SEMANTIC INFORMATION RETRIEVAL USING ONTOLOGY AND SPARQL FOR CRICKET

S. M. Patil[1], D. M. Jadhav[2]
[1]Information Technology Department, BVCOE, Navi Mumbai, Maharashtra, India
[2]Information Technology Department, PIIT, New Panvel, Maharashtra, India.

ABSTRACT

In this paper we present semantic information retrieval using Ontology and SPARQL and its application to cricket domain. In the first stage, domain ontology is designed for the cricket domain. Data is extracted from the specific domain and stored in the domain ontology. Ontology storage is done by using web ontology language (OWL). New relations are added in the knowledge base by using the reasoner. Pallet reasoner is used for the inferencing purpose. The searching is done on the ontology with the help of SPARQL. SPARQL query language is able to answer the complex query asked by users. Graphical user interface is provided for simplifying the SPARQL searching process. Traditional search is provided for comparison purpose. Domain specific information extraction improves the retrieval performance of the system. Detailed evaluation is provided to compare the performance gain against the traditional methods. System is able to achieve the usability, scalability and retrieval performance. The system is implemented using latest technology like Ontology, RDF, inference, SPARQL.

KEYWORDS: *Ontology, OWL, SPARQL, Inference, Semantic Web.*

I. INTRODUCTION

The amount of data available on World Wide Web has increased tremendously. Searching the useful information on a huge data is the topic in discussion today. Current web mostly relies on keyword based search. Most of the systems are implemented using vector space model. The performance of the system depends on matching the keyword within the available data. Such a model misses the actual semantic information in text [1].

The disadvantage of the traditional search can be overcome with the proposal of semantic web. Semantic web is also called the intelligent web or next generation web or Web 3.0. Semantic web is approach towards understanding the meaning of the contents. Semantic information is stored in the form of ontology. Ontology is nothing but specification of a conceptualization. Today Ontologies are the backbone of the semantic web. Information extraction and retrieval is benefitted with advent of ontology. Semantic data is published in the form of language like RDF, OWL, and XML. After obtaining the semantic information from the plain text next step is finding the required information. Semantic indexing the semantic data and doing the keyword search is one of the techniques [2].

In general search engines aims towards achieving the usability, scalability and retrieval performance. Currently studies on semantic search deals with keyword based, form based and natural language based query interface. Out of this keyword query interface is the most user friendly one. Combining the usability of keyword interface with power of semantic technology is one of the challenging task in semantic searching.

In this paper we present semantic information retrieval using Ontology and SPARQL for cricket. Central ontology is designed for the cricket domain which is used during information extraction, ontology mapping and inference. Information is extracted from the cricket domain and stored in the ontology. The searching is done on the ontology data with the help of SPARQL. To simplify the searching process, graphical user interface is provided. Traditional search is designed to compare the performance of the system against the SPARQL search. Detailed evaluation is provided to compare the performance gain against the traditional methods. The system consists of automated crawling, information extraction module, ontology mapping module, inference module, SPARQL search module.

The rest of the paper is organized as follows: literature survey in section-II, section-III gives detailed problem statement, section-IV about system implementation; section-V provides the results and discussion whereas conclusion and future work of paper is given in section VI.

II. Literature Survey

Classical search engines are depends of vector space model. Generally vector space model carried out in three stages namely document indexing, term weighting, similarity coefficients. Document indexing is done in the first stage. Importance of the term within the document is calculated with the help of term frequency in the second stage. In the third stage, documents and queries are represented by vectors of term weight and retrieval is done by cosine similarity. This method does not require any extraction method. Problem with vector space model is documents are represented poorly, search keyword must be precise, document with similar context but different term won't be associated. This method gives very low precisions and recall values [1].

Traditional methods are unable to understand the meaning of the contents. With the introduction of semantic web, knowledge representation is very much easier. Semantic to plain text can be given by using ontology. Information is extracted from specific domain and stored in the ontology instances. The retrieval model is based on an adaptation of the classic vector-space model, including an annotation weighting algorithm, and a ranking algorithm. They are able to achieve the greater performance as compared to classical search engine [3], [4].

Semantic indexing is one of the techniques to improve precision and recall values. Semantic index is created on ontological data and keyword search is performed. Semantic indexing approach is based on the lucene indexing. Basic idea is to extend the traditional full text index with extracted data. Ranking is modified so that document containing ontological data get higher rates. Performance of the system is improved as compared to ontology search using vector space model. But still it is not able to achieve the 100% precision and recall values [2].

SPARQL can be used to query ontological data. Using SPARQL we can achieve the 100% precision values. Problem with SPARQL is that, it is not made for end users as we require the knowledge of domain ontology and syntax of the language. Therefore, Semantic Web community works on simplifying the process of query formulating for the end-user. In Jan 2008, W3C made the SPARQL as recommendation for semantic query language [5]. With SPARQL we get the answer of the very complex query [6].

