# Semantics Oriented Web Searching*

Junzhong Gu

*Institute of Computer Application (ICA), East China Normal University*
*3663 Zhongshan Road N., 200062 Shanghai, China*
*jzgu@ica.stc.sh.cn*

***Abstract***

*Current Web search can't fully satisfy users' requirements. This is also due to the fact that it is term based rather than concept based or even object oriented. It is studied here, how to extend a search function semantically in order to improve its functionality and performance. It's proved that semantics oriented Web searching is a good solution to enable concept/object oriented Web search, faceting search, as well as associate and relationship search. The knowledge supporting for semantics extension is proposed and a corresponding ontology solution is designed. At last, a case study named SmartSearch, launched at ECNU-ICA since 2011, is presented. The case study shows that semantics oriented Web search is powerful and that the search results will be efficiently improved with a semantic extension.*

***Key Words:*** *ontology, Web Search, semantics extension, concept oriented Web search, associate search, faceting search*

## 1. Introduction-Web Searching

Internet is the largest and most diverse source of knowledge (information) in the world now. It is a challenge to find the information especially in the form of Web pages from huge amounts of data. Search engines, such as Google, Bing and Baidu[1], receive several hundred million queries per day. They have to be efficient in order to respond to each user within seconds and effective so that returned results are relevant to information needs of users.

Search engines crawl the Web periodically in order to obtain the latest possible content of the Web. Inverted index files are the state-of-the-art data structure for efficient retrieval. Finally, queries are processed over the inverted index using a retrieval (ranking) function that computes scores for documents based on their relevance to the query and documents with highest scores are returned as the query result. Additionally, result pages containing *title*, *url* and snippet (*i.e.*, two or three sentence summary of the document) information are prepared and displayed.

Unfortunately, users always complain about the search results, because most of them are garbage.

For example, in order to get information about the famous Great Wall, I used Google to search Internet with the query phrase "Great Wall" on 27. November 2012. It returned about 286,000,000 links to me. This is a huge amount, but most of the links for me are garbage. Figure 1 is the first page returned, where the 5th result is a link to www.greatwall.com.cn, which is a home page of a computer company.

* This work is supported by the Shanghai Scientific Development Foundation with Grant No. 11530700300.
[1]A search engine (http://www.baidu.com/)


**Figure 1. The First Result Page Searching "Great Wall"**

The 6th is a home page www.greatwall.cn/, which is also about the computer manufacturer.

The 7th is a link to www.greatwall.org/, which is about a school "Great Wall Chinese School". Other results involve Great Wall Motors, Great Wall Marathon and so on. The first result searched by Baidu is about Great Wall Motors-an automobile manufacturer.

Why did this happen? From my point of view, the reason is that the search is based on terms, rather than being concept oriented, or even object oriented. The phrase "Great Wall" in the query has various meanings, *e.g.*, it denotes the object of ancient Chinese architecture, or it is used as a trade mark or as the name of a school. A common feature of the returned documents is that their texts contain the phrase "Great Wall". No distinction is made with respect to the actual semantics of the phrase. Thus too much garbage is returned to the user.

If we restrict the search phrase "Great Wall" to the concept "architecture", the results will focus on Great Wall in the sense of architecture. Furthermore, with the restriction to the object -China's Great Wall, the results will exactly match the users' intention:

"*China's Great Wall as the world's longest architectural structure and is widely renowned as one of the seven great wonders of the world.*"

As shown, a semantics based search would be more ideal than the current semantic-free term based search. Therefore semantics oriented Web search is proposed here. A semantics oriented search engine should have the ability to automatically bind and transform the term(s) appearing in user query to the corresponding concept, even to the corresponding object following the user's intention.

In current Web search, in most cases a query is expressed by one or two words (most of them are in one word). Therefore we assume that a user query is expressed by a term (*e.g.*, wall) or a phrase (*e.g.*, Great Wall). (For convenience, we use term to denote both.)

A smart Web searching can automatically implement the following mappings (noted as $\delta_1$ and $\delta_2$):

$$\delta_1(\text{query term})=\text{concept}$$

$$\delta_2(\text{concept})=\text{object}$$

After the mapping, the search is concept oriented, and then object oriented. The returned search results will correspond more precisely to user requirements.

