Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 5, May 2015, pg.305 – 313

RESEARCH ARTICLE

A Research on RDF Analytical Query Optimization using MapReduce with SPARQL

Pravinsinh Mori[1], A R Kazi[2], Sandip Chauhan[3]

[1]M.E in C.E, Kalol Institute of Technology & Research Centre, India

[2]Asst. Prof. in Kalol Institute of Technology & Research Centre, India

[3]HOD of ME(CE) in Kalol Institute of Technology & Research Centre, India

[1] mori_pravin123@yahoo.co.in, [2] kazi_ce@yahoo.com; [3] sandymba2006@gmail.com

Abstract— Large scale data (Big Data) analysis is an important topic in Cloud Computing, which required complex data analysis. Big data analysis can be done efficiently done using MapReduce process. RDF is popular type of Web data storage that store and retrieve huge amount of useful data. The size of RDF data collections has increased rapidly making the issue of scalable processing on data. The MapReduce computation model provides limited static optimization techniques used in relational databases. So, dynamic optimization techniques for such join intensive tasks on MapReduce need to be investigated. Scalable RDF based data can be retrieved as per requirement it is more time consuming. To improve query execution process on RDF using MapReduce is key technique. Proposed system gets RDF data and process data mapping according to mapping function. After applying the Bloom Filter on mapped data it applies Indexing function on it. Finally we aggregate the result using aggregate functions specified in the select clause and generate new RDF. Generated RDF provides faster and efficient query execution as compare to raw RDF.

Keywords— MapReduce, Bloom Filter, Hadoop, SPARQL, Directed Graph

I. INTRODUCTION

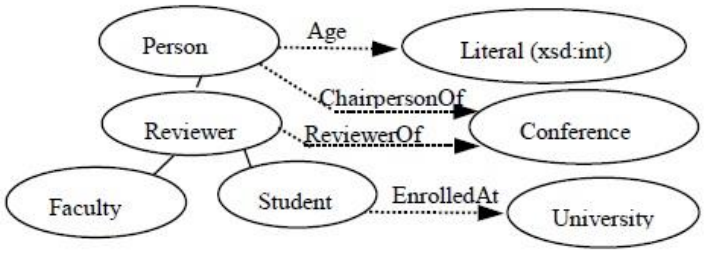
Cloud Computing provide users with massive amount of Big data storage and analysis of High-level data. Big data is defined as the ability to wring business value from the large volume, variety, and velocity of information becoming available. As per the recent report on big data implementation shows that many organizations find the **variety** element of big data a much bigger challenge than **volume** or **velocity.** Many fields require organization to Manage and analysis of huge amounts of data. Such fields include social network to analysis, financial-risk management, threat finding in complex network systems, and medical and biomedical large databases. These are all examples of big data analytics, in which dataset sizes enlarge exponentially. These application fields create operational challenges not only in terms of sheer size but also in time to solution, because quickly answering queries is essential to obtaining market advantages, avoiding vital security issues, or preventing life-threatening health problems. Semantic graph databases seem to be a promising resolution for storing, managing and querying the large and different datasets of these application fields. Such datasets present an abundance of relations among many elements. Semantic graph databases

organize the data in the form of subject-predicate-object triples following the Resource Description Framework (RDF) data modelconvex hull model in high dimensional spaces.

II. INTRODUCTION RDF AND MAP REDUCE

A. RDF:

In this section, we primary introduce RDF (Resource Description Framework) [7]. Semantic web technologies are being developed to current data in standardized way such that such data can be retrieved and understood by both human and machine. Historically, WebPages are published in simple html files which are not suitable for reasoning. Instead, the machine treats these html files large amounts of keywords. Researchers are developing Semantic web technologies that have been standardized to deal with such inadequacies. The most well-known standards are Resource Description Framework (RDF) , SPARQL Protocol and RDF Query Language (SPARQL). RDF is the model for storing and representing data and SPARQL is a query language to retrieve data from an RDF dataset.