From the above details, it is known that keyword based search are very user friendly but they are not suitable for large information repository and not able to capture all the semantic of the query. Aim of the system presented in this paper is to keep the user friendly of the search query as well as improve the retrieval performance. The framework is applied to cricket domain to observe the performance of the system. Remarkable increase in the performance of the system is possible with SPARQL.

III. Problem Statement

Semantic information retrieval is done with the help of Ontology and SPARQL. System is implemented for the cricket domain. Data is extracted from the cricket domain and stored in the ontology. Ontology is stored in OWL format which is based on RDF and RDFS. Inference is applied on semantic data by using pallet reasoner, which check the consistency of the database and able to add new relations. Searching is done with the help of SPARQL. Graphical user interface is provided to enter the SPARQL query.

IV. System Implementation

Fig. 1 shows the architecture of the overall system. System is divided into different modules for better the management. The important module within the system are crawling, plain text search, Information extraction, Ontology design, Ontology mapping, Inference, semantic search with SPARQL. The system is implemented in Java with the support of Jena framework and pellet reasoner. Implementations details of the modules are as follows.

4.1 Crawling

This is the first module in our system. It takes the web URL as input and crawl all the pages of the website. After crawling, the web pages are stored on local machine by removing the unwanted content from HTML pages. HTML parser removes the unwanted spaces, extra lines, HTML tags, image tags, scripts, comments from the pages. System crawls only the commentary pages as these pages have the most of the match information. For testing purpose we have crawled the specific series in the cricket domain. For example IPL2012, IPL2011, World Cup 2011, etc. The crawling process is shown in fig.2.