Since 2010, a project for semantics extended Web search is launched at the Institute of Computer Application (ICA), East China Normal University. Its target is to study the approach to satisfy concept and object oriented Web search.

## 2. Semantics Oriented Web Search

A user query can be illustrated as an expression, such as

"Great + Wall"

It means that the searched documents contain a term "Great", immediately followed by the term "Wall" in text. Plus (+) is used here as a logical operator "AND" (conjunction)[2]. Similar operators are "OR" (disconjunction), noted as minus ("-")[3], *etc*. The interpretation of these expressions does not include any semantic interpretation of the terms involved.

The semantic interpretations which users are more interested in include the following cases:

1) Binding a query term to a particular concept or object

A term based query (*e.g.*, "Great Wall") can be semantically extended with the binding to a particular concept, for example, in the form of:

Great Wall & <Concept>

Great Wall & <Object>

Here <Concept> means a particular concept, *e.g.*, architecture, trade mark, and so on. <Object> refers to a particular object. & is used as an operator and means as *bind to*, *e.g.*,

Great Wall & *architecture*

This expression means, that a user wants to find information about "Great Wall" in the domain (concept) of "architecture". Bound to a particular concept, searched results will focus on the desired target.

The binding can be extended to a particular object. For example, "Great Wall" can be bound to the object, *Chinese Great Wall*. Then the search results will be confined exactly to information about the famous ancient Chinese work of architecture-*Chinese Great Wall*.

The binding can be automatically implemented. It is based on knowledge inference according to the analysis of user behaviors, query context, user profile, and so on.

2) Associate and Relationship search

Users are more interested in associate search as well as relationship search, rather than simple information on individuals. For example, users are interested in finding the relationship between objects.

- Finding the relationship between objects

Users can find the relationship between two objects or object(s) corresponding to a further object with a particular relationship. For example:

  - ✧ Finding some object related to another object, for example:

[2] Assume that here conjunction as a default operator, such as "Great Wall" means "Great + Wall".
[3] Minus (-) here means normally, exclude.

<YAO Ming, father-of, ?>

It means, a user wants to find the child's name of the famous Chinese basketball player, YAO Ming, from the Internet.

✧ Finding the relationship between objects, *e.g.*,

<YAO Ming, YE Li, ?REL>

That means a search for the relationship between YAO Ming and YE Li[4].

- Query by Example

"Query by Example" is a user-friendly interface. Users prefer to query with examples, because it's intuitionistic and convenient.

For example, taking <Obama, USA> to describe a relationship between Mr. Obama and United States, it matches the relationship *president-of*. If a user wants to know, who is the President of Russia, then he/she can issue a query expression:

<Obama, USA>, <?, Russia>

Here, the left hand side of the expression is an example denoting the association between Obama and United States. The right hand side is a question to find a similar relationship. As a result, the search engine will return a list of information about President **Putin**, the President of Russia.

3) Faceting Search

Users are usually only interested in some aspects of a particular concept or object. For example, if a query "Shanghai" is issued to a search engine, Web pages containing the term "Shanghai" will be returned to user. But possibly the user only wants to get information on some aspects of Shanghai, such as its population and climate. In this case a multiple choice menu for faceting search can be presented to a user: after a user query (*e.g.*, "Shanghai") has been processed, the returned result contains a choice menu listing all possible facets that might be interesting for the user, as shown in Figure 2.

Users can then choose interested facets by marking corresponding boxes (□) or by directly clicking on the facet link inside the pane. Faceting can also be implemented as an automatic routine without manual selection. The routine is based on analyzing user behaviors, user profiles, as well as his/her search history. Thus, the smart search engine can automatically select facets and focus on the aspects of the concept/object which user may be interested in.

