

Towards data analysis for Weather Cloud Computing

Victor Chang

IBSS, Xi'an Jiaotong Liverpool University, Suzhou, China
Victor.Chang@xjtlu.edu.cn

Abstract— This paper demonstrates an innovative data analysis for weather using Cloud Computing, integrating both system and application Data Science services to investigate extreme weather events. Identifying five existing projects with ongoing challenges, our aim is to process, analyze and visualize collected data, study implications and report meaningful findings. We demonstrate the use of Cloud Computing technologies, MapReduce and optimization techniques to simulate temperature distributions and analyze weather data. Two major cases are presented. The first case is focused on forecasting temperatures based on [studying trends from](#) the historical data of Sydney, Singapore and London to compare the historical and forecasted temperatures. The second case is to use five-step MapReduce for numerical data analysis and eight-step process for visualization, which is used to analyze and visualize temperature distributions in the United States, before, during and after the time of experiencing polar vortex, as well as in the United Kingdom during and after the flood. Optimization was used in experiments involved up to 100 nodes between Cloud and non-Cloud and compared performance with and without optimization. There was an improvement in performance between 20% and 30% under 60 nodes in Cloud. Results, discussion and comparison were presented. We justify our research contributions and explain thoroughly in the paper how the three goals can be met: (1) forecasting temperatures of three cities based on [evaluating the trends from](#) the historical data; (2) using five-step MapReduce to achieve shorter execution time on Cloud and (3) using eight-step MapReduce with optimization to achieve data visualization for temperature distributions on the US and UK maps.

Index Terms— System and application for weather computation; temperature forecasting and distribution; MapReduce; weather data visualization; polar vortex; weather Data Science.

1. Introduction

Data Science is an interdisciplinary area that allows experts in different domains to study and work together [1-2]. Outputs of all different types of work are in the form of the data, which comes in different types of formats. This allows experts in different disciplines to investigate the meaning of data which can be from the same or different disciplines. There are five common characteristics for Data Science: volume, velocity, variety, veracity and value. Volume refers to the size and quantity of the data that have been processed and stored. Velocity is the rate in which the data has been created, processed and analyzed. Variety refers to the different types and formats of data available and ready for further analysis. Veracity is the accuracy and validity of the data analysis. Value is the added value offered by the Data Science, such as allowing organizations to stay competitive and efficient [1-3]. It has become apparently obvious that the processing, analysis and presentation of data outputs will be essential to a growing number of sectors involved.

In order to analyze data successfully, the infrastructure, platform and software as a service approach should be adopted. With regard to infrastructure and platform, system dependency for Data Science allows the developers to design and build systems and understand the relationship between different clusters, jobs, nodes and virtual machines with the aim to work out the best possible recommendations for different scenarios [3-4]. For software as a service, software dependency for Data Science allows the developers to design and develop software and understand the relationship between libraries, algorithms, APIs, commands, outputs and user interfaces [5-7]. For successful development, the dependency on both system and software are essential to ensure that services can meet expected demands and deliver required tasks. The MapReduce framework can be used as an example. A function called `map()` can be written that can send jobs to different nodes and return results back to the system, whereby the intelligent software can use `reduce()` function to map to the related groups together and produce outputs based on the collaboration of `map()` and `reduce()` functions. The collaborative system and software dependency can achieve development of different types of services known as "Everything as a Service" (EaaS). The concept of EaaS is based on the development and integration of Infrastructure, Platform and Software as a Service to ensure the joint delivery of all the system and software requirements. For example, Business Intelligence as a Service

- Dr. Victor Chang is with International Business School Suzhou, Xi'an Jiaotong Liverpool University, Suzhou, China. He is affiliated with Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK. E-mail: ic.victor.chang@gmail.com.

Please note that all acknowledgments should be placed at the end of the paper, before the bibliography (note that corresponding authorship is not noted in affiliation box, but in acknowledgment section).

(BlaaS) is developed based on the pipeline-method architecture that allows the output of the first service becomes the input of the second service [8].

Weather as a service is a good example that need strong collaboration between system and software dependency for Data Science. Often such services require supercomputers, intelligent algorithms, data services, visualization technique to jointly deliver. Weather computing will require an excellent dependency between systems, between different parts of the software and between system and software to ensure that results are fast, accurate, responsive and interactive [9-10]. Vigorous scientific processes aided by the use of advanced technology help meteorologists to make weather predictions. Weather information is particularly useful for the general public to make relevant plans, such as going for trips in the sunny and bright weathers and avoiding trips in snowy and flood conditions. In the event of extreme weathers, this information is vital to the general public. For example, the UK experienced the wettest winter flood in 250 years between December 2013 and February 2014 [11]. Thousands of the residents could evacuate to suitable places in advance to avoid destructive impacts by flood, which damaged houses, towns and buildings in various parts of the UK. The US experienced one of the coldest winters due to the impact of the polar vortex in the same period as the UK, causing several states to experience sub-temperatures below -20 C and -30 C. In the similar period, Southern China experienced one of the warmest winters with the mean temperature of 21 C [12]. Some scientists suggest that the extreme weather conditions will become a norm rather than a possibility of less than 1% due to the global warming and intense human activities [13-15]. What are the impacts of the unpredictable weathers? Unpredictable weathers have become more common in the past few years in the UK. These included the driest summer in 2006 and 2013, big freeze in the UK in 2009 and 2010 winters and the wettest summer in 2011. Some months have "abnormal" temperatures such as the warmest April in 2013 (average of 16.7 C) and the potentially the

2 The background theories

This section describes the background theory associated with weather science. Related work is described as follows. First, Campbell and Diebold present their weather forecasting model [18]. They define the term "weather derivatives" and apply the concept to financial derivatives. This includes the use of volatility to illustrate the concept of time-series waves, which corresponds to the daily average temperature between 1996 and 2001 in four cities. Temperatures can be modeled as a normal distribution curve to show the likelihood of the temperatures. Temperature forecasting can adopt financial forecasting method to estimate temperatures in Atlanta, Chicago, Las Vegas and Philadelphia. Results confirm that the most of expected temperatures is within 95% confidence interval of the actual temperatures. Second, Plale et al. [19] explain how two major

second coolest summer in 2011 (13.6 C).

The first group of scientists using computational weather forecasting was John von Neumann and his colleagues, who undertook the first experimental weather forecast on the ENIAC computer in the late 1940s [10]. Within a decade of their work, numerical models became the foundations of weather science and also a discipline in computer science. According to [14, 16], each year the US had an economic loss averaging more than US\$13 billions due to the extreme weathers. This estimated figure is likely to be up due to the increased frequency of extreme weather events. The improved ways of conducting weather modeling and forecasting are essential to the development of weather science. The e-Science community has demonstrated weather applications. However, thousands of CPUs and expensive infrastructure have to be deployed [17]. The use of Cloud Computing can reduce costs and the scale of deployment while being able to compute weather simulations. We will illustrate an example how to achieve performance and visualization to analyze weather data and conduct performance evaluation. Pioneering methods such as Cloud Computing, Big Data Analytics and dependency between Cloud and Big Data should be investigated to make weather science affordable, accessible and technically viable.

Our paper presents a case on how to use Cloud Computing to process the data and Big Data Analytics to present the results based on the integrated dependency approach. The breakdown of our paper is as follows. Section 2 presents the related models for weather forecasting and the background theories supporting these models. Section 3 illustrates the architecture, system design and deployment to perform weather computing, Section 4 shows the results of weather computing with three case studies and compare performance between Cloud and non-Cloud. Section 5 demonstrates the data visualization dealing with the extreme weathers in the US and UK. Section 6 compares our work with similar approaches and summarizes our contributions. Section 7 presents Conclusion.

weather forecasting systems, CASA (Collaborative Adaptive Sensing of the Atmosphere) and LEAD (Linked Environment for Atmospheric Discovery) can interact with each other's data in real time. They present the system architecture between each system and core technologies. They explain how LEAD can use workflow applications to derive their forecast, even though they do not show technical details. They state the use of XML, Meteorological command and control and Blackboard can achieve interactions.

Third, Li [20] describe the pipeline method that utilizes the e-Science principles and use Hadoop algorithm to process applications and data in their public cloud. This is a pioneering approach for Cloud Computing with results supported their method. However, their method has not been applied to other e-Science related specialization such as Weather Science. Fourth, Droege-meier et al. [12] demonstrate the use of Service-Oriented

Grid Computing to enable dynamic weather computation. The architecture they use is a LEAD system. They demonstrate the case of radar reflectivity with several examples about Arkansas. These include the precipitation intensity at the eleven, nine and five hours of forecast. They include the precipitation forecast on January 29, 1999 and March 29, 2000. They also explain their workflows approach towards the ingesting and analysis of their data. Fifth, Demirkan and Delen [21] use their service-oriented decision support system to analyze data, including geospatial data, with the analytics and big data in the cloud approach. They have explained the architecture and their high-level method, without revealing much of their implementation process and performance evaluation. Last, Gao et al [22] explain more detailed geospatial data approach by using Hadoop to process big geo-data. They explain their architecture and two proposed algorithms, which are their main contributions to analyze big geo-data. They present all datapoints and map them on the USA map to show their outputs. They have only done two performance tests.

However, these five existing projects demonstrate current challenges to be resolved. First, Campbell and Diebold [18] do not explain enough theories about how the financial derivatives can be applied to temperature forecasting. There is a need to consolidate. Second, the work in [19] is not an interaction but information exchange. There is a need to use any services to process data and analyze the results. Third, both CASA and LEAD are expensive and are not systems that can be designed and implemented realistically, since funding may restrict the scale of design and implementation that not every scientist can afford using it. Fourth, Demirkan and Delen's systems are not robust enough without performance evaluation [21]. Although Gao et al [22] have designed and implemented their algorithms to process big geo-data, their definition of big data is quantity of data and not the size of each of thousands of data.

This motivates us to build an inexpensive system that can perform weather simulation and forecasting. The aim is to process, analyze and visualize collected data and to study any implications, such as forecasting of 2014 weathers based on 2012 data, study the impacts of extreme weathers in 2014 and identify whether there is a satisfactory performance evaluation. We have collected data between Year 2012 and 2014, with focused to cities like Sydney, Singapore and London. We have studied extreme weathers occurred in the USA and UK in 2014, with data collected in specific dates. We propose an improved architecture based on the pipeline method [20], which can be designed and implemented based on the public cloud to store our code and private cloud for architecture to meet the requirements of affordable cost to achieve data, processing analysis and visualization enabled by our proposed techniques discussed between Section 3 and 5. We explain the mathematical formula used and bridge the gap between financial derivatives to weather derivatives. We use weather data from the National Weather Service [23], the Met

Office [24] and BBC Weather [25] to demonstrate the examples in the temperatures in the US and UK in response to the extreme weathers experienced between December 2013 and March 2014.

2.1 Background for forecasting theories

This section presents theories behind weather forecasting, with three forecasting methods as follows.

1. The mean-based method which use selective estimate of the sample mean as a forecast.
2. Medium-based method uses a median estimator, which is the medium over a period of time within a 95% of confidence interval (CI).
3. Autoregressive method to use advanced mathematical modeling to determine the estimate.