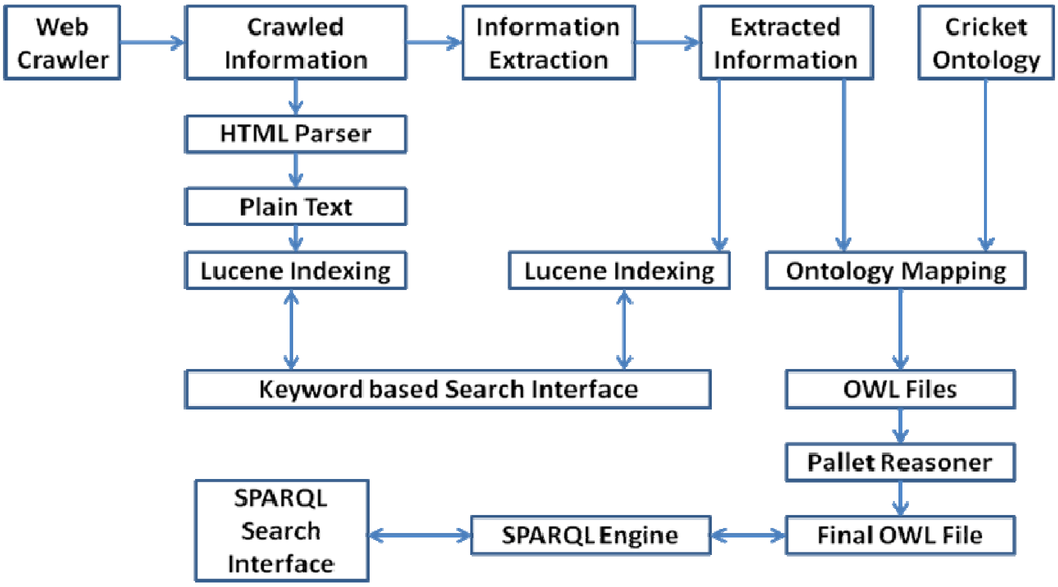


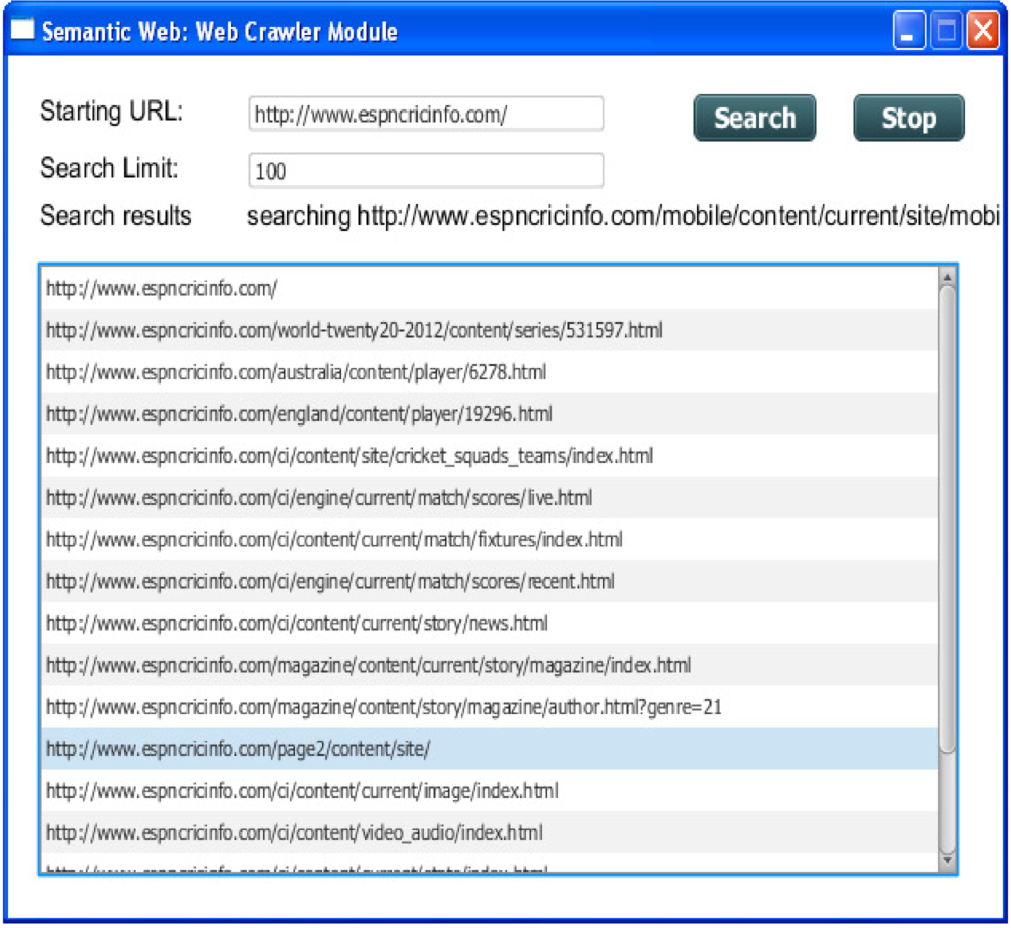
Fig. 1: System Architecture.

4.2 Plain text search

In this module indexing is done on the plain text. Index preparation is done with the help of Apache Lucene. Apache Lucene is a high-performance, full-featured text search engine library written entirely in Java. Apache Lucene is open source project freely available [15]. Input for indexing is plain text which is available from the previous module that is crawling and HTML parser. Searching is keyword based searching. Search gives the list of documents in which the given keyword matches with words in the documents. This is just like traditional search engine. This search engine is used for comparison purpose only.

4.3 Ontology Design

Central ontology is designed for cricket domain. Domain ontology is used during information extraction, creating the OWL files and inference. First step in designing the ontology is identifying the different classes in the particular domain. Class in the ontology may have number of instances. Instance may belong to none, one or more classes. Class may be a subclass of another class. All classes are subclasses of owl: Thing the root class. In the cricket domain identified classes are Ball, Event, Inning, Location, Match, Over, Player, Series, Stadium, Team, etc. After identifying the classes next step is identifying the properties of the classes. Class characteristics are specified by Properties. They are attributes of instances and sometimes act as data values or link to other instances. Properties can be object properties or datatype properties. Datatype properties are relations between instances of classes and RDF literals whereas Object properties are relations between instances of two classes. Identified Object properties in cricket domain are ballBy, ballTo, hasInning, hasMatch, hasOver, hasPlayer, hasStadium, etc. Datatype properties are hasBall, hasName, hasCity, hasDate, hasEvent, hasRR, hasRRR, etc. Protégé is used for ontology design [7], [14]. Fig.3 shows the class hierarchy in cricket domain.



Starting URL: http://www.espncricinfo.com/

Search Limit: 100

Search results searching http://www.espncricinfo.com/mobile/content/current/site/mobi

```
http://www.espncricinfo.com/
http://www.espncricinfo.com/world-twenty20-2012/content/series/531597.html
http://www.espncricinfo.com/australia/content/player/6278.html
http://www.espncricinfo.com/england/content/player/19296.html
http://www.espncricinfo.com/ci/content/site/cricket_squads_teams/index.html
http://www.espncricinfo.com/ci/engine/current/match/scores/live.html
http://www.espncricinfo.com/ci/content/current/match/fixtures/index.html
http://www.espncricinfo.com/ci/engine/current/match/scores/recent.html
http://www.espncricinfo.com/ci/content/current/story/news.html
http://www.espncricinfo.com/magazine/content/current/story/magazine/index.html
http://www.espncricinfo.com/magazine/content/story/magazine/author.html?genre=21
http://www.espncricinfo.com/page2/content/site/
http://www.espncricinfo.com/ci/content/current/image/index.html
http://www.espncricinfo.com/ci/content/video_audio/index.html
http://www.espncricinfo.com/ci/content/current/stats/index.html
```