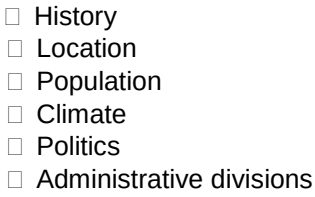


**Figure 2. Proposed Faceting Menu for the Concept City**

Usually, faceting is an interaction style where users filter a set of items by progressively selecting items from valid values of a faceted classification system. Faceting enhances search

[4] YAO Ming, Chinese basketball player, who became an international star as a centre for the Houston Rockets of the National Basketball Association (NBA). YE Li is his wife.

results with aggregated information over all the documents found in the search rather than on the entire index. A way for achieving this is using facet queries in which a complex query is specified as a facet. To select a facet, the user chooses corresponding items from the selection menu (*e.g.*, masking the □ as in Figure 2). New results based on the facet are displayed on the page. Different concepts have different facets, which will be bound automatically by the ontology system. It will be presented in the next section.

## 3. Smart Search Engine

A search engine is normally composed of a crawler, a query interface, a corpus management system, an index system and some other components as illustrated in Figure 3.

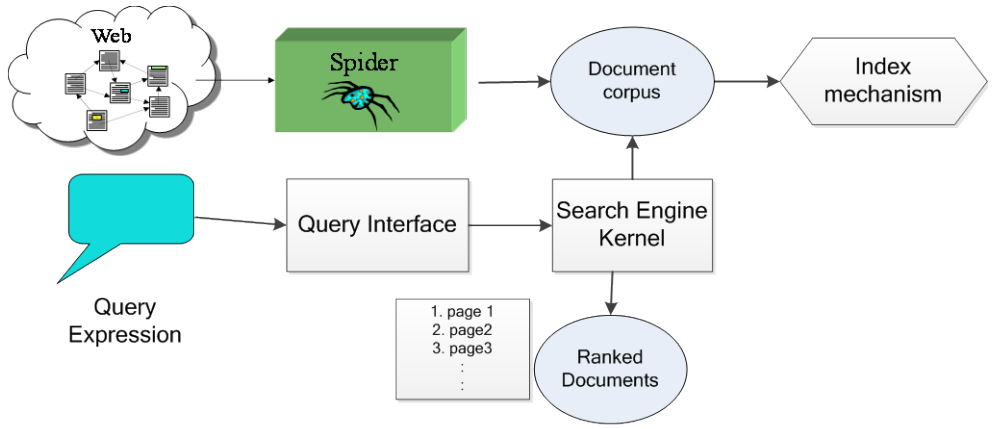


**Figure 3. Search Engine**

As shown in the figure, a user issues his/her query in the form of query expression to the query interface. After a pre-processing the request is passed to the Search Engine Kernel, which checks the Documents corpus for suitable documents. If documents are found, it ranks and returns them to the user. Otherwise, it crawls Internet to find documents using a spider, then ranks and returns them to the user.

For the semantic extension, the knowledge based support for Web search is designed by us, as shown in Figure 4. The user interface is replaced by a Semantics Extended Query Interface. For supporting the semantics extension we are engaged in the development of three knowledge based modules. They are a semantics support system, an inference system and an ontology system, as shown in Figure 4.

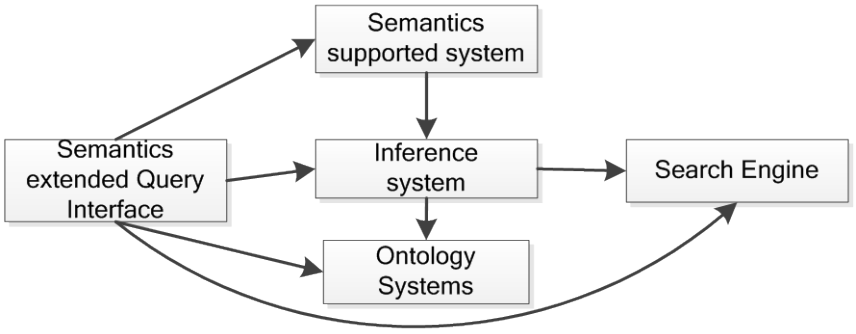


**Figure 4. Knowledge based Support for Web Search**

### 3.1. Knowledge Base

A knowledge base (KB) is required for the semantically extended Web search. A KB is the description of concepts and their relationships which users may be interested in.

Knowledge is classified here as static and dynamic (knowledge). Static knowledge refers to the information stored in the knowledge base, and dynamic knowledge is mined directly from Internet.

For example, for a city, its geographic information, climate, history, population, administration, as well as its neighbors (cities near it), sister cities abroad, province (state) it is located in, and further aspects may be interesting for a user. Such information can be stored in the knowledge base.

