ISSN 2305-8277 (Online), 2012, Vol. 1, No. 2, 42-56.

IMPLEMENTATION OF INFORMATION RETRIEVAL (IR) ALGORITHM FOR CLOUD COMPUTING: A COMPARATIVE STUDY BETWEEN WITH AND WITHOUT MAPREDUCE MECHANISM[*]

RIKTESH SRIVASTAVA[†]
Skyline University College, UAE

ABSTRACT

The procedure of Information Retrieval (IR) algorithm appears disingenuously modest when observed from the viewpoint of terminological explanation. However, the implementation mechanism of the IR Algorithm is quite complicated and particularly when implemented to gratify the definite organizational requirements. In this research, the Information Retrieval Algorithm is developed using the MapReduce mechanism to retrieve the information in a Cloud computing environment. The MapReduce algorithm was developed by Google for experimental evaluations. In the present study, the algorithm portrays the results in terms of number of buckets required to generate the output from the large chunk of data in Cloud computing. The algorithm is the part of the complete Business Intelligence tool to be implemented and the results to be delivered for Cloud computing architecture.

Keywords: IR Algorithm, Cloud Computing; MapReduce, Business Intelligence; Name nodes; Data nodes; Main Server; Secondary Server; Database Server

INTRODUCTION

Cloud computing is evolving as a novel prototype for extremely scalable, fault-tolerant, and compliant computing on enormous clusters of computers. Cloud architectures provide highly obtainable storage and compute capacity through dissemination and replication. Cloud computing as a developing technology is anticipated to restructure the information retrieval procedures in the near future. A typical cloud application would have a data owner outsourcing data services to a cloud, where the data is stored in a keyword-value form, and users could retrieve the data with several keywords (Qin, Chiu, Jie & Guojun, 2012). Due to this reason, MapReduce mechanism finds its suitability to design and implement the IR Algorithm. Also importantly, Cloud architectures adapt to changing requirements by dynamically provisioning new (virtualized) compute or storage nodes (Xu, Meina, Xiaoqi & Junde, 2009). Also numerous services and dynamically scalable virtualized resources are added to the cloud (James, 2010). Almost at every instance of time and Cloud

[*] The views or opinions expressed in this manuscript are those of the author(s) and do not necessarily reflect the position, views or opinions of the editor(s), the editorial board or the publisher.
[†] Corresponding author Email: riktesh.srivastava@gmail.com

computing makes the resources available universally with better flexibility (Jingfang & Xing, 2005).

The need for improvements in information services including information retrieval is now mandatory due to the rapid growth of virtualized resources in cloud (Jingfang & Xing, 2005). All the cloud resources are distributed whereas the existing search engines such as Yahoo, Google, and MSN are centralized systems (Htoon & Thwin, 2008). Centralized systems are suffering from the different drawbacks including less scalability, frequent server failures and information retrieval issues as mentioned by (Watters, 1999).

Document virtualization is also becoming popular over the last few years (Kirpal, Kishorekumar & Revathy, 2010). Existing distributed IR models are also unable to search inside a virtualized physical node with multiple virtual systems running in parallel in the form of a grid. Htoon & Thwin (2008) proposed a distributed IRmodel to resolve the issue of accurate and fast allocation of required information but still many issues are unsolved. A modified IR model is the need of the time which can work efficiently with virtualized resources Jingfang & Xing (2005). This paper is an attempt to design the IR algorithm with the employment of MapReduce mechanism. The algorithm is verified and simulated results are evaluated based on the following criteria's:

1. The algorithm takes the number of Search requests as input.
2. The algorithm then breaks the Search requests into number of chunks required for the information retrieval from the public cloud.
3. Based on the two assumptions, the algorithms does the mapping functionalities and determines the number of buckets required to perform the reduce function of the algorithm.

Thus, the main aim of the algorithm is to regulate the amount of buckets (packets) required to accomplish the MapReduce algorithm without any deterrent. The algorithm (as depicted in the Annexure A) of the paper is being tested on the large number of requests based on different chunks of data.

