

# **National Institute of Technology,Raipur**

**Department of Computer Science & Engineering**



## **Applications of Hadoop in Distributed Computing**

### **A Term Project on Network Programming**

**GitHub Project Link:**

[https://github.com/Angelion1/Hadoop\\_Application\\_Websites\\_Interconnections](https://github.com/Angelion1/Hadoop_Application_Websites_Interconnections)

**Submitted By:**

Roll no: 14115011  
Roll no: 14115035  
Roll no: 14115037  
Roll no: 14115071  
Roll no: 14115083

Name: Anshul Verma  
Name: Hari Shrawgi  
Name: Harshdeep Kaur  
Name: Prerana Agrawal  
Name: Shriya Thawait

## **Abstract**

There has been a constant trend of our technology moving towards distributed computing. With the advent of Big-Data and Cloud, it seems inevitable that distributed computing would soon take the centre-stage.

In this era of data proliferation there is a need for intelligent use of our limited resources. We as computer scientists of this new generation must accustom ourselves with the new paradigms of computer science. In an attempt to do the same, in this project we have aimed to learn and harness the power afforded by the Apache Hadoop framework. Being familiar with JAVA through our Network Programming class has been a boon for us as Hadoop works brilliantly in tandem with JAVA.

In this project we have simple demonstrated the use of Hadoop to implement a task which is very time consuming when implemented sequentially. The application developed in this project deals with the domain of web-crawling and website indexing. Finding the various interconnections among websites is often the first step in many web applications like spidering. Thus, it is of prime importance that this step be time efficient.

Thus, we chose and implemented a Hadoop implementation of the task. MapReduce framework of the Apache Hadoop library has been used by us to divide and parallelize the task.

We were able to setup a single node cluster of Hadoop and successfully run our application. As a result of the application we found out the various links from our college website.

Github Repository of the project

[https://github.com/Angelion1/Hadoop\\_Application\\_Websites\\_Interconnections](https://github.com/Angelion1/Hadoop_Application_Websites_Interconnections)

## LIST OF FIGURES

FIGURE 1: BASIC OUTLINE OF THE APPLICATION .....	5
FIGURE 2: BASIC WORKING OF MAPREDUCE THROUGH SIMPLE WORD COUNT EXAMPLE.....	6
FIGURE 3: STARTING HADOOP SERVICES.....	8
FIGURE 4: INFORMATION OF OUR SINGLE NODE CLUSTER .....	9
FIGURE 5: SCREENSHOT DURING THE EXECUTION OF THE CODE.....	9
FIGURE 6: FINAL OUTPUT WHICH SHOWS THE INTERLINKING OF WEBSITES .....	10

## Table of Contents

Abstract .....	2
Introduction.....	4
1. Motivation.....	4
2. Project Objective.....	4
3. Outline.....	4
Chapter 1: Apache Hadoop.....	5
1. Introduction to Hadoop .....	5
2. MapReduce .....	6
3. Features of MapReduce .....	6
4. Use of MapReduce in our task.....	6
Chapter 2: Implementation .....	7
1. Introduction.....	7
2. Input .....	7
3. Mapper .....	7
4. Reducer .....	8
Results.....	8
Conclusion .....	10
References.....	10

# Introduction

## 1. Motivation

The main motivation for taking up this project has been to learn a technology whose relevance has significantly increased in the past and is pitted to be the future [1].

We believe that Distributed computing is a very important sub-domain of computer science which will become more and more important as the era Artificial Intelligence ushers in. With an inundation of data, advancements in silicon are failing to keep up [2]. Thus, it is logical that leveraging computing resources all around the world will soon be as common as sharing pen-drives.

The following points outline the need for distributed computing:

- File sizes have continued to increase at a faster rate than computing power
- CPU cores have reached the upper limit of frequency
- Distributed computing when implemented efficiently can be faster
- Distributed computing reduces the cost of computing dramatically

## 2. Project Objective

This project aims to showcase a use-case of Hadoop. In this project, we aim to implement a Hadoop application which can divide and optimise the computation of a simple web-finder program.

## 3. Outline

In this project we are developing a Hadoop application which applies MapReduce paradigm to the problem of finding closely linked web-sites. We basically have a list of websites, and we want to find the interconnections between these web-sites via hyperlinking. This task is highly optimisable using MapReduce technique of Hadoop. Using MapReduce, we would divide up the tasks into sub-tasks which will be performed at different nodes and then will be combined together to give the final output. This technique leverages all the resources available at our disposal.

