Project Report

on

Air Pollution Analysis by City (2015-2020)

Submitted by

Srijan Sood

24MCC20018

Under the guidance of

Mr. Rishabh Tomar

in partial fulfilment for the award of the degree of

MASTER OF COMPUTER APPLICATIONS CLOUD COMPUTING & DEVOPS

Chandigarh University

April 2025

Certificate

This is to certify that **Srijan Sood**, a student of **Master of Computer Applications (MCA) – Cloud Computing and DevOps**, has successfully completed the **Minor Project** titled **"Air Pollution Analysis by City (2015-2020)"** under the esteemed guidance of **Mr. Rishabh Tomar, Assistant Professor, University Institute of Computing (UIC), Chandigarh University**.

This project was undertaken as a part of the academic curriculum and is submitted in **partial fulfilment of the requirements** for the MCA program. The work presented in this project is a result of **independent research, diligent effort, and dedication**, demonstrating the student's ability to apply theoretical knowledge to practical problem-solving.

The project successfully implements **Big Data processing using Hadoop and MapReduce**, demonstrating an efficient approach to analysing word frequency in large-scale textual data. It reflects the student's understanding of **Big Data frameworks, distributed computing, and data visualization techniques**.

I hereby confirm that this project is an **original work** carried out by the student and has **not been submitted elsewhere** for the award of any other degree, diploma, or certification.

Project Guide:
Mr. Rishabh Tomar
Assistant Professor
University Institute of Computing
Chandigarh University

Acknowledgement

I would like to express my sincere gratitude to **Chandigarh University** and the **University Institute of Computing (UIC)** for providing me with the opportunity to undertake this project, **" Air Pollution Analysis by Cities(2015-2020)."**

I extend my heartfelt appreciation to my esteemed mentor, **Mr. Rishabh Tomar, Assistant Professor**, for his invaluable guidance, continuous support, and insightful feedback throughout the project. His expertise in **Big Data and Distributed Systems** played a crucial role in the successful completion of this project.

I am also grateful to my friends and peers for their encouragement and discussions, which helped refine my approach. Lastly, I thank my family for their unwavering support and motivation during this research.

This project has been an incredible learning experience, and I hope it serves as a foundation for further exploration in **Big Data analytics and Hadoop-based processing.**

Srijan Sood
MCA – Cloud Computing and DevOps
Chandigarh University

Contents

Abstract

Air pollution has become a major global concern due to its adverse effects on human health and the environment. The Air Quality Index (AQI) is a widely used measure to assess pollution levels across different cities and regions. This project aims to analyse air pollution data by computing the average AQI for various cities using Hadoop and MapReduce, which enables efficient processing of large datasets.

The project follows a Big Data approach, where air quality data is stored in the Hadoop Distributed File System (HDFS) and processed using the MapReduce programming model. The Mapper function extracts city names and AQI values from the dataset, while the Reducer function computes the average AQI for each city. The results are then stored in an output file and further analysed using Python and Matplotlib for graphical visualization.

The dataset used in this project contains real-world air quality data, including multiple attributes such as particulate matter concentrations (PM2.5, PM10), nitrogen dioxide (NO_2), sulphur dioxide (SO_2), and other pollutants. However, this analysis specifically focuses on AQI values to provide a clearer understanding of pollution levels.

One of the key advantages of using Hadoop and MapReduce is the ability to handle and process large-scale environmental datasets in a distributed computing environment. Traditional data processing tools often struggle with large datasets, whereas Hadoop efficiently distributes the workload across multiple nodes, ensuring faster execution and scalability.

The results of this project provide valuable insights into air pollution trends across different cities, helping researchers and policymakers make data-driven decisions. The final output, visualized through bar charts, allows for an intuitive comparison of AQI values, making it easier to identify highly polluted areas.

This project demonstrates the practical application of Big Data technologies in environmental monitoring and showcases the potential of Hadoop for handling large-scale data analytics. Future enhancements could include real-time AQI monitoring and deeper analysis of pollution sources, contributing further to the field of environmental data science.

