Available online www.jocpr.com

Journal of Chemical and Pharmaceutical Research, 2014, 6(5):936-943

Research Article

ISSN : 0975-7384
CODEN(USA) : JCPRC5

Big data learning resources integration and processing in cloud environments

Sun Dapeng

North China University of Water Resources and Electric Power, Zhengzhou, China

---

ABSTRACT

This paper discusses about educational data integration and processing and how it can be used to improve the functional activities of business of education through students, teachers and the way classes are arranged. A key contribution of this paper is the description of a wide array of course resources, e.g., virtual machines, sample projects, and in-class exercises, and how these resources support the learning outcomes and enables a hands-on experience with Big Data technologies.

Keywords: x-learning, learning resource, Map Reduce, Educational Data Mining

---

INTRODUCTION

Educational institutions usually use two well-known environments to educate students. One is the traditional environment, the other is the x-learning environments including distance learning (d-Learning), electronic learning (e-Learning) and mobile learning (m-Learning). The rapid growth of information and communication technologies and rising computer knowledge of the students make possible appearance of these new educational forms.

More and more educational organizations recently put their educational resources online after ten years of being in business, transforming these educational resources to retailers around the globe by x-learning environments. These educational resources are currently stored in a SQL Database, and we have been happy with it. However, since the teachers and students started learning the educational resources online, the database is not able to keep up and the teachers and students are experiencing delays [1]. As teachers and students base and educational resources grow rapidly, we spend money buying more and more Hardware/Software, but to no avail. Losing teachers and students is our primary concern.

At present, as cloud computing has become an attractive technology due to its dynamic scalability and effective usages of the resources, researchers pay more attentions to its applications. These new environments support the creation of a new generation of applications that are able to run on a wide range of hardware devices, like mobile phones, tablet computer, or PDAs, while storing their big data learning resources inside the cloud, such as text, picture, multimedia, video and other learning resources. Then x-learning environments have evolved from a monolithic application perspective to a modular application based on cloud computing.

A lot of problems had been studied, such as the technology for future distance education cloud, integration of hardware and network, integration of learning resources, persistent storage of learning resources as concerned as cloud computing applied in the x-learning environments. There are several cloud computing services providers that offer support for educational systems. Among them are Amazon, Google, Yahoo, Microsoft etc. In [2] are presented the main advantages of using cloud computing in schools. But they did not provide the practicable solution to big data learning resources integration and processing in cloud environments as learning resources become more and more big. The most difficult problem with scaling x-learning systems is typically the different data nodes which are the integration of multimodal data. The rapid growth of learning resources requires new information technologies to solve the problem. One of the major strengths of this paper is its ability to define data quality and learning resources

integration transforms in educational workflows. The purpose of this paper is to present the big data X-learning resources solution for integration and processing and the lightweight architecture partially based on cloud computing environment.

STATE OF THE ART AND RELATED WORK

The educational system is currently facing several issues such as identifying students need, personalization of training and predicting quality of student interactions. In order to solve these problems and improve the reuse of educational resources in digital format appears the concept of "learning object"(LO). A LO includes not only educational content, but also metadata descriptions that describe the own object and make easier its use and location in other contexts, LOs standards are used. An example of the metadata's standard is the LOM (Learning Object Metadata) XML scheme [3], which was developed by LTSC which contains only the object meta-data and allows access to learning materials hosted in the connected repositories. The objects stored in these repositories are characterized according to international standards for learning objects meta-data (LOM). The meta-data fields describe the object and the possibilities for its use, so that objects may be located using keywords, retrieved, and examined to see whether they suit learners' needs. It is additionally possible to add any material to a personal collection, which helps in organizing teaching materials for each of courses.

We use the term "learning resource" to imply a defined package of structured, factual information that is linked with a specific educational context. Here, context is defined as a set of circumstances in which an educational resource is used or may be used. The Can Core Learning Resource Metadata Initiative, which was developed by Creative Commons (CC) and the Association of Educational Publishers (AEP), enhances the ability of educators, researchers and students around the world to search and locate materials from online collections of educational resources.