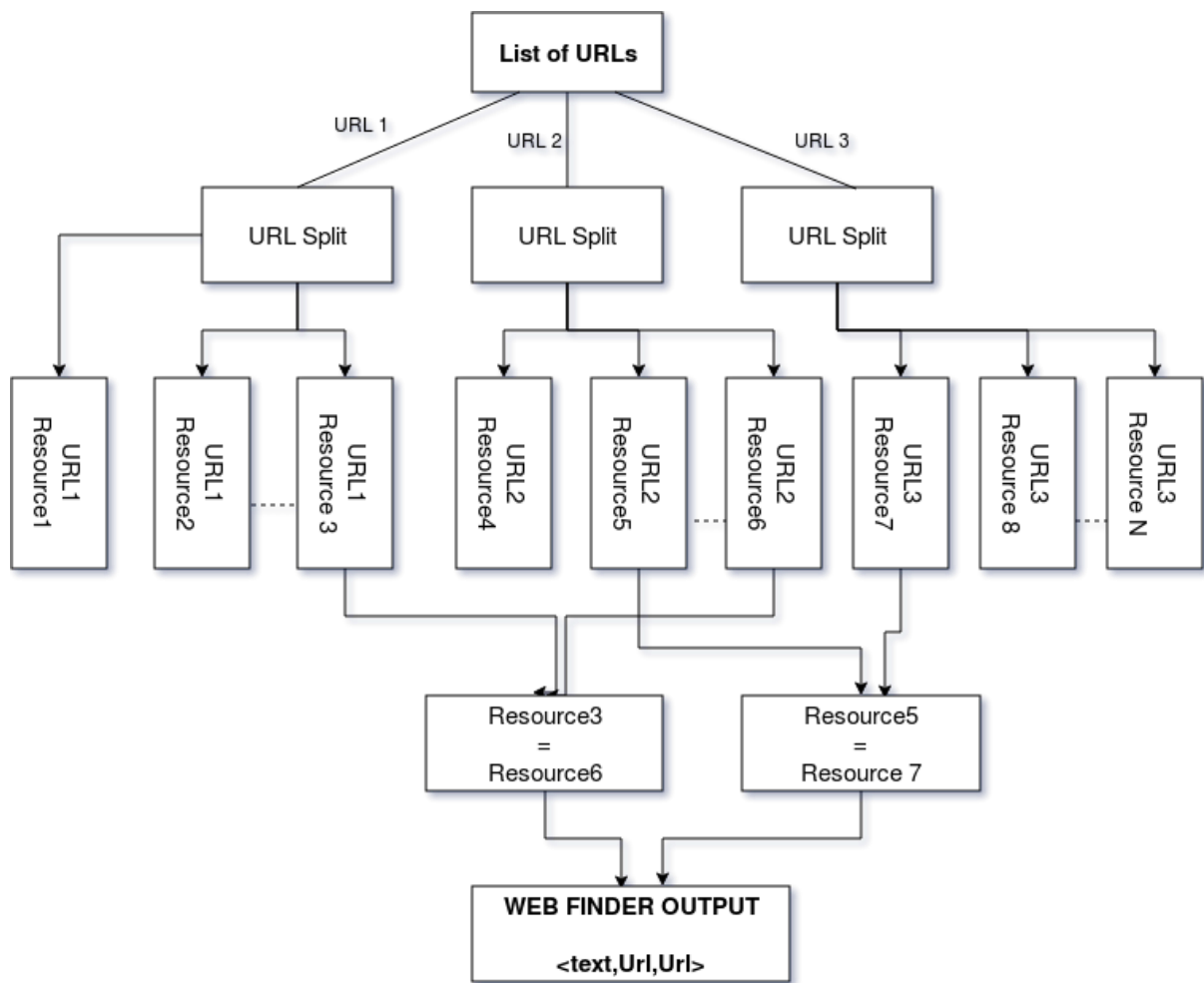


Figure 1: Basic outline of the application

## Chapter 1: Apache Hadoop

### 1. Introduction to Hadoop

Apache Hadoop is a software framework that supports data intensive distributed applications; it enables applications to work with thousands of nodes and petabytes of data.

