# Ron Hadoop

## Expanding Rcapabilities using Apache Hadoop

**Rochsyid Anggara**
**Balai Diklat Tambang Bawah Tanah**

**Yohanes Gultom**
**Faculty Of Computer Science**
**University Of Indonesia**
**Depok, Indonesia**
**Yohanes.gultom@ui.ac.id**



December 07, 2016

**Abstract** R has recently became a new standard programming language for statistical computation and visualization for data scientist. However, R faces challenge when processing big data due to its reliance on computer's RAM. In the other hand, Apache Hadoop has been acknowledged as standard framework for highly scalable distributed system which is suitable to handle big data. Therefore, combining R with Hadoop is one of the most promising solution to enable convenient yet powerful big data statistical computation and visualization. In this paper, several options to run R on Hadoop are reviewed: Hadoop Streaming, RHive, RHIPE and RHadoop. Additionally, there are also options available for specific cases: tm.plugin.dc and Oracle Big Data Connectors.

**Keywords** R · Hadoop · RHive · RHIPE · RHadoop · Big Data

## 1 Introduction

### 1.1 R

R is an opensource programming language and environment for statistical computing and graphics/visualization designed by Ross Ihaka and Robert Gentleman [9]. R provides a wide variety of statistical and graphical techniques and highly extensible [1]. It's currently supported by more than 1000 add-in packages contributed by community. Some R real-world use cases are credit risk analysis, market data analysis, scientific reports .etc.

Despite of sharing similar functionality with older statistical tools (SPSS, SAS, Matlab or Ms. Excel), R is gaining more and more market shares[1] due to its flexibility and openness. R is implemented as programming language (unlike SPSS or Ms. Excel) so it is flexible to be used for almost anything: machine learning, data mining or interactive data visualization. Moreover, R is an opensource project which is designed to be extended easily. RStudio, an opensource R Integrated

Yohanes Gultom
University of Indonesia E-mail: yohanes.gultom@ui.ac.id

[1] http://r4stats.com/articles/popularity/

Development Environment (IDE), is also distributed freely as a tool to develop R packages (extensions) Figure 1.