We use the term "knowledge object" to describe the subject matter content or knowledge to be taught. A knowledge object (KO) consists of a set of fields (containers) for the components of knowledge required to implement a variety of instructional strategies. These components include: the name, information about, and the portrayal for some entity; the name, information about, and the portrayal for parts of the entity; the name, information about, values, and corresponding portrayals for properties of the entity; the name, and information about activities associated with the entity; and the name and information about processes associated with the entity. In the following paragraphs we will attempt to clarify these components. Some of metadata standards and the added extended metadata are shown in Figure 1.

IEEE LOM	Can Core	LOM Core Metadata		KO as Extended Metadata	
1. General	1. General	1. Contributor	10. Relation	1. General	10. Knowledge Object
2. Life Cycle	2. Life Cycle	2. Coverage	11. Rights	2. Life Cycle	10. 1 ID
3. Meta-Metadata	3. Meta-Metadata	3. Creator	12. Source	3. Meta-Metadata	10. 2 Creator
4. Technical	4. Technical	4. Date	13. Subject	4. Technical	10. 3 Date
5. Educational	5. Educational	5. Description	14. Publisher	5. Educational	10. 4 Topic
6. Rights	6. Rights	6. Format	15. Title	6. Rights	10. 5 Subtopic
7. Relation	7. Relation	7. Identifier	16. Type	7. Relation	10. 6 Language
8. Annotation	8. Annotation	8. Language		8. Annotation	10. 7 Summaries KO
9. Classification	9. Classification	9. Publisher		9. Classification	10. 8 Creator
					10. 9 Creator

Figure 1. Learning Object Metadata, Learning Resource Metadata and Knowledge Object Metadata

KEY TECHNOLOGIES AND ARCHITECTURE

Currently x-learning systems produce huge amounts of learning resources from observations, experiments, simulations, models, and higher order assemblies, along with the associated documentation needed to describe and interpret the big data X-learning resources, which are stored in large data warehouses in digital form [4]. More and more large-scale x-learning problems are facing similar processing challenges on learning resources datasets which are a group of learning resources structures used to store and describe multidimensional arrays of big data learning resources, where cloud environments could potentially help [5].

Until recently, the choice of database architecture was largely a nonissue. Relational databases were the defector standard and the main choices were Oracle, SQL Server or an open source database like MySQL. Although Mainframe Hierarchical Databases are very much alive today, The Relational Databases (RDBMS) (SQL) have dominated the Database market, and they have done a lot of good.

With the advent of big data learning resources, scalability and performance issues with relational databases became commonplace. For online processing, NoSQL databases have emerged as a solution to these problems. NoSQL is a catch-all for different kinds of database architectures — key-value stores, document databases, column family

databases and graph databases. Each has it's own relative advantages and disadvantages. NoSQL Databases offered an alternative by eliminating schemas at the expense of relaxing ACID principles. Some NoSQL vendors have made great strides towards resolving the issue; the solution is called eventual consistency. However, in order to get scalability and performance, NoSQL databases give up "query ability" (i.e. not being able to use SQL) and ACID transactions. More recently a new type of database has emerged that offers high performance and scalability without giving up SQL and ACID transactions. This class of database is called NewSQL, a term coined by Stonebreaker. NewSQL provides performance and scalability while preserving SQL and ACID transactions by using a new architecture that drastically reduces overhead.

Architecture of Big Learning Objects Integration and Processing

The NoSQL and NewSQL movement has produced a host of new big data learning resources integration and processing solutions that attempt to solve the scalability challenges without increased complexity. Solutions such as MongoDB, a self-proclaimed "scalable, high-performance, open source NoSQL database", attempt to solve scaling by combining replica data sets with sharding clusters to provide high levels of redundancy for large data sets transparently for applications. Undoubtedly, these technologies have advanced many systems scalability and reduced the complexity of requiring developers to address replica sets and sharding.

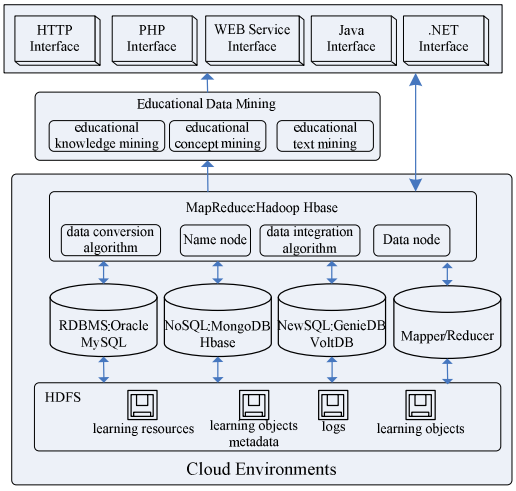

Figure 2. X-Learning resources integration and processing architecture in cloud environment

