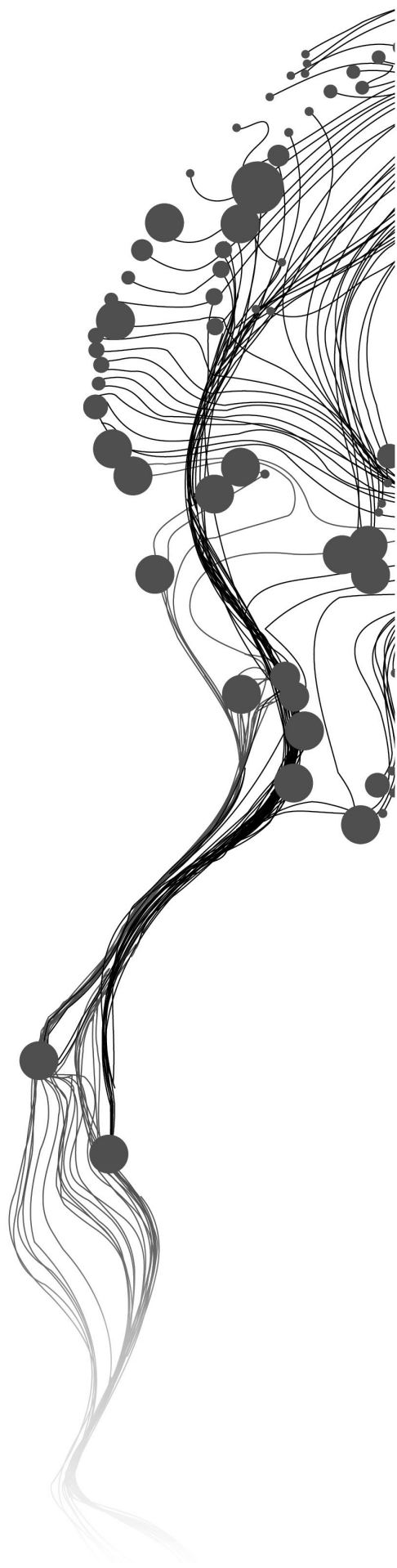


ADAPTING SPRING INDEX MODEL TO THE CLOUD: A CASE OF GOOGLE EARTH ENGINE

RIDDHI MUNDE
March, 2015

SUPERVISORS:

Dr. Frank Ostermann
Dr. Raul Zurita-Milla



ADAPTING SPRING INDEX MODEL TO THE CLOUD: A CASE OF GOOGLE EARTH ENGINE

RIDDHI MUNDE

Enschede, The Netherlands, March, 2015

Thesis submitted to the Faculty of Geo-information Science and Earth
Observation of the University of Twente in partial fulfilment of the requirements
for the degree of Master of Science in Geo-information Science and Earth
Observation.

Specialization: GFM

SUPERVISORS:

Dr. Frank Ostermann
Dr. Raul Zurita-Milla

THESIS ASSESSMENT BOARD:

Prof. Dr. Menno-Jan Kraak (chair)
Ir. Edward Verbree, Delft University of Technology, Netherlands

Disclaimer

This document describes work undertaken as part of a programme of study at the Faculty of Geo-information Science and Earth Observation of the University of Twente. All views and opinions expressed therein remain the sole responsibility of the author, and do not necessarily represent those of the Faculty.

ABSTRACT

The research explores the prospects of cloud computing approach for adapting the Spring Index model on a continental scale. The phenomenon of first day of occurrence of leaf gives information about climate change when studied over a large spatial and temporal scale. However, studies over large scales are constrained by lack of adequate technology to process large volume of data. Here, cloud computing can help bridge the gap created due to inadequate computing power. Google Earth Engine(GEE) , a cloud computing platform for geo spatial analysis, is taken as a case for implementation.

The Spring Index model predicts the first day of leaf based on temperature and longitudinal data at a given location. The model when computed over a large gridded dataset, gives prediction results for a wider spatial scale. In addition, the model computed over data from decades gives a rich source of results for analysis that may further be used to identify peculiar patterns in occurrence of phenomenon of first day of leaf. To achieve the aim, the SI model was adapted from a desktop-based model to a cloud-computing platform of GEE. The results of the GEE-based SI-x model were analysed to find decadal trend in the occurrence of phenomenon. Further, the gridded form of predicted results was correlated with point location form of volunteered data.

The key outcome of the research includes a re-designed, generic algorithm that enables SI-x model adaptation from desktop paradigm to the cloud platform, GEE. The SI-x model in GEE was successfully run on a continental scale for USA. The results were obtained in less than a minute after computations over 100 million pixels. The implementation demonstrates the feasibility of cloud computing approach, though, with a further scope for optimization. Also, the correlation study seem to indicate that the scale can be an influential factor when correlating gridded form with point form of data. Hence, a careful consideration is needed while choosing a scale, again, this requires further study for conclusive results.

Keywords

Cloud computing, Google Earth Engine, Spring index model,Daymet

ACKNOWLEDGEMENTS

This research project would not have been possible without the support of many people. I wish to express my gratitude to Dr. Frank Ostermann and Dr.Raul Zurita-Milla for their guidance, support and motivation at every stage of my research.