The rest of the paper is divided as follows: Section 2 elucidatesabout the MapReduce mechanism. Section 3 elaborates about Cloud computing architecture in detail. Section 4outlines the elementary considerations for the IR algorithm using MapReduce mechanism. Section 5describes the IR algorithm and description of the different functions used in the actual Java code. Section 6illustrates the outcomes of the code execution. Section 7 particularizes the inference and commendations based on the experimentation. The paper also includes Annexure A which includes the Java code snippet for IR Algorithm.

MAPREDUCE MECHANISM

The concept of MapReduce was introduced by Google in 2004 and is the backbone of many larger data computations. MapReduce is fundamentally a divide and conquer algorithm which breaks down the problem in to small components and processing it in parallel to accomplish efficient computation on a larger data set. The MapReduce mechanism includes steps:

1. Map
2. Reduce

Map

In Map step, the Main node acquires the input, partitions it up into smaller sub-problems, and distributes them to data nodes. A data node may do this over in turn, leading to a multi-level tree structure. The data node processes the smaller problem, and passes the response back to its main node.

Reduce

In Reduce step, the main node then collects the responses to all the sub-problems and merges them in severalways to outline the output – the respond to the problem it was initially trying to resolve. The overall structure of MapReduce mechanism is depicted in Figure 1.

CLOUD COMPUTING ARCHITECTURE

The cloud computing architecture used for the experiment includes three different types of servers, namely:

1. Main Server
2. Secondary Server
3. Database Serve

The cloud architecture has both master nodes and slave nodes. In this enactment, a main server is one that gets client requests and handles them. The master node is present in main server and the slave nodes in secondary server.Search requests are forwarded to the MapReduce algorithm present in main server. MapReduce takes care of the searching and indexing procedure by instigating a large number of Map and Reduce processes. Once the MapReduce process for a particular search key is completed, it returns the output value to the main server and in turn to the client. The complete architecture is depicted in Figure 2.

Insert Figure 1 about here

Insert Figure 2 about here

As mentioned in Figure 2, the information required by the client is send directly to the Main Server. For simplicity, the Main server is termed as Name node and stores the Meta data about the information. The Meta data includes the size of the file, exact location of the file, block locations amongst others. Each of the information (file) is replicated in number of Secondary Servers, named as Data nodes. Data nodes are actually responsible to track the data from the data centers.

The complete functionality of the MapReduce algorithm operates as follows:

1. The client requests arrive at the Main Node.
2. The Main node has the MapReduce algorithm in place and does the task of mapping. In nutshell, Name node keepstrajectory of complete file directory structure and the placement of chunks. Thus Name node is the essential control point for the complete system. To read a file, the client API will calculate the chunk index based on the offset of the file pointer and make a request to the Name node. The Name node will reply which Data nodes has a copy of that chunk. From thispoint, the client contacts the Data node directly without going through the Name node.
3. The client pushes its changes to all Data nodes, and the change is stored in a buffer of each Data node. After changes are buffered at all Data nodes, the client send a “commit” request, and client gets the response about the success.

The above-mentioned three steps are depicted in Figure 3.

After accomplishment of the three steps stated above, all modifications of chunk distribution and metadata alterations will be transcribed to an operation log file at the Name node. This log file preserves an order list of operation which is significant for the Name node to recover its view after a crash. The Name node also keeps its persistent state by frequently check-pointing to a file.

Insert Figure 3 about here

IR ALGORITHM WITH AND WITHOUT MAPREDUCE MECHANISM

As the study conducted in the research is the comparative analysis of performance of IR algorithm with and without MapReduce mechanism, this segment of the paper elaborate the flowchart of implementation of both the algorithms in detail.

Flowchart of IR Algorithm without MapReduce mechanism

The IR algorithm implementation without MapReduce works in three fold:

a. The requests are broken into number of parts.
b. Each of these parts are processed in sequential order at different data centers and response is send back to the main server.
c. The main server which has IR Algorithm joins each of the response and sends back to the user.

Insert Figure 4 about here

Flowchart of IR Algorithm via MapReduce mechanism

In this section, the IR Algorithm using the MapReduce implementation for the cloud computing environment is being developed and executed. The proposed algorithm is used in IR Algorithm to retrieve results from the World Wide Web, and the outcomes depicted in the next section shows that MapReduce mechanism are used to improve the rapidity of information search. The proposed algorithm is an iterative method that makes use of the three methods, namely, map() reduce() and combine(), in the main server, to show the results. Categorization is used to retrieve and order the results according to the user choice to personalize the search.

