document model. It is crucial problem to the Semantic Web crawler that how to acquire more relevant resources during shorter time. There are two approaches for the Focused Semantic Crawler to obtain semantic documents. The first is to search through semantic search engine, such as Swoogle, Watson, etc, and extract more URIs based on these semantic documents. Another is to select several URIs that point to the semantic document that contains a large number of semantic links as begin for crawling. For the vertical search, the critical problem is to obtain semantic documents related to theme by parsing them, and find additional source to crawl through extracting URI from current documents. The Focused Semantic Crawler of Sniper improves Slug[8] . It represents the content of documents by extracting the entities, and then, compares the content with the computer domain ontology we extract from WordNet, if a semantic document is regarded as domain-dependent on computer, it will be stored into the documents repository. These documents can be used as materials for other modules. In the course of fetching documents, there are some isolated semantic documents that have not pages which point to and not point to other pages. We proposed a method to describe the content of document by integrate semantic cluster in lexicon with Word Sense Disambiguation. In this Method, we use the concept of maximum probability density clustering algorithm to accurately improve classification of the content of the document. We regularly adjust the parameters of path prediction algorithm through related samples and access patterns that by mining the user log.

1.2 Ontology Mapping

Gruber[9] has defined the ontology as “an explicit specification of a conceptualization” and the purpose of ontology is “knowledge share”, but it is impossible to construct a global ontology that covering everything, as the knowledge of the world is infinite, and the subjectivity and distribution during the construction of ontology. In fact the different users construct their own ontology according to their application requirements, so there is a lot of ontology that the contents they have described is overlapping or related, and the representation model and ontology language that they have used is various, this is called heterogeneous problem of ontology. Ontology mapping can solve this problem by establishing alignment for the same or similar elements between different ontology. As a knowledge-based semantic search engine Sniper need to map the large number of heterogeneous ontology on domain ontologies. The method of ontology mapping is various. GMO[10] uses bipartite graphs to represent ontology, and measures the structural similarity between graphs by a new measurement. RiMOM[11] and presents a dynamic multi-strategy ontology alignment framework and propose a strategy selection method to automatically combine the matching strategies based on two estimated factors. ASMOV[12] is a novel algorithm that uses lexical and structural characteristics of two ontology to iteratively calculate a similarity measure between them. The value of concept similarity more think about the property of concept during the process of Ontology mapping at present, less consider the influence of instance. Ontology Mapping algorithm of Sniper based on OLA[13] algorithm that proposed by universities in Montreal, Canada and We improved this algorithm. The Conceptual similarity algorithm of OLA is the sum of label similarity, instance similarity, super and subclasses similarity and the similarity of properties. Sniper improves OLA algorithm. Such as, match type of entity strictly and exactly, expand the instance similarity, etc.

1.3Parser

It is essential to parse the semantic documents for building index, and it is convenient for us to expediently display and manage the semantic structure information. The semantic documents were divided into two categories, ontology document and semantic data document. The majority of documents are ontology documents; the other is HTLM Web that after semantic tagging and the web has some semantic information. From 1998 Tim Berners-Lee proposed the Semantic Web to the present, the development of ontology language from RDF that possesses simple semantic relations, RDFS that can simply describe ontology to DAML+OIL, OWL that possesses the ability of stronger reasoning and describing, so the ability of description of ontology language is becoming richer and richer. Therefore, we need to classify the Semantic Web Documents according to ontology language and call different parsing model to parse the semantic documents. The tool of parsing semantic document is Jena, developed by HP Labs Semantic Web application framework for java. The related information of concept or instance extracted from the semantic documents, including the relation of concepts and instances, the relation of concepts and concepts, and the relation of concepts or instances and semantic documents, etc, and stored data into the repository. The work of parsing is divided into two parts, parsing standards and parsing the semantic documents. Parsing standards is significant to parse other semantic documents, such as RDF, RDFS, DAML, OWL, etc. These standards are Semantic Web standards that W3C recommended, and they are principal to build Semantic Web. Parsing the semantic documents includes pre-processing, parsing and stores the data into repository.

1.4Indexer

Sniper will show all information that related with the user's query in a whole page, including semantic structure information and resource recommended to the user. So we need to build index for the data after parsing, the data after Ontology Mapping and the data after Semantic Cluster in order to quickly response to user's queries. The indexer of Sniper consists of Semantic Cluster of entity and Index Model.