1. Introduction

Air pollution is one of the most pressing global challenges, affecting millions of people worldwide. It is caused by a variety of factors, including vehicle emissions, industrial activities, and natural phenomena such as wildfires and dust storms. The **Air Quality Index (AQI)** is a widely used measure to assess and communicate air pollution levels to the public. High AQI values indicate poor air quality, which can lead to severe health issues such as respiratory diseases, cardiovascular problems, and reduced life expectancy.

With the rapid urbanization and industrialization of cities, monitoring air quality has become more critical than ever. Traditional methods of air quality monitoring often struggle with handling large volumes of data collected from multiple sources. This is where **Big Data technologies** like **Hadoop and MapReduce** become essential. These technologies enable the efficient storage, processing, and analysis of massive datasets, making it possible to gain meaningful insights from large-scale environmental data.

This project focuses on analysing air pollution levels across various cities by computing the **average AQI** for each city using Hadoop's **MapReduce framework**. The project workflow involves three main steps:

1. **Data Collection and Storage**: The AQI dataset is stored in **Hadoop Distributed File System (HDFS)** for scalable and distributed data handling.
2. **Processing using MapReduce**: A **Mapper function** extracts AQI values for each city from the dataset, while a **Reducer function** computes the average AQI per city.
3. **Data Visualization**: The results are retrieved and plotted using **Python and Matplotlib** to provide an intuitive representation of pollution trends across different cities.

The key objective of this project is to showcase the effectiveness of **Hadoop and MapReduce** in handling large datasets while providing insights into air pollution patterns. The results can be useful for environmental researchers, policymakers, and urban planners to take informed actions to reduce pollution levels.

2. Tools and Technologies Used

This project utilizes various tools and technologies to handle data collection, processing, and visualization effectively. The key components include:

2.1 Big Data Frameworks

- **Apache Hadoop**

  Apache Hadoop is an open-source framework designed for handling large-scale data processing. It provides:

  HDFS (Hadoop Distributed File System): A fault-tolerant and scalable storage solution.

  MapReduce: A distributed computing model used for processing large datasets efficiently.

2.2 Programming Languages

- **Java:** Java is used to implement the MapReduce program that processes AQI data.

  It includes:

  Mapper Function to extract relevant data.

  Reducer Function to compute and aggregate AQI values.

  Driver Class to control execution.

- **Python:** Python is used for data processing and visualization after executing the MapReduce job. It helps in:

  Manipulating the processed AQI data.

  Creating **graphical representations** using Matplotlib.

2.3 Data Sources

- **Kaggle (2015-2020 AQI Dataset)**

  The dataset used in this project is sourced from Kaggle, containing air quality index (AQI) readings collected between 2015 and 2020 across multiple cities. It includes:

  City-wise air quality indicators such as PM2.5, PM10, NO2, SO2, and CO levels.

  Temporal variations in AQI values over different years.

3. Implementation Steps

The implementation of this project follows a structured approach, leveraging **Hadoop, MapReduce, and Python** to perform word frequency analysis. The steps include **data collection, storage, processing, and visualization**. Below is a detailed breakdown of the execution:

3.1 Setting Up the Environment

To begin the project, the necessary software and tools were installed and configured. This includes:

- Installing **Apache Hadoop** and setting up the **Hadoop Distributed File System (HDFS)** for data storage.
- Configuring the environment to execute **MapReduce jobs** for distributed data processing.
- Setting up a **Linux-based execution environment** to manage Hadoop operations.

Commands to initialize Hadoop:

```
hadoop-3.3.6/bin/hides namenode -format
export PDSH_RCMD_TYPE=ssh
start-all.sh
hdfs dfs -mkdir /input
hdfs dfs -put /home/hrushi/Downloads/city_day.csv /input
```

3.2 Fetching Data

The AQI dataset, covering air pollution metrics from **2015 to 2020**, was sourced from **Kaggle**. This dataset includes:

- City-wise AQI values for various pollutants such as **PM2.5, PM10, NO2, SO2, and CO**.
- Time-series data representing **daily and yearly air quality trends**.
- Additional metadata describing **geographical and seasonal variations** in air quality.