I would like to acknowledge the contributions from the Google Earth Engine developer team for all the help during the development work related to Google Earth Engine. I also acknowledge the contributions and work of Ms.Emma Izquierdo-Verdiguier in taking my work ahead.

Special thanks also to me sister Siddhi and my dear friends- Eduardo, Manuel, Gustavo, Leila, Amy, Amie and many others for all the support and lovely memories.

I wish to express my love and gratitude to my beloved families..They are always a pillar of support and giving love.Especially, to my father for being a true mentor of my life.

Last but not the least, my loving husband,Arun, for all the love and care that kept me motivated throughout.

TABLE OF CONTENTS

Abstract	i
Acknowledgements	ii
1 Introduction	1
1.1 Motivation	1
1.2 Research objectives and questions	3
1.3 Thesis outline	3
2 Related work	5
2.1 Study of onset of spring	5
2.2 Trends in Onset of Spring from literature	5
2.3 Cloud computing	6
2.4 Google Earth Engine	7
2.5 Map reduce concept	8
3 Datasets	11
3.1 Datasets for GEE-based SI-X model	11
3.1.1 Daymet temperature data	11
3.1.2 Volunteered data	12
3.2 Pre-processing of data	12
4 Adaptation of SI-x model to cloud computing	15
4.1 Conventional SI-x model design	15
4.2 Conceptual design for cloud based SI-x	16
4.3 Methodology for correlation	20
5 Implementation of SI-x in GEE	21
5.1 Results from GEE based SI-x model implementation	21
5.1.1 Results from time series data from area of interest	23
5.1.2 Results from workload vs. time analysis	23
5.2 Results of Correlation with volunteered data and its interpretation	25
5.3 Further work	26
6 Discussion	27
6.1 Conceptual design of cloud based SI-x	27
6.2 Google Earth Engine implementation	27
6.2.1 Discussion on the SI-x model	29
6.3 Correlation of point and gridded data	29
7 Conclusion and Recommendation	31
7.1 Conclusion	31
7.2 Recommendations	32

References	33
A Appendix-A	37

LIST OF FIGURES

2.1	Concept of Map reduce framework	8
3.1	Sample of Daymet dataset of average daily maximum temperature for September,1989. Source: Retrieved from http://daymet.ornl.gov	12
3.2	Steps involved in pre-processing of the historical Lilac data	13
3.3	Merged table for station locations and observation data for Lilac in Google Fusion table.	13
3.4	Visualisation of the observations for the first day of leaf of Lilac species from 1980-2003	14
4.1	Block diagram of a typical workflow of the SI-x in MATLAB	16
4.2	Workflow of algorithm to implement SI-x in cloud computing.	17
5.1	Screenshot from Google Earth Engine visualization panel. The map shows occurrence of spring for the year 2000 starting from Day 1. Each pixel refers to first day of leaf. The white colour indicates the early days of year while greener colours indicate later days of the year.	21
5.2	Screenshot from Google Earth Engine visualization panel. The map shows occurrence of spring for the year 2005 starting from Day 1. Each pixel refers to first day of leaf. The white color indicates the early days of year while greener colors indicate later days of the year.	22
5.3	Screenshot from Google Earth Engine visualization panel. The map shows occurrence of spring for year 2010 starting from Day 1.	22
5.4	Screenshot from Google Earth Engine visualization panel. The map shows occurrence of spring for year 2013 starting from Day 1.	22
5.5	Screenshot for Year 2000 with the Area of Interest (AOI) for time-series analysis. The co-ordinates for polygon as ((-86,31), (-83,33)).	23
5.6	Graph of mean day of year values for each year starting from 1980 until 2013. . .	24
5.7	Graph for number of images v.s time for each block in SI-x in GEE.	24
5.8	The above figure shows correlation between number of images and time taken to process the functions in Block 3 and 4 of SI-x in GEE.	24
5.9	Correlation of volunteered data with SI-x model results from GEE. The SI-x model results were obtained at 1km and 20km scale.	25
5.10	Implementation of SI-x model in GEE for 190 days for the year 2000	26

Chapter 1

Introduction

1.1 MOTIVATION

Phenology can be related to climate change as the study of the life cycle of plants and animals. This lifecycle consists of pheno-phases which occur with changes in the season. The occurrence of a pheno-phase can be related to start of a particular season. An example of such a phenomenon is the pheno-phase in plants for the first appearance of a leaf denoted as “first day of leaf” and further generalised as first day of spring or onset of spring. Monitoring “first day of leaf” over long periods of time suggests the tendency of occurrence of spring season (Menzel, 2000). Such studies, when implemented in large space and time, can characterize the decadal variability in occurrence of a season (Cleland et al., 2007). In the end, these studies contribute to the study and the identification of the effects of climate change.

There are several approaches towards the prediction of the start of spring. Some studies have used satellite imagery to identify seasonal changes in vegetation (Stöckli & Vidale, 2004). Vegetation indices are fair indicators of the changes occurring in the phenological behaviour of plants. Besides, several historical phenology networks have collected ground observations of different pheno-phases in plants and animals. At present, such a second source of information has led to the emergence of publicly accessible geographic information over the web as part of the phenology networks. Since this form of gathering information is on a volunteer basis, it is termed as Volunteered Geographic Information (Goodchild, 2007). In future references, this