1.4.1 Semantic Clustering

The objects of Semantic Cluster are mainly for concepts and instances. So we also call it semantic entity cluster. The entities were clustered in order to recommend the most relevant resources to the user. We integrate all similar entities into a cluster by semantic distance of entities. The goal of semantic entity cluster is to show the recommended the most similar resource for the user's query, and enhance the hit rate of query results that meet the user's query intent. The measure of semantic entity cluster adopts conceptual similarity based on WordNet. The conceptual similarity takes into the distance of two entities in the WordNet, the amount of information of entity, and hierarchy of entity. It is certain influent to conceptual similarity that exist multiple connection paths between two entities. If the value of similarity is greater than a certain threshold, we will form entity cluster. We build index for cluster in order to provide the most similar resource for user. And we can quickly find the entity from the index to correspond with the query keyword and obtain a sorted set according to the similarity.

1.4.2 Index Model

Index is another way to represent data. The order of the indexing data is different from the physical storage order on disk. The role of the index is able to provide fast query. We designed four index structures according to the needs of Sniper, and the index model consist of these index. The four index structure is linked together and support fast query of Sniper. The four index structure is ontology mapping index, keyword index, concepts or instance index and cluster index.

The ontology mapping index is to enhance the performance of Ontology Mapping. In the process of Ontology Mapping, the number of pairs of concept matching is the Cartesian product of the number of concepts in two ontologies, if the number of concept is the large number of in two ontologies, then the calculation of the similarity of the two concepts is a long time process. Therefore we build a B-tree index for ontology mapping in order to reduce the time of matching process. If the string similarity above a certain threshold, we put the concepts in the same node, thus reducing the concept of the matching process, greatly reducing the time Ontology Mapping process.

The keyword index is build for concept and instance. It can make the query keywords quickly correspond to concepts or instances. We use the B+tree as the structure to locate concepts and instance. It is enough to contain the whole concepts and instances due to the features of B+tree.

Concepts or instance index are consist of inverted index, mapping index and semantic index. The inverted index includes the frequency of entities, the position of entities and the document that contain the entity, etc. The mapping index is built for the data after ontology mapping. The semantic index is built for the data after parsing.

The semantic clustering index is for the data after semantic entity cluster. So Sniper can recommend the most correlative entity with the query keyword.

1.4.3 The structure of index

The structure of inverted index of entity and path

The inverted index of entity and path is similar to the traditional inverted index of term and document. The structure of traditional inverted index as pictured in Figure 2.

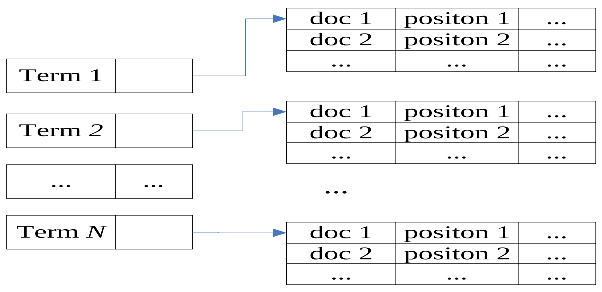


Fig. 2 the structure of traditional inverted index

The entry of inverted index is term, and can find the collection of documents that contain the term. This collection is called hits and the hits include the information of the position of term in document, etc. we refer to the idea of traditional inverted index and designed the inverted index of entity and path as depicted in Figure 3. The entry of inverted index of entity and path is term, too, and term corresponds to the entity. We can find the collection of path that contain the term by querying the inverted index of entity and path and the path include the information of the position of term and length of path. The path is combination of multiple words that have semantic relation.

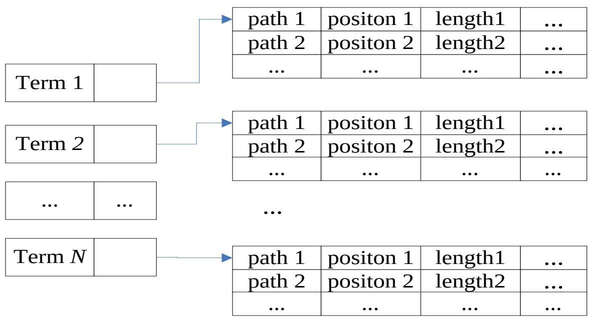

Term 1
Term 2
...
Term N
path 1 | positon 1 | length1 | ...
path 2 | positon 2 | length2 | ...
... | ... | ... | ...
...