But the problem is that hosting MongoDB or any other persistent storage solution requires keeping the hardware capacity on hand for any expected increase in traffic. The obvious solution to this is to host it in the cloud environment, where we can utilize someone else's hardware capacity to satisfy our demand. Unless you are utilizing a hybrid-cloud with physical hardware you are not getting direct attached storage. The problem with this is that I/O in the cloud is very unpredictable, primarily because it requires traversing the network of the cloud provider. Solutions such as MapReduce which is widely considered to be one of the core programming models for big data learning resources integration and processing in the cloud environment enables building highly distributed programs that run on failure-tolerant and scalable clusters of commodity machines.

Figure 2 shows the architecture of the learning resources integration and processing in cloud environment. As it may be seen, it is composed of different integration layers, allowing the developer to use different subsystems to integrate learning resources into cloud environment.

Big Data Learning Resources with Educational Data Mining

Educational data mining can help both students and educational institutions for improving the quality of education. It includes the mining of student data or other data related to education, such as courses assignments, marks and student background. Educational data mining allows having a better perspective on the educational progress, and at the same time to analyze the information related to the specifics of the programs, courses, and course assignments [6]. This innovative approach allows the decision making process to use the what-if scenario when analyzing the student data, and other education related information in order to improve educational processes. The data related to the educational progress is retrieved from the educational records, imported into the data mining system, analyzed, and exported back. Educational data mining allows identifying and locating details about educational processes that need improvements, or those that perform very well and could be used as good examples. Educational data mining can assist in the design of the educational content. It can help in improvements in student academic performance.

Educational data mining uses many techniques such as decision tree, rule induction, neural networks, k-nearest neighbor, naïve Bayesian, data mining algorithms and many others. By using these techniques, many kinds of knowledge can be discovered such as association rules, classifications and clustering. Data mining algorithms can help in discovering pedagogically relevant knowledge contained in databases obtained from x-learning systems. These findings can be used both to help teachers with managing their class, understand their students' learning and reflect on their teaching and to support learner reflection and provide proactive feedback to learners.

Big Data Learning Resources with NewSQL

NewSQL is a class of modern relational database management systems that seek to provide the same scalable performance of NoSQL systems for online transaction processing (read-write) workloads while still maintaining the ACID guarantees of a traditional database system. Some examples of NewSQL systems are: VoltDB, NuoDB, Google Spanner, GenieDB and Clustrix. These are designed to operate in a distributed cluster of shared-nothing nodes, in which each node owns a subset of the learning resources. Though many of the new databases have taken different design approaches, there are two primary categories evolving. The first type of system sends the execution of transactions and queries to the nodes that contain the needed learning resources. SQL queries are split into query fragments and sent to the nodes that own the learning resources. These databases are able to scale linearly as additional nodes are added.

The purpose of VoltDB is to go radically faster than traditional relational databases, such as MySQL, DB2 and SQL Server on a certain class of applications. It is an ACID-compliant in-memory database and represents new types of databases that focus on maintain the guarantees that traditional relational databases offer, but also provides a scalable and fault-tolerant system. As VoltDB, the system provides queries in standard SQL language, and executes the queries before the data arrives in data warehouse systems.

GenieDB is a commercial storage engine for MySQL developed by GenieDB Inc. At the time of this writing, no peer-reviewed publication is available describing the storage engine or its back-end storage strategy, but some features and capabilities can be inferred from the white papers available from their commercial website. GenieDB appears to provide two levels of functionality. The lower-level GenieDB datastore is described as a distributed database providing immediate consistency across a number of nodes. Replication is made more efficient, where possible, through use of a reliable broadcast protocol. The datastore is accessed through a type of "NoSQL API" that, we assume, is similar to APIs for most key value systems The GenieDB MySQL storage engine is then built on top of the GenieDB datastore to provide relational access, implementing the MySQL table handler.