After obtaining the dataset, **data preprocessing** was performed to ensure consistency, including:

- **Handling missing values** by replacing them with average AQI values.
- **Standardizing data formats** for efficient processing.
- **Filtering relevant columns** required for analysis.

3.3 Storing Data in HDFS

Once preprocessed, the dataset was uploaded to the **Hadoop Distributed File System (HDFS)**. HDFS was chosen due to its:

- **Scalability:** Capable of storing large datasets efficiently.
- **Fault tolerance:** Ensures data integrity even in the event of node failures.
- **Distributed architecture:** Optimized for parallel processing with MapReduce.

Commands for uploading data:

```
hdfs dfs -put /home/hrushi/Downloads/city_day.csv /input
```

3.4 Implementing the MapReduce Job

To analyze AQI trends, a **MapReduce** program was developed using **Java**. The implementation follows a **three-phase** approach:

1. Mapping Phase

- The **Mapper function** reads AQI data and extracts relevant information such as **city name, date, and AQI values**.
- Each data record is processed as a key-value pair, where the **city name serves as the key**, and the **AQI value acts as the associated data**.

2. Shuffling & Sorting Phase

- The **Hadoop framework automatically groups** AQI values by city, sorting them in preparation for aggregation.

3. Reducing Phase

- The **Reducer function** calculates the **average AQI** for each city over the given time period.
- The results are written back to **HDFS** for further analysis and visualization.

Java code for MapReduce job (AQIAnalysis.java):

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.FloatWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import java.io.IOException;

public class AQIAnalysis {

    public static class AirPollutionMapper extends Mapper<LongWritable, Text, Text,
FloatWritable> {
        private Text city = new Text();
        private FloatWritable aqi = new FloatWritable();

        @Override
        public void map(LongWritable key, Text value, Context context) throws
IOException, InterruptedException {
            String[] fields = value.toString().split(",");
            if (fields.length >= 15) {
                try {
                    city.set(fields[0]);
```

```
                    if (fields[13] != null && !fields[13].isEmpty()) {
                        aqi.set(Float.parseFloat(fields[13]));
                        context.write(city, aqi);
                    }
                } catch (NumberFormatException e) {}
            }
        }
    }

    public static class AirPollutionReducer extends Reducer<Text, FloatWritable, Text,
FloatWritable> {
        private FloatWritable result = new FloatWritable();

        @Override
        public void reduce(Text key, Iterable<FloatWritable> values, Context context)
throws IOException, InterruptedException {
            float sum = 0;
            int count = 0;
            for (FloatWritable val : values) {
                sum += val.get();
                count++;
            }
            if (count > 0) {
                result.set(sum / count);
                context.write(key, result);
            }
        }
    }

    public static void main(String[] args) throws Exception {
        if (args.length != 2) {
            System.err.println("Usage: AQIAnalysis <input path> <output path>");
            System.exit(-1);
        }
```

```
        Configuration conf = new Configuration();
        Job job = Job.getInstance(conf, "Air Pollution AQI Analysis");
        job.setJarByClass(AQIAnalysis.class);

        job.setMapperClass(AirPollutionMapper.class);
        job.setReducerClass(AirPollutionReducer.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(FloatWritable.class);

        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```

Commands for compiling and running the MapReduce job:

```
hadoop com.sun.tools.javac.Main AQIAnalysis.java
jar cf aqi_analysis.jar AQIAnalysis*.class
hadoop jar aqi_analysis.jar AQIAnalysis /input /output
```

3.5 Retrieving Processed Results

After completing the **MapReduce** job, the processed results were retrieved from **HDFS**. These results contain:

- **City-wise AQI averages**, allowing comparison of air quality across different regions.
- **Temporal trends** indicating improvements or deterioration in air quality over time.