Subject	Property	Object
IDBC2005	rdf:type	Conference
John	Age	24
John	rdf:type	Student
John	ReviewerOf	IDBC2005
Mary	rdf:type	Faculty
Mary	ChairpersonOf	IDBC2005

Figure 1: RDF data for reviewers model [1]

Resource Description Framework (RDF) [1] is a language for present the information or data about resources in the WWW. The resources are not limited to web pages but can also contain things that can be identified on the web. The design of metadata in the generic RDF format makes it appropriate for automatic consumption by a diverse set of applications. The RDF data represented as a set of <subject, property, object> triples.

As per above figure, consider RDF data about research paper checkers. The RDF classes and the triple instances are shown in Figure 1. Assuming the RDF data is stored in the database as the representation of ' checkers ', user can issue the following query to find reviewers who are students with age less than 27

```
SELECT t.r reviewer
FROM TABLE(RDF_MATCH(
 '(?r  ReviewerOf  ?c)
  (?r  rdf:type    Student)
  (?r  Age         ?a)',
  RDFModels('reviewers'),
 NULL, NULL)) t
 WHERE t.a < 27;
```


The various arguments to RDF_MATCH are as follows: The first argument captures the graph pattern to search for. It uses SPARQL-like syntax [13] and variables are prefixed with a '?' character. The second argument specifies the model(s) to be queried.

The third argument specifies the rule bases (if any). Here the NULL argument indicates absence of rule bases. The fourth argument specifies user-defined namespace aliases (if any). Here the NULL argument indicates that no user-defined aliases are used, however default aliases such as rdf: are always available.

B. MapReduce:

Large amount of data processing frameworks, such as MapReduce[1] supply capabilities of extensibility, fault tolerance, and parallel data processing. MapReduce[1] framework is composed of a master and many number of workers that conduct MapReduce[1] jobs. A MapReduce job includes Map segment and Reduce segment. At the time perform a MapReduce job, the master will divide the data to be processed into multiple same-sized blocks, and create *M* Map works and *R* Reduce works. MapReduce[1] has emerged as accepted a way to tie together the power of large clusters of computers. MapReduce[1] allows programmers to consider in a data-centric fashion: they focus on applying transformations on sets of data, allow the details of distributed execution, network communication and fault tolerance to be handled by the MapReduce. MapReduce[1] is typically applied on big batch-oriented computations that are concerned primarily with time to job completion.

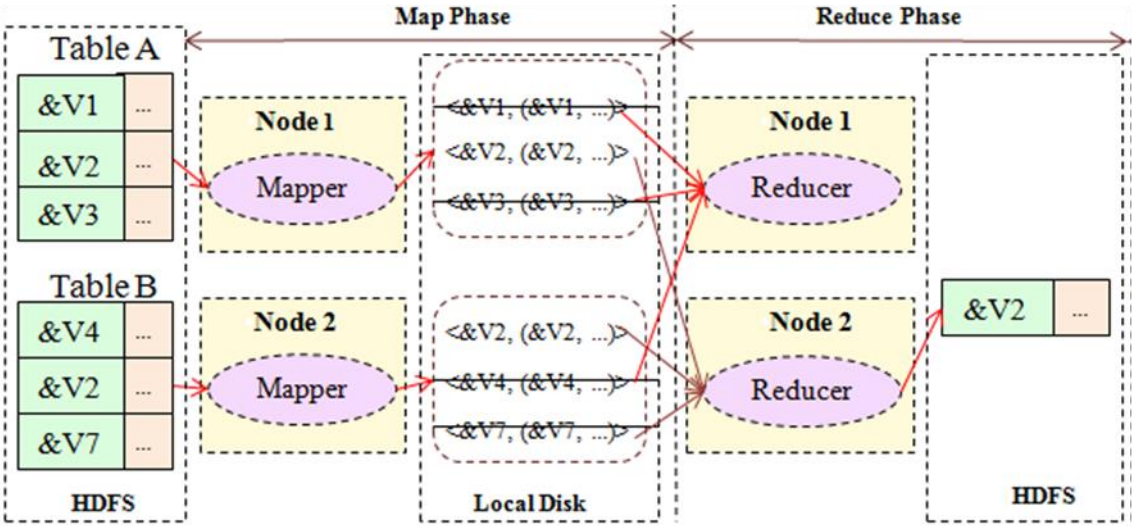


Figure 2 Hadoop MapReduce process [8]