Some knowledge can be mined dynamically in Internet when it cannot be found in the knowledge base. For example:

- What's the relationship of YAO Ming and YE li?
- Who is related to Alice similar to the relationship between YAO Ming and YE Li?

Such information can be acquired and mined from Internet dynamically. It will be stored in the KB for later usages. A temporal KB is designed for it. Data in the temporal KB will be synchronized with the KB later.

The corresponding mechanism can be illustrated as in Figure 5.

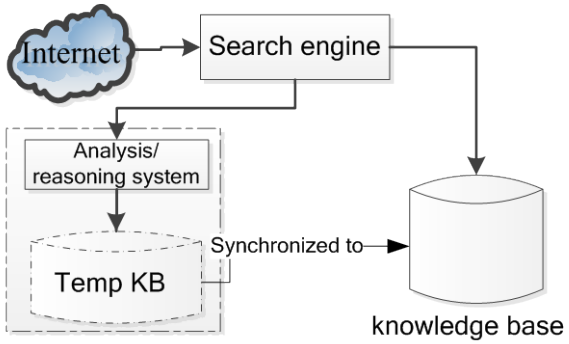


**Figure 5. Knowledge Base for Static and Dynamic Knowledge**

### 3.2. Ontology System

Knowledge is represented in the ontology system. An ontology can be represented by a directed graph, where a node represents a concept, and an arc between two nodes represents their relationship. When a term is bound to a concept, it can be interpreted by the ontology system. For instance, as soon as a query "Shanghai" is bound to a concept *city*, its different aspects, such as the location of Shanghai, population of Shanghai, area of Shanghai, climate of Shanghai and so on, can be deduced. This deduction is named as attributes extension, because location, population, climate *etc.*, are attributes of the concept *city* in the ontology system. In continuation, as an object in the ontology "city", Shanghai can be extended to China, or Nanjing Road, because in the ontology hierarchy, Shanghai is located in China and Najing Road is a street in Shanghai. Therefore, a simple query can be automatically extended to China or Nanjing Road in order to exactly satisfy a user's intention. This function is very important, because on the one hand users are often interested in particular aspects of Shanghai, and on the other hand, users are not always so clear which aspects they are really interested in at the beginning.

In *SmartSearch*, as soon as a user inputs his/her query "Shanghai", the search results will be returned and displayed together with a recommendation pane. The recommendation pane after the reasoning according to the query includes the following items:

1. The attribute list such as "location", "population", "climate", and so on.
2. The generic relationship between the concepts, for example, *is-a*, *part-of*.
3. Special relationships for particular concepts, for example, *located-in* for *city*, *father-of* for concepts *person*, and so on.

For Example, the attributes of *city* will be shown as in Figure 6a:

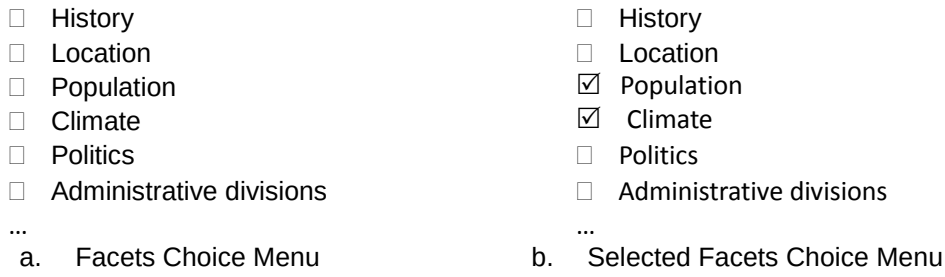


**Figure 6.**

A user can choose his/her interesting facets such as population, and climate, as shown in Figure 6b. Then after refreshing the search, the search result will be a list of documents about population and climate of Shanghai.

Obviously, the ontology system plays an important role here. A supporting ontology system is being developed together with *SmartSearch*. The knowledge base is also implemented in an ontology mechanism.

## 4. SmartSearch-A Case Study

*SmartSearch* is a prototype of semantically extended Web search. The architecture of *SmartSearch* can be shown as in Figure 7.