Big Data Learning Resources with NoSQL

MapReduce is a programming model for processing large datasets including big data learning resources datasets. With the MapReduce programming model, programmers only need to specify two functions: Map and Reduce. The map function takes an input pair and produces a set of intermediate key/value pairs. It is an initial transformation step, in which individual input records can be processed in parallel. The Reduce function adds up all the values and produces a count for a particular key. It is an aggregation or summarization step, in which all associated records must be processed together by a single entity. It merges together these values to form a possibly smaller set of values. Typically just zero or one output value is produced per Reduce invocation. MapReduce functions are as follows.

$$\text{Map:}(\text{in\_key},\text{in\_value}) \rightarrow \{\text{key}_j, \text{value}_j \mid j=1 \cdots k\}$$

$$\text{Reduce:}(\text{key}, [\text{value}_1, \text{value}_2, \cdots, \text{value}_m]) \rightarrow (\text{key}, \text{final\_value})$$

The input parameters of Map are in_key and in_value. The output of Map is a set of <key,value>. The input parameters of Reduce is (key, [$value_1$, ..., $value_m$]). After receiving these parameters, Reduce is to merge the data which were get from Map and output (key, final_value).

Apache Hadoop which is an open source implementation of the Google's MapReduce parallel processing framework is an open-source project for reliable, scalable, distributed computing and data storage [7]. Hadoop not only has a distributed processing platform but also has a sequential and batched file system called Hadoop Distributed File System (HDFS) Google had developed a distributed file system, called Big Table to successfully store a large amount of structured data. The Hadoop components that are analogous to Google's components described below are:

1. The MapReduce programming model
2. Hadoop's Distributed File System (HDFS).

HDFS is a flat-structure distributed file system that store large amount of data with high throughput access to data on clusters. HDFS has master/slave architecture, and multiple replicas of data are stored on multiple compute nodes to provide reliable and rapid computations [8]. Its master node is called JobTracker or NameNode which is a simple master server, and TaskTrackers or DataNodes which are slave servers.

HBase is a solution similar to BigTable and is developed by the Hadoop team. HBase and BigTable adopt column-oriented approach to store data instead of row-oriented process in the relational database. The advantage of column-oriented access is that a record can have a variable number of columns [9]. HBase takes the advantage of a distributed file system and partitions a table into many portions which are accessed by different servers in order to achieve high performance.

Big Data X-Learning Resources Integration in Cloud Environments

We decide to run the x-learning system in SQL, NoSQL and NewSQL simultaneously by segmenting our online user base. Our objective is to find the big data X-learning resources integration and processing solution. We choose SQL MySQL, NoSQL MongoDB and NewSQL VoltDB. Because MongoDB has an integrated caching mechanism, and it can automatically spread data across multiple nodes. VoltDB is an ACID compliant RDBMS, fault tolerant, scales horizontally, and possesses a shared-nothing & in-memory architecture. At the end, all systems are able to deliver. We won't go into the intricacies of each solution because this is an example and comparing these technologies in the real-world will require testing, benchmarking, and in-depth analyses.

X-learning systems include variety of educational data nodes; each node may store various types of educational data. In order to unify the format, transform various data format into learning resource's format which eliminated semantic ambiguity [10]. Data integration is required to merge multi-modal data to improve actionable insights (shown in Figure 3).

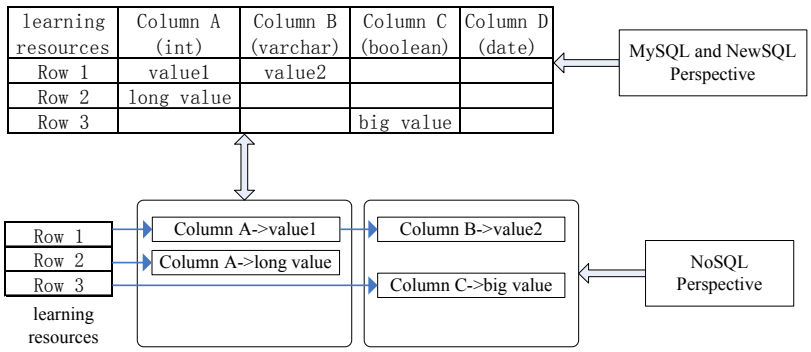

Figure 3. X-Learning Resources Integration in Cloud Environments

For example, a database is built to store student information and the courses that each student takes. A possible design in the relational model for this data is to have one table for student, one for course, and one that maps a student with his courses (shown in Figure 4). One problem with this design is that it contains extra duplicated data; in this case the mapping table student_course repeats the Std_ID multiple times for each different course. NoSQL approach, however, is flexible enough to map one student with a list of courses in only one record without this duplicated data. Figure 5 shows the solution using a document-store database.