Fig. 3 the structure of the inverted index of entity and path

We query the inverted index of entity and path by a keyword, the results is a collection of path that contains the query keyword. When the number of query keywords is multiple, the results will be the collection for each query keyword. Then we can obtain intersection of collections and the path in the intersection collection that contains multiple query keywords. According to the position of query keyword in the path, we calculate the distance between query keyword, and return the most relevant semantic paths.

The structure of entity index

The purpose of entity index is to record the position of entity in path index file that consist of entity and the path set that contain the entity. We establish a map of entity and the position of entity in path index file so as to quickly find the set which contains the entity. As showed in picture Figure 4

Id	word	*offset*
1	mouse_click	*112828*
2	CD_drive	*2988156*
3	*computer_monitor*	*3085219*
4	mother_board	*3125057*
5	disk_cache	*3209141*
6	firewall	*3348142*
7	Intelnet	*3577474*
...	...	...

Fig. 4 the structure of entity index

Each entity has a unique ID and map the entity to the offset of entity in path index file. Such as, we query the "firewall" ,we can obtain the all path that contain "firewall" by fetching the set from path index file in position "3348142". So we can quickly find the paths that contain entity.

1.4.4 The process of construct index

Domain Ontology

The Domain Ontology is constructed by the expert in the field and reflects the general view of the field. It describes the semantic information of concepts by relations of concepts. The hierarchy of concepts is not a tree, it is a reticular structure because the relation of concept not only hyponymy but also concept connect with concept by other relation. The Domain Ontology in this article is extracted from the WordNet language ontology. WordNet was constructed by Princeton University psychologists, linguists and computer engineers, based on cognitive linguistics. It dose not only order the words in alphabetical, and according to the meaning of words to form a "word network". We only extract the words that belong to computer field from WordNet in order to obtain computer domain ontology. The process of construct computer domain ontology as follow: first, we extract vocabulary of computer field from the computer topic of Open Directory[1]. Second, we extract the synsets from WordNet refer to the computer vocabulary that from Open Directory. We extract synsets start with "entity", use the attribute relation of WordNet find next synsets,

[1] http://www.opdir.com/. Open Directory uses a hierarchical ontology scheme for organizing site listings. Listings on a similar topic are grouped into categories, which can then include smaller categories.

until the attribute relation end or synsets not in computer field. We store the all computer synsets and relation into rdf file to form computer domain ontology.

Path

The main work in inverted index of entity and path is to obtain paths from domain ontology. The hierarchy of concepts in the domain ontology is not a structure of tree, it is a reticular structure. According to the position of node in the reticular structure we divide the nodes into leaf node and non-leaf node. We get two leaf nodes from computer domain ontology as endpoint of the path, extract all possible paths that connected the any two leaf node. So we obtained all path of domain ontology.

Construct Index

The index is consisted of the path index file and entity index.

The process of construct the path index file, as follow:

1. Divide the words from path into the set of keyword.
2. Record the position of keyword in path and belong to which path. The purpose of record the position of keyword in path is to calculate the distance between keywords in path so as to obtain the path that more related to the query keywords.
3. Construct the inverted index for keywords and path.
4. Write the indexed information of keywords and path into the file of inverted index of path.

1. In the above 4, we also record the position of keyword in the path index file in order to construct entity index. The entity index map keyword with its position in the path index file.
2. The keywords in entity index correspond to the entity in the computer domain ontology, and the keyword correspond s to the unique keyword position in the path index file.

1.4.5 Experiment

Test environment: DELL OPTIPLEX 740, AMD Athlon(tm) 64 X2 Dual Cord Processor 3600+ 1.90GHz CPU, 2G DDR, Microsoft Windows XP operating system.

First, we test the performance of the inverted index of path by input single query keyword, double query keywords and three query keywords. The results of test are display in Tab 1, Tab 2 and Tab 3. It should be explained that the capacity in table is the capacity of path set correspond to each keyword.