Fig. 2: Crawling

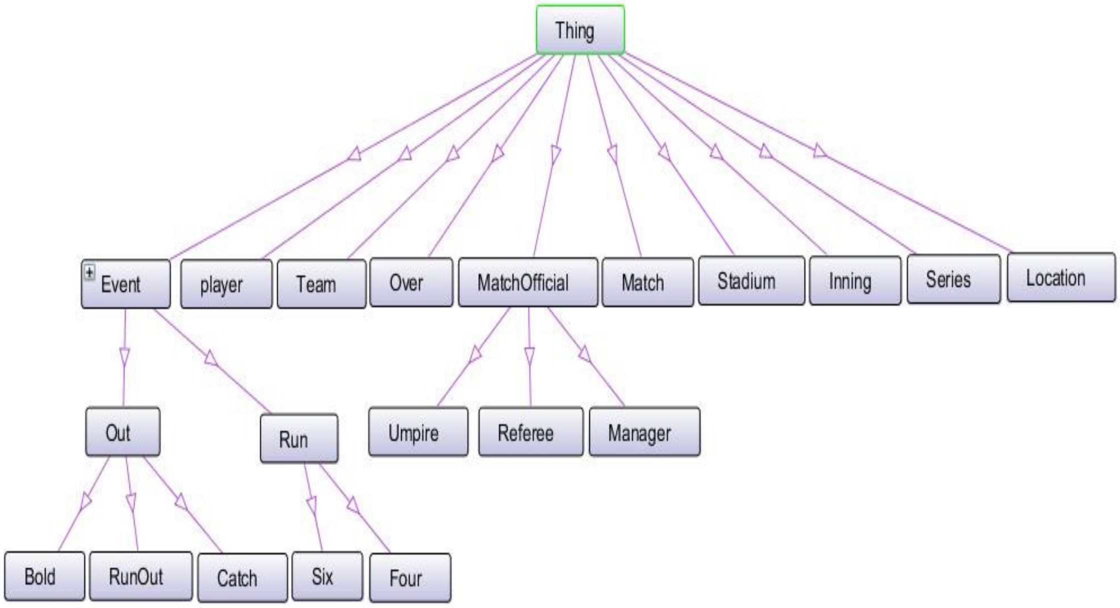


Fig. 3: Class hierarchy

4.4 Information extractions

It is a process of adding structured information from unstructured resources. In this phases system uses the data crawled from the cricket websites. Input for this module is basic information and narrations of the matches as only the commentary pages are crawled. This module collects the information like series, type of the match, teams involved, narrations, over by over data of the match, etc. Information extraction becomes easy after studying the structure of web pages. System is able to crawl almost all the information of the match as espncricinfo.com website is structured very nicely. Information collected in this phase is given to ontology mapping.

The algorithm for the information extraction is given below.

Step 1: Start

Step 2: Create ontology file for storing the semantic data in OWL format.

Step 3: Copy original cricket ontology in the Ontology file.

Step 4: Get the first file from the crawler module.

Step 5: Search seriesText class in the HTML page and create the isSeries event and store in the Ontology and extracted file.

Step 6: Locate the match number, create the hasMatch event and store in the file.

Step 7: Search teamText class, create ontology events and store in the file.

Step 8: Search statusText class, create event and store in the file.

Step 9: Search playedAt class, create hasStadium and hasDate event and store in the ontology file.

Step 10: Identify commsTable class, get the ball by ball details and create events for run, wicket, four, six, bowler, batsman, etc and save in the ontology file.

Step 11: Get the over by over details from the commsTable class, extract the details like overNumber, run in that over, wicket in that over, run rate, required run rate, etc. Create the event and save it in the file.

Step 12: Repeat the Step 4-11 for all the files in the crawler directory.

Step 13: Stop.

For example, consider the following piece of commentary.

Indian Premier League - 1st match

Chennai Super Kings v Mumbai Indians

Mumbai Indians won by 8 wickets (with 19 balls remaining)

Played at MA Chidambaram Stadium, Chepauk, Chennai

4 April 2012 - day/night (20-over match)

7:10 pm it's the start of the match. Bowler and batsman are ready for the first delivery.

0.1 Warne to Sachin, **SIX**, superb shot, takes it on the full and tucks it to deep midwicket.

Information extraction modules extract the following data from above piece of commentary.

Series: Indian Premier League
Match: 1st match
Team1: Chennai Super Kings
Team2: Mumbai Indians
Status: Mumbai Indians won by 8 wickets (with 19 balls remaining)
Playedat: MA Chidambaram Stadium, Chepauk, Chennai
PlayedOn: 4 April 2012

Ball: 1
Ball By: Warne
Ball To: Sachin
hasOverNumer: 1
Event: SIX
Description: superb shot, takes it on the full and tucks it to deep midwicket.