In this example, if the user wants to query the average grade of all students together, that is one simple work for the SQL table, which only works on one column grade and gets the average value of all grades. Meanwhile, the operation will be much more complicated with the nested layers in NoSQL collection. On the other hand, if the system only serves displaying the data, meaning listing the courses and grades for each student (including student name) then the opposite is true.

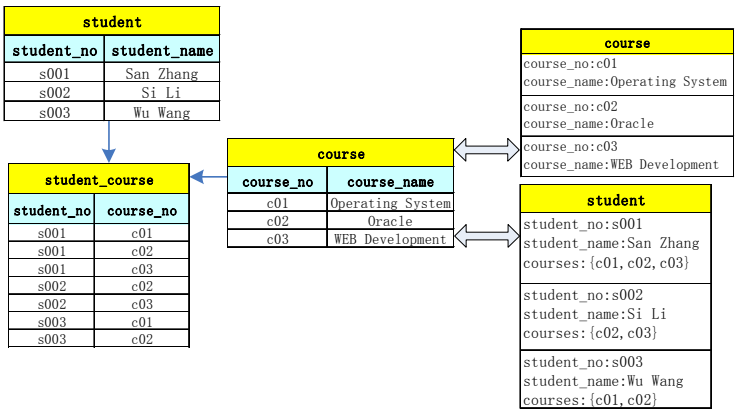

Figure 4. X-Learning Resources Integration with extra duplicated data

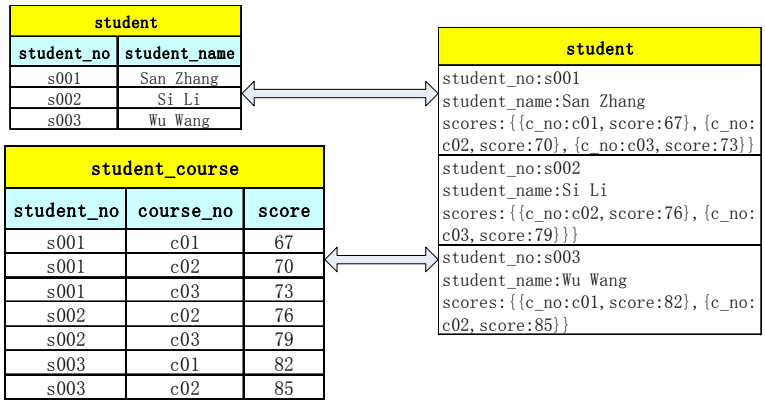

Figure 5. X-Learning Resources Integration without duplicated data

Big Data X-Learning Resources Integration and Processing in Cloud Environments

We need to analyze learning resources and merge the MySQL learning resources warehouse with the data from the x-learing data sources, and run analytical reports. That's where Hadoop comes in. We configure a Hadoop system and merge the data from the three data sources [11]. A better idea is to big data learning resources integration across learning resources conversion algorithm. The following is an overview of heterogeneous learning resources conversion algorithm (shown in Figure 6). We use Hadoop's Mapreduce in conjunction with the open source R programming language to achieve the big data X-learning resources integration and processing in cloud environments.

Input: educational data;
Output: learning resources LRs;
1. if (the educational data is relational database)
set the first line of the flag: FLAG= 0, then read the educational data into cluster by row in the table;
2. if (it is a NoSQL data resource)
set the first line of the flag: FLAG=1, then read the educational data into cluster by column in the table;
3. if (it is a NewSQL data resource)
set the first line of the flag: FLAG=2, then read the educational data into cluster by column in the table;
4. Repeat 1-3 until all the educational data in data resource network is stored to cluster ;
5. End;

Figure 6. Learning Resources Conversion Algorithm

We have defined an alternative integration model which consists of the following five models.

learning resources data(Id,Datum): It is an extensional predicate corresponding to the learning resources dataset.

model(J,M): It is an intensional predicate recording the global model M at iteration J. Initialization is performed through the function predicate init model(M) that returns an initial global model.

map(M, R, S): It is a predicate that receives as input the current model M and a learning resources data element R, and generates a statistic S as output.

reduce: It is an aggregation function that aggregates several per-record statistics into one statistic.

update(J, M, AggrS, NewM): It is a predicate that receives as input the current iteration J, the current model M and the aggregated statistic AggrS, and generates a new model NewM.