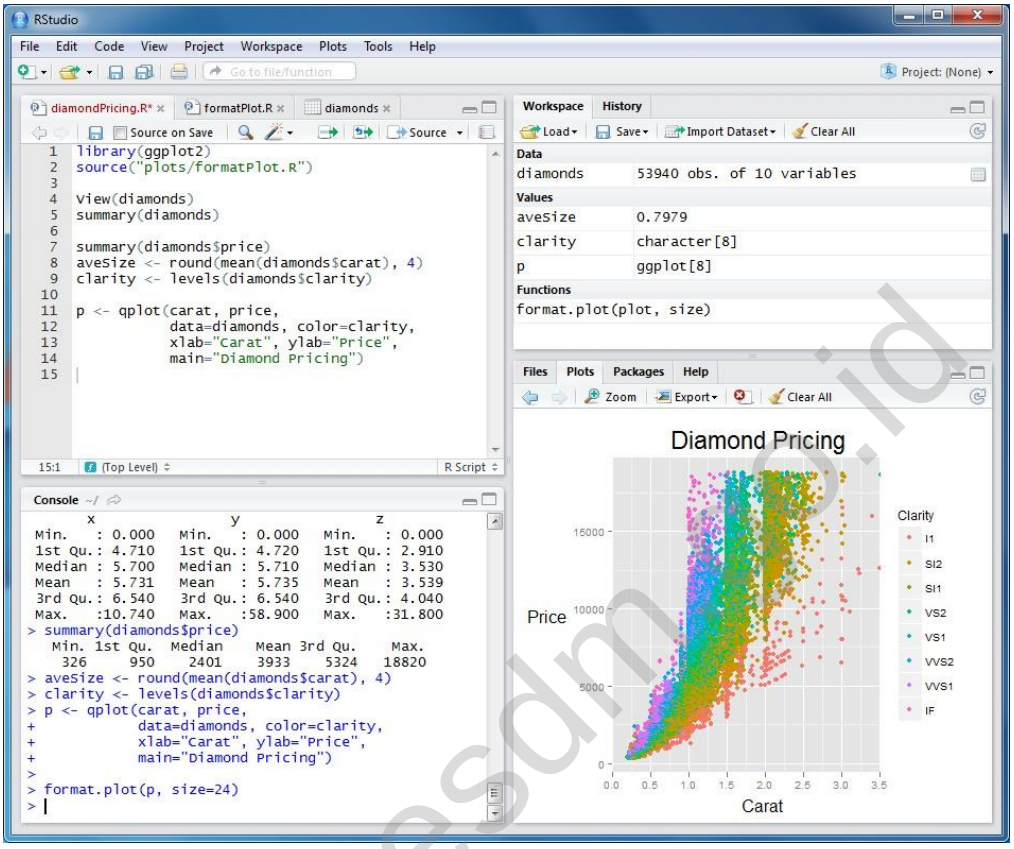


**Fig. 1** RStudio Desktop Application



## 1.2 Big Data

Big data is a trending buzzword used to describe high volume (terrabytes, petabytes of data) of various (unstructured, semi-structured and structured) data growing in high velocity (mostly real-time) which need to be processed within short time to maximize its value [6]. Examples of unstructured data are social media updates, blog articles, chat logs .etc. While semi-structured data is a self-describing structures such as Extensible Markup Language (XML) files, Javascript Object Notation (JSON) files, system logs .etc. Databases, such as Relational, Document-based, Key-value stores, are examples of structured data.

Some examples of big data are:

- Data generated by Facebook in one day[2]:
  - 500 TB of content (text, images, videos)
  - 2.5 billions ”shares”
  - 2.7 billions ”likes”
- Data generated by New York Times in a day [7]:
  - 50 GB of uncompressed log files
  - 50-100 millions of website clicks

In small amount, these kind of data may be worthless from business perspective. However, if they comes in big amount and are visualized properly (in charts or diagrams), it may provide various of insights proven useful for business [8].

## 1.3 Apache Hadoop

Apache Hadoop is an opensource framework written in Java allowing distributed processing of large datasets across cluster of commodity computers using simple programming model [2]. It is currently considered a de-facto standard in big data processing and storage. It provides theoretically unlimited scalability and is supported by major vendors in the software industry [7]. Big companies, like Yahoo, Facebook and Alibaba, have been investing intensively in Hadoop-based clusters to handle their big data related tasks such as searching, statistics or recommendation engine.

Hadoop is composed by 3 primary modules: Hadoop MapReduce, Hadoop Distributed File System (HDFS) and Hadoop Yet Another Resource Negotiator (YARN).

### *1.3.1 MapReduce*

MapReduce is a software framework or library originally proposed by Google for large scale processing of data sets [10]. It is composed of 2 (abstract) functions map and reduce which run on each nodes in clusters as shown in Figure 2. map function maps (eg. grouping or categorization) distributed data to proper nodes to be processed by further. reduce function applies aggregation operations on the mapped data (eg. summation, average) and return the result to the client/output.

On Hadoop cluster, MapReduce functions run on slave nodes called Task Nodes which are controlled and monitored by a Job Tracker (usually implemented on Master Node). Task Nodes are reading and writing the data directly from HDFS so they are tightly coupled.

### *1.3.2 HDFS*

Hadoop Distributed File System (HDFS) is Hadoops rack-aware distributed file system which was inspired and derived from the concept of Google File System (GFS) and built on the UNIX OS [9]. Being ”rack-aware” means HDFS is providing data-loss protection switch or power failure.