To ensure the accuracy and effectiveness of the proposed method, the combined use of the mean square prediction error and the mean percentage prediction error is the choice, as reported by Wolski, Spring and Hayes [26] in their attempt to use weather science theories into their network technology. Our objective is to make a prediction model a generic model, for example, for each predictive method f at measurement t ,

$$\text{Prediction}_f(t) = \text{METHOD}_f(\text{value}(t), \text{history}_f(t)) \quad (1)$$

where

- $\text{value}(t)$ = the measured value at time t ,
- $\text{prediction}_f(t)$ the predicted value made by method f for measurement value $(t+1)$
- $\text{history}_f(t)$ = a finite history of measurements, forecasts and residuals generated previously to time t using method f .
- METHOD_f = in forecasting method f

The values supplied by the weather data authorities are regarded as a time series by a forecasting method. Each method maintains a history of previous records and accurate information. This leads to the definition of the error residual.

$$\text{Err}_f(t) = \text{value}(t) - \text{prediction}_f(t-1) \quad (2)$$

where the method f generates the error residual err_f associated with a measurement and a forecast of that measurement. This model is the most suitable dealing with fewer instances of t at a time.

2.2 Mean-based method

This section presents the mean-based method for weather forecasting. The running average is the term to define an estimate of the mean values and is adopted for the mean-based method. It is based on the measurement of the historical data to predict the value of the next measurement [8, 27-28]. The running average is presented as

$$\text{RUN_AVG}(t) = \frac{1}{t+1} \sum_{i=0}^t \text{value}(i) \quad (3)$$

Formula (3) computes the average of the measure-

ments taken at time t while using all the previous measured data as a predictor of the measurement at $t + 1$. The weight in each measurement decreases with time linearly. The reason is that the running average takes the entire historical measurements into account while modeling each forecast. An average taken over a fixed-length history can be used as a predictor and it also means fixing the weight given to each measurement. The fixed-length or sliding window average can be calculated as

$$SW_AVG(t, K) = \frac{1}{K+1} \sum_{i=t-K}^t value(i) \quad (4)$$

where $K = 0$ is an integer indicating the number of samples used in the sliding window. If $K = 0$, SW_AVG then uses the last measurement as the only predictor, which then becomes

$$LAST(t) = SW_AVG(t, 0) \quad (5)$$

The value of K may vary over time, so it is important to set K dynamically to adapt it to the time series. A gradient-descent strategy is used to achieve this [16]. If $K(t)$ represents the value of K at time t , the formula becomes

$$err_i(t) = (value(t) - SW_AVG(t, K(t) + i))^2 \quad (6)$$

Since the values of t and K can change over a period of time, it affects the values of SW_AVG . A new terminology, $ADAPT_AVG$, is used [26] to illustrate the different variations in the value of SW_AVG . $ADAPT_AVG$ can list all the possibilities in regard to the change of t and $K(t)$. Formula (7) and (8) are in Appendix. The value of K can be adjusted at each time step, particularly when the lowest error that can be computed while using formula (6), which uses square error. Absolute percentage error can be used as a measurement as an alternative [27]. The value of $K(t)$ should be part of the historical value of $ADAPT_AVG$, and $K(0)$ is the starting value. K should also be restricted to ensure a high extent of accuracy. In this case, K is set between 5 and 50 based on the preliminary mathematical modeling that we perform, which can yield to the lowest errors.

2.3 Median-based method

The median value prediction can be useful for measurements that contain random and asymmetric outliers. The proposed median-based method follows up the method introduced by [28], who explain the use of their method. The median over the sliding window of a fixed length is used as the forecast as follows:

- Sort K = the sorted sequence of the K most recent measurement values,
- Sort $K(j)$ = the j th value in the sorted sequence.

$$MEDIANT(t, K) = \begin{cases} Sort_K((K+1)/2) & \text{if } K \text{ is odd} \\ (Sort_K(K/2) + Sort_K(K/2+1))/2 & \text{if } K \text{ is even} \end{cases} \quad (9)$$

Formula (9) explains the value of the median is dependent on the t and K , and the value of Sort K is dependent on K is an odd or even number. Similar to formula (7), there is a value for $ADAP_MED(t)$ presented in formula (10) in Appendix, where $K(t)$ is the value at time t and $err_i(t)$ is represented as follows.

$$err_i(t) = (value(t) - MEDIANT(t, K(t) + i))^2 \quad (11)$$

In formula (11), $K(t+1)$ is determined by formula (7). Predictions by median-based methods are useful and removing the possibility of outlying data points. On the other hands, this method lack the smoothing ability of the averaging based methods [29], resulting in forecasts with certain extent of jitters produced. However, input data from the previous years, and/or months, and/or weeks is required to ensure that forecasting can produce better quality of results.

2.4 Autoregressive method

Autoregressive, integrated, moving average (ARIMA) models have been used in financial services to predict the likely outcomes of the selected stocks and options [29]. However, ARIMA requires non-linear simultaneous equations to fit all models to specific time series, which makes it difficult and challenging [27]. On the other hand, a purely autoregressive model (AR) model is a suitable method. The reason is that AR only needs a linear system of equations which use the Lavinson Recursion approach to achieve forecasting [24]. Formula (12) presents the p th order autoregressive model.

$$AR(t, P) = \sum_{i=0}^p a_i \times value(t-i) \quad (12)$$

The use of AR models can explain how financial forecasting can be applied to weather science of making useful forecasting based on historical data. However, a challenge is that the time series is moving, but it can be stationary. For example, the same or highly temperature can stay for a few days. This means time series is "stationary", or with slow-moving movement. When such a case happens, the sequence $\{a_i\}$ minimizes the error can be determined by the linear system:

$$\sum_{i=0}^N a_i \cdot r_{i,j} = 0, \quad j = 1, 2, \dots, N \quad (13)$$

where r_{ij} is the autocorrelation function for the series of N measurements taken. We calculate the $\{a_i\}$ coefficient sequence over a sliding window of the K as the most recent measurement, rather than the entire series of the size K . The reason is that we can recomputed the autoregressive coefficients $\{a_i\}$ using the previous K measurements as a good approximation of the complete time series [31]. The model tracks the prediction error and uses AR model only if the error is lower than the competing predictors. The autocorrelation can be expensive to compute, choosing arbitrary fixed values will

be useful, where the values of $p=15$ and $K=60$ can yield the maximum output and efficiency according to [31]. Thus, our inputs for p and K also follow this.

2.5 Variance Gamma Process (VGP) Model

Variance Gamma Process (VGP) model has been effective to smooth out outliers and ensure that all the predicted values have consistencies. A commonly used technique is to blend Monte Carlo Method (MCM) with VGP, error corrections can be achieved as demonstrated by Ribeiro and Webber [32]. Variance-Gamma Process (VGP) has been adopted by a few research projects [32-35]. Brigo et al. [32] explains how VGP reduces inconsistency with their algorithm and demonstrate their improved risk modeling process. Error corrections in any forms of predictions are essential. When errors are identified, rectifications need to be found and applied automatically wherever possible [35]. The reason is that slight discrepancies in predictive analysis can lead to adverse impacts such as financial loss. VGP is a technique of MCM used for error corrections. VGP offers two benefits:

- Pricing and risk analysis can be simulated by using raw data which includes good and out of range data.
- It removes out of range of data, recomputes simulations and presents the improved simulations.

Double-gamma bridge sampling (DGBS) is another method used in VGP for error corrections by eliminating outliers and reducing gaps or uncertainties in the list of predicted values [35] presented by formula (14), which has been validated by Avramidis et al [36], who have illustrated their algorithm. Our approach is to integrate examples in [32-36] for error corrections of the three major methods represented by formula (3), (9) and (12) for a periodic time, for example, every 10 minutes.

$$X(t; \theta, \sigma, \nu) = \gamma_p(t; \mu_p; \nu_p) - \gamma_n(t; \mu_n; \nu_n) \quad (14)$$

where

$$\mu_p = (1/2)\sqrt{\theta^2 + 2\sigma^2/\nu} + \theta/2$$

$$\mu_n = (1/2)\sqrt{\theta^2 + 2\sigma^2/\nu} - \theta/2$$

$$\nu_p = ((1/2)\sqrt{\theta^2 + 2\sigma^2/\nu} + \theta/2)^2 \cdot \nu$$

$$\nu_n = ((1/2)\sqrt{\theta^2 + 2\sigma^2/\nu} - \theta/2)^2 \cdot \nu$$

2.6 Prediction Selection

This section describes the formulas related to prediction selection. Our approach maintains all the predictive methods simultaneously, rather than attempting to select a method. This ensures that we do not miss out outputs of any predictive modeling. Error measures in formula (2) can be used to justify the overall fitness of each method. The method illustrating the best predictive per-

formance at any time can be used to forecast the measurement at time $t+1$. As discussed in Section 2.1, 2.2 and 2.3, formulas are presented to demonstrate the calculations of the errors, which require structured model to compute.

$$MSE_f(t) = \frac{1}{t+1} \sum_{i=0}^t (err_f(i))^2 \quad (15)$$

$$MPE_f(t) = \frac{1}{t+1} \sum_{i=0}^t |err_f(i)|/value(i) \quad (16)$$

Amongst structured formulas, the mean square error and the mean percentage are the commonly used [14-15, 17]. The mean square prediction (MSE) error and the mean percentage prediction (MPE) error are introduced and presented in formula (15) and (16) respectively, where the fitness metrics for each method f is at the time t with further definitions below.

$$MIN_MSE(t) = predictor_f(t) \quad (17)$$

if $MSE_f(t)$ is the minimum overall method at time t .

$$MIN_MPE(t) = predictor_f(t) \quad (18)$$

if $MPE_f(t)$ is the minimum overall method at time t .

Interpretations of formulas (17) and (18) are as follows. At time t , the method yielding the lowest mean prediction error is used to forecast the next measurement by MIN_MSE . Similarly, the method that yields the lowest mean percentage error at time t is used to forecast the next measurement by MIN_MPE . Both MSE and MPE are used since the differences to yield the better prediction between them can be small.

Table 1: Summary of forecasting methods and key inputs

| Predictor | Descriptions | Parameters |
|----------------------------|---|-----------------------------|
| RUN_AVG | Running average | |
| SW_AVG | Sliding window average | $K = 20$ |
| LAST | Last measurement | |
| ADAPT_AVG | Adaptive window average | $\max = 50$, $\min = 5$ |
| MEDIAN | Median filter | $K = 20$ |
| ADAPT_MED | Adaptive window median | $\max = 50$, $\min = 5$ |
| AR | Autoregression | $K = 60$, $p = 15$ |
| MIN_MSE | Adaptive minimum MSE | |
| MIN_MPE | Adaptive minimum MPE | |
| Error corrections [35, 36] | Use ADAPT_AVG, ADAPT_MED and AR values. Get the mean, compare values and finalize every 10 minutes. | |

To sum up all the discussions in Section 2, Table 1 presents all the predictive methods and the key input values. Three case studies will be presented in Section 4 to support our forecasting methods, whereby results of experiments will be presented in Section 4.4.