*SmartSearch* is implemented based on the open source system *Hadoop*[5]. A Search Engine Kernel is built based on Apache *Nutch*[6] and *Solr*[7]. A newly designed Semantically Extended Query Interface is presented to users for accepting their queries. After text processing the user query will be passed to the Search Engine Kernel. The Query Semantics Extension Module and Knowledge Support System are used basically for the semantic extension. In addition, interfaces for third part APIs allow for the integration of *SmartSearch* with other software, *e.g.*, domain based ontology, relationship mining and analysis, data filtering, and so on.

[5] http://hadoop.apache.org/
[6] http://nutch.apache.org/
[7] http://lucene.apache.org/solr/

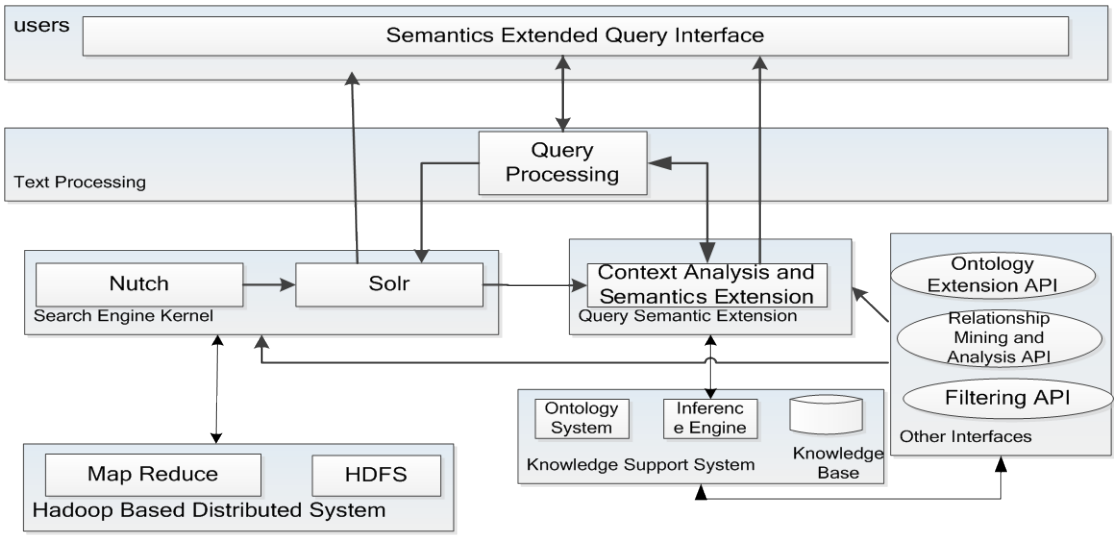


**Figure 7. System Architecture of SmartSearch**

### 4.1. Hadoop and MapReduce

Apache Hadoop is a framework for running applications on large cluster built of commodity hardware. Hadoop implements a computational paradigm named Map/Reduce, where the application is divided into many small fragments of work, each of which may be executed or re-executed on any node in the cluster. In addition, it provides a distributed file system (HDFS) that stores data on the compute nodes, providing very high aggregate bandwidth across the cluster. (http://wiki.apache.org/hadoop/)

A crawler is in charge of capturing suitable Web data traversing the Internet, day and night, from time to time. Captured data should be processed, indexed, and archived in time, such that massive computing power is required. The semantics extension costs even more computing power. The computing should scalable, distributed and based on a large cluster built of commodity hardware. For this reason Apache *Hadoop* is preferred.

As widely known, collected data are indexed in inverted index files for later efficient retrieval. For index processing, Map-Reduce is a suitable computational paradigm.

Furthermore the organization of the corpus and the semantic knowledge have more NoSQL features.

Therefore, *Hadoop* is chosen by us as the base system for *SmartSearch*.

### 4.2. Nutch and Solr

Nutch is an effort to build an open source Web search engine based on Lucene and Java for the search and index component. Nutch is coded entirely in the Java programming language, but data is written in language-independent formats. (http://en.wikipedia.org/wiki/Nutch)

Solr is an open source enterprise search platform from the Apache Lucene project. …Solr uses the Lucene Java search library at its core for full-text indexing and search, and has REST-like HTTP/XML and JSON APIs that make it easy to use from virtually any programming language. (http://en.wikipedia.org/wiki/Apache_Solr)

The Search Engine Kernel of *SmartSearch* is built on *Nutch* and *Solr*. These two software packages are widely used and have been proved to be efficient. They can be flexibly upgraded by us to satisfy the requirements of semantics extensions. Therefore our Search Engine Kernel is built on this basis.