[2] http://techcrunch.com/2012/08/22/how-big-is-facebooks-data-2-5-billion-pieces-of-content-and-500-terabytes-ingested-every-day/

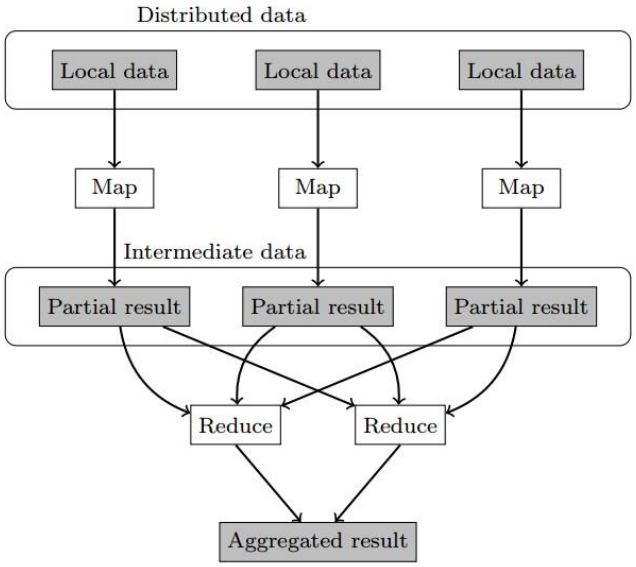


**Fig. 2** MapReduce Diagram

HDFS relies on a client/server architecture consisting a Name Node and a number of Data Nodes which manage storage itself [7]. Name node, acts as the file system namespace, is usually implemented on a Master Node along with Job Tracker.

### *1.3.3 YARN*

Yet Another Resource Negotiator (YARN) is a framework for job scheduling and cluster resource management [7]. It basically acts as a Master Node that hosts the implementation of Job Tracker and Name Node. The fundamental idea to split up the two major functionalities of the Job Tracker, resource management and job scheduling/monitoring, into separate daemons.

## 1.4 R on Hadoop

Despite of its capabilities, R faces problem when handling big data since, by default, it stores and manipulates data in memory [6]. Due to their popularity, combining R and Hadoop as integrated solution become an interesting option to be explored further. By utilizing Hadoop distributed processing capability, R is expected to be able to do statistical computing and visualization of big data in effective and affordable manner.

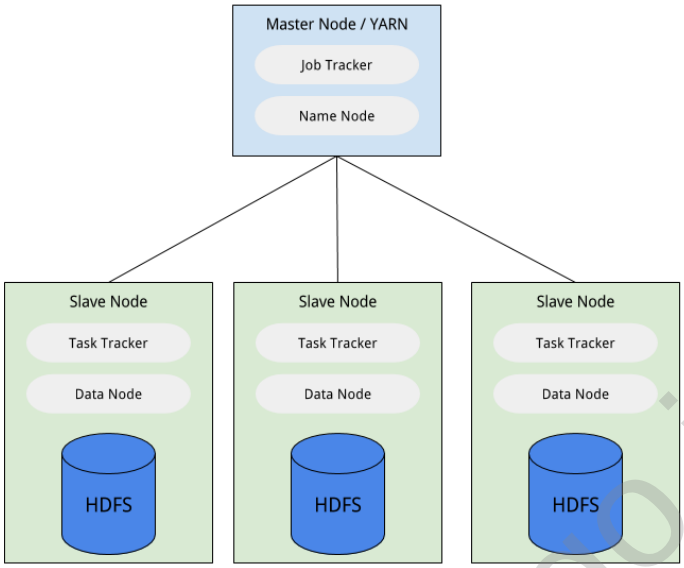


**Fig. 3** Apache Hadoop Architecture

## 2 R-Hadoop Integration

Several options are available to integrate R and Hadoop: Hadoop Streaming, RHive, RHIPE and RHadoop. For specific use cases there are also some alternatives: Oracle Big Data Connectors and tm.plugin.dc.

### 2.1 Hadoop Streaming

The most basic way to integrate Hadoop MapReduce with other computation technology is by using a built-in feature from Hadoop itself: Hadoop Streaming. Streaming is a built-in Hadoop feature allowing users to run MapReduce jobs using any script or executable that reads data from standard input and writes the results to standard output as the mapper or reducer [7]. Therefore, we can create an R scripts as a mapper and reducer utilizing standard input and output and run it using streaming.