III. RDF VS. REATIONAL DATABASE

In a usual Relational Database (RD) support structure database schema. In RD there are several databases table that are populating with information about their relation. In relational database the example as the Student information table, Employee data, Salary table that all are combine and join after we get School useful information. It required vey long process and complicated. Now we talk about RDF. There have been a lot of attempts to shred RDF data into relational model. In approach involves a single triple-store relation with three columns subject, predicate and object. After that RDF triple becomes a single tuple, which for a popular dataset. SPARQL is popular query languages which is used to query RDF data. RDF data mainly supported Web based heavy data. RDF should be able of rapidly processing large

amounts of data and should also generate more knowledge from the existing data. In RDF data that is gathered from sources in different geographical locations is naturally distributed. It mainly to work on compress the uncompressed data, also filtered data as per prediction, subject and object based.

IV. STATE OF THE ART ON MAP REDUCE TECHNIQUES

MapReduc is a programming model, which provides easy way of parallelizing complex tasks. MapReduce is encouraged from the functional programming language that provide map and reduce primitives. The framework surrounding MapReduce model manages the concerns related to work division, fault-tolerance, data area over a distributed file system and provides abstraction to implement the programming logic in Map and Reduce. This model is like to a split and combined model of parallelization, where a given task that needs to be performed on a set of data is managed by executing the task at a time on the chunk of data splits and later aggregating the results of all the tasks to provide the final result. The Map is analogous to the divide phase and the Reduce is analogous to the aggregation.

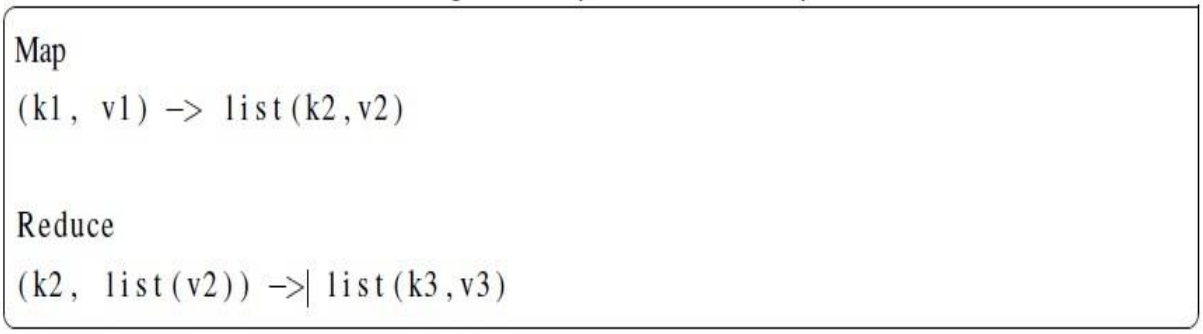


Figure 3 Map Reduce Concept [6]

SPARQL queries that to take the form of sub graph match. This types of data access is therefore the motivate use-case for which that architect to our system. RDF data is partitioned across a cluster of machines, and SPARQL directed graph patterns are searching for in parallel by each machine. Data may need to be exchange between machines in order to deal with the information that some patterns in the data set may span partition boundaries. The Hadoop to manage all parts of query processing that require multiple machines to work together.

SPARQL is the W3C standard to query the RDF data model. It follows the same principles (interoperability, extensibility, decentralization, etc.) and a similar graph notation. Intuitively, a SPARQL graph pattern comprises named terms but also variables for unknown terms. The pattern returns solutions when it matches an RDF sub graph after variable substitution. This require substitution of RDF terms for the variables is then the solution for the query.In general, SPARQL graph patterns that involve pathsthrough the RDF graph convert to subject and object joins in the SQL, and patterns that involve multiple attributes about the same entity involve subject and subject joins in the SQL (theabove example is both types of joins). Although othertypes of joins are possible, subject-subject and subject-objectjoins are by far the most common.