Hadoop contains:

1. The projects Hadoop Common: the common utilities that support the other Hadoop subprojects.
2. Hadoop distribute file system(HDFS),
3. Hadoop MapReduce: a software framework for distributed processing of large data sets on compute clusters.
4. Avro: A data serialization system.
5. Chukwa: A data collection system for managing large distributed systems.
6. HBase: A scalable, distributed database that supports structured data storage for large tables.
7. Hive: A data warehouse infrastructure that provides data summarization and ad hoc querying.
8. Mahout: A Scalable machine learning and data mining library.

9. Pig: A high-level data-flow language and execution framework for parallel computation.

In our project we mainly focus on the MapReduce framework.

## 2. MapReduce

MapReduce is a programming model and software framework introduced by Google to support distributed computing on large data sets on clusters of computers. Hadoop MapReduce is an open source project from Apache, it allows us to develop parallel applications without any parallel programming techniques, and the applications could be easily deployed and executed, it works with the HDFS and processes huge datasets (e.g. more than 1TB) on distributed clusters (thousands of CPUs) in parallel.

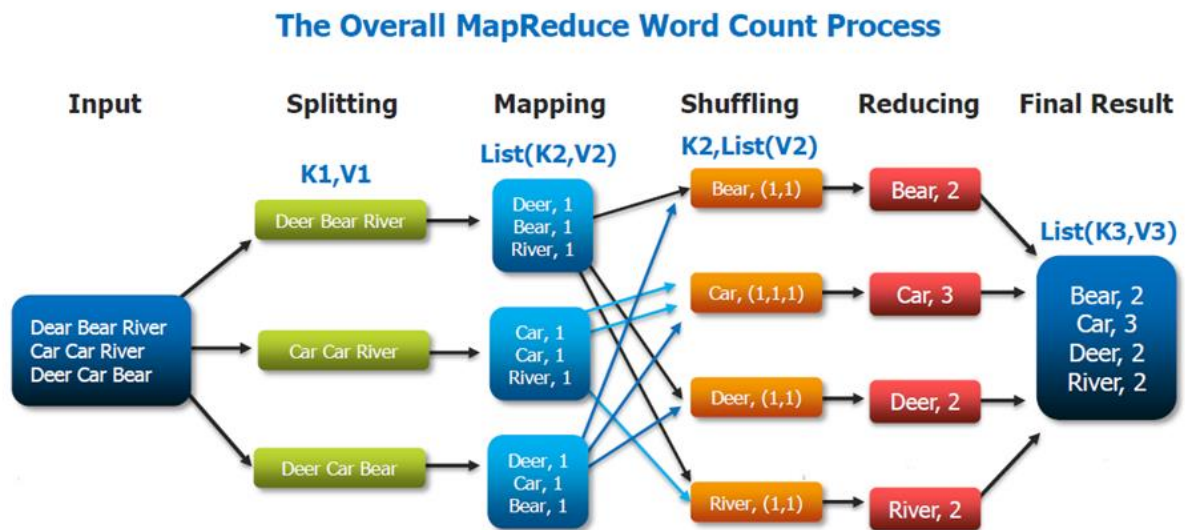


Figure 2: Basic working of MapReduce through simple Word Count example

We can easily understand the working of MapReduce by using word count example. Figure 1. represents the MapReduce solution to the word count problem.

## 3. Features of MapReduce

MapReduce framework has several advantageous features, some of which are as follows:

- Automatic Parallelization & Distribution
- Fault Tolerance
- Locality Optimization
- Backup tasks

## 4. Use of MapReduce in our task

Our task was to optimise the task of finding links between websites. In a sequential programming technique, we would need to iterate over each website and find all of the

incoming as well as outgoing links. Then we would need to find all the common links and create a set of combinations that have the same source or destination. This is easily a very time taking process if done sequentially.

Further complexity arises when we define depth of the web. That is how far the web crawler will search for an interconnection. When the depth of search is increased to 10, it gets computationally very expensive.

Thus, using MapReduce can help in sharing the load and reducing the time complexity of the overall process. The next chapter is dedicated to our implementation of MapReduce applied to the task.

## Chapter 2: Implementation

### 1. Introduction

We have used Java as the programming language to implement our project. Map-Reduce framework from the Apache Hadoop library has been used to parallelize the task of finding the interconnections between different web sites.