Assuming that we already have R programs `map.R` and `reduce.R` that respectively do map and reduce function against text data, we can invoke Hadoop MapReduce using streaming in unix environment (eg. Linux or Mac OS) as described by Figure 4. This command also assumes that the input data is already

```
$ ${HADOOP_HOME}/bin/Hadoop jar
        ${HADOOP_HOME}/contrib/streaming/*.jar
        -inputformat org.apache.hadoop.mapred.TextInputFormat
        -input input_data.txt
        -output output
        -mapper /home/tst/src/map.R
        -reducer /home/tst/src/reduce.R
        -file /home/tst/src/map.R
        -file /home/tst/src/reduce.R
```

**Fig. 4** Example of Hadoop Streaming with R command

available on HDFS input_data.txt. By defining -file options, all R scripts will be distributed to Hadoop slave nodes where map and reduce process take place.

The major advantage of this integration method is easy configuration. What we need to do is simply configure R in every Hadoop slave nodes and upload the input data to HDFS [7]. There is no additional packages needed for R neither Hadoop.

The biggest problem with this approach is performance overhead due to the usage of standard input/output [13]. Another trade-off of using this solution is that programmers are unable to access Hadoop modules such as HDFS or HBase (Hadoop database) from R programs.

### 2.2 RHive

RHive is an opensource R package which provides interface to interact with Apache Hive, a data warehouse software facilitates querying large datasets residing in distributed storage such as HDFS [3]. Through RHive, one can execute HiveQL, an SQL-like query language, directly from R against input data residing in HDFS. This can be achieved by calling rhive.query() function which the basic usage is shown in snippet Figure 5. The query will be converted to a MapReduce process in Hadoop by Hive and the output of the query will be stored in R Data Frame by RHive [8].

```
exampleQuery <- sprintf("select CarNum, highSpd, Correlation
        from rawdata
        where Correlation < %d
        and infoDate > %s
        and infoDate < %s ", interval, sDate, eDate)
result <- rhive.query(exampleQuery)
```

**Fig. 5** Example of HiveQL query through RHive

At the time this paper is written, RHive has been removed from CRAN (official R archive network) due to an issue with RHive maintainer's email. So currently

we can only retrieve archived versions of RHive from CRAN. If we need to get the latest version, we should retrieve it from its GitHub repository[3]. A basic guide to install RHive on simple Hive and Hadoop setup is also available in the repository.

The advantages of this solution are the usage of HiveQL and the integration with powerful Apache Hive. HiveQL hides MapReduce complexity from R programmers under the familiarity of SQL-like syntax. Hive capability is also well-proven since it has been used by Facebook to analyze its huge social media databases[4].

Compared to Hadoop Streaming, RHive's disadvantage lies on the fact that it requires Apache Hive and additional R packages. The required R packages are rjava , rserve and runit which need to be installed on Hadoop slave nodes except rjava (only on local machine). The effort to install, configure and maintain the Hive needs to be taken into consideration

## 2.3 RHIPE

RHIPE is an opensource R package developed by Saptarshi Guha to run Hadoop MapReduce wholly from R [5]. RHIPE, stands for "R and Hadoop Integrated Programming Environment", provides R programmers with all facilities Java programmers have when programming Hadoop [7]. Additionally, RHIPE is able to do Divide and Recombine algorithm which is a new statistical approach to analyze large complex data [12].

Basic R program structure of RHIPE MapReduce job is shown in Figure 6. The first most function after library loading is `rhinit()` which initializes the process. Two most important functions in the structure are `rhmr()` and `rhex()` — the first one is responsible to create Hadoop MapReduce job and the latter executes the job. RHIPE also provides functions to communicate with Hadoop during the job execution such as `rhcollect()` (writes data to Hadoop MapReduce) and `rhstatus()` (returns the job's status) [7]. Besides that, RHIPE also has several functions to deal with HDFS described in its online documentation[5] such as `rhput()`, `rhget()`, `rhread()`, `rhwrite()` .etc.

Compared to RHive, RHIPE is currently relatively easier to install because it doesn't require additional server (eg. Apache Hive) and it can be installed together with `datadr` and `trelliscope` using R's `devtools`[6]. The publisher of RHIPE, Tessera[7], even provides Vagrant[8] Virtual Machine (VM) which has RHIPE preconfigured in it and ready to be used. So now RHIPE installation procedure is not as difficult as described by Oancea (2014) anymore.