4.5 Ontology mapping

It is process of mapping unstructured, structured, semi-structured data into ontology instances. Information extraction module has done lots of work by extracting most of structured information. After mapping data in ontology instances, OWL individual for each event is created. If IE module not able to extract some attributes of the event then also we create an instance with empty property. Jena framework is used for ontology mapping. Jena framework provides the facility for reading, writing and manipulating the data in RDF and OWL format [8]. Code of snippet in Appendix shows the ontology mapping for the above crawled information.

4.6 Inference

New relation can be added from the existing database with the help of class and subclass relations. Pellet provides the complete OWL-DL reasoner with good performance and number of unique features. Pellet provides reasoning services, including consistency checking, concept satisfiability, classification, and realization [9]. Basic idea is to load the ontology schema from OWL file to Pellet reasoner, compute class subsumption tree and store it back in OWL instances.

4.7 Semantic Search

Semantic searching is done with the help of SPARQL. SPARQL is an RDF query language which is able to retrieve and manipulate data stored in Resource Description Framework format. SPARQL allows users to write unambiguous queries. SPARQL is just like the SQL only. SPARQL contains capabilities for querying required and optional graph patterns along with their conjunctions and disjunctions [5]. The results of SPARQL queries can be results sets or RDF graphs. Query is specified in the triple pattern which is subject, object and predicate. Query syntax is given below.

```
//Prefix declaration

//Select clause

//Where clause
```

The example of SPARQL query is given below which select all players from cricket database.

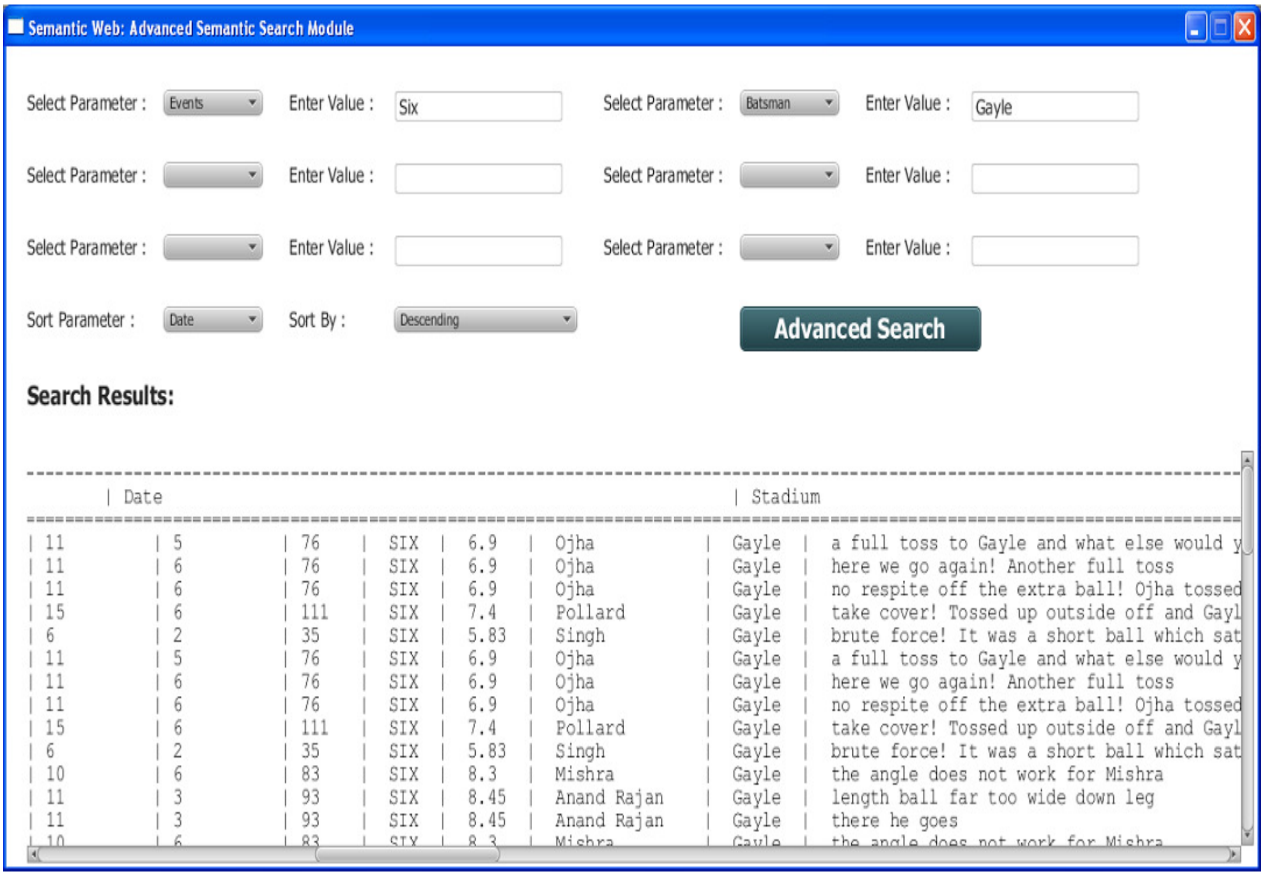
```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX ns:
```

```
<http://www.semanticweb.org/ontologies/2012/3/cricket.owl#>
select  ?team ?date  ?over  ?event ?Bowler ?Batsman where {
?m rdf:type ns:Match .      ?m ns:hasTeam ?a  .
?a ns:hasName ?team.
?m ns:hasDate ?date . ?m ns:hasTeam ?t .
?m ns:hasInning ?i .  ?i ns:hasOver ?o .
?o ns:hasOverNumber ?over .
?o ns:hasBall ?b .    ?b ns:hasEvent ?event .
?b ns:ballBy ?c .     ?c ns:hasName ?Bowler .
?b ns:ballTo ?e .     ?e ns:hasName ?Batsman .
```