Insert Figure 5 about here

RESULTS

The Results of the entire experiment are depicted in this segment of the paper. Few imperative points significant here are:

1. Experiment is conducted between 5000 to 20000 requests/s
2. The experiments represent the outcome for the pool of four Bucket sizes, 1000, 2000, 3000 and 4000

TABLE 1

Comparative study of IR Algorithm with and without MapReduce mechanism

Number of Requests/s	Bucket Size=1000		Bucket Size=2000	
Choice of the IR Algorithm	Response time without MapReduce Algorithm (in s)	Response time via MapReduce Algorithm (in s)	Response time without MapReduce Algorithm (in s)	Response time via MapReduce Algorithm (in s)
5000	54213	53893	53922	53893
6000	66923	64883	65126	64893
7000	77343	75893	75898	75882
8000	87903	86893	87961	86893
9000	99123	97893	98956	97871
10000	129924	108894	119862	108894
11000	148264	120894	130700	120894

12000	159924	132894	149879	132894
13000	163434	144894	166973	144846
14000	156894	156894	176756	156894
15000	163268	168894	185683	168894
16000	192876	180894	192876	180894
17000	208734	192894	208342	192894
18000	229869	204894	238672	204894
19000	250980	216894	237803	216894
20000	277987	228894	249800	228894
5000	53922	53893	54798	53893
6000	65126	64893	66098	64882
7000	75898	75882	79876	75882
8000	87961	86882	87762	86893
9000	98956	97893	99877	97893
10000	119862	108894	110872	108872
11000	130700	120833	139813	120871
12000	149879	132894	158090	132894
13000	166973	144894	179898	144894
14000	176756	156894	190232	156882
15000	185683	168894	209873	168882
16000	192876	180894	219098	180894
17000	208342	192846	239098	192894
18000	238672	204894	248803	204882
19000	237803	216894	270892	216894
20000	249800	228894	308767	228894

Insert Figure 6(A) about here

Insert Figure 6(B) about here

Insert Figure 6(C) about here

Insert Figure 6(D) about here

EVALUATING THE PERFORMANCE

Dissimilar sets of requests were delivered, each of altered size, and accomplished the MapReduce jobs in singlenode clusters. The corresponding times of execution were calculated and the conclusion of executing the experiment was that running MapReduce in clusters is by far the more effectual for a large volume of requests.

The two important inferences from the study lead to two obviousresults:

- In a cloud environment, the MapReduce structure upsurges the adeptness of throughput for large number of requests. In contrast, one wouldn't unescapably see such an increase in throughput in a non-cloud system.
- When the data set is small, MapReduce do not affectsubstantial increase in throughput in a cloud system.

Therefore, consider a combination of MapReduce-style parallel processing when planning to process a large amount of requests in the cloud system.

REFERENCES

Htoon, H., & Thwin, M. M. T. (2008, May). Mobile agent for distributed information retrieval system. In *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, 2008. ECTI-CON 2008. 5th International Conference on* (Vol. 1, pp. 169-172). IEEE.

James F. (2010). Article 'research on demand', *Research Information*, posted on 6 January (2010).

Jingfang, X., & Xing, L. (2005). Design and implementation of a scalable distributed information retrieval. *J. J Tsinghua Univ (Sci& Tech), 1842*-1846.

Kirpal A. Venkatesh, Kishorekumar Neelamegam, R. Revathy. (2010). Using MapReduce and load balancing on the cloud Hadoop MapReduce and virtualization improves node performance. *IBM Corporation,* Retrieve from: http://www.ibm.com/developerworks/cloud/library/cl-mapreduce/cl-mapreduce-pdf.pdf

Liu, Q., Tan, C. C., Wu, J., & Wang, G. (2012, March). Efficient information retrieval for ranked queries in cost-effective cloud environments. *In INFOCOM, 2012 Proceedings IEEE* (pp. 2581-2585). IEEE.

Watters, C. (1999). Information Retrieval and the Virtual Document. *Journal of the American Society for Information Science, 50* (11), 1028–1029.