[3] https://github.com/nexr/RHive

[4] https://www.facebook.com/notes/facebook-engineering/hive-a-petabyte-scale-data-warehouse-using-hadoop/89508453919

[5] http://www.stat.purdue.edu/ sguha/rhipe/doc/html/functions.html

[6] https://cran.r-project.org/web/packages/devtools/README.html

[7] http://tessera.io

[8] https://www.vagrantup.com/

```
library(Rhipe)
rhinit(TRUE, TRUE);
map<-expression ({
        lapply (map.values, function(mapper) {
                ## mapper code
        })
})
reduce<-expression(
        pre = {
                ## pre-reduce code
        },
        reduce = {
                ## reduce code
        },
        post = {
                ## post-reduce code
        },
)
x <- rhmr(
        map=map,
        reduce=reduce,
        ifolder=inputPath,
        ofolder=outputPath,
        inout=c( text , text ),
        jobname= a job name )
)
rhex(z)
```

**Fig. 6** RHIPE basic usage's structure

### 2.4 RHadoop

RHadoop is an opensource project developed by Revolution Analytics[9] which recently acquired by Microsoft. Unlike RHIPE, RHadoop is actually contains more than one R package. Currently it consists of 5 packages[4] which is more than previously described by Oancea (2014). Those 5 R packages are:

1. **rhdfs**: provides basic connectivity (file management) to the HDFS. This package only needs to be installed in the R client machine(s).
2. **rhbase**: provides basic connectivity (database management) to the HBASE (Hadoop distributed database) which is a column-oriented database to store big data [9]. This package requires Apache. Thrift[10] and only needs to be installed in the R client machine(s).
3. **plyrmr**: provides common data manipulation operations, as found in popular packages such as plyr[11] and reshape2[12], on very large data sets stored on Hadoop. This package needs to be installed on every Hadoop nodes.

[9] http://www.revolutionanalytics.com/
[10] https://thrift.apache.org/
[11] https://cran.r-project.org/web/packages/plyr/index.html
[12] https://cran.r-project.org/web/packages/reshape2/index.html

4. **rmr2**: allows R programmer to perform statistical analysis in R using Hadoop MapReduce on HDFS. This package needs to be installed on every Hadoop nodes.
5. **ravro**: adds the ability to manipulate Avro[13] files from local and HDFS. This package only needs to be installed in the R client machine(s).

These 5 packages fit in Apache Hadoop environment as described ny Figure 7.

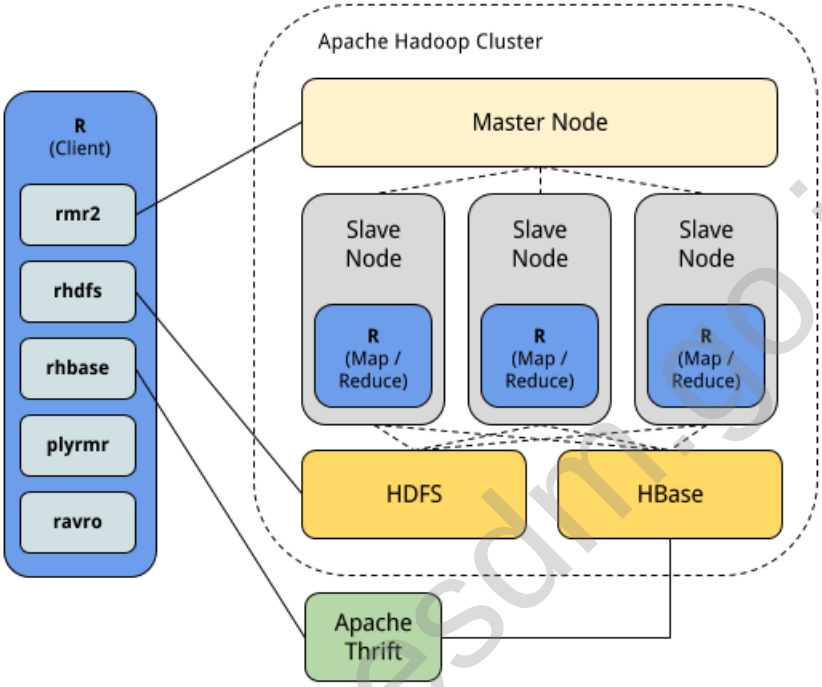


**Fig. 7** RHadoop architecture

Code listing Figure 8 shows the structure of rmr2 basic usage. The only function needed to define the MapReduce job is mapreduce() which accepts map and reduce functions along with input/output configurations. In terms of arguments, this function is very similar with RHIPE's rhmr() shown in Figure 6.