Command to retrieve results:

```
hadoop fs -getmerge /output/* air_quality_output.csv
```

3.6 Analyzing and Visualizing Results

With the processed data, **Python-based visualization** techniques were applied to generate meaningful insights. The key steps include:

- **Loading the processed AQI data** into Python using the Pandas library.
- **Creating bar charts and line graphs** using Matplotlib to visualize AQI trends.
- **Comparing air pollution levels** across different cities and time periods.

Python code for visualization (plot_aqi.py):

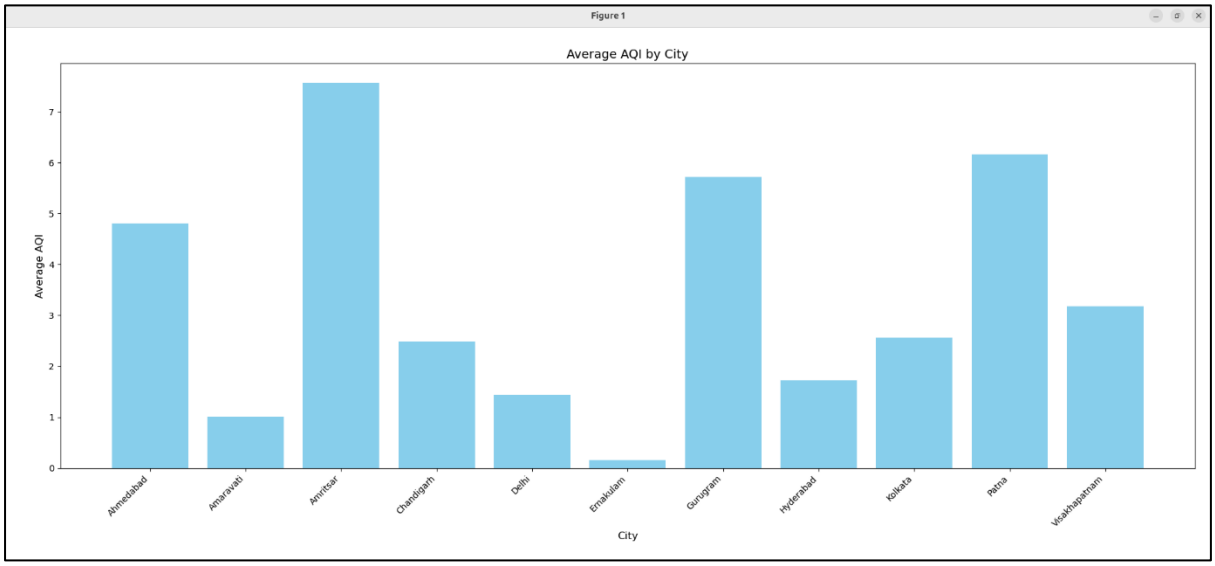
```
import matplotlib.pyplot as plt
import pandas as pd

file_path = "air_quality_output.csv"

data = pd.read_csv(file_path, sep="\t", names=["City", "Average_AQI"])
data["Average_AQI"] = data["Average_AQI"].astype(float)

plt.figure(figsize=(12, 6))
plt.bar(data["City"], data["Average_AQI"], color="skyblue")

plt.xlabel("City", fontsize=12)
plt.ylabel("Average AQI", fontsize=12)
plt.title("Average AQI by City", fontsize=14)
plt.xticks(rotation=45, ha="right")

plt.tight_layout()
plt.show()
```

These visualizations help in identifying **pollution hotspots** and understanding long-term air quality trends.