### 2. Input

The application accepts a list of URLs as input, i.e., one or multiple text file(s) where each line represents a single URL. The program has three parameters, the input folder, the output folder, and the maximum depth limit for spidering the websites. This limit is an optional parameter and 1 by default, meaning that for each base URL, one level of links is followed. If one increases the parameter the case as described in the previous chapter arises, but Hadoop is able to handle it well.

### 3. Mapper

The mapper receives tuples of the type:

*<Integer (line number), Text (the line contents, i.e., the URL)>*

Each URL is loaded recursively, similar to how a bot from a search engine would do it. When reading a web page (using Java's URLConnection object), we look for links/usages of other resources, such as images, CSS, JavaScript, links, frames, and iframes. The latter three are traced recursively up to a given maximum depth (the third, optional, command line parameter of the program). For each linked resource, the mapper will output a tuple as follows:

*<Text (with the URL to the resource), Text (with base URL from the input)>*

## 4. Reducer

The reducer receives elements of the form:

*<Text (with the URL to the resource), Iterable<Text> (all base URLs linking to the resource)>*

The reduction step is very simple: All input pairs which just contain a single element in the values, i.e., which stand for resources only linked from a single one of the originally specified URLs, are discarded. This leaves only resources linked from multiple input URLs. These resources are thus shared amongst different sites. The reducer returns tuples as follows:

*<Text (with the URL to the shared resource), List<Text> (with all originally specified sites linking to the shared resource)>*

This will help us to understand how different websites are connected and which resources are vital, i.e., which dependencies are needed by several sites to work properly.

## Results

Following are the screenshots at various timepoints during the execution of the application on Ubuntu.

```
17/11/07 22:27:11 INFO namenode.FSDirectory: ACLs enabled? false
17/11/07 22:27:11 INFO namenode.FSDirectory: XAttrs enabled? true
17/11/07 22:27:11 INFO namenode.FSDirectory: Maximum size of an xattr: 16384
17/11/07 22:27:11 INFO namenode.NameNode: Caching file names occurring more than 10 times
17/11/07 22:27:11 INFO util.GSet: Computing capacity for map cachedBlocks
17/11/07 22:27:11 INFO util.GSet: VM type = 64-bit
17/11/07 22:27:11 INFO util.GSet: 0.25% max memory 889 MB = 2.2 MB
17/11/07 22:27:11 INFO util.GSet: capacity = 2^18 = 262144 entries
17/11/07 22:27:11 INFO namenode.FSNamesystem: dfs.namenode.safemode.threshold-pct = 0.9990000128746033
17/11/07 22:27:11 INFO namenode.FSNamesystem: dfs.namenode.safemode.min.datanodes = 0
17/11/07 22:27:11 INFO namenode.FSNamesystem: dfs.namenode.safemode.extension = 30000
17/11/07 22:27:11 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
17/11/07 22:27:11 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
17/11/07 22:27:11 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
17/11/07 22:27:11 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
17/11/07 22:27:11 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 millis
17/11/07 22:27:11 INFO util.GSet: Computing capacity for map NameNodeRetryCache
17/11/07 22:27:11 INFO util.GSet: VM type = 64-bit
17/11/07 22:27:11 INFO util.GSet: 0.02999999932947746% max memory 889 MB = 273.1 KB
17/11/07 22:27:11 INFO util.GSet: capacity = 2^15 = 32768 entries
17/11/07 22:27:11 INFO namenode.FSImage: Allocated new BlockPoolId: BP-672635260-127.0.1.1-1510073831475
17/11/07 22:27:11 INFO common.Storage: Storage directory /tmp/hadoop-anshul/dfs/name has been successfully formatted.
17/11/07 22:27:11 INFO namenode.FSImageFormatProtobuf: Saving image file /tmp/hadoop-anshul/dfs/name/current/fsimage.ckpt_00000000000000000000 u
sing no compression
17/11/07 22:27:11 INFO namenode.FSImageFormatProtobuf: Image file /tmp/hadoop-anshul/dfs/name/current/fsimage.ckpt_00000000000000000000 of size
323 bytes saved in 0 seconds.
17/11/07 22:27:11 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
17/11/07 22:27:12 INFO util.ExitUtil: Exiting with status 0
17/11/07 22:27:12 INFO namenode.NameNode: SHUTDOWN MSG:
/*****
SHUTDOWN MSG: Shutting down NameNode at anshul-X555LJ/127.0.1.1
*****/
anshul@anshul-X555LJ:~/hadoop/hadoop-2.7.4$ sbin/start-dfs.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /home/anshul/hadoop/hadoop-2.7.4/logs/hadoop-anshul-namenode-anshul-X555LJ.out
localhost: starting datanode, logging to /home/anshul/hadoop/hadoop-2.7.4/logs/hadoop-anshul-datanode-anshul-X555LJ.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /home/anshul/hadoop/hadoop-2.7.4/logs/hadoop-anshul-secondarynamenode-anshul-X555LJ.out
anshul@anshul-X555LJ:~/hadoop/hadoop-2.7.4$ bin/hdfs dfs -mkdir /user
anshul@anshul-X555LJ:~/hadoop/hadoop-2.7.4$ bin/hdfs dfs -put /home/anshul/hadoop/distributedComputingExamples/hadoop/webFinder/input /input
anshul@anshul-X555LJ:~/hadoop/hadoop-2.7.4$ bin/hadoop jar /home/anshul/hadoop/distributedComputingExamples/hadoop/webFinder/target/webFinder-f
ull.jar /input /output
```