Even we can write very complicated query on database. Graphical user interface is provided for simplifying the query writing process. Graphical user interface allows users to select certain fields and get the result instead of writing the complicated query. The algorithm for advanced search is given below.

Fig. 4 shows the SPARQL query interface.



Select Parameter : Events Enter Value : Six Select Parameter : Batsman Enter Value : Gayle

Select Parameter : Enter Value : Select Parameter : Enter Value :

Select Parameter : Enter Value : Select Parameter : Enter Value :

Sort Parameter : Date Sort By : Descending

Search Results:

```
---------------------------------------------------------------------------------------------------
       | Date                                                 | Stadium
===================================================================================================
| 11        | 5         | 76    | SIX | 6.9  | Ojha           | Gayle | a full toss to Gayle and what else would y
| 11        | 6         | 76    | SIX | 6.9  | Ojha           | Gayle | here we go again! Another full toss
| 11        | 6         | 76    | SIX | 6.9  | Ojha           | Gayle | no respite off the extra ball! Ojha tossed
| 15        | 6         | 111   | SIX | 7.4  | Pollard        | Gayle | take cover! Tossed up outside off and Gayl
| 6         | 2         | 35    | SIX | 5.83 | Singh          | Gayle | brute force! It was a short ball which sat
| 11        | 5         | 76    | SIX | 6.9  | Ojha           | Gayle | a full toss to Gayle and what else would y
| 11        | 6         | 76    | SIX | 6.9  | Ojha           | Gayle | here we go again! Another full toss
| 11        | 6         | 76    | SIX | 6.9  | Ojha           | Gayle | no respite off the extra ball! Ojha tossed
| 15        | 6         | 111   | SIX | 7.4  | Pollard        | Gayle | take cover! Tossed up outside off and Gayl
| 6         | 2         | 35    | SIX | 5.83 | Singh          | Gayle | brute force! It was a short ball which sat
| 10        | 6         | 83    | SIX | 8.3  | Mishra         | Gayle | the angle does not work for Mishra
| 11        | 3         | 93    | SIX | 8.45 | Anand Rajan    | Gayle | length ball far too wide down leg
| 11        | 3         | 93    | SIX | 8.45 | Anand Rajan    | Gayle | there he goes
| 10        | 6         | 83    | SIX | 8.3  | Mishra         | Gayle | the angle does not work for Mishra
```

Fig. 4: SPARQL query interface

V. RESULTS AND DISCUSSION

To evaluate the performance of the system, we have crawled all the matches of IPL (Indian Premier League)-2012. IPL-2012 has in all 71 matches. Information extraction module is able to find out 20343 events within the 71 matches. The first index is created on the data crawled by web crawler. This index is traditional plain text index. This is index is based on vector space model. Index is created with the help of Apache Lucene. This searching is referred as `TRADITIONAL SEARCH.`

Second index is created on the data extracted from the information extraction module. Indexing is done like the traditional plain text search. Again indexing is done using the concept of vector space model with the help of Apache Lucene. Input to indexing is extracted data by information extraction module. This is search is referred as *EXTRACTED SEARCH*.

Third type of searching is directly done without any type of indexing. Searching will be done on the ontology data with the help of SPARQL search. Searching is referred as SPARQL search. Complex of the complex query can be written using SPARQL. Simple queries can be executed using the graphical user interface provided by the system. Graphical user interface contains the some drop down list from which parameter can be chosen or values can be entered. This search is referred as *SPARQL SEARCH*.