4. Result and Analysis

```
hrushi@Ubuntu:~$ hadoop jar aqi_analysis.jar AQIAnalysis /input /output
2025-03-31 18:07:28,314 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
2025-03-31 18:07:29,314 INFO client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager at /0.0.0.0:8032
2025-03-31 18:07:29,920 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to rem
edy this.
2025-03-31 18:07:29,956 INFO mapreduce.JobResourceUploader: Disabling Erasure Coding for path: /tmp/hadoop-yarn/staging/hrushi/.staging/job_1743440428942_0016
2025-03-31 18:07:30,353 INFO input.FileInputFormat: Total input files to process : 1
2025-03-31 18:07:30,428 INFO mapreduce.JobSubmitter: number of splits:1
2025-03-31 18:07:30,652 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1743440428942_0016
2025-03-31 18:07:30,652 INFO mapreduce.JobSubmitter: Executing with tokens: []
2025-03-31 18:07:30,895 INFO conf.Configuration: resource-types.xml not found
2025-03-31 18:07:30,895 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
2025-03-31 18:07:30,987 INFO impl.YarnClientImpl: Submitted application application_1743440428942_0016
2025-03-31 18:07:31,066 INFO mapreduce.Job: The url to track the job: http://Ubuntu:8088/proxy/application_1743440428942_0016/
2025-03-31 18:07:31,067 INFO mapreduce.Job: Running job: job_1743440428942_0016
2025-03-31 18:07:39,242 INFO mapreduce.Job: Job job_1743440428942_0016 running in uber mode : false
2025-03-31 18:07:39,244 INFO mapreduce.Job:  map 0% reduce 0%
2025-03-31 18:07:46,474 INFO mapreduce.Job:  map 100% reduce 0%
2025-03-31 18:07:52,519 INFO mapreduce.Job:  map 100% reduce 100%
2025-03-31 18:07:54,549 INFO mapreduce.Job: Job job_1743440428942_0016 completed successfully
2025-03-31 18:07:54,681 INFO mapreduce.Job: Counters: 54
        File System Counters
                FILE: Number of bytes read=144070
                FILE: Number of bytes written=840547
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=2574161
                HDFS: Number of bytes written=208
                HDFS: Number of read operations=8
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
                HDFS: Number of bytes read erasure-coded=0
        Job Counters
                Launched map tasks=1
                Launched reduce tasks=1
                Data-local map tasks=1
                Total time spent by all maps in occupied slots (ms)=4248
                Total time spent by all reduces in occupied slots (ms)=3940
                Total time spent by all map tasks (ms)=4248
                Total time spent by all reduce tasks (ms)=3940
                Total vcore-milliseconds taken by all map tasks=4248
```

```
hrushi@Ubuntu:~$ hadoop fs -cat /output/*
2025-03-31 18:09:58,952 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Ahmedabad       4.806593
Amaravati       1.007455
Amritsar        7.5658884
Chandigarh      2.4896328
Delhi   1.4383389
Ernakulam       0.1540132
Gurugram        5.7224383
Hyderabad       1.7249397
Kolkata 2.5662947
Patna   6.16372
Visakhapatnam   3.181565
hrushi@Ubuntu:~$
```



The bar chart above represents the **Average AQI of Cities based on the Kaggle Air Pollution Data (2015-2020)** based on the processed dataset.

5. Conclusion

The AQI analysis project utilized a structured approach combining Hadoop, MapReduce, and Python to efficiently process and visualize large-scale air quality data. The structured implementation enabled a detailed exploration of air pollution trends, offering valuable insights for policymakers and environmental researchers. Below is a structured summary of the key outcomes:

1. **Data Processing with Hadoop and MapReduce**:
    - The project leveraged Hadoop's scalability and fault tolerance to store and process large datasets, ensuring that air quality data from multiple cities over a five-year period was handled efficiently.
2. **City-wise AQI Analysis**:
    - The Mapper extracted relevant AQI data for each city, while the Reducer calculated the average AQI values across the dataset, allowing for a comprehensive city-wise comparison.
3. **Visualization with Python**:
    - The processed AQI data was visualized using Python's Pandas and Matplotlib libraries, creating clear and informative bar charts that depict AQI levels across cities.
4. **Scalability and Efficiency**:
    - The use of Hadoop's Distributed File System (HDFS) ensured that the data could be efficiently stored and accessed across a distributed network, making the solution scalable for handling much larger datasets.
5. **Impact and Application**:
    - This project serves as a valuable example of how big data technologies can be applied to environmental data analysis. The insights gained from the AQI analysis can inform decision-making processes in environmental management and urban planning.