Xu, K., Song, M., Zhang, X., & Song, J. (2009, August). A cloud computing platform based on p2p. In IT in Medicine & Education, 2009. *ITIME'09. IEEE International Symposium on* (Vol. 1, pp. 427-432). IEEE.

APPENDIX

FIGURE 1
MapReduce structure

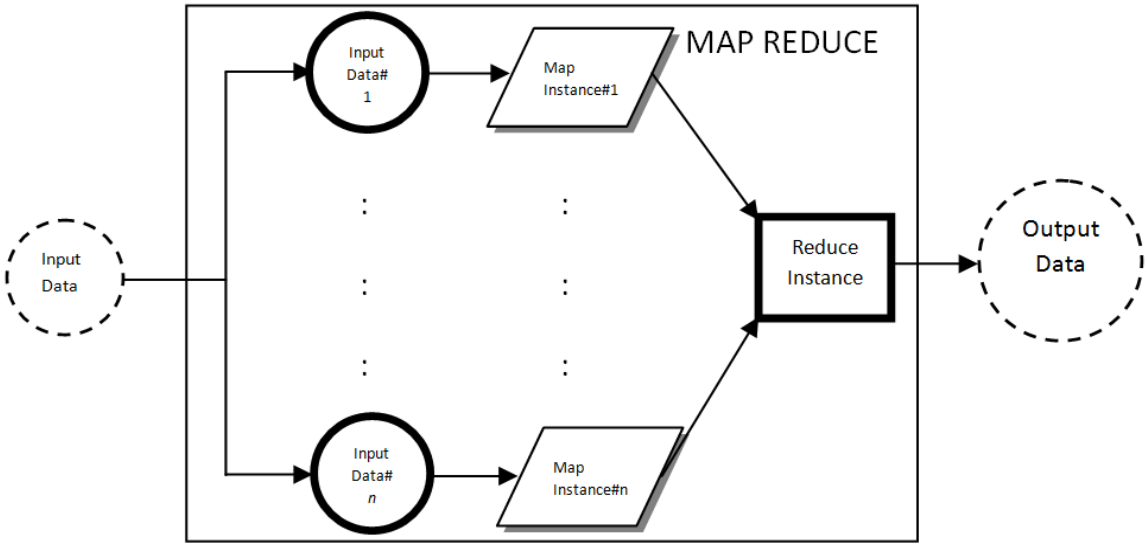

FIGURE 2
Implementation of Information Retrieval (IR) Algorithm in a Cloud computing Environment