Nested Triplegroup algebra (NTGA): That supports special TripleGroup based operators for efficient processing of RDF graph patterns. The triple relation is loaded using NTGA's TG_LoadFilter which coalesces two operations: TG_Load - lo:ads RDF statements into TripleGroups and TG_Filter - eliminates irrelevant TripleGroups that fail to satisfy the SPARQL FILTER construct i.e. value-based filtering. The value-based filtering is processed during the load phase since Hadoop does not currently support efficient indexing scheme to

selectively retrieve records from HDFS. The TG_GroupBy operator is equivalent to the grouping operator in relational algebra, and generates TripleGroups based on the common Subject. The TripleGroups thus generated are filtered using the TG_Groupfilter operator to eliminate TripleGroups with missing edges that violate the structural constraints specified in the given query. For example, the following TripleGroup tg1

tg1 = { (&V2, type, VENDOR),
(&V2, country, US),
(&V2, delDays, 6) }

Does not satisfy the structure-based filtering operation:

TG_GroupBy(TG{type, country, homepage})

Due to the missing edge corresponding to the Predicate homepage. After the completion of structure-based filtering, they process the joins between the star patterns using the TG_Join operator. The TG_Join operator is semantically equivalent to the relational join operator except that it is defined on TripleGroups.

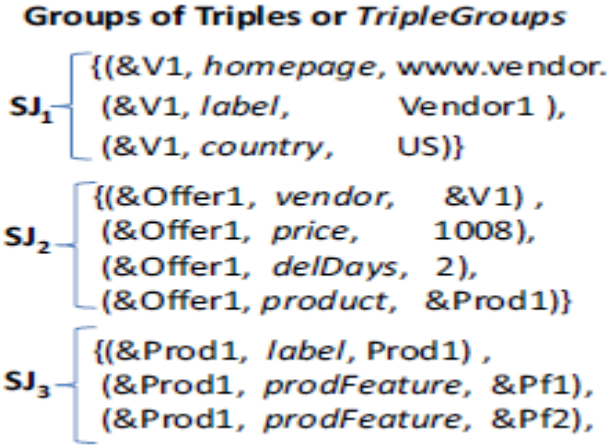


Figure 4 TripleGroups generated using "Group T by Sub"[2]

V. PROPOSED WORK

In this work, the main idea behind how to distributed RDF large data usin MapReduce technique. RDF is mainly store data as in the triple form (subject, object, predicate). In the subject and object binary relation with predicate. Subject always directed with object or with subject using predicate. The r system we first optimized our RDF using the String Synopsis filter (remove duplicates objects or literals). Then we generate the map of subject, object and predicates. Then we count the total no of items in each subject, object, and predicate. After this steps we compare with each other subject, object, and predicates.

Then one of them which get higher count of subject, predicate and object we create the new RDF. This RDF is known as optimized RDF because as compare to raw RDF this is better because as per higher no of occurrence values are on top and indexing already done. This all process done using the MapReduce.

Then after the insert SAPRQL queries jobs on the new RDF. SPARQL queries first to check in bloom filter the value are available on set or not. Bloom filter always work on RAM. So it reduce the IO cost and fast execution of queries.

In that which records in the set that are available it execute and which are not in set. That queries is skipped and jump to next queries. In the proposed system can run query execution time faster as compare to raw RDF.