Figure 3: Starting Hadoop Services



```

anshul@anshul-X555LJ: ~/hadoop/hadoop-2.7.4
17/11/07 22:31:27 INFO mapreduce.Job: Job job_local1454762259_0001 completed successfully
17/11/07 22:31:27 INFO mapreduce.Job: Counters: 35
File System Counters
    FILE: Number of bytes read=78859192
    FILE: Number of bytes written=80094254
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=190
    HDFS: Number of bytes written=18568
    HDFS: Number of read operations=13
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=4
Map-Reduce Framework
    Map input records=3
    Map output records=490
    Map output bytes=37227
    Map output materialized bytes=38214
    Input split bytes=104
    Combine input records=0
    Combine output records=0
    Reduce input groups=220
    Reduce shuffle bytes=38214
    Reduce input records=490
    Reduce output records=135
    Spilled Records=980
    Shuffled Maps=1
    Failed Shuffles=0
    Merged Map outputs=1
    GC time elapsed (ms)=135
    Total committed heap usage (bytes)=688914432
Shuffle Errors
    BAD_ID=0
    CONNECTION=0
    IO_ERROR=0
    WRONG_LENGTH=0
    WRONG_MAP=0
    WRONG_REDUCE=0
File Input Format Counters
    Bytes Read=95
File Output Format Counters
    Bytes Written=18568
anshul@anshul-X555LJ:~/hadoop/hadoop-2.7.4$

```

Figure 4: Information of our single node cluster

```

anshul@anshul-X555LJ: ~/hadoop/hadoop-2.7.4
17/11/07 22:29:33 INFO webFinder.WebFinderMapper: Now tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem3.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:34 INFO mapred.LocalJobRunner: map > map
17/11/07 22:29:37 INFO webFinder.WebFinderMapper: Finished tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem3.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:37 INFO webFinder.WebFinderMapper: Now tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem4.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:37 INFO mapred.LocalJobRunner: map > map
17/11/07 22:29:40 INFO mapred.LocalJobRunner: map > map
17/11/07 22:29:42 INFO webFinder.WebFinderMapper: Finished tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem4.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:42 INFO webFinder.WebFinderMapper: Now tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem5.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:46 INFO webFinder.WebFinderMapper: Finished tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem5.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:46 INFO webFinder.WebFinderMapper: Now tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem6.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:46 INFO mapred.LocalJobRunner: map > map
17/11/07 22:29:49 INFO mapred.LocalJobRunner: map > map
17/11/07 22:29:50 INFO webFinder.WebFinderMapper: Finished tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem6.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:50 INFO webFinder.WebFinderMapper: Now tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem7.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:54 INFO webFinder.WebFinderMapper: Finished tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem7.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:54 INFO webFinder.WebFinderMapper: Now tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem8.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:55 INFO mapred.LocalJobRunner: map > map
17/11/07 22:29:58 INFO mapred.LocalJobRunner: map > map
17/11/07 22:29:58 INFO webFinder.WebFinderMapper: Finished tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BT/BIOTECH_Sem8.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:29:58 INFO webFinder.WebFinderMapper: Now tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BM/BM3rd.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:30:00 INFO webFinder.WebFinderMapper: Finished tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BM/BM3rd.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:30:00 INFO webFinder.WebFinderMapper: Now tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BM/BM4th.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:30:02 INFO webFinder.WebFinderMapper: Finished tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BM/BM4th.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:30:02 INFO webFinder.WebFinderMapper: Now tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BM/BM5th.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.
17/11/07 22:30:04 INFO webFinder.WebFinderMapper: Finished tracing URL 'http://www.nitrr.ac.in/downloads/syl_new/BM/BM5th.pdf' for base url 'http://www.nitrr.ac.in/aboutcse.php'.