In conclusion, the project highlights the power of Hadoop and Python in handling large datasets and producing meaningful, data-driven insights. By leveraging these technologies, it provides a scalable and efficient solution for air quality analysis, which can be further expanded to improve environmental monitoring and policy planning.

6. References

i. **Apache Hadoop Documentation**
Apache Hadoop. (n.d.). *Apache Hadoop*. Retrieved from https://hadoop.apache.org/

ii. **Kaggle - Air Quality Dataset**
Kaggle. (2020). *Air Quality Dataset*. Retrieved from https://www.kaggle.com/

iii. **Hadoop: The Definitive Guide**
White, T. (2015). *Hadoop: The Definitive Guide (4th ed.)*. O'Reilly Media.

iv. **Matplotlib Documentation**
Matplotlib. (n.d.). *Matplotlib*. Retrieved from https://matplotlib.org/

v. **Pandas Documentation**
The Pandas Development Team. (n.d.). *Pandas Documentation*. Retrieved from https://pandas.pydata.org/

vi. **MapReduce: Simplified Data Processing on Large Clusters**
Dean, J., & Ghemawat, S. (2004). *MapReduce: Simplified Data Processing on Large Clusters*. Communications of the ACM, 51(1), 107-113.

vii. **Central Pollution Control Board (CPCB) - Air Quality Index (AQI)**
Central Pollution Control Board, Ministry of Environment, Forest and Climate Change, Government of India. (n.d.). *Air Quality Index (AQI) - CPCB*. Retrieved from https://cpcb.nic.in/

7. Plagiarism Report:

SmallSEOTools

Plagiarism Scan Report By SmallSEOTools

Report Generated on: Mar 31,2025

0% Plagiarized Content

0% Exact Plagiarized

0% Partial Plagiarized

100% Unique Content

Total Words: 299 Total Characters: 2033 Plagiarized Sentences: 0 Unique Sentences: 15 (100%)

Content Checked for Plagiarism

Air pollution is one of the most pressing global challenges, affecting millions of people worldwide. It is caused by a variety of factors, including vehicle emissions, industrial activities, and natural phenomena such as wildfires and dust storms. The Air Quality Index (AQI) is a widely used measure to assess and communicate air pollution levels to the public. High AQI values indicate poor air quality, which can lead to severe health issues such as respiratory diseases, cardiovascular problems, and reduced life expectancy.

With the rapid urbanization and industrialization of cities, monitoring air quality has become more critical than ever. Traditional methods of air quality monitoring often struggle with handling large volumes of data collected from multiple sources. This is where Big Data technologies like Hadoop and MapReduce become essential. These technologies enable the efficient storage, processing, and analysis of massive datasets, making it possible to gain meaningful insights from large-scale environmental data.

This project focuses on analyzing air pollution levels across various cities by computing the average AQI for each city using Hadoop's MapReduce framework. The project workflow involves three main steps:

1. Data Collection and Storage: The AQI dataset is stored in Hadoop Distributed File System (HDFS) for scalable and distributed data handling.
2. Processing using MapReduce: A Mapper function extracts AQI values for each city from the dataset, while a Reducer function computes the average AQI per city.
3. Data Visualization: The results are retrieved and plotted using Python and Matplotlib to provide an intuitive representation of pollution trends across different cities.

The key objective of this project is to showcase the effectiveness of Hadoop and MapReduce in handling large datasets while providing insights into air pollution patterns. The results can be useful for environmental researchers, policymakers, and urban planners to take informed actions to reduce pollution levels.

No Plagiarism Found