A. **FLOW OF WORK:**

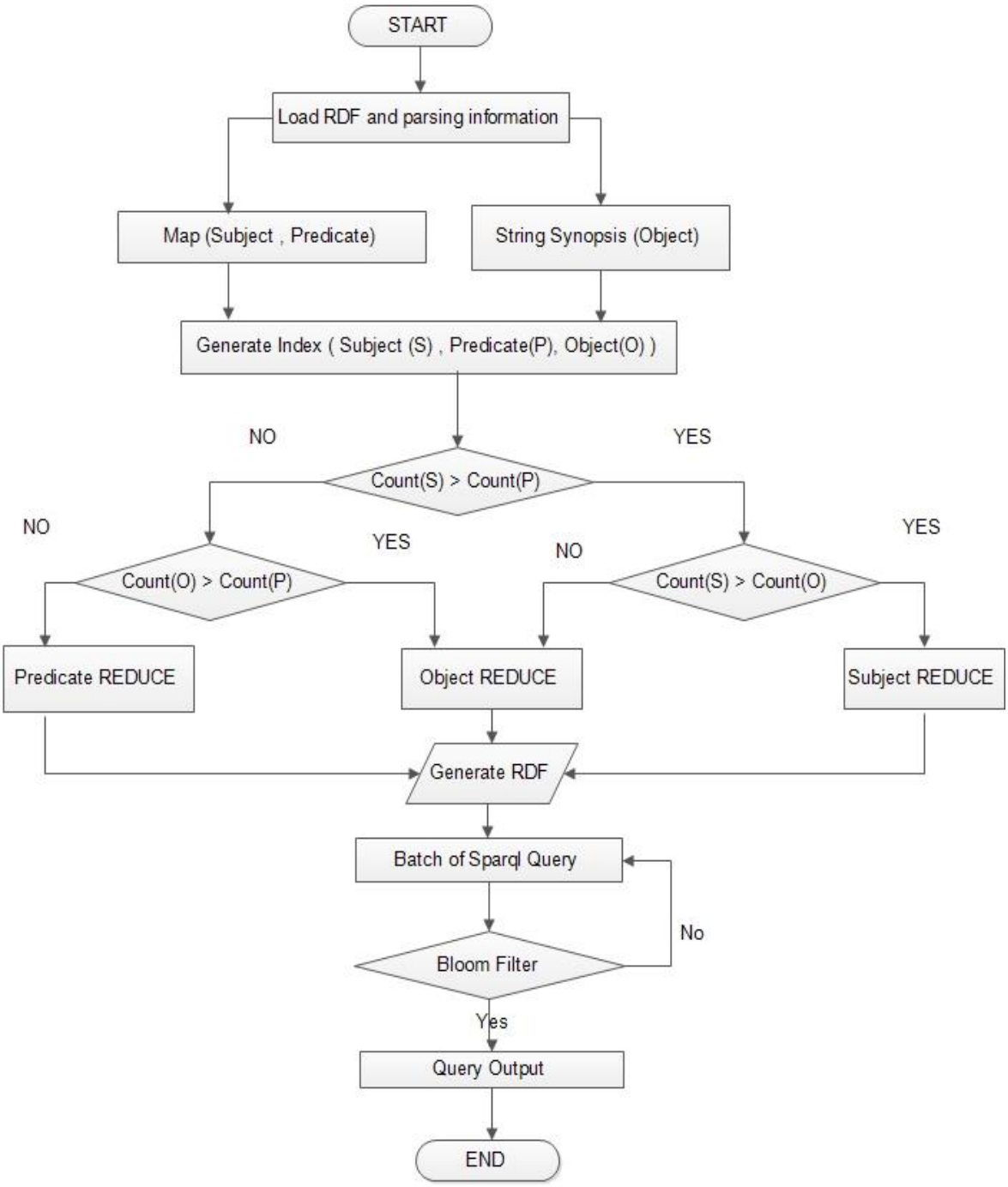


Figure 5 Flow of Proposed Work .

B. **THE PROPOSED ALGORITHM:**

1. First, Load RDF.
2. Parsing the Triple Information (Subject, Predicate, Object).
3. Apply the Mapping Function Map => (key=x, Triple value)
   Where X= Subject, Predicate, Object
4. The String Synopsis (Remove Variable values from Objects).

5. Indexing Function on generated Maps (Subject, Predicate, Object).
6. Compare Counts.

```
If Count (S) > Count(P) Then
        If Count(S) > Count(O) then
                Generate MAP Subject
                Else
                Generate MAP Object
    Else If Count (O) > Count (P)
                Generate MAP Object
                Else
                Generate MAP Predicate
```

7. RDF Generate above result.
8. Execute SPAQRL queries job.
9. Check data in Bloom Filter.
10. Query Output.

VI. EXPERIMENTAL WORK

This section describes the current status of implementation along with appropriate screen shots.

First, read the raw RDF and then apply the algorithm using MapReduce and compare the count of subject, object and predicates. In that which value occurrence higher it go to higer end at the new RDF..