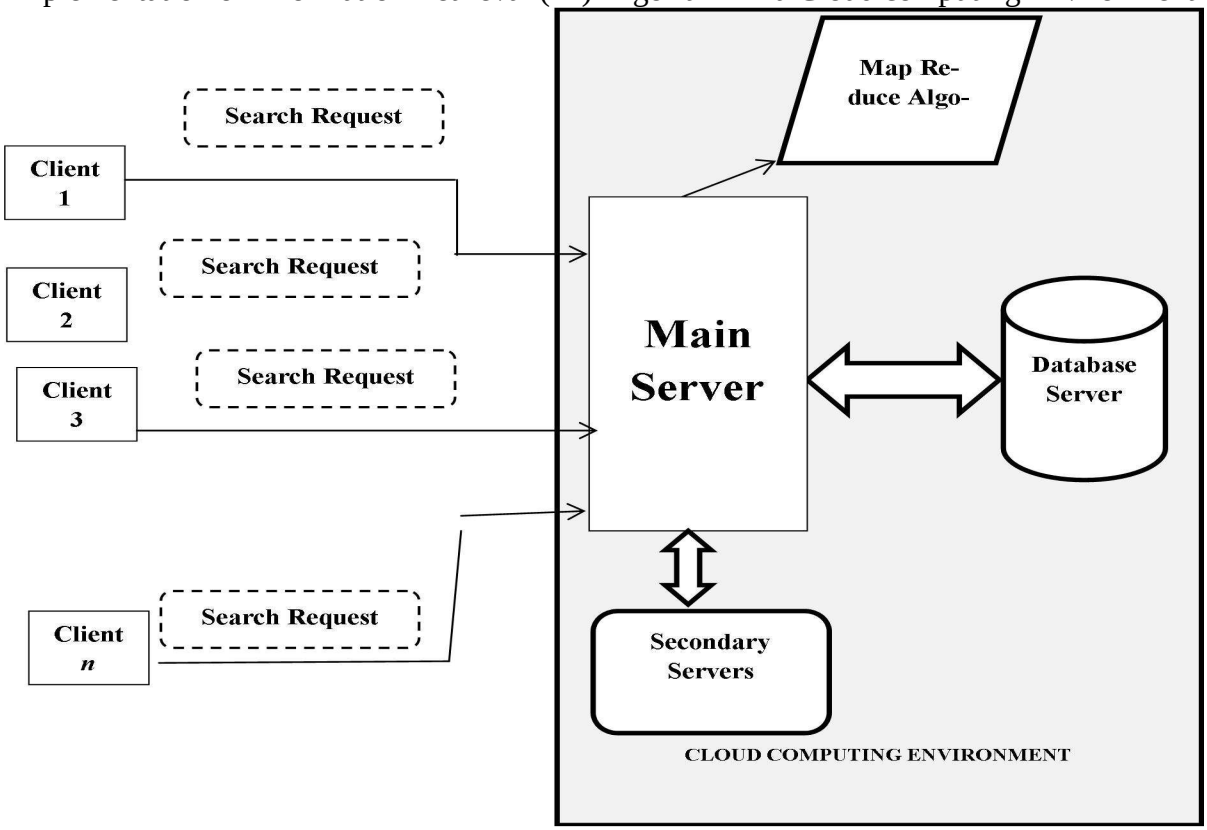

FIGURE 3
Operational Steps of the IR Algorithm using MapReduce in a Cloud Computing Environment