3 The architecture

This section presents the architecture to perform weather science computing. The first section explains the overall system design and deployment for performing weather computing and platform architecture in the Virtual Machines (VMs). The second section presents the parallel MapReduce algorithms to process weather data and optimize computation.

3.1 The process for weather computing and system architecture

This section presents the process involved in weather computing and the system architecture. Weather data is extracted from data sources (such as National Weather Service) and then stored in the data warehouse with the managed clean data. Figure 1 shows the process involved and the system used for the weather computing. Data sources from the National Weather Service, the MET Office and the BBC Weather. Our service can download their daily data of temperatures, which converts daily temperature data into text formats. The daily data is 2 GB and stored in our system recording the temperature of the day for 24 hours. The processed data are stored and managed in the data warehouse, which also cleans the data and gets it ready for computational analysis.

Figure 2 shows MapReduce data flow process which involves with (a) map and merge operations and (b) reduce operations. The MapReduce data flow architecture is based on the improved version of Flame-MR model proposed by [39] and recommendations from [40-42]. It involves two major steps with three-worker model to allow a better collaboration. The first step in Worker 1 involves with dividing inputs and mapping them into

sections, sorting them before passing out as the output of Worker 1, which then become the input of Worker 2. The second step merges all the sorted sections in Worker 2 into a larger output. In Worker 3 corresponding to Figure 2 (b), which then focuses on reduced operations, which reduces all outputs of Worker 2 into structured and collected output as the final output. More descriptions about how MapReduce can work with NoSQL will be presented in Section 3.3.

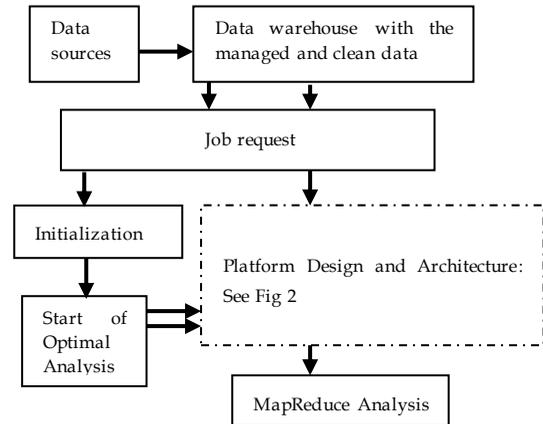


Figure 1: The process involved and system design for weather science computing

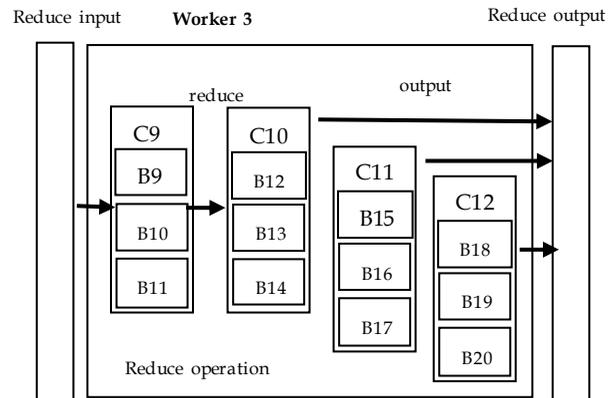


Figure 2: (b) MapReduce dataflow with reduce operations

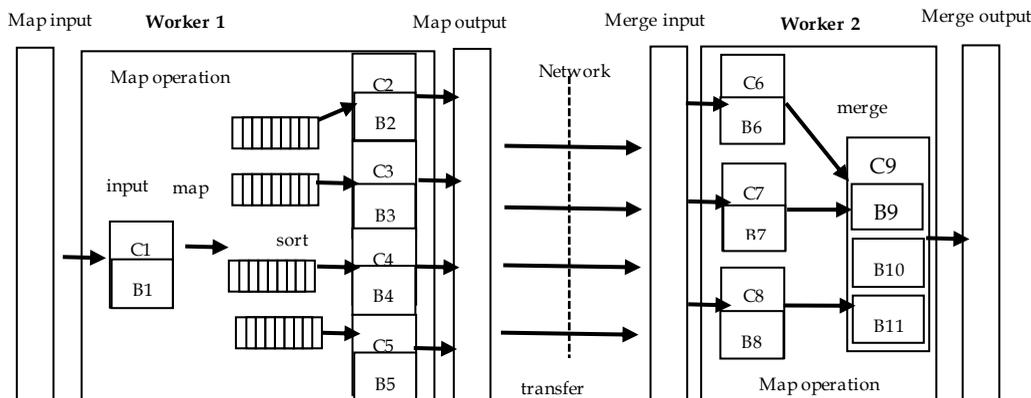


Figure 2: MapReduce workflow diagram- (a) MapReduce dataflow with map and merge operations

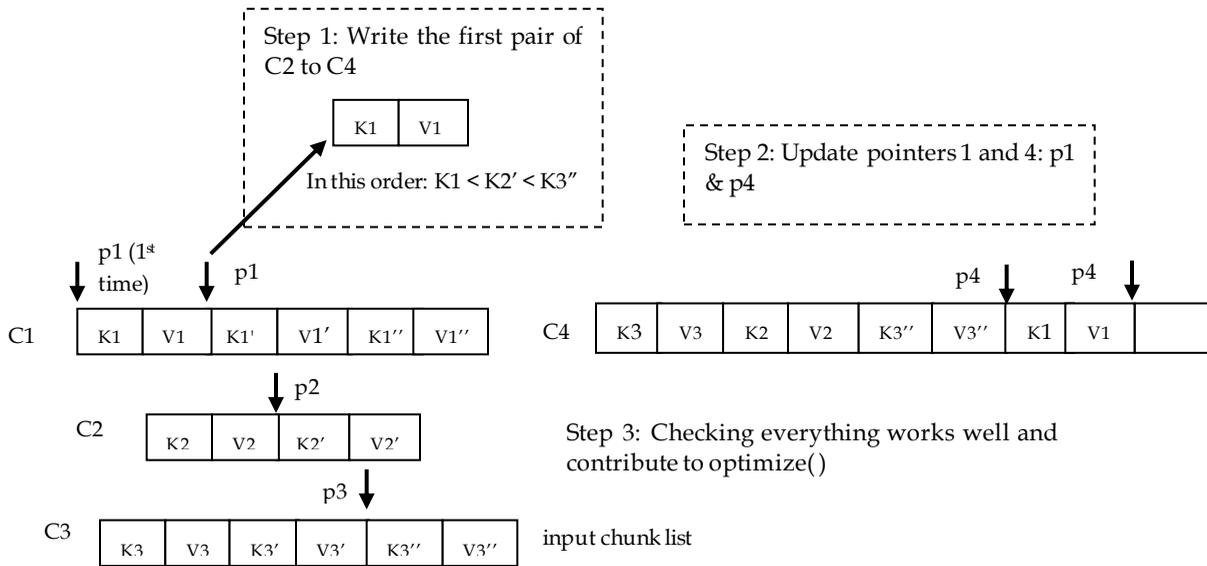


Figure 3: Techniques behind optimization

3.2 Techniques for optimization

Optimization is required to speed up Map and Reduce processes in MapReduce, so that all jobs can be processed and completed. It is an essential scientific process for our Weather data processing. To ensure optimization to take place successfully, Merge and Reduce operations are used and described as follows.

K-way merger for Map function: This algorithm contains 3 steps. C represents “chunk” that contains all data, K is the key; V refers to the values and p is the pointer to C, K and V during the Map and Reduce operations. The principle can be summarized by illustration of Figure 3. Input chunks of C1, C2 and C3 are merged into C4. Each chunk contains a list of key-value pairs such as $\langle K_i, V_i \rangle$, or $\langle K1, V1 \rangle$ in the first pair of C1, with the pointer in place [44]. There are 3 steps to make this happen. In step 1, the pair is copied into chunk C4 illustrated in the left part of Figure 3. In step 2, pointers of $p1$ and 4 are updated so that positions of C2 and 4 are moved forward and also corresponding K and V positions in C4. Step 3 checks everything works well. For example, checking whether C1 is in the first place as $\langle K1', V1' \rangle$ for update and become the lowest key in the list. The benefit allows processing of multiple data chunk happen at once rather than multiple times, this can reduce the number of merging operations to achieve optimization, since speed can be enhanced and percentage for errors can be reduced.

Pipeline outputs for Reduce function: Pipeline outputs aim to streamline all data chunk outputs and minimize the write operations (to execute reduce function). Our approach is to store output pairs to a data chunk which is written to HDFS. All outputs are limited to a threshold size, which can ensure the reduce outputs do not go

beyond the remaining available memory, or without implementing an additional algorithm to free up memory. It can also work with HPC pipeline such as HDFS to ensure a good optimization. Both k-way merger for Map function and pipeline outputs for Reduce function can be presented as optimize() to achieve optimization of data processing. More results will be presented in Section 5.

3.3 Parallel MapReduce and NoSQL algorithm

Cloud Computing can use parallel algorithms to optimize performance with NoSQL databases. Referring to Figure 2, each virtualized application has their services up-and-running. However, a parallel algorithm is required to optimize all VMs for performance enhancement. MapReduce is commonly used as in Cloud, HPC and Cluster Computing. It can be fully used in Cloud Computing while treating each VM as a node in the cluster [40-42]. NoSQL databases can be used to reduce possibility of security hacks and improves integration between data queries, data structure and complexity [43]. This section demonstrates parallel MapReduce and NoSQL algorithm.

Presented in Section 3.2, MapReduce framework can be presented and summed up as $\langle \text{key}, \text{value} \rangle$ to demonstrate the input and output of data processing. The advantage is to simplify the programming model for parallel computing. As shown in Figure , there are five stages in which MapReduce is involved [41]:

- (1) Input: Details were explained in Section 3.2. The application input is treated as a group of $\langle \text{key}, \text{value} \rangle$ to process the data and provide the locations of both input and output.

- (2) Map: The framework calls the Map function that users define to process each <key, value> key value pairs. It also creates “new intermediate <key, value> key value pairs” simultaneously, as explained in Section 3.2. This can also be used as `map()` in our algorithm.
- (3) Shuffle: The frame gets all related <key, value> key value pairs in Map output for each Reduce through HTTP/HTTPS to ensure that Reduce input is output in sequence that Map has already sequenced, as presented in Figure 2.
- (4) Reduce: This stage contains full of intermediate data and implementation of the user-defined Reduce function for each unique key. The input parameter is “<key, {list of values}>”, and the output is a new < key, value > key value pairs. All lists of values are reduced to each single number. This can also be used as `reduce()` in our algorithm.
- (5) Output: This involves with writing the result output from Reduce in the location of the output directory to ensure a MapReduce process is completed.