Second, apply the queries on the newly generate RDF and Bloom Filter check the data available in set or not and query output.

1. READ RDF DATASET:

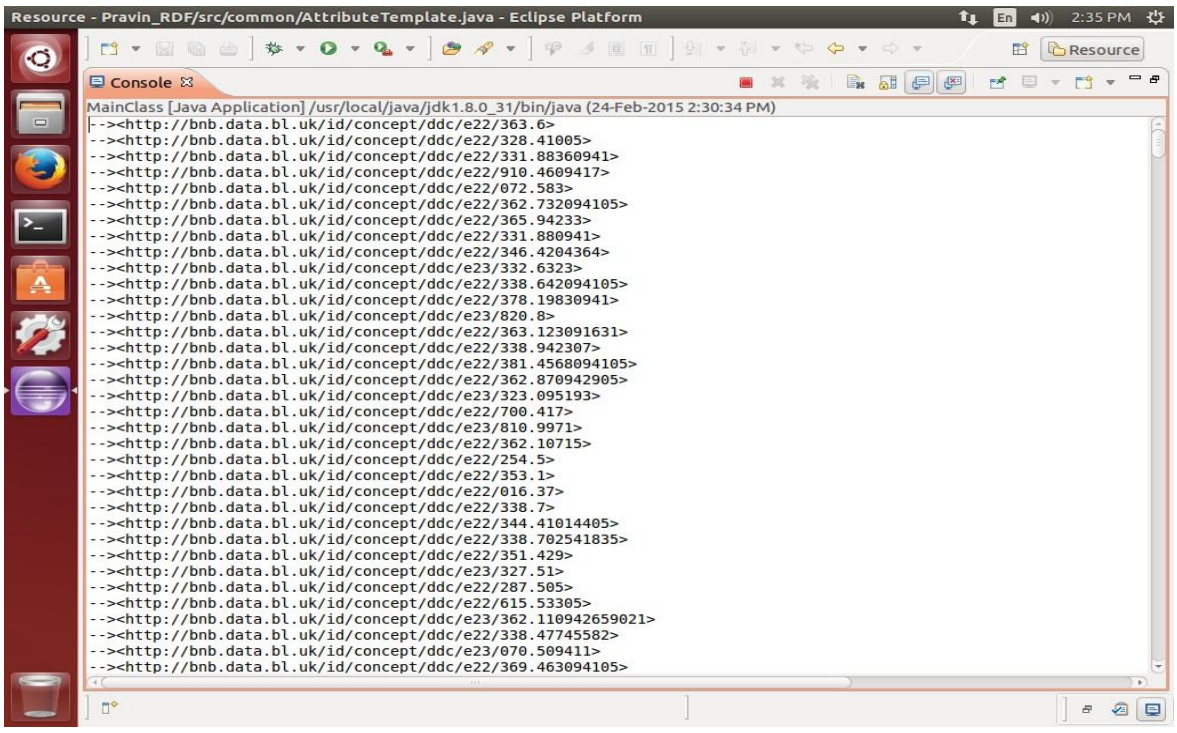


Figure 6 Read Dataset

2. MAP REDUCE PROCESS:

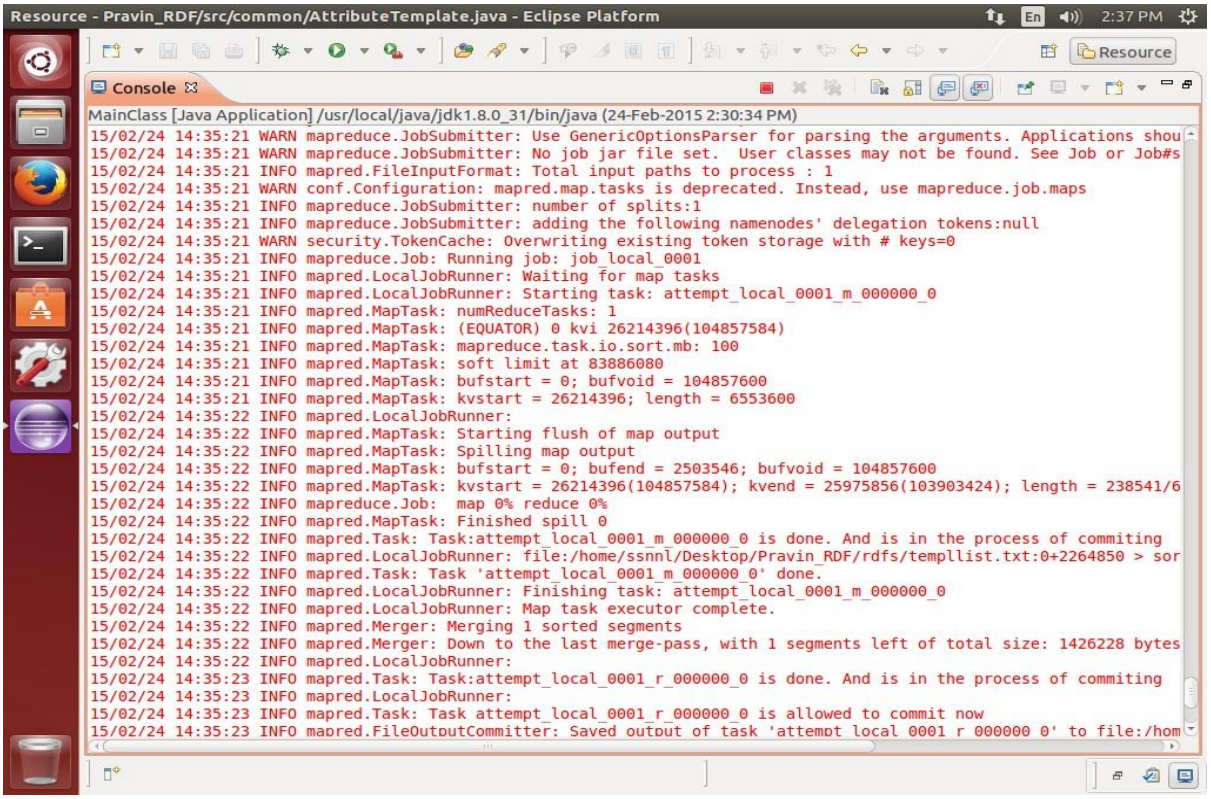


Figure 7 Map Reduce process stage 1

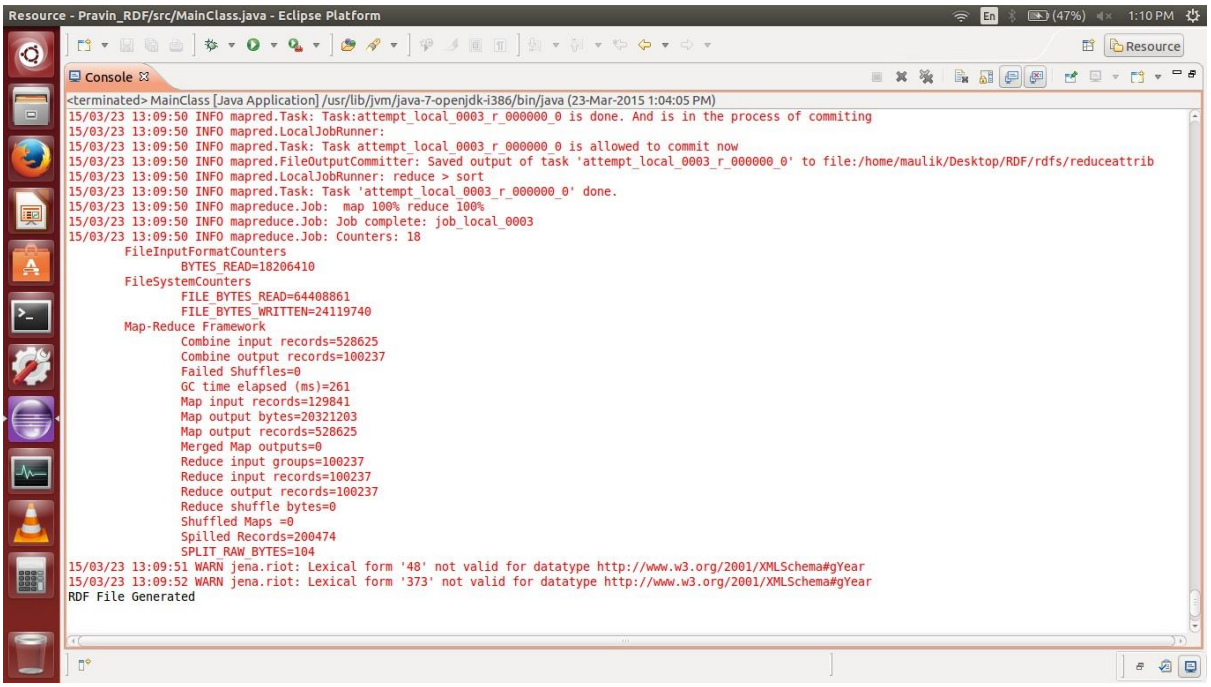


Figure 7 Map Reduce process stage 2 and RDF generate

We discover improved % Accuracy or Reduced % Error Rate from the use of Map Reduce method to generate the RDF and queries execute to optimized execution.

VI.A.1.1 CONCLUSION AND FUTURE WORK

In this paper, the goal of Hadoop MapReduce method using to retrieve the large scale of RDF dataset processed. RDF data process using the MapReduce to extract the data and then it check to count of each function and occurrence of the value. After it generate the new RDF on based of the given algorithm. The main goal to reduce time execution of the SAPRQL queries. SAPRQL queries can get the data through the RDF dataset. SPARQL queries execute faster if it apply indexing function on it and also reduce query execution time.