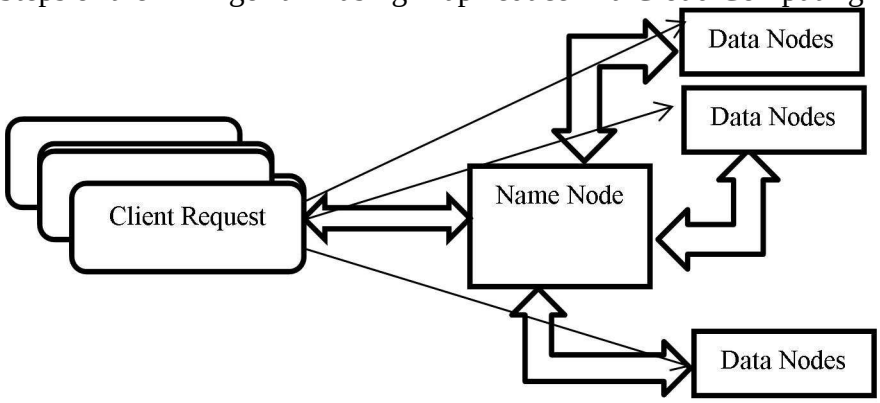

FIGURE 4
IR Algorithm without MapReduce mechanism

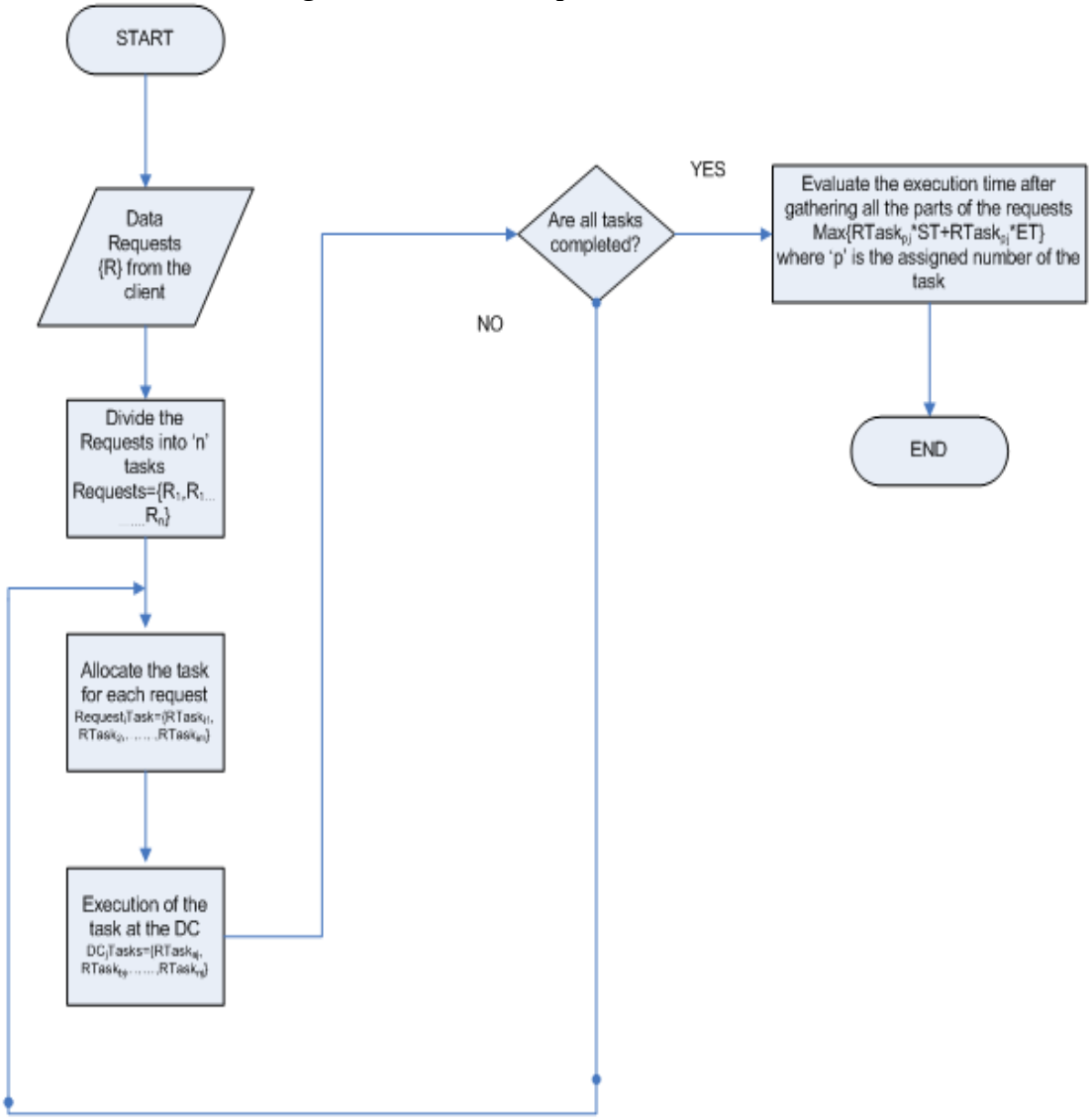

FIGURE 5
IR Algorithm with MapReduce mechanism