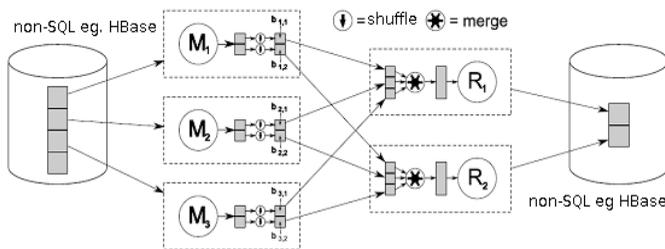


Figure 4: MapReduce operations with NoSQL

Table 2: Algorithm 1 parallelized constructing the reduce function in similarity matrix

```

i_content = retrieve(HBase(i));
For j = i to n do
  j_content = retrieve(HBase(j));
  sim = computeSimilarity(i_content, j_content);
  storeSimilarity(i, j, sim) into HBase table;
End For
Output <key, null>
End.

```

Table 2 shows the first parallel computing algorithm to reduce function in similarity matrix and store them together in HBase. With regard to the code in Table 2, the application input is treated as a group of <key, value>. HBase is developed as Apache Hadoop project and runs on top of Hadoop Distributed Filesystem (HDFS) to offer BigTable-like capabilities for Hadoop. It is consistent with read and write operations, which make processing more streamline with the pipeline architecture. HBase makes APIs (such as Java) to be easy to access and use [45]. In this way, it reduces the time that users receive the outputs. HBase can be used to block cache and

Bloom Filters, so that it reduces possibility of getting a prolong job request due to massive cache accumulated through data processing [45-46]. The command `retrieve(HBase(i))` can directly get the data from HBase. Detailed set up will be presented in Section 4.4. HBase with MapReduce can support the use of the Laplace matrix that contains the k minimum vectors and k -means the clustering, where Farivar et al. [47] explain how to achieve parallelization of K -means clustering. Originated from GPU, k -means clustering can be utilized for Cloud. We adopt this approach and use it in our MapReduce framework. In summary, our parallel design for the clustering algorithms includes four steps:

1. Reading all input data and their values
2. Split the data and tasks are reduced
3. Merge all the results from the Reduced process
4. Present the final output

Our approach adopts the essence from the MapReduce into “five-step process” visualization and examples will be presented in Section 5.1.

4. Demonstrations of weather computing

This section explains how to use the theories and system development described in the previous section to produce predicted results. We use the historical data and [analyze the trends](#) to predict the following months and the following years. We compare the predicted results which keeps the differences between the predicted and actual temperatures to be within 1%. Modeling results in Section 2, parameters in Table 1 and system architecture in Section 3 are used to demonstrate weather forecasting. Other variables without parameters can be calculated by mathematical formulas. Results can then be presented in visualization shown in Figure 5, 6 and 7. Three case studies are presented: Sydney, Singapore and London to support the validity of our proposed solution. We undertook these forecasting from a different period of time between 2010 and 2014 to show that our forecasting method can work across different periods and different locations. We also use the historical data to perform the forecast.

4.1 The Sydney case study of 20GB

This case study demonstrates [analyzing the trends](#) of the historical results between January 2010 to March 2012 to forecast the mean temperature between January 2012 and March 2014. The three-month period between January 2012 and March 2012 can provide relevant parameters for temperature forecasting. Figure 5 shows the historical data between January 2010 and March 2012 on the left and the forecasted temperatures based on the two-year historical data on the right. It analyzes 20 GB of data (text files) using MapReduce altogether. All the datapoints are computed together which look like graphical visualization as in Figure 5.

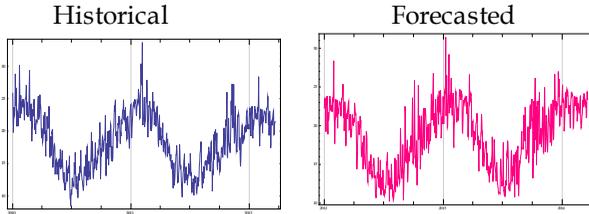


Figure 5: The mean temperature in Sydney between January 2010 and March 2012 (left; historical) and between January 2012 and March 2014 (right; forecasted)

The forecasted results inherits the characteristics of the historical results with more spikes and had 95% and above accuracy. Data was carefully checked before making the forecasted results. The two-year data before each forecast can be useful for cross-checking all the key outputs presented in Table 1. Referring to Section 2.5 and Table 1, mean square prediction error (MSE) and the mean percentage prediction error (MPE) are used to present the extent of errors in the forecasting. The use of MSE and MPE ensures no outputs are omitted for any predictive modeling. Results for Sydney case study are shown in Table 3 with acceptable values for MSE and MPE. With regard to error measurement, the lower the values are, the more accurate the results can present. While applying error correction technique by [35, 36], MSE and MPE can be kept under 0.0027.

Table 3: Forecasting error statistics for Sydney

| Predictor | MSE | MPE |
|-----------|----------|----------|
| RUN_AVG | 0.002582 | 0.002676 |
| SW_AVG | 0.001951 | 0.002169 |
| LAST | 0.002878 | 0.002194 |
| ADAPT_AVG | 0.002208 | 0.002416 |
| MEDIAN | 0.002086 | 0.002227 |
| ADAPT_MED | 0.001980 | 0.002048 |
| AR | 0.002278 | 0.002111 |
| MIN_MSE | 0.001927 | 0.002183 |
| MIN_MPE | 0.002023 | 0.002075 |

All the key outputs for MSE and MPE modeling are low but above 0.001. The reason is that although the overall trends of the predicted and actual temperatures are similar, the extent of the fluctuations means that the forecasted results yield a certain level of errors which cannot be removed in the temperature forecasting. This happens when out-of-range outlier has been experienced and rounds undertaking MSE and MPE will be required to smooth the final outputs.

4.2 The Singapore case study of 20 GB

Similar to Section 4.1, Figure 6 shows the historical data between January 2010 and March 2012 on the left and the forecasted temperatures between January 2012 and March 2014 on the right. The only exception is the temperature between the end of December 2012 and early January of 2013, where the forecasted temperature was adjusted to the forecast warning of higher temperature at the end of Year 2012 in October 2012. It analyzes 20

GB of text data (text files) using MapReduce altogether. Graphical visualization in Figure 6 contains all the data-points for historical and forecasted outputs. Table 4 show a small forecasting error statistics for Singapore.

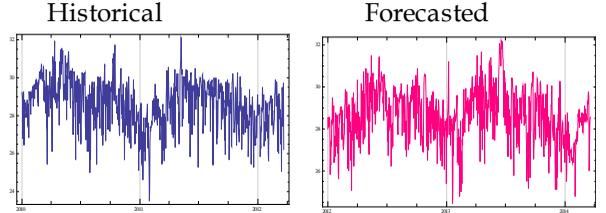


Figure 6: The mean temperature in Singapore between January 2010 and March 2012 (left; historical) and between January 2012 and March 2014 (right; forecasted)

Table 4: Forecasting error statistics for Singapore

| Predictor | MSE | MPE |
|-----------|----------|----------|
| RUN_AVG | 0.002281 | 0.002472 |
| SW_AVG | 0.001753 | 0.001974 |
| LAST | 0.002675 | 0.001993 |
| ADAPT_AVG | 0.002011 | 0.002213 |
| MEDIAN | 0.001882 | 0.002025 |
| ADAPT_MED | 0.001777 | 0.001946 |
| AR | 0.002076 | 0.001911 |
| MIN_MSE | 0.001724 | 0.001985 |
| MIN_MPE | 0.001828 | 0.001978 |

Similar to Section 4.1, MSE and MPE modeling have been used for the Singapore case study. Results are shown in Table 4. Similar to Section 4.1, fluctuations in both predicted and actual temperatures are noted are kept within 8 C of range. This may explain why all the values in MSE and MPE in Table 4 are lower than in Table 3. While applying error correction technique by [33, 34], MSE and MPE can be kept under 0.0025.

4.3 The London case study of 20GB

Similar to the last two case studies, Figure 7 shows the historical data between January 2010 and March 2012 on the left and the forecasted temperatures between January 2012 and March 2014 on the right. The forecasted data looks similar to the historical data, except there are greater extents of fluctuations. There is a revised forecasted result in the early 2014 due to the warning of the warm and wet winter in October 2013 by the UK Met Office [11].

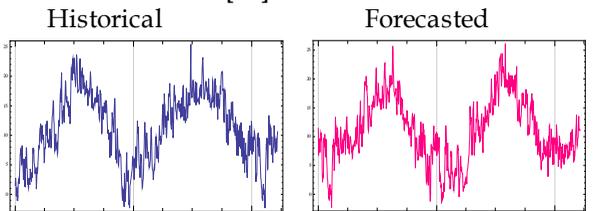


Figure 7: The mean temperature in London between January 2010 and March 2012 (left; historical) and between January 2012 and March 2014 (right; forecasted)

Reasons for the forecasted temperatures can be supported by the modeling results of the forecasted models.

It analyzes 20 GB of text data (text files) using MapReduce altogether. Graphical visualization in Figure 7 contains all the datapoints for historical and forecasted outputs. Table 5 show a small forecasting error statistics for London.

Table 5: Forecasting error statistics for London

| Predictor | MSE | MPE |
|-----------|----------|----------|
| RUN_AVG | 0.001974 | 0.002232 |
| SW_AVG | 0.001932 | 0.002365 |
| LAST | 0.001961 | 0.002393 |
| ADAPT_AVG | 0.001936 | 0.002335 |
| MEDIAN | 0.001924 | 0.002310 |
| ADAPT_MED | 0.001926 | 0.002414 |
| AR | 0.001971 | 0.002406 |
| MIN_MSE | 0.001923 | 0.002475 |
| MIN_MPE | 0.001948 | 0.002496 |

4.4 Cloud and non-Cloud comparisons for 40GB data

This section describes the comparison between Cloud and non-Cloud comparison. The Grid computing available at Southampton was used as a resource for comparison. Figure 8 shows a generic platform for Grid/Cloud for benchmark and performance comparison to process raw data of 40 GB each. It has identical hardware infrastructure: 100 CPU with 3.0 GHz speed each, 100 GB memory, 10 TB and 10 Gbp network speed. The difference between the Grid and Cloud includes:

- Grid requires one job to be completed before processing the next. Our pipeline approach allows initiation of the next job before the completion of the current one to reduce run time.

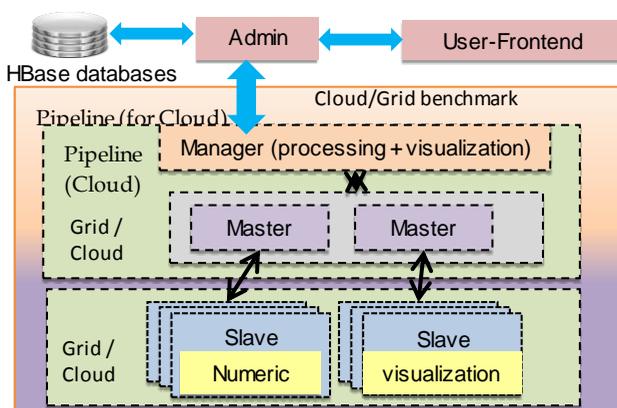


Figure 8: A generic Grid/Cloud benchmark platform for comparison.