In code listing Figure 9 we can see simple R MapReduce program that interacts directly with HDFS using rhdfs function hdfs.init() (to initialize communication with HDFS) and from.dfs() (to retrieve file from HDFS) [9]. This basic example can also be used to test RHadoop installation[14].

Since RHadoop consists of 5 R packages, it provides more functionalities than RHIPE. They are interface to HBASE (eg. hb.list.tables(), hb.insert, hb.delete.table),

[13] https://avro.apache.org
[14] http://hadooptutorial.info/rhadoop-installation-on-ubuntu/

```
library(rmr2)
map<-function(k,v) {
        ## mapper logic
}
reduce<-function(k,vv) {
        ## reducer logic
}
mapreduce(
        input = data.txt ,
        output= output ,
        textinputformat = rawtextinputformat,
        map = map,
        reduce = reduce
)
```

**Fig. 8** RHadoop rmr2 basic usage structure

```
library(rhdfs)
library(rmr2)

hdfs.init()
sample<-1:10
small.ints<-to.dfs(sample)
out<-mapreduce(
        input = small.ints,
        map = function(k,v) keyval(v,v^2)
)
from.dfs(out)
df<-as.data.frame(from.dfs(out))
print(df)
```

**Fig. 9** RHadoop basic example

data manipulation operations (eg. melt, decast, count, sample) and Avro files manipulation [9]. These additions makes RHadoop better than RHIPE in terms of functionalities.

In terms of installation, RHadoop is as simple as RHIPE. Even though it doesn't utilize helper package such as devtools, we can download all RHadoop packages from its GitHub repository[15] and easily install RHadoop packages from RStudio which is the most common web-based IDE for R [8]. There is no preconfigured VM provided by Revolution Analytics so far but we can use Hortonworks Sandbox VM[16] which has RHadoop preconfigured.

[15] https://github.com/RevolutionAnalytics/RHadoop/wiki/Downloads
[16] http://hortonworks.com/products/hortonworks-sandbox/

### 2.5 Oracle Big Data Connectors

In a special case where the big data environment is dominated by products such as Oracle RDBMS, there is a set of components called Oracle Big Data Connectors [11] that can be used to integrate R and Hadoop.

Architecture of the proposed solution is composed of:

1. Oracle RDBMS + Oracle Big Data Connectors
   - Oracle SQL Connector for Hadoop
   - Oracle Loader for Hadoop
   - Oracle Data Integrator
2. Hadoop
   - Apache Hive: provide HiveSQL SQL-like query for HDFS
   - Apache Sqoop: data migration between HDFS-RDBMS
3. Analytics with R
   - Oracle R Connector with Hadoop (ORCH) package

Despite of quite complex installation and configuration, the proposed architecture managed to solve a churn analysis over a big data of telecommunication company [11]. So this may be an alternative enterprise solution where reliable support is required.

### 2.6 tm.plugin.dc

Text mining is one of the most common use case for R. tm package is one of the standard text mining package for R[10]. However, tm is also having problem dealing with big data as it relies on R. Therefore, Theul and Feinerer (2012) propose a plugin tm.plugin.dc to solve this issue.

This plugin depends on 2 R packages [10]:

1. **DSL (Distributed Storage and List)**: handling distributed files using similar approach with Message Passing Interface (MPI).
2. **HIVE** (Hadoop Interactive): provides interface to Hadoop Streaming explained in 2.1.

This plugin is also able to expand the capability or R to handle big data of text corpus by utilizing Hadoop MapReduce capability [10].