The TABLE-I gives the evaluation queries whereas TABLE-II gives result of the evaluation. We could answer first three queries because the winner of the series, name of player and name of team have recorded in the ontology data. Next three queries (Query 4-6) could answer because ontology is storing the bowler and batsman of every event occurred in the matches. Next three queries (Query 7-9) are able to answer as the winner and loser of the matches are storing in the ontology data. TABLE-II gives the percentage of relevant (precise) result produced for the given query.

TABLE-I: Evaluation Queries

Query-1	All sixes in IPL 2012
Query-2	All wickets in IPL
Query-3	All sixes by Gayle
Query-4	All wins by Mumbai Indians
Query-5	All wins by Mumbai Indians in Mumbai
Query-6	All Matches of Chennai super kings in Chennai in May 2012
Query-7	Find all six on Zaheer's bowling against Kolkata.
Query-8	Find all six in second over and second ball of each match.
Query-9	Find Sachin's six at Mumbai in April 2012

TABLE-II: Evaluation Results

	TRADITIONAL SEARCH	*EXTRACTED SEARCH*	*SPARQL SEARCH*
Query-1	12%	20%	99%
Query-2	12%	18%	100%
Query-3	10.5%	16%	98%
Query-4	10%	16.5%	98%
Query-5	7.5%	12%	97%
Query-6	7.5%	8%	95%
Query-7	5%	6.5%	92%
Query-8	5%	4%	90%
Query-9	4.5%	2.5%	90%

The graph of the search evaluation is shown in the fig.5. The graph is drawn for query against the average precision value.

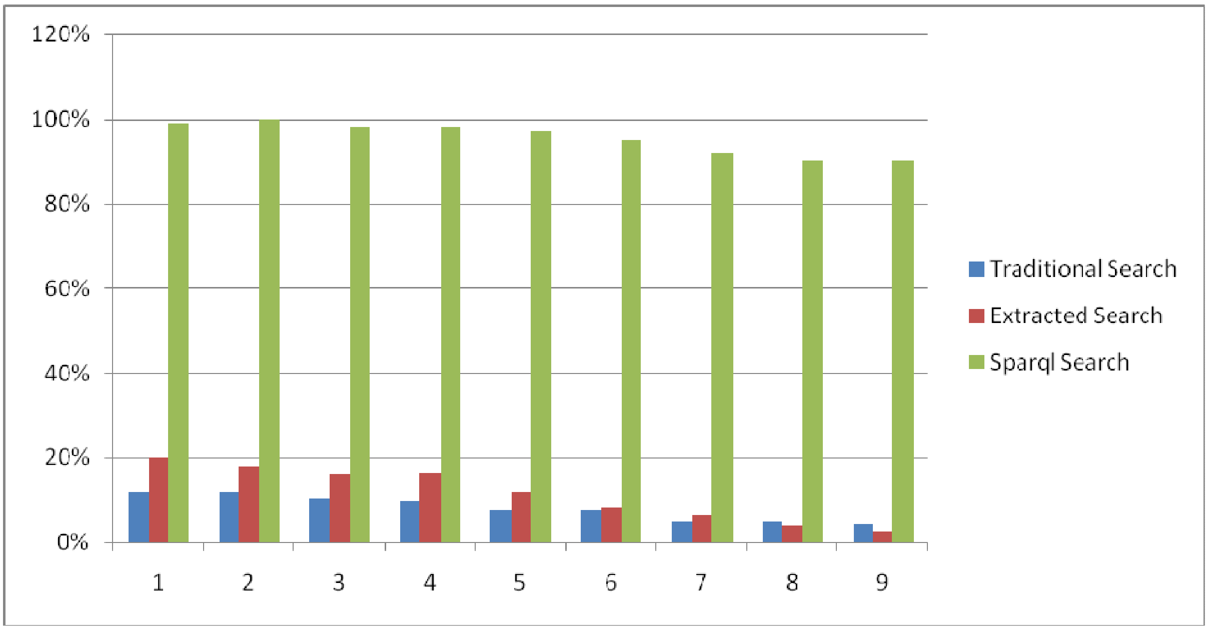


Fig.5: Search evalaution graph

VI. Conclusion and Future Work

We have presented the semantic information retrieval framework and its application to Cricket domain. The system is implemented using the most cutting edge technology like Ontology, OWL, Inference, information extraction, Ontology development and mapping, SPARQL. Considerable increase in the performance of the system using domain specific information extraction is observed. With the help of inference, performance is further improved. Very complex query asked by the user can be answered using SPARQL. Graphical user interface made easy to construct the SPARQL query otherwise it is very complicated to write the query. System is able to achieve the greater precision and recall values.

With the successful implementation of the system for cricket domain, we can extend the system for other domain with the changes in the domain ontology and information extraction. Other modules can be easily used in the new system without any changes. System can be extended for storing the semantic information from multiple languages. The concept of semantic information retrieval can be applied for image retrieval also. As shown in the result section, we can achieve better recall and precision values for the other domain and for multilingual databases.