With regard to the benchmark platform, “Admin” is the first step to access other functions. While choosing “Manager”, it is a function to process all the data. It adopts the Master and Slave architecture. The Master sends jobs, processes data and then uses Slaves to compute all the requests in numerical outputs or visualization. The use of pipeline for our Cloud only also ensures that different jobs can be initiated before starting and

completion, thus reducing execution time for weather computations. Three experiments were conducted on each of Grid and Cloud with the execution time for job completion being recorded. To this experimental set-up, each node consisted of a pair of master and slave and up to 10 nodes were used for experiments. Only London case study was chosen to compare the performance between Cloud and non-cloud due to the close proximity of the execution time in all three case studies. There are two types of master-slave pairs as follows:

- Numerical computation: It computes all the key values in MSE and MPE.
- Visualization: It collects numerical outputs and presents them as visualization.

Experiments were involved with measuring the execution time three times on both numerical computation and visualization for nodes between 10 and 100. Results are shown in Figure 9, whereby Cloud has a better performance for numerical computation between 59 (10 nodes) and 38 (100 nodes) seconds faster when the number of nodes has increased. When the number of nodes is 60 and above, the rate of the decreased execution time has been slower. On the other hand, the visualization was between 32.12 and 26.21 seconds for Cloud whereas the rate of the decrease in the execution time in the non-Cloud was faster before number of nodes had reached 50. The explanations can be that MapReduce framework has optimized the performance for numerical computation for Cloud. In the case of visualization, the outputs of the MapReduce are directly extracted and presented in the form of visualization for Cloud and the non-Cloud approach needs to calculate all the output data and convert them from numerical modeling into visualization.

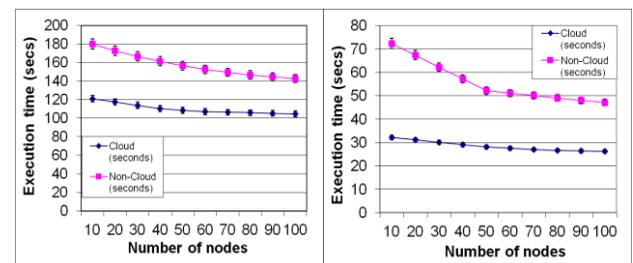


Figure 9: The average execution time for numerical computation (left) and visualization (right) for Cloud and non-Cloud for nodes between 10 and 100

4.5 The execution time of running each case study

This section presents performance evaluation of conducting three case studies between Sections 4.1 and 4.3. The architecture is based in Figure 8 to process the data five times. Techniques to process data have been explained in Section 3. The aim is to determine the numerical results recorded in Table 3, 4 and 5. Execution time is used as a benchmark for the performance of the computational work [18, 20], which includes the time to complete the task (which is our definition) or a lower CPU load to complete the task. Figure 10 then shows that the mean execution time for obtaining results for

Table 3, 4 and 5 is 102.36, 103.48 and 103.42 seconds for three case studies, with 2.4% standard deviations.



Figure 10: The average execution time for weather computing in three case studies for processing 20GB of text data

4.6 Summaries of all case studies

This section presents three case studies that the analyzing trends of the historical data can be used for temperature forecasting. Three cities, Sydney, Singapore, and London were chosen to demonstrate that the use of MSE and MPE can be useful to check the accuracy and the forecasting error statistics in our proposed approach. All the output indicators for MSE and MPE are low and their explanations have been justified. We explained a generic Cloud/Grid platform for performance comparison. Experiments were conducted, whereby the Cloud approach had better performance in both numerical computation and visualization shown in Figure 9.

5. Data Visualization dealing with extreme weathers

This section describes the use of technologies and APIs in Section 3 to achieve data visualization. Some parts of the world experienced extreme weather conditions between November 2013 and March 2014, for example, the polar vortex in the North America. The objective is to illustrate different regions of the US could experience different ranges of temperature. The contribution is to transform numerical computing into visualization, so that interpretation can be easier including those without the specialization in this area. This approach is different from results presented in Section 4, which emphasize on analyzing all the historical data and use them for forecasting. The approach for this section is to collect all the historical data in details, categorize them into different groups and each group has its own colors indicating the range of the temperature. The amount of the data to analyze is as significant as in each case study presented in Section 4. Each specific task requires different types of algorithms. The algorithms involved in data processing and visualization are as follows.

5.1 Algorithms for generic data visualization

This section explains the algorithms associated with

data visualization, which includes four-function steps: read, process, update and visualize corresponding to different stages of the MapReduce framework discussed in Section 3.2. Map and Shuffle stages in MapReduce correspond to the core step in data processing which explain why we use five-step functions. All steps take place in sequence explained as follows:

- **read(data)** reads all numerical data from the US Weather Service. This is referred to “input” stage of the MapReduce framework.
- **process(data)** processes the data from the read operation. Collected data is recorded as datasets, which can be varied dependent on the quantity of the input data. This is referred to the combined “Map” and “Shuffle” stage of the MapReduce framework.
- **update(data)** updates all the data values for read(data) and process(data). This is referred to the “Reduce” stage of the MapReduce framework.
- **visualize(data)** represents the numerical values in the phase of process(data). The intensity of each output is represented by the numerical values collected and calculated. This is referred to “output” stage of the MapReduce framework.
- **optimize(data)** – this function can accelerate speed and completion for all the functions above to ensure successful job done. It makes use the parallel processing and MapReduce optimization framework described in Section 3.3.

The term dataset is to describe each set of collected values for “read, process, update and visualize” procedures. Each dataset contains a row of all the information associated with these four procedures. This needs infrastructure in Section 3 and 4 and four-step MapReduce process in Section 5.1 to process and interpret the data prior data analysis. The main reason is to improve on performance, since computed outputs should be presented in visualization which demands on performance. The optimized Map function is presented in Table 6.

Table 6: The optimized Map function design

```
map()
m=output(current clusterID, point);
optimize(m)
```

The Map function can read the datasets from the weather data with 200 GB altogether. Datasets with considerable sizes (such as 200 GB) should be considered to use HBase or another non-SQL database. It can calculate the nearest class center to the input data point. The output is presented by <key, value>, which includes <cluster category ID, record at attribute vector>, or <cluster ID, point> in this case, and can be summed up by a variable m. The reasons to do so is to support the handling of data processing involved with thousands of records <key j, value j> produced in Map process, which can be handled by map(). To achieve optimization, the process

optimize(m) can constantly accelerate processing of variable and returns the output to Reduce step.

The Reduce function is to calculate the new clusterID associated with the Map function and is useful for the next round of MapReduce job. The form of the input data <key, value> is <clusterID, {record}>. All the records with the same key can receive a Reduce task which can achieve the followings:

- Accumulate the number of points with same key
- Sum the records, get the mean value and a new clustering center description file
- The output result <key, value> pair is <cluster ID, average vector >

Table 7: The optimized Reduce function design

```
reduce()
sum[i] += current point. point[i];
r=out(key, sum[i]/num);
optimize(r)
```

The syntax for the optimized Reduce design is presented in Table 7. The iteration will not stop except each class cluster center is not changed any more. Key in the output <key, value> has been identified from optimize(m). The challenge is to calculate which value corresponding to m of different time scales (since m can be changed all the times) from the cluster. Thus, reduce() is used to identify the value. One for loop is used to get the sum and the mean (divided by the number of iteration), which corresponds to the "value" in output <key, value>. Next, the operation out(key, mean) gets the final outcome for Reduce function, and can be presented by a variable r, which is recorded with different r values of different time scales until the completion. The function optimize(r) will accelerate processing of the outputs from Reduce function.

5.2 Overview and algorithms for specific data visualization involved with color data and map

This section describes visualization with specific needs, including data visualization to get numerical outputs to the right color codes and map. Since outputs in Sections 4 and 5.1 are in numerical formats, it is necessary to transform them into visualization to the correct color and map to ensure accuracy. This is an important part since the data processing and analysis have been undertaken from the hardware design and infrastructure and all the way to software framework, processing and visualization.

5.2.1 Transforming the temperature data into visualization

Weather data has been collected, analyzed and presented in Section 4. However, a lot of mathematical modeling and simulations based on numerical data

have been presented, an innovative and easy-to-use algorithm is required to allow the data to be processed and presented in the form of analytics, so that all the results can be presented as visualization to allow users and stakeholders to understand all the numerical analysis with ease. To illustrate the first step for visualization, Figure 11 shows the color codes for categorizing and displaying different temperature ranges on the collected data. Five categories were used to check warmer temperatures, and another five to check cooler temperatures with one category to take the mean temperature. That is why eleven categories are used. All temperature data can be read, saved, compared and categorized all the numerical values into eleven category. The highest temperature is in red and the lowest is in dark blue. The color of the temperature depends on the mean temperatures. If the mean is 10C and below, more regions and colors will be more towards the blue and its variants and vice versa. Other factors such as the weather conditions of the day can affect the color codes. If it is windy, raining or snowy, the color codes will be more towards blue and its variants. If the weather is sunny and warm, it is more towards yellow, orange and red.

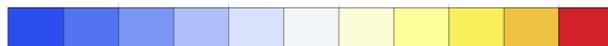


Figure 11: Color codes for displaying different temperature ranges

5.2.2 Algorithms for temperature visualization

This section explains the algorithms for temperature visualization, which requires nine-step process due to additional work to present data and map them at the right place. The following nine-step process have been defined and programmed as follows:

- read(data): All the numerical data have been read.
- store(data): All the data have been stored temporarily in the memory until reshuffle of data is completed
- shuffle(data): Data have been compared with each other and then rearranged
- check(data): This function is to ensure that all comparisons conducted by shuffle(data) is accurate.
- categorize(data): Based on the result in shuffle(data), all data can be categorized into eleven categories according to their numerical values
- color(data): Each category is mapped to its own color code. The lowest is assigned to dark blue and the highest is assigned to red. Those in the middle is mapped to each of remaining nine color codes.
- map(data): it only works for USA, UK and a few countries with original maps from Google APIs and other internet sources.
- visualize(data): all color codes can be presented on the weather map.
- optimize(data) – accelerate the process as explained in Section 5.1. This is an optional step,

making it an eight-step function with or without optimization.

Compared with Section 5.1, `stored(data)`, `shuffle(data)` are part of the `process(data)`; `check(data)` and `categorize(data)` are part of `update(data)` and `color(data)` is part of `visualize(data)`. All these minor steps are used to ensure `process(data)`, `update(data)` and `visualize(data)` can be fully comprehensive to carry out the tasks. To carry out the tasks, Table 8 how steps of required functions to carry out visualization.

Table 8: steps to carry our visualization

```

read(data);
shuffle(data);
store(data);
check(data);
categorize(data);
output(categorize(data));
If (output(categorize(data) = 11)),
  color(data);
  check(data);
  map(data);
  visualize(data);
  optimize(data); // optional
else
  exit;
end;

```

Since there are 11 categories, `output(categorize(data))` is to check whether there are 11 categories. If there is, the output is returned to 11 and carries on to other steps such as `color(data)` to get the respective color codes, `check(data)` to update all results and `visualize(data)` to display all outputs with the right color codes. If there are no 11 categories, then the entire procedure will exit and end. The function `categorize(data)` divides all data into 11 groups according to their numeric order. The function `color(data)` is to provide each category with the respective color codes for Figure, whereby dark blue on the far left that represents color code 1, with all the way right red on the far right that represents color code 11. The term `status(color(data))` can identify what color code that each category is assigned to. The first five categories are on the lower end of temperature values with variant of blue colors and the last five are on the upper end of temperature values with yellow, orange and red colors to indicate the higher temperatures. The category six stands for range of the mean of all the numeric values. Once the all the color codes have been identified and updated by `check(data)`, `visualize(data)` can display and represent all the values on the map of the selected country enabled by `map(data)` with USA, UK and a few selected countries. For example, it can then match all visualization color code to the map. Each color code represents a different temperature range. More examples will be illustrated in Section 5.4 and 5.5 to demonstrate Data Science for weather cloud computing.