We have defined an alternative processing model which consists of the following three functions: and an alternative integration model which consists of the following three functions map function: It receives read-only global state value (i.e., the model) as side information and is applied to all learning resources in parallel.

reduce function: It aggregates the map-output into a single aggregate value. This function is commutative and associative.

update function: It receives the combined aggregate value and produces a new global state value for the next iteration or indicates that no additional iteration is necessary.

The learning resources in cloud are abstracted into data nodes in the Hadoop, all the data nodes constitute the learning resource network. Data nodes increases, failure will cause the learning resource network changes [12]. In order to update the big data learning resources network automatically, we use the following learning resources integration algorithm (shown in Figure 7) to maintain the learning resources dynamically.

```
Input: new data node n;
Output: learning resources XML w;
1. Scan Hadoop XML,
if n==0, and no node failure,
 go 8;
 if n!=0, go 2;
 if node failure, go 3;
2. for(i=n;i>0;i--)
 locate the position, and find its neighbor, add all the edges, go 4;
3. for(i=n;i>0;i--),
 locate the position, and find its neighbor, delete all the edges, go 5;
4. Calculate the load of the node, and submit the load of the node to
learning resource migration algorithm to get the node's actual load, go 7.
5. Calculate the failed node's learning resources, includes the resource
name, quantity, etc., go 6;
6. According to the content of 5 , calculate the learning  resources needed
and it's amount, the results submit to learning  resource conversion
algorithm;
7. Monitoring the added and failed node, go 1.
8. return w;
```

Figure 7. Learning Resources Integration Algorithm

CONCLUSION

Many application scenarios require processing massive datasets in a highly scalable and distributed fashion and different types of big data resources have been designed to address this challenge raised by big data: Volume, Variety, Velocity, Variability and Integration. This paper presents a set of guidelines and a wide array of learning resources to integrate the study of three core types of big data resources: MapReduce, NoSQL, and NewSQL. The paper also reports data conversion algorithm and data automatically updated algorithm of integrating the proposed units into SQL table course.

REFERENCES

[1] Emilio Julio Lorenzo; Roberto Centeno; Miguel Rodriguez-Artacho; Proceedings of the First International Conference on Technological Ecosystem for Enhancing Multiculturality, **2013**, 127-132.
[2] Y Md. Anwar Hossain Masud; Xiaodi Huang; *World Academy of Science, Engineering and Technology*, **2012**, 74-78.

[3] Sai Sabitha; Deepti Mehrotra; Abhay Bansal; *International Journal of Education and Learning*, **2014**(3), 1-12.
[4] Vaishnavi.J.Deshmukh; Sapna.S.Kaushik; Amit.M.Tayade; *International Journal of Advanced Research in Computer Science and Software Engineering*, **2013**(3), 1-9.
[5] Yasin N. Silva; Suzanne W. Dietrich; Jason M. Reed; Proceedings of the 45th ACM technical symposium on Computer science education, **2014**, 139-144.
[6] Pooja Gulati; Archana Sharma; *International Journal of Computer Science and Information Technology & Security*, **2012**(2), 648-650.
[7] Srinivasa Rao; K.Thammi Reddy; MHM.Krishna Prasad; *Information Technology and Computer Science*, **2014**(4), 37-42.
[8] Peng Hu; Wei Dai; *International Journal of Database Theory and Application*, **2014**(7), 37-48.
[9] Tyler Harter; Dhruba Borthakur; Siying Dong; Amitanand Aiyer; Proceedings of the 12th Conference on File and Storage Technologies, **2014**,1-14.
[10] Zengqiang Ma; ShaZhong; Xingxing Zou; Yacong Zheng; *Journal of Chemical and Pharmaceutical Research*, **2014**(2), 145-150.
[11] ZhiyunFeng; Maozhu Jin; Renpei Yu; *Journal of Chemical and Pharmaceutical Research*, **2014**(2), 187-192.
[12] Trimurti Lambat; Sujata Deo; Tomleshkumar Deshmukh; *Journal of Chemical and Pharmaceutical Research*, **2014**(4), 888-892.