ACKNOWLEDGEMENT

The authors would like to thank the reviewers for their precious comments and suggestions that contributed to the expansion of this work.

REFERENCES

[1] RDF Primer. W3C Recommendation. http://www.w3.org/TR/rdf-primer, 2004.296.
[2] Padmashree Ravindra," Towards Optimization of RDF Analytical Queries on MapReduce," Department of Computer Science, North Carolina State University,IEEE 2014.
[3] Jiewen Huang, Daniel J. Abadi, Kun Ren , "Scalable SPARQL Querying of Large RDF Graph, " VLDB Endowment, Vol. 4, No. 11
[4] Vikas V Deshpande, Kemafor Anyanwu, "Towards Scalable RDF Graph Analytics on MapReduce", Department of Computer Science North Carolina State University Raleigh, 2010.
Techniques," Informatica 31 (2007) 249-26.
[5] Martin Przyjaciel-Zablocki, Alexander Schatzle, Eduard Skaley, "Map-Side Merge Joins for Scalable," IEEE, 2013.
[6] Shih-Ying Chen, Po-Chun Chen "An Efficient Join Query Processing based on MJR Framework ," Department of Computer Science and Information Engineering National Taichung University of Science and Technology Taichung, Taiwan 2012 IEEE.
[7] Jiewen Huang, Daniel J. Abadi , Kun Ren ,"Scalable SPARQL Querying of Large RDF," Yale University, VLDB Endowment, Vol. 4, No. 11 , September 3rd 2011 Seattle Washington.
[8] Prasad Kulkarni ,"Distributed SPARQL query engine using MapReduce " , Master of Science Computer Science School of Informatics University of Edinburgh 2010..
[9] Martin Przyjaciel-Zablocki, Alexander Sch¨atzle ,"RDFPath: Path Query Processing on Large RDF Graphs with MapReduce" ,Springer-Verlag Berlin Heidelberg 2011.
[10] Malini Siva, A. Poobalan, "Semantic Web Standard in Cloud Computing," ISSN: 2231-2307, Volume-1, Issue-ETIC2011,January 2012..
[11] Jose M. Gimenez-Garcıa , Javier D. Fernandez," MapReduce-based Solutions for Scalable SPARQL Querying , " Open Journal of Semantic Web (OJSW)Volume 1,Issue 1 , 2014.
[12] Konstantina Palla , "A Comparative Analysis of Join Algorithms Using the Hadoop Map/Reduce Framework" , Master of Science School of Informatics University of Edinburgh, 2009