```

Figure 5: Screenshot during the execution of the code

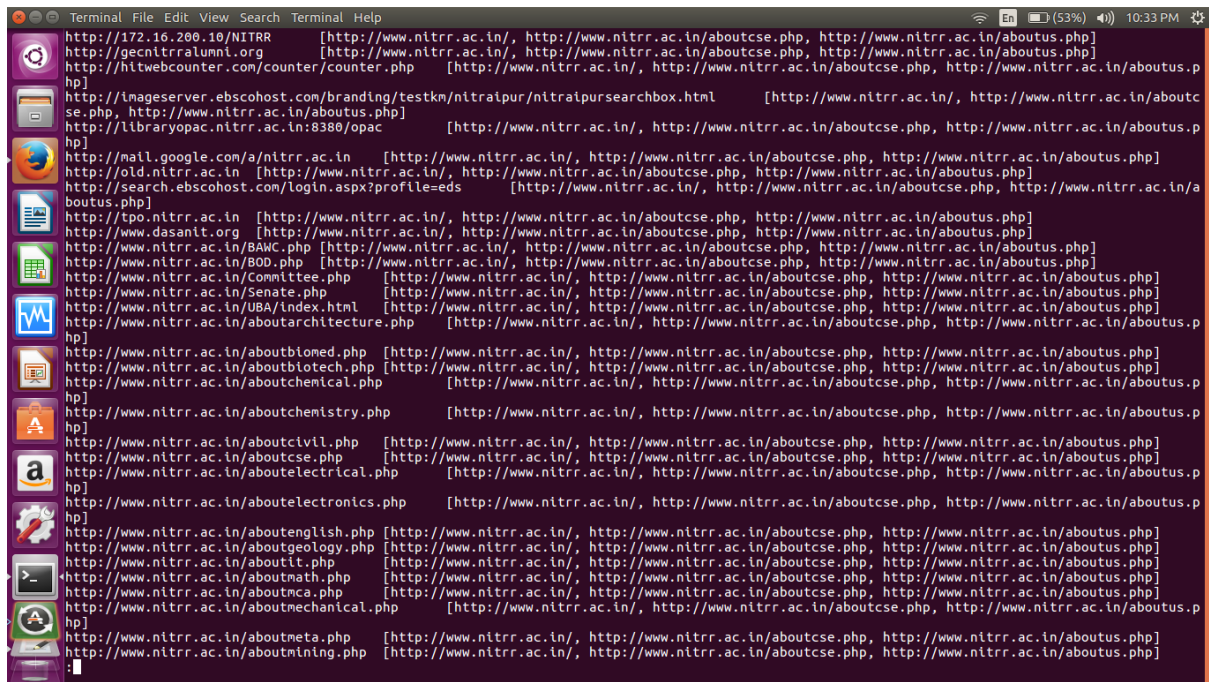


Figure 6: Final Output which shows the interlinking of websites

## Conclusion

We have implemented the defined project and have learnt the underlying concepts of Hadoop and its application to distributed computing. We hope to continuously add to this knowledge in the future and tackle tasks that are more difficult to divide and parallelize.

## References

- [1] Eugenia Sawa. Retrieved from <https://blogs.microsoft.com/jobs/story-library/5-reasons-why-distributed-systems-are-the-future/>
- [2] benjismith, Shane. 2009. Retrieved from <https://stackoverflow.com/questions/522438/why-does-moores-law-necessitate-parallel-computing>
- [3] Chu Yan Shing, Wu Bing Chuan. 2011. *Cloud Computing technologies and applications*. Term Project Report, Department of Computer Science and Engineering. The Chinese University of Hong Kong, Hong Kong, China.
- [4] StackOverflow. 2017. Retrieved from <https://stackoverflow.com/>
- [5] Github. 2017. Retrieved from <https://github.com/> [for various project parts]