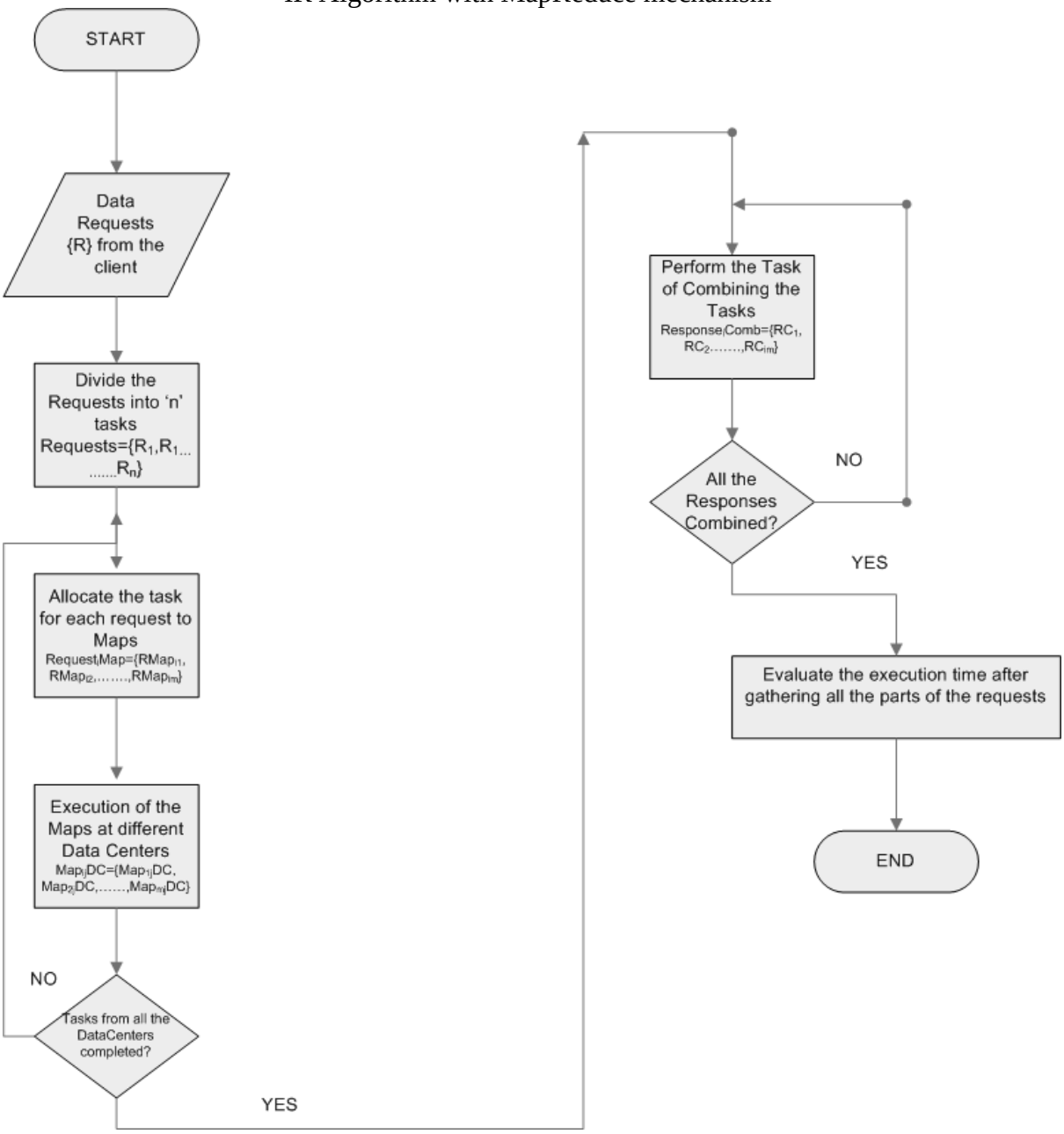

FIGURE 6(A)
IR Algorithm with Bucket Size=1000

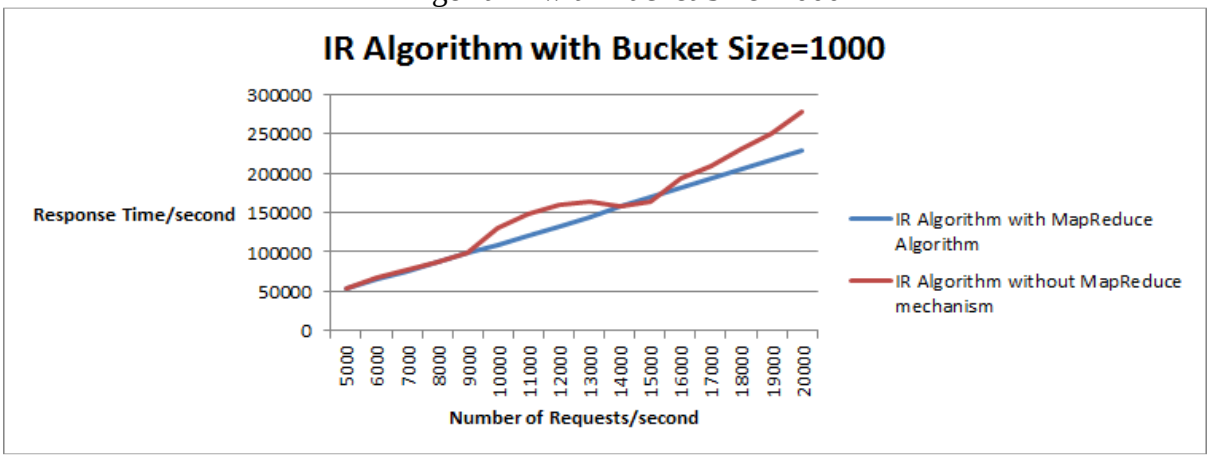

FIGURE 6(B)
IR Algorithm with Bucket Size=2000