5.3 Case study 1: Data visualization of the US temperatures

Section 5.4 and 5.5 demonstrates the applied examples of combining system and application Data Science to investigate extreme weathers in the US and the UK. Experiments of data processing were introduced earlier. This section describes the outputs of the four-step MapReduce approach presented by data visualization with two case studies presented. The first case study is focused on the United States, which experienced the impact of the polar vortex. Data was collected at four particular occasions which represents different period of polar vortex [23]:

- January 10, 2014: At the beginning
- February 7, 2014: Towards the middle
- March 6, 2014: Towards the end
- April 9, 2014: The beginning of spring and the end of polar vortex

200 GB of data was collected and processed for each date to present the mean temperature of the date. Performance results will be shown in Section 5.5 for Cloud. Analysis and discussion are as follows.

5.3.1 Data visualization of the US temperatures on January 10, 2014

Figure 12 shows temperature distributions in different parts of the US at the beginning of the polar vortex. The majority of the regions fell below 0 C and had temperatures between 0 C and - 9 C. Only a minority of regions could stay above 0 C, with the exception in Florida.

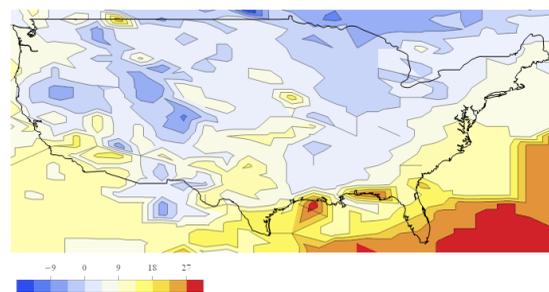


Figure 12: Temperature distributions in the US at the beginning of the polar vortex, January 10, 2014

5.3.2 Data visualization of the US temperatures on February 7, 2014

Figure 13 shows temperature distributions in different parts of the US towards the middle of the polar vortex on February 7, 2014. There was a clear north and south divide in the temperature, where the north of the US fell between 0 C and -20 C in temperatures. The south had the average temperature between 0 C and 27 C. It was reported to have a largest temperature difference between the north and the south in the most recent decade [23].

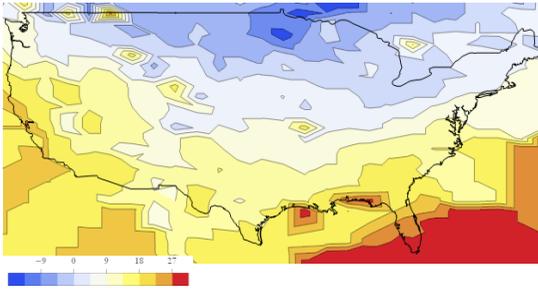


Figure 13: Temperature distributions in the US towards the middle of the polar vortex, February 7, 2014

5.3.3 Data visualization of the US temperatures on March 6, 2014

Figure 14 shows temperature distributions in different parts of the US towards the end of the polar vortex on March 6, 2014. The north-east experience the sub-temperature between 0 C and - 20 C, whereas a significant area in the US had the average temperature between 9 C and 18 C, and between 18 C and 23 C.

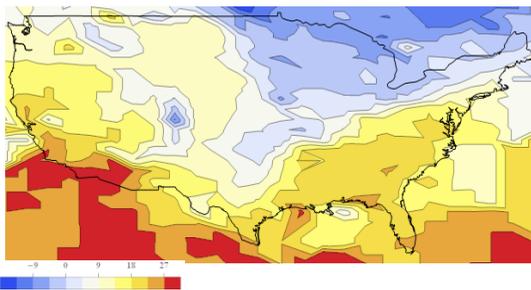


Figure 14: Temperature distributions in the US towards the middle of the polar vortex, March 6, 2014

5.3.4 Data visualization of the US temperatures on April 9, 2014

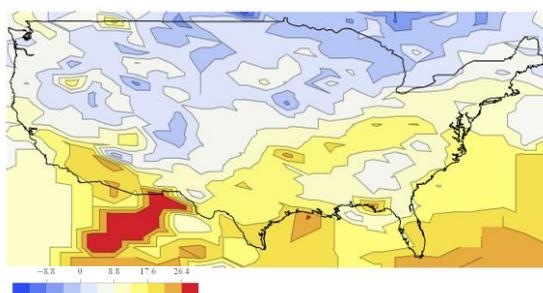


Figure 15: Temperature distributions in the US towards the end of the polar vortex, April 9, 2014

Figure 15 shows the distribution of temperatures in different parts of the US at the beginning the spring and before the submission of this article on April 9, 2014. The temperature distribution still shows the south-north divide but the temperatures were distributed more equally than the previous four dates. The temperature range included between -4.4 C and 0 C, between 0 C and 4.4 C, between 4.4 C and 8.8 C, between 8.8 C and

13.2 C and between 13.2 C and 17.6 C.

5.3.5 Summary of the data visualization

Four dates were chosen to study the impact to temperature distributions in different parts of the US in the different stage of the polar vortex that last for about four months. The use of data visualization can help the general public to understand temperature distributions easily and realize the importance of weather science. Hence, residents in the US to make suitable plans for their visits and day-to-day lives. It helps the travelers to decide whether and when they should visit their destinations in the US. It helps researchers to monitor the temperatures in different stages of the polar vortex and understand the impacts that it can bring to the US and all the people (stakeholders, residents, travelers, businessmen, US local governments and researchers) that are involved.

5.4 Case study 2: Data visualization of the UK temperatures

This case study is to illustrate that the use of data visualization can be applied to the UK. Since the UK has experienced the warm and wet winter, the UK government has focused more effort on rescue actions. Comparing to the National Weather Service, less data is available through the Met Office. An alternative is to record the regional temperatures in the entire UK on the daily basis. Since it took more effort for manual extraction, only two dates were recorded and presented for this analysis. The first date was on February 7, 2014, in which was the peak of the UK flood that caused severe flooding in several towns. The second date was April 10, 2014, in which all floods receded in Easter 2014. 200 GB of data was collected and required for each date.

5.4.1 Data visualization of the UK temperatures on February 7, 2014

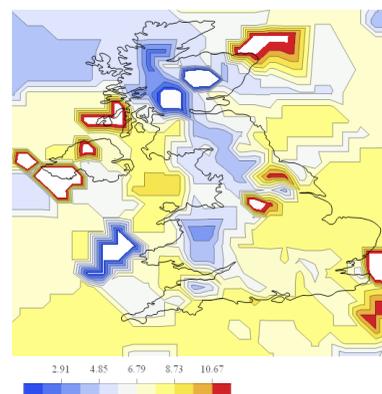


Figure 16: Temperature distributions in the UK at the peak of the flooding period, February 7, 2014

Figure 16 shows temperature distributions in different parts of the UK at the peak of the flooding on February 7, 2014. In 2009, 2010 and 2012 winter, a majority of the UK experienced snow and the sub-zero temperatures. In contrast, Figure shows the opposite – the majority had above sub-zero temperatures, and had temperatures between 6.79 C and 10.67 C. A likely reason to explain the vast abundance of flood could be that it should be snow instead of flood [24].

5.4.2 Data visualization of the UK temperatures on April 10, 2014

Figure 17 shows temperature distributions in different parts of the UK after the flooding period, in which the majority of the flood had receded. Most the regions experienced temperatures between 4 C and 12 C.

However, the mean temperature between February 7 and April 10 are not greatly significant. In other words, the UK has experienced approximately five months of extended warm winter or cold spring since November 2013. Although there is no direct correlation from Boykoff’s forecast [13], the unusual behaviors in the UK weathers may indicate the influence due to global warming. The likely reason to explain is due to the effect of polar vortex, which draws cold air in the North America and pushes the warm air to the other side of the Atlantic Ocean. The use of data visualization allows the general public and scientists to understand the temperature distributions in the US and UK in one single glance, so that they can make respective planning for their work and travel. Precautions and preventive measures can also be enforced to reduce further damage as a result of extreme weather.

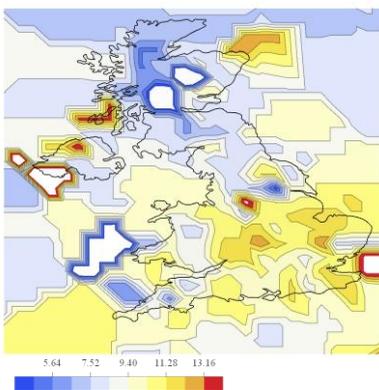


Figure 17: Temperature distributions in the UK after the flooding period, April 10, 2014

5.5 Performance on 200 GB data

This section describes experiments for weather data processing of 200 GB of text formats, as a result of our collection of temperature data in specific dates presented in Section 5.3 and 5.4. The aim is to test the performance and find out any technical issues before presenting final outputs in the form of visualization. The four-step procedures without optimization were been adopted and the infrastructure in Section 3, University

of London Computing Center (ULCC) was used, with hardware infrastructure set up and explained in [48]. The architecture is based on Figure 8. Experiments were conducted five times and results were recorded for both sets of experiments, one of which was based on Section 5.1 and one was based on Section 5.2.

5.5.1 Performance for five-step approach in Section 5.1

Results show that 200 GB of data can use the five-step processes (generic approach) to optimize the performance, including read(data), process(data), update(data), visualize(data) and optimize(data). These five-step approach can be used in data processing and analysis for performance evaluation. All could be completed between 126.5 and 172.8 seconds. “Process” data take the longest since it has Map and Shuffle steps involved. “Update” data is the second longest since it corresponds to Reduce function. “Visualize” takes the least amount of time since its primary function is to present and display the outputs. We use facilities in Section 4.4 to test performance between Cloud and non-Cloud, where Table 9 presents results for Cloud for node =10. We then performed for experiments of up to having 100 nodes, which means connecting up to different 100 data sections (whereby each section contains clusters of machines that each has its VMs) in the ULCC.

Table 9: Performance results of steps involved for data visualization (node =10; no optimization)

| Steps | Average execution time (sec) | Standard deviation (sec) |
|------------------|------------------------------|--------------------------|
| read | 136.8 | 5.47 |
| process | 172.8 | 7.45 |
| update | 168.6 | 7.37 |
| visualize | 126.5 | 4.78 |
| Total | 604.7 | 25.07 |

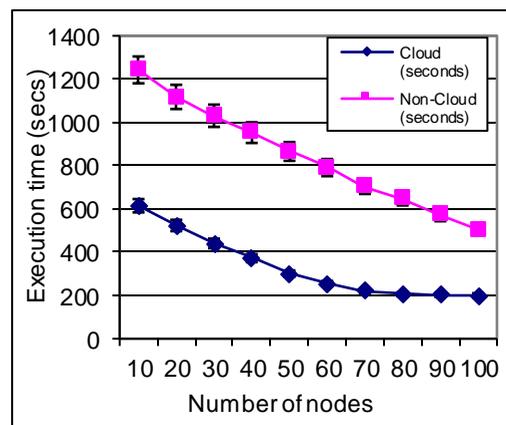


Figure 18: Cloud and non-Cloud data processing 200 GB of weather data without optimization.