### 4.3. Other Interfaces

There is also a 3rd API interface in *SmartSearch*. Some other projects are launched in ECNU-ICA, *e.g.*, a project named Human Relationship Mining. It works for mining relationship between persons, with some features similar to the *Renlifang* Search[8] launched in Microsoft Research Asia. The associated and relationship search in *SmartSearch* is depended on the algorithms developed by the Human Relationship Mining project. Therefore a corresponding interface for its API is designed in *SmartSearch*. Besides, the Filtering mechanism is useful here. A corresponding API has an interface supplied by the 3rd API interface in *SmartSearch*. Likewise the Ontology Extension API is supplied by the 3rd API interface module.

### 4.4. Query Semantics Extension Module and Knowledge Support System

A Query Semantics Extension (QSE) Module and Knowledge Support System (KSS) is designed for semantics extensions. QSE is used to analyze a user query according to the context, such as user profiles, user behaviors, and usage history (*e.g.*, Cookies). On this basis binding the query term to a concept/object can be automatically achieved.

The KB is designed as a self-expanding system. That is, information mined from Internet and further processed, such as the relationship between two persons which has been found, will be appended to the KB automatically for later usage.

The ontology system is named ICA Ontology. It is scalable, reflective, self-growing and manageable.

The Inference Engine is a rule based inference system.

Having received the query expression ($q$) from the Text Processing Module, QSE analyses $q$, following the Algorithm *SmartSearch*: Case 1: $q$ is not a Relationship Search: binding the query term to a particular concept/object and searching according to extended query; Case 2: $q$ is Relationships search: making corresponding relationship search; Case 3: Query by example: search relationship according the example.

### 4.5. The Search Algorithm

A user query q will be processed as follows:

Step 1: $q$ is bound to a particular concept. It will follow the obvious assignment in user query. Or with the help of the inference system, $q$ is bound to a concept automatically via inference, on the analysis of usage log, user behaviors, user profiles, and so on. That is

$$q \rightarrow \text{concept } \mathbb{C}$$

Step 2: The concept C is mapped to a particular ontology:

$$\mathbb{C} \rightarrow \text{ontology O}$$

Step 3. Faceting according to the semantics of the ontology O:

$$\sigma(q) \rightarrow \text{faceting query menu}$$

Step 4. $q$ is extended to $q'$ with the concept binding and facets selection.

Step 5. Send $q'$ to search engine to carry out the search, return results to users.

[8] http://renlifang.msra.cn/

Similarly, supported by the inference system the associate and relationship search can be executed, *i.e.*,

$$\rho(q) \rightarrow \text{relationship}$$

The Algorithm *SmartSearch* is as following:

Algorithm *SmartSearch*:

1. Get query *q* from user interface;
2. Check the user query *q*:
   Case 1: *q* is not a Relationship Search:
   Binding the query term to a concept/object;
   Modify *q* to a concept/object oriented query *q'*;
   Search (with or without faceting extension) according to query *q'*;
   Case 2: *q* is Relationships search
   If Searching relationship between objects
   then
   Finding the relationship between objects
   else
   Finding object(s) related another object according to the assigned relationship
   Case 3 Query by example:
   Searching according the given example
3. Return search results

### 4.6. A Search Sample

A prototype of *SmartSearch* has been developed at ECNU-ICA. For example: a user issues a query to the system with a term "王菲" (Wang Fei), *i.e.*, name of a famous Chinese Singer. *SmartSearch* processes the query with the semantic extension and finally returns the first page as shown in Figure 8. In the middle of the returned page there is a ranked page list. On the left hand side of the page there is a recommendation pane. In the pane, facets (named as "属性" in Chinese) are listed, such as her birth date (出生日期), blood type (血型), and her height (身高). In addition, recommended terms related with the query term are proposed. The right pane is a relationship diagram describing the persons related with the singer. Obviously, *SmartSearch* is powerful and more efficient.