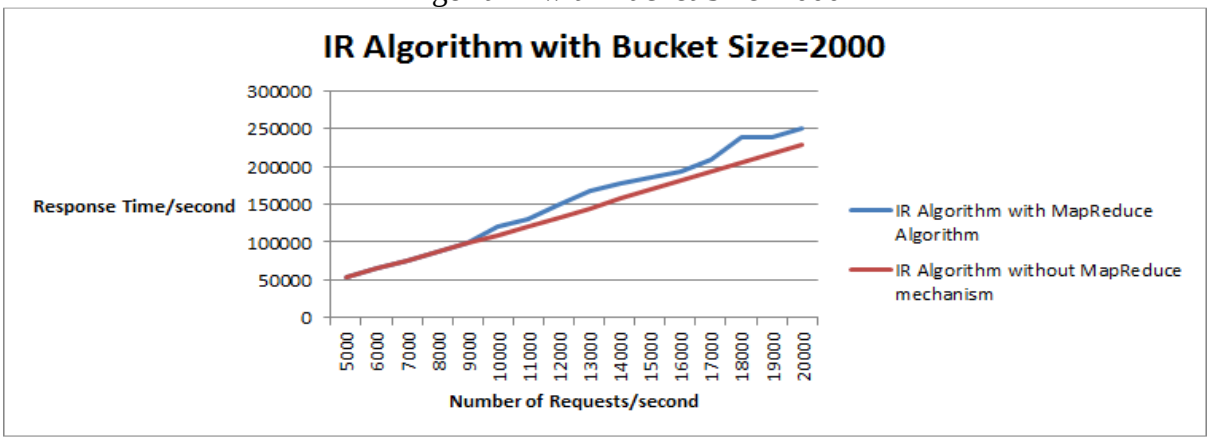

FIGURE 6(C)
IR Algorithm with Bucket Size=3000

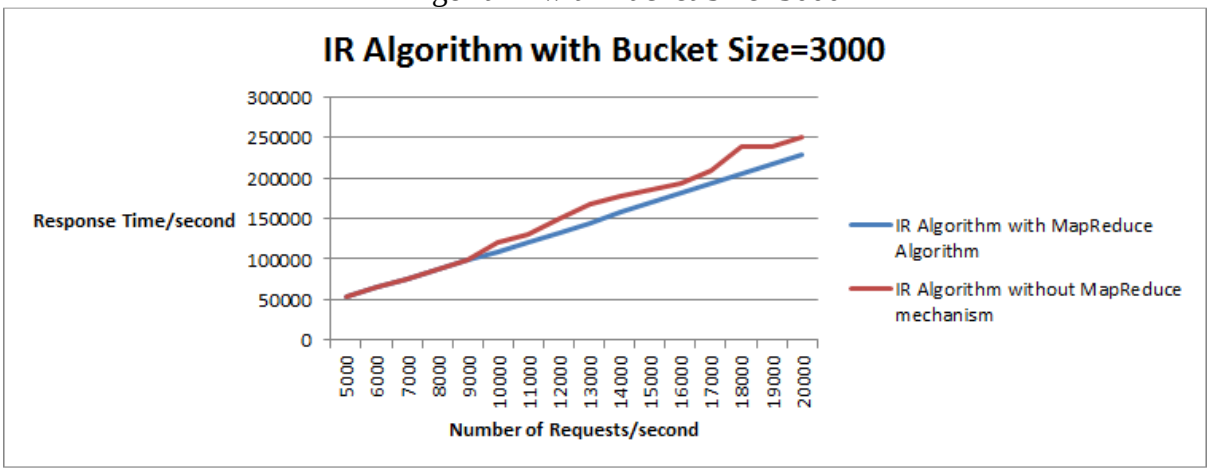

FIGURE 6(D)
IR Algorithm with Bucket Size=4000