Figure 18 shows the results between Cloud and non-Cloud of processing 200 GB weather data without optimization. Similar to results in Section 4.4, the Cloud

has a better performance with less execution time at all times. There was a rapid fall of execution time when the node has increased from 10 to 70 for Cloud and from 10 to 100 for non-Cloud. There were small differences in execution time when the node has increased from 70 to 100, whereby the Cloud using four-step MapReduce approach has taken 200.1 seconds and non-Cloud has taken 544.2 seconds with 100 nodes.

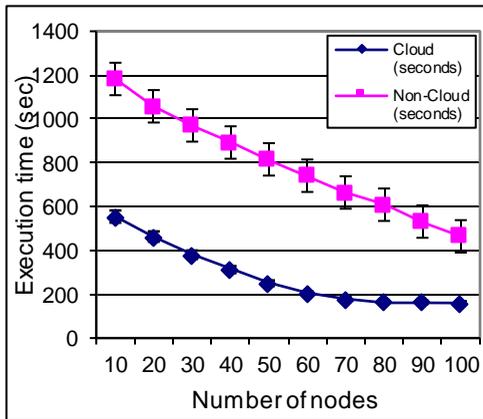


Figure 19: Cloud and non-Cloud data processing 200 GB of weather data with optimization

Figure 19 shows results between Cloud and non-Cloud of processing 200 GB weather data with optimization. Execution time for Cloud had an average of 50 seconds lower than non-optimization until the number of nodes reached 70. Execution time had an average of 40 seconds lower than non-optimization between 80 and 100 nodes. For non-Cloud, execution time had an average of 100 seconds lower than non-optimization until the number of nodes reached 70, and then had an average of 80 seconds lower between 80 and 100 nodes. Results in Figure 18 and 19 show satisfactory performance evaluation by using our algorithms to optimize processing, analysis and visualization of our data, which is our major contribution.

5.5.2 Performance for eight-step approach without/with optimization in Section 5.2

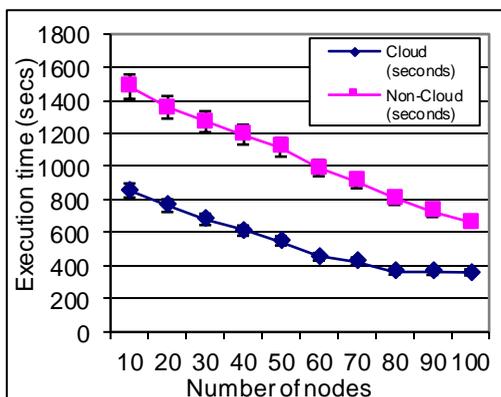


Figure 20: Cloud and non-Cloud data processing 200 GB of weather data with optimization

200 GB of weather data for visualization with optimization

This section explains performance evaluation for eight-step process (optimization as an optional step) for analyzing and visualizing data in Section 5.2, with the same set-up as in Section 5.1. Experiments were performed five times to take the mean values for execution time. Figure 20 shows the execution time to perform visualization in Section 5.3 and 5.4 without optimization. Results show that Cloud has the lower execution time and the difference between Cloud and non-Cloud gets smaller when the number of nodes has increased.

Execution time gets longer than results in Figure 18 and 19 due to more steps involved to perform visualization of data. Figure 21 shows execution time without using optimization step. All the execution time was between 20-30% higher than Figure 20. The impact on execution time was much reduced when the number of nodes was 60 and above for Cloud and non-Cloud kept on going down further for the increased number of nodes. In summary, experiments in both Section 5.1 and 5.2 demonstrate our research contributions to process, analyze and visualize large data in an acceptable amount of time. Optimization developed by our proposed work can accelerate performance with an improvement between 20% and 30% under 60 nodes.

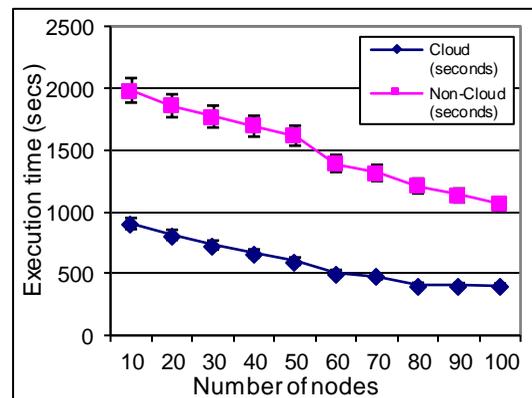


Figure 21: Cloud and non-Cloud data processing 200 GB of weather data for visualization without optimization

6. Discussion: Comparison with other methods

This section presents comparison with other methods. The distinctive advantages include the rapid computational time and acceptable accuracy without the reliance on CPU-intensive infrastructure. We explain the background theory and the models used for forecasting. We describe the technologies used to support weather science, including the system design, deployment, platform architecture and two services on offer. We present three case studies: Sydney, Singapore and London to support the validity of our approach. We use the historical data between January 2010 and March 2012, and between January 2011 and March 2013 to compute the forecast temperatures between January 2012 and March

2014. We can explain that all case studies can support our hypotheses and demonstrations.

We also illustrate the use of data visualization to help analyze and present the temperature in extreme weather conditions, a step beyond current Data Science since our work has applied to weather science investigating extreme weathers. We have the US case study, which has five chosen dates of collecting data and present the results. We have the UK case study for two selected dates and their results. The emphasis is to present the performance of weather forecasting and data visualization to demonstrate that large data can be processed with better performances than the non-Cloud approach. The results from all the experiments could justify our contributions. The five-step MapReduce approach is designed to offer computation and visualization of the US and UK weather data. The pipeline method allows us to use private cloud to process up to 2 terabytes of data each time for weather science free of charge, which can save us money in data processing in the public cloud. We demonstrate a cost-effective way for weather science.

We also compare our approach with other methods in this section. First, Campbell and Diebold [18] present their weather derivatives. As discussed in Section 2, they do not explain how the financial derivatives can be used to temperature forecasting. The examples in their papers are only based in America and it is useful to try examples outside the America to see whether their models can be applicable to different parts of the world for weather forecasting. Second, both Droegemeier et al. [10] and Plale et al. [19] use the multi-million LEAD systems as discussed in Section 2. Such a system is expensive and not every scientist can afford to have. There are insufficient technical details except the use of XML, Meteorological command and control and Blackboard, including details in computational time, accuracy and results of other weather forecasting data. While there are few literature addressing how to use Cloud Computing to perform weather modeling and data processing, Villegas et al [49] explain their approach using Cloud Computing with their rationale and architectures. However, they only focus on Infrastructure as a Service and only measure the execution time when more CPUs are included. Gao et al [22] develop MapReduce algorithms on top of Hadoop and process the geo-data from their data nodes. They present their datapoints on the map of the USA to show their focus of research for each data node. The difference is they focus on displaying parks, schools, museums, coffee shops, major roads and rivers, we focus on displaying temperature distribution across the map of the USA and UK. Shekhar et al. [50] present a similar but simpler example like in [22]. They use tables to represent maps and demonstrate how to work out between tables and datapoints in the street map. More experimental results should be discussed. Our approach has explained the rationale, formulas, system architecture, MapReduce framework, results and interpretations.

7. Conclusion

This paper demonstrates an innovative system and application data science for weather Cloud Computing, integrating both system and application dependability to investigate weathers. It started with the problems in existing weather computing and practices, followed by system design, formulation for forecasting, MapReduce for weather forecasting and visualization, performance, algorithms for visualization, results and analysis. The architecture is a state-of-the-art platform to use parallel MapReduce data processing the private cloud. We explained the forecasting techniques used and presented three case studies to support. The previous temperatures between January 2010 and March 2010 were used to forecast the temperatures between January 2012 and March 2014 for Sydney, Singapore and London. We demonstrated that data visualization could be used for extreme weathers. Algorithms associated with data visualization have been explained, including its functions, code design and the steps corresponding to MapReduce Framework. Comparison between Cloud and non-Cloud approach had been explained, including the platform that could perform fair Cloud and non-Cloud experiments. Results supported there was a better performance in Cloud. In order to optimize performance for visualization, we presented four-step MapReduce algorithms and explained the function in each step with its code design. We showed execution time required by each step. We compared data processing performance between Cloud and non-Cloud which showed that Cloud had a better performance LEAD between 10 and 100 nodes.

Data visualization was used to study the temperatures in the US for their polar vortex period. Four periods before, during, and the end of the polar vortex were chosen for analysis. The temperatures in the UK during and after the flood were also investigated and discussed. Data visualization can help the general public to understand temperature distribution easily and the importance of the weather science. Experiments related to the performance of weather computing and data visualization were undertaken and all execution time could be completed within seconds. Our research contributions had been explained. The two major cases of weather computing demonstrated in this paper can support our contributions to make weather science research to be adaptable and affordable for scientists.

We demonstrate our proofs-of-concept to support the development of weather science and ensure three goals can be met. First, we aim to provide a forecasting method to show that temperatures in Sydney, Singapore and London can be forecasted based on our proposed solution. Second, data processing and analysis can be performed by our proposed five-step MapReduce with a shorter execution time in the Cloud-based solution. Third, data visualization can be achieved with the eight-step process with/without optimization to ensure that data can be analyzed, mapped to the right lo-

cations and visualized with different color codes indicating temperature ranges. Additionally, the implementation can be affordable, results of data processing and analysis have been validated by good performance evaluation.

Our research contributes to 5 V's in Data Science. Our work is relevant to volume, since 20 GB, 40 GB and 200 GB of data in each instance can be processed and analyzed from time to time with a low execution time. Our work is relevant to velocity since we can extract and update our weather data on hourly basis which can be ready for further analysis and investigation. Our work supports variety since different formats of data can work together. Veracity has been demonstrated since the level of accuracy is high supported by our results and analysis. Our work creates value since we can help investigate extreme weathers and inform the general public about the impacts to their schedules due to the sudden change of the weathers.