Tab 1 is the comparison of response time of ranked and inverted index of path whit single query keyword. From the Tab 1, we can conclude that the query responded time of the inverted index of path is proportional to the capacity of path. The query responded time of the ranked index of path has a little change because the process of the query the ranked index is same.

Table -1:The Comparison Of Response Time Of Ranked And Inverted Index Of Path Whit Single Query Keyword

words	**Rank time**	**Inverse time**	**capacity**
typewriter	1453(ms)	0(ms)	1kb
briefcase computer	1469(ms)	16(ms)	10kb
computer peripheral	1484(ms)	172(ms)	100kb
computer	1471(ms)	1765(ms)	1161kb

Tab 2 is the comparison of response time of ranked and inverted index of path whit double query keywords. Tab 3 is the comparison of response time of ranked and inverted index of path whit three query keywords. From the Tab 2 and Tab 3, the query responded time of the inverted index of path is shorter than the query responded time of the ranked index of path.

Table -2:The Comparison Of Response Time Of Ranked And Inverted Index Of Path Whit Double Query Keywords

words	**Rank time**	**Inverse time**	**capacity**
Typewriter keyboard	1469(ms)	234(ms)	(1+135)kb
briefcase computer Portable computer	1438(ms)	65(ms)	(10+27)kb
computer peripheral printer	1485(ms)	185(ms)	(100+8)kb

Table-3: The Comparison Of Response Time Of Ranked And Inverted Index Of Path Whit Double Query Keywords

words	**Rank time**	**Inverse time**	**capacity**
Typewriter Keyboard typewriter keyword	1453(ms)	293(ms)	(1+135+35)kb
briefcase computer Portable computer PC	1469(ms)	97(ms)	(10+27+20)kb
computer peripheral Printer dot printer	1484(ms)	236(ms)	(100+8+30)kb

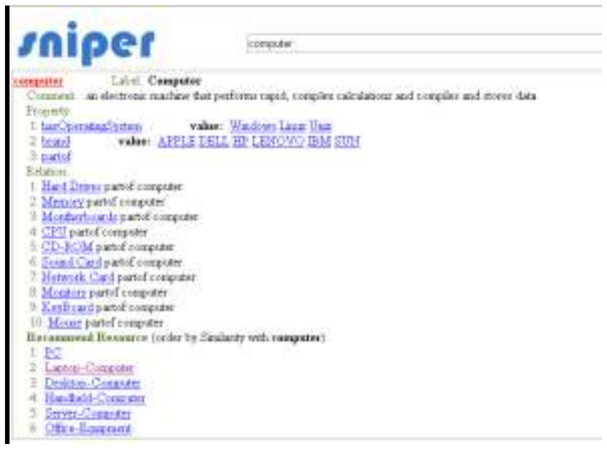
Second, we analyse the results of the path query. We input the query keywords computer and cpu, the query result is a path-“server→computer→cpu→cpu_board”. When we input the query keywords computer_network, client and foreground, the query result is a path-“client→computer_network→network→System”. So we can see that the path can better meet the query requirement of user and realize the semantic expansion of query keywords.

The multiple keywords that the user’s input for query describe the user’s query object in all aspects. So the inverted index of path is more suitable for the query of multiple keywords. From the experimental results we can see that the inverted index of path not only eliminate ambiguity of query keywords but the other words in path is complementary for query keywords, and enhance the user's query satisfaction.

Semantic search engine-sniper

Sniper is a knowledge-based semantic search engine for computer field. The goal of Sniper is to show an accurate and general knowledge of computer field for user's query. The search results of Sniper are displayed in the form of knowledge in order to improve user's satisfaction. If the query keyword is a single word, Sniper will match keyword based on etyma and correspond with the entity. The all information related to keyword will be obtained through the index model. If the keyword has more senses, the related information will be showed in classification according senses. If the keyword is not a single word, the query object can be described by these keywords in all aspects. These keywords are interrelated with the query object, so the concepts correspond with these keywords will be strong semantic interrelation. We match these keywords to the entities so as to obtain semantic combination of interrelated entities that better represent the intent of user's query, and return more accurate query results.