References

[1] Jhk G. Salton and M.J. McGill, Introduction to Modern Information Retrieval, McGraw-Hill, New York, 1983.

[2] Soner Kara, Ozgur Alan, Orkunt Sabuncu, Samet Akpınar, Nihan K. Cicekli, Ferda N. Alpaslan, “An Ontology-Based Retrieval System Using Semantic Indexing”, IEEE, ICDE workshop 2010.

[3] Vallet, D., Fernández, M., Castells, “An Ontology-Based Information Retrieval Model”, 2nd European Semantic Web Conference (ESWC 2005).

[4] Paralic, J., Kostial, I., “Ontology-based Information Retrieval”, Information and Intelligent Systems, Croatia (2003).

[5] SparQL Protocol for RDF, K. Clark, Editor, W3C Recommendation, 15 January 2008, http://www.w3.org/TR/2008/REC- rdf-sparql-protocol-20080115.

[6] Jun Zhai, Kaitao Zhou, “Semantic Retrieval for Sports Information Based on Ontology and SPARQL”, International Conference of Information Science and Management Engineering, 2010.

[7] Natalya F. Noy and Deborah L. McGuinness, Stanford University, Stanford, “Ontology Development 101: A Guide to Creating Your First Ontology”.

[8] Jena, A.: Semantic Web Framework for Java, http://jena.sourceforge.net/ontology/index.html.

[9] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur , Yarden Katz, “Pellet: A Practical OWL-DL Reasoner”, publiced by Elsevier B.V, 2007.

[10] Wen-tau Yih, “Template-based Information Extraction from Tree-structured HTML documents”, Citeseerx, 1997.

[11] M. Grobe, “RDF, Jena, SparQL and the Semantic Web”, in Proceedings of the ACM SIGUCCS fall conference on User services conference, 2009.

[12] J. Huang, D. J. Abadi, and K. Ren. “Scalable SPARQL querying of large RDF graphs”. In VLDB, 2011.

[13] Jens Lehmann, Lorenz Buhmann, “AutoSPARQL:Let Users Query Your Knowledge Base”, 2011.

[14] Protégé, http://www.protege.stanford.edu/.

[15] Apache Lucene, http://lucene.apache.org/core/.

Appendix: Code snippet for Ontology Mapping.

```
<rdf:type rdf:resource="http://www.semanticweb.org/ontologies/2012/3/cricket.owl#Series"/>
<hasName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">Indian Premier League
</hasName>
<owl:NamedIndividualrdf:about="http://www.semanticweb.org/ontologies/2012/3/
cricket.owl#1st_match">
<rdf:typerdf:resource="http://www.semanticweb.org/ontologies/2012/3/cricket.owl#Match"/>
<hasDate rdf:datatype="http://www.w3.org/2001/XMLSchema#string">4 April 2012 </hasDate>
<hasStatus rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
        Mumbai Indians won by 8 wickets (with 19 balls remaining)
</hasStatus>
<hasTeam rdf:resource="http://www.semanticweb.org/ontologies/2012/3/cricket.owl#Chennai Super Kings
"/>
<hasStadium rdf:resource="http://www.semanticweb.org/ontologies/2012/3/cricket.owl#MA Chidambaram
Stadium, Chepauk, Chennai"/>
<hasTeam rdf:resource="http://www.semanticweb.org/ontologies/2012/3/cricket.owl#Mumbai Indians"/>
 </owl:NamedIndividual>
<rdf:type rdf:resource="http://www.semanticweb.org/ontologies/2012/3/cricket.owl#Ball"/>
<hasRun rdf:datatype="http://www.w3.org/2001/XMLSchema#string"> SIX</hasRun>
<hasDescription rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
          Superb shot, takes it on the full and tucks it to deep midwicket
</hasDescription>
<hasOverNumber rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">
          1
</hasOverNumber>
 <hasBallNumber rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">
          1
</hasBallNumber>
<ballBy rdf:resource="http://www.semanticweb.org/ontologies/2012/3/cricket.owl#Warne"/>
 <ballTo rdf:resource="http://www.semanticweb.org/ontologies/2012/3/cricket.owl#Sachin"/>
```

AUTHORS

S. M. Patil has received his graduation from Shivaji University, Kolhapur and post-graduation in Computer Engineering from Bharati Vidyapeeth University, Pune. He is having 14-years of work experience in teaching in Engineering College. He has published 15 national and international papers on various topics. Currently seeking the opportunity for doing Ph.D from a best recognized college of India. His main research interest is Image Processing.

D. M. Jadhav has received his graduation in Computer Engineering from Pune University, Pune. Currently he is pursuing his post-graduation in Information Technology. He has published one international paper on semantic information retrieval. His research interest is in Information Retrieval.