**Figure 8. A Sample Search Page returned by SamrtSearch**

Figure 9 is another sample search. It's a sample of Query by Example, to find information about the correspondence in Russia comparing with Mr. Obama in USA, *i.e.*, the query is as shown in: <Obama, USA>, <?, Russia>

For convenience, the query is expressed in the form "Russia #Example#(USA Obama)", *i.e.*, in Chinese "俄罗斯 #例子#（美国 奥巴马）".

The results returned are information about Mr. Putin. Obviously the query user may not know who is the president of Russia. He/she only gave an example, *i.e.*, the relationship similar to USA and Mr. Obama. The right pane in the return page is a relationship graph describing different presidents of Russia in different time.


**Figure 9. A Sample Search Page of Query by Example**

## 5. Conclusion

In this paper, the semantic extension Web search, as well as its design and implementation are introduced. The first implementation of *SmartSearch* is designed for China Telecom Shanghai. After the semantics extension with knowledge supporting, Web search in *SmartSearch* is concept/object oriented. And faceting search and associate/relationship search are achieved. Some details of our work will be reported in our other papers later.

## References

[1] R. Rao, From IR to Search AND BEYOND, ACM Queue, vol. 2, no. 3, **(2004)**.
[2] B. J. Jansen and U. Pooch, "A Review of Web Searching Studies and a Framework for Future Research", Journal of the American Society for Information Science and Technology, vol. 3, no. 52, **(2001)**.
[3] A. Spink, D. Wolfram, B. J. Jansen and T. Saracevic, "Searching the Web: The Public and Their Queries", Journal of the American Society for Information Science and Technology, vol. 3, no. 52. **(2001)**.
[4] L. Huang, "A Survey on Web Information Retrieval Technologies", http://www.ecsl.cs.sunysb.edu/tr/rpe8.ps.Z.
[5] T. White, "Hadoop: The Definitive Guide", Second Edition, O'Reilly Media, Inc., **(2011)**.
[6] R. Kuć, Apache Solr 3.1 Cookbook, Packt Publishing Ltd., **(2011)**.
[7] S. Zheng, J. Yang, X. Lin and J. Z. Gu, "Dependency tree based Chinese relation extraction over web data", Proceedings of 7th International Conference on Knowledge", Information and Creativity Support Systems, Melbourne, VIC, Australia, **(2012)** November 8-10.
[8] D. Sontag, K. Collins-Thompson and P. N. Bennett, "Probabilistic Models for Personalizing Web Search", WSDM'12, Seattle, Washington, USA, **(2012)** February 8-12.
[9] D. E. Rose and D. Levinson, "Understanding User Goals in Web Search", WWW 2004, New York, New York, USA, **(2004)** May 17-22.

[10] C. d'Amato, N. Fanizzi, B. Fazzinga, G. Gottlob and T. Lukasiewicz, "Combining Semantic Web Search with the Power of Inductive Reasoning, Scalable Uncertainty Management", Lecture Notes in Computer Science, Edited Salem Benferhat, John Grant, Springer Berlin Heidelberg, vol. 6379, **(2010)**, pp. 137-150.
[11] A. Broder, "A taxonomy of web search", SIGIR Forum, vol. 2, **(2002)**, pp. 36.
[12] U. Lee, Z. Liu and J. Cho, "Automatic Identification of User Goals in Web Search", WWW2005, Chiba, Japan, **(2005)** May 10-14.
[13] S. Goel, J. M. Hofman1, S. Lahaie, D. M. Pennock and D. J. Watts, "Predicting consumer behavior with Web search", PNAS Early Edition, www.pnas.org/cgi/doi/10.1073/pnas.1005962107.
[14] M. A. Hearst, "Next Generation Web Search: Setting Our Sites", Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, vol. 38, **(2000)**.

## Authors

**Junzhong GU** is a Professor of Computer Science, Head of Institute of Computer Application, East China Normal University, China. He received the M.S. degree in Computer Science from East China Normal University in 1982. He works at East China Normal University since 1982. He worked as visit professor at GMD, and University Mannheim, Germany (1987-1989, and 1991-1993). His research interests now include context aware computing, distributed data management, Web searching and multimedia information processing.