In order to better illustrate the process, we enter the query word computer in the search interface of Sniper. The query results as shown in Figure 5.



computer Label: Computer
Comment: an electronic machine that performs rapid, complex calculations and compiles and stores data
Property:
1. hasOperatingSystem value: Windows Linux Unix
2. brand value: APPLE DELL HP LENOVO IBM SUN
3. partof
Relation:
1. Hard Drives partof computer
2. Memory partof computer
3. Motherboards partof computer
4. CPU partof computer
5. CD-ROM partof computer
6. Sound Card partof computer
7. Network Card partof computer
8. Monitors partof computer
9. Keyboard partof computer
10. Mouse partof computer
Recommend Resource (order by Similarity with computer)
1. PC
2. Laptop-Computer
3. Desktop-Computer
4. Handheld-Computer
5. Server-Computer
6. Office-Equipment

Fig-5:The resulted page of Sniper

Computer, in the figure, represent concepts in different ontology, if click on it, the page will be show the information of computer in different ontology. The information include label, comment, URI, URL of document, etc. Similarly, other elements have also gathered information from different ontology. We can see from the page that not only shows the Data type Property of computer but also demonstrates the value of Data type Property, for example, the has Operating System that have values of Windows, Linux and Unix. There are many concepts have relation of part of to computer, such as, Hard Drives, Memory, etc, and have related resource recommended to the user. So this result page can greatly satisfy the user's query intent.

CONCLUSION

This paper presents a semantic search engine for the semantic document of computer field, Sniper. We have presented the system structure, basic functions and its realization. Sniper has the following characters:

- Sniper is a semantic search engine based on Semantic Web for the knowledge of computer field.
- Sniper is a ontology-based knowledge extraction framework. Sniper extracts the knowledge dispersed anywhere can be retrieved and align ontology into a domain one.
- Sniper integrates the concepts and instances of different ontology by ontology mapping technology, so Sniper refines the query result and returns the most relevant information to the user's query.
- The index structure of sniper is an inverted index structure of path that based on domain ontology, which enhance the user's query satisfaction.

Sniper goes well in our laboratory at present, but there is a lot of work needs to do in the future, such as extending the semantic relation of ontology to be more consistent with human understanding, enhancing the power of keyword query supporting, extracting the entities from the WWW, and so on. It is also a future work to expand Sniper to other domains.

Acknowledgement:

Fund project: scientific research of Hei Long Jiang agricultural reclamation bureau（HNK125BZD-04-21）.

REFERENCES

1. Brin S, Page L; The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems, 1998.
2. Powerset website. [Online]. Available: http://www.powerset.com/

3. Hakia website. [Online]. Available: http://www.hakia.com/
4. Harth A, Umbrich J, Decker S; MultiCrawler Crawling and Idexing Semantic Web data. Proc. 2006 In 5th International Semantic Web Conference.
5. Ding L, Finin T, Joshi A, Pan R, Cost RS, Peng R, et al; Swoogle : a search and metadata engine for the semantic web. Proc. 2004 Information and knowledge management,
6. D’Aquin M, Sabou M, Dzbor M, Baldassarre C, Gridinoc L, Angeletou S, et al.; Watson: A Gateway for Next Generation Semantic Web Applications. Poster session of the International Semantic Web Conference, ISWC 2007.
7. Cheng G, Ge W, Qu Y; Falcons: searching and browsing entities on the semantic web. In Proceedings of the 17th international conference on World Wide Web , 2008; 1101-1102.
8. Dodds L; Slug: A Semantic Web Crawler. Jena User Conference, Bristol , 2006.
9. Gruber TR; A translation approach to portable ontology specifications. Knowl. Acquis., 1993.
10. Hu W, Jian N, Qu Y, Wang Y; GMO: A Graph Matching for Ontologies. Ontologies, K-CAP Workshop On Integrating, Banff, Canada, 2005
11. Juanzi L, Jie T, Yi L, Qiong L; RiMOM: a dynamic multistrategy ontology alignment framework," IEEE Transactions on Knowledge and Data Engineering, 2009.
12. Jean-Mary YR, Shironoshita EP, Kabuka MR; Ontology matching with semantic verification. Journal of Web Semantics, 2009.
13. François J, Kengue D; OLA in the OAEI 2007 Evaluation Contest. ISWC + ASWC Workshop on Ontology Matching, 2007