## 3 Conclusions

Researches have shown that using R on Hadoop is a proven solution for the issue of big data statistical computation and visualization. It is not only effective but also efficient and affordable because almost are open source and runs on commodity hardwares.

From the described options on using R with Hadoop, a general characteristics may be concluded:

1. **Hadoop Streaming**
   - Suitable when batch execution/processing is acceptable

- Easy and quick to be configured
- Only provide crude MapReduce without any integration on R client
- Not suitable when high-performance is required

2. **RHive**
   - Very suitable when SQL-like programming interface is required
   - No control on MapReduce process as it is not explicitly done in R programs
   - Quite difficult to setup and configure from scratch
3. **RHIPE**
   - Enables MapReduce from R programs
   - Provides access to manipulate HDFS but not to HBase
   - Supports Divide and Recombine (D&R) algorithm
   - Quite easy to be installed using devtools
   - Vagrant VM is available
4. **RHadoop**
   - Enables MapReduce from R programs
   - Provides access to manipulate HDFS and HBase
   - Composed of 5 packages which can be selected base on needs
   - Quite easy to be installed through RStudio
   - Hortonworks Sandbox VM is available
   - Most general and popular solution
   - Enterprise support by Microsoft may be available in the future

## 4 Open Problems

In this paper, general characteristics and setup process of each options to run R on Hadoop are described and can be used as a brief guidelines to choose most suitable solution. However, performance of each options have not been explored in this paper. So in the future works, comparison of each options in term of performance in solving various common big data problems needs to be evaluated.

Detailed step of installations and example of program of each options are also not yet provided in this paper. These details are also necessary to give better picture of the implementation. Therefore further research on how to implement each of each options, including the hardware requirements and corner cases, are also available for future works.

Despite of currently being a de-facto solution, Apache Hadoop is not an only solution to handle big data. There are other emerging solutions for distributed system such as Apache Spark[17], Apache Storm[18] and HPCC System[19] that can be explored to expand the capability of R as well. It also serves as an open problem for next researches.

## References

1.R Foundation, What is R?, https://www.r-project.org/about.html (2015)
2. Apache Foundation, What is Apache Hadoop?, https://hadoop.apache.org/ (2015)

[17] http://spark.apache.org/
[18] http://storm.apache.org/
[19] https://hpccsystems.com/

3. Apache Foundation, Apache Hive, https://hive.apache.org/ (2015)
4. Revolution Analytics, RHadoop, https://github.com/RevolutionAnalytics/RHadoop/wiki (2015)
5. Guha, S., RHIPE: R and Hadoop Integrated Programming Environment, https://github.com/tesseradata/RHIPE (2012)
6. Gahlawat, A. Big Data Analysis using R and Hadoop. IJCEM International Journal of Computational Engineering & Management, Vol. 17 Issue 5. p 9-14 (2014)
7. Oancea, B., Dragoescu, R. M. Integrating R and Hadoop for Big Data Analysis. Revista Romn de Statistic nr. 2 p 83-94 (2014)
8. Cho, W et al. Big Data Analysis with Interactive Visualization using R packages. Big Data Science August 0407, 2014, Beijing, China. (2014)
9. Rotte, A. et. al. Big data analytics made easy with RHadoop. IJRET: International Journal of Research in Engineering and Technology. 9-15 (2015)
10. Theul, S. et. al. A tm Plug-In for Distributed Text Mining in R. Journal of Statistical Software (2012)
11. Momtselidze, N., Kuksin, A. Hadoop Integrating with Oracle Data Warehouse and Data Mining. Journal of Technical Science and Technologies. 21-25 (2014)
12. Guha, S., Hafen, R., Rounds, J., Xia, J., Li, J., Xi, B. and Cleveland, W. S. Large complex data: divide and recombine (D&R) with RHIPE. Stat, 1: 5367. doi: 10.1002/sta4.7 (2012)
13. Ding, M. et al. More convenient more overhead: The performance evaluation of Hadoop streaming. RACS '11 Proceedings of the 2011 ACM Symposium on Research in Applied Computation Pages 307-313 (2011)