References

- [1] H., Chen, R. H., Chiang, & V. C., Storey (2012). Business Intelligence and Analytics: From Big Data to Big Impact. *MIS quarterly*, 36(4), 1165-1188.
- [2] V., Mayer-Schönberger, & K., Cukier (2013). Big data: A revolution that will transform how we live, work, and think. Houghton Mifflin Harcourt.
- [3] V., Chang, V., & G., Wills, A model to compare cloud and non-cloud storage of Big Data. *Future Generation Computer Systems*, (2016).
- [4] A., Sutcliffe, User-centred requirements engineering. Springer Science & Business Media, (2012).
- [5] R., Biddle, S., Marshall, J., Miller-Williams, E., Tempero, Reuse of debuggers for visualization of reuse. In *Proceedings of the 1999 ACM symposium on Software reusability*, 1999, May, pp. 92-100.
- [6] A. J., Ko, R., Abraham, L., Beckwith, A., Blackwell, M., Burnett, M., Erwig, ... & S., Wiedenbeck. The state of the art in end-user software engineering. *ACM Computing Surveys (CSUR)*, (2011), 43(3), 21.
- [7] J., Nichols, B. A., Myers, M., Higgins, J., Hughes, T. K., Harris, R., Rosenfeld, M., Pignol, Generating remote control interfaces for complex appliances. In *Proceedings of the 15th annual ACM symposium on User interface software and technology*, 2002, October, pp. 161-170.
- [8] V., Chang, Business Intelligence as a Service, *Future Generation Computer Systems*, 37, (2014), pp 512-534.
- [9] T., Gunarathne, C., Herath, E., Chinthaka, S., Marru, Experience with adapting a WS-BPEL runtime for eScience workflows. In *ACM Proceedings of the 5th Grid Computing Environments Workshop*, 2009, November.
- [10] K. K., Droegemeier, K., Brewster, M., Xue, D., Webber, D., Gannon, B., Plale, J., D., Reed, L., Ramarrishnan, J., Alameda, R., Wilhelmson, T., Baltzer, B., Domenico, D., Murray, R., Clark, S., Yalda, S., Graves, R., Ramachandran, J., Rushing, E., Joseph, Service-oriented environments for dynamically interacting with mesoscale weather. *Computing in Science & Engineering*, 7(6), 12-29, 2005.
- [11] J., Slingo, S., Belcher, A., Scaife, M., MacCarthy, A., Saulter, K., McBeach, A., Jenkins, C., Huntingford, T., March, J., Harnford, S., Parry (2014), The Recent Storms and Floods in the UK, Met Office, white paper, Feb.
- [12] Van Hinsber, G., China weather in December, China High-lights, accessible on <http://www.chinahighlights.com/weather/december.htm>, edited on March 6, 2016.
- [13] M., Boykoff, Flogging a dead norm? Newspaper coverage of anthropogenic climate change in the United States and United Kingdom from 2003 to 2006, *Journal compilation © Royal Geographical Society*, 39 (2), 2007.
- [14] D. P., Loucks, E., van Beek, Water Resources Systems Planning and Management, Facts about Water, white paper, 2005.
- [15] A., Shah, Climate Change and Global Warming Introduction, *Global Issues*, Nov issue, 2013.
- [16] M. T., Chahine, T. S., Pagano, H. H., Aumann, R., Atlas, C., Barnet, J., Blaisdell et al., AIRS: Improving weather forecasting and providing new data on greenhouse gases. *Bulletin of the American Meteorological Society*, 87(7), 2006.
- [17] M., Xue, D., Wang, J., Gao, K., Brewster, & K. K., Droegemeier, K. K. "The Advanced Regional Prediction System (ARPS), storm-scale numerical weather prediction and data assimilation". *Meteorology and Atmospheric Physics*, 82(1), 139-170, 2003.
- [18] S. D., Campbell, F. X., Diebold, Weather forecasting for weather derivatives, *Journal of the American Statistical Association*, 100 (469), 2005.
- [19] B., Plale, D., Gannon, J., Brotzge, K., Droegemeier, J., Kurose, D., McLaughlin, R., Wilhelmson, S., Graves, M., Ramamurthy, R. D., Clark, S., Yalda, D. A., Reed, E., Joseph and V., Chandrasekar, Casa and lead: Adaptive cyberinfrastructure for real-time multiscale weather forecasting. *IEEE Computer*, 39(11), 56-64, 2006.
- [20] J., Li, M., Humphrey, D., Agarwal, K., Jackson, C., van Ingen, Y., Ryu. escience in the cloud: A modis satellite data reprojection and reduction pipeline in the windows azure platform. In *2010 IEEE International Symposium on Parallel & Distributed Processing (IPDPS)*, pp. 1-10, 2010, April.
- [21] H., Demirkan, & D., Delen. (2013). Leveraging the capabilities of service-oriented decision support systems: Putting analytics and big data in cloud. *Decision Support Systems*, 55(1), 412-421.
- [22] S., Gao, L., Li, W., Li, K., Janowicz, & Y., Zhang, (2014). Constructing gazetteers from volunteered big geo-data based on Hadoop. *Computers, Environment and Urban Systems*.
- [23] National Weather Service, <http://www.nws.noaa.gov> accessible on March 5, 2016.
- [24] The Met Office, accessible from <http://www.metoffice.gov.uk/climate/uk/stationdata/> accessed on March 5, 2016.
- [25] BBC Weather <http://www.bbc.co.uk/weather/> accessed on March 5, 2016.
- [26] R., Wolski, N. T., Spring, J., Hayes, The network weather service: a distributed resource performance forecasting service for metacomputing, *Future Generation Computer Systems*, 15(5), 757-768, 1999.
- [27] R. E., Mayagoitia, A. V., Nene, P. H., Veltink, Accelerometer and rate gyroscope measurement of kinematics: an inexpensive alternative to optical motion analysis systems. *Journal of biomechanics*, 35(4), 537-542, 2002.
- [28] L., Yang, I., Foster, J. M., Schopf, Homeostatic and tendency-

- based CPU load prediction", In Proceedings, the IEEE International Parallel and Distributed Processing Symposium, Apr, pp. 9-pp, 2002.
- [29] R. A., Haddad, T. W., Parsons, Digital signal processing: theory, applications, and hardware. Computer Science Press, Inc, 1991.
- [30] T. J., Ulrych, T. N., Bishop, Maximum entropy spectral analysis and autoregressive decomposition, *Reviews of Geophysics*, 13(1), 183-200, 1975.
- [31] R., Wolski, Forecasting network performance to support dynamic scheduling using the network weather service. In Proceedings, The Sixth IEEE International Symposium on High Performance Distributed Computing, Aug, pp. 316-325, 1997.
- [32] C. Ribeiro, N., Webber, Valuing Path Dependent Options in the Variance-Gamma Model by Monte Carlo with a Gamma Bridge, Working Paper, No.02-04, Warwick Business School, September 2002.
- [33] P., Carr, H., Geman, D., Madan, M., Yor, The fine structure of asset returns: An empirical investigation, *Journal of Business*, 75 (2), 305-332, 2002.
- [34] D., Brigo, A., Dalessandro, M., Neugebauer, F., Triki, A Stochastic Processes Toolkit for Risk Management, Working Paper, King's College London, November 2007.
- [35] H. G., Zimmermann, R., Neuneier, R., Grothmann, Active Portfolio-Management based on Error Correction Neural Networks, Working Paper, Siemens AG, 2006.
- [36] D. B., Madan, P. P. Cam, E. C. Chang. 1998. The variance gamma process and option pricing. *European Finance Review* 2:79-105.
- [37] A. N., Avramidis, P., L'Ecuyer, P. A., Tremblay, Efficient simulation of gamma and variance-gamma processes. In IEEE Proceedings of the 2003 Winter Simulation Conference, 2003, Vol. 1, pp. 319-326.
- [38] A. N., Avramidis, P., L'Ecuyer, P. A., Tremblay, Efficient simulation of gamma and variance-gamma processes. In IEEE Proceedings of the 2003 Winter Simulation Conference, 2003, Vol. 1, pp. 319-326.
- [39] J., Veiga, R. R., Expósito, G. L., Taboada, & J., Touriño. (2016). Flame-MR: An event-driven architecture for MapReduce applications. *Future Generation Computer Systems*, 65, 46-56.
- [40] M., De Kruijf, K., Sankaralingam, MapReduce for the cell broadband engine architecture. *IBM Journal of Research and Development*, 53(5), 10-1, 2009.
- [41] C., Hoffa, G., Mehta, T., Freeman, E., Deelman, K., Keahey, B., Berriman, J., Good, On the use of cloud computing for scientific workflows. In IEEE Fourth International Conference on eScience, eScience'08., Dec, pp. 640-645, 2008.
- [42] B., Sotomayor, R. S., Montero, I., Martín Llorente, I. Foster, Resource leasing and the art of suspending virtual machines, In the 11th IEEE International Conference on High Performance Computing and Communications, HPCC'09, June, pp. 59-68, 2009.
- [43] V., Chang, Y. H., Kuo, & M., Ramachandran. (2016). Cloud computing adoption framework: A security framework for business clouds. *Future Generation Computer Systems*, 57, 24-41.
- [44] H. C., Yang, A., Dasdan, R. L., Hsiao, D. S., Parker, Map-reduce-merge: simplified relational data processing on large clusters, In Proceedings of the 2007 ACM SIGMOD international conference on Management of data, June, pp. 1029-1040, 2007.
- [45] J., Huang, X., Ouyang, J., Jose, M., Wasi-ur-Rahman, H., Wang, M., Luo, ... & D. K., Panda. High-performance design of hbase with rdma over infiniband. In 2012 IEEE 26th International on Parallel & Distributed Processing Symposium (IPDPS), pp. 774-785, 2012, May.
- [46] A. S., Balkir, I., Foster, A., Rzhetsky. A distributed look-up architecture for text mining applications using MapReduce. In 2011 IEEE International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1-11, 2011, November.
- [47] R., Farivar, D., Rebolledo, E., Chan, R. H., Campbell, A Parallel Implementation of K-Means Clustering on GPUs, In PDPTA Conference, July, pp. 340-345, 2008.
- [48] V., Chang, & G., Wills. (2016). A model to compare cloud and non-cloud storage of Big Data. *Future Generation Computer Systems*, 57, 56-76.
- [49] D., Villegas, N., Bobroff, L., Rodero, J., Delgado, Y., Liu, A., Devarakonda, L., Fongc, S. M., Sadjadi & M., Parashar, Cloud federation in a layered service model. *Journal of Computer and System Sciences*, 78(5), 1330-1344, 2012.
- [50] S., Shekhar, V., Gunturi, M. R., Evans & K., Yang. (2012, May). Spatial big-data challenges intersecting mobility and cloud computing. In Proceedings of the Eleventh ACM International Workshop on Data Engineering for Wireless and Mobile Access, pp. 1-6.

Biography of the author

Dr. Victor Chang is an Associate Professor at IBSS, Xi'an Jiaotong Liverpool University, Suzhou, China. He was a Senior Lecturer at Leeds Beckett University. He helps organizations in achieving good Cloud design, deployment and services. He has more than 100 peer-reviewed publications up-to-date. He won a European Award on Cloud Migration in 2011, a best paper in 2012 and in 2015, Best Project in Research in 2016. With 16 years of IT experience, he is an experienced practitioner and scientist in Cloud Computing and Big Data in England. He is the founding chair of two international workshops which have been described as the best workshops by organizers and participants. He is the founding chair of IoTBD 2016 www.iotbd.org and COMPLEXIS 2016 www.complexis.org. He is an Editor-in-Chief of IJOI and OJBD journals. He's an Editor of Future Generation Computer Systems (FGCS). He has four books on Cloud Computing. He won another European award in 2016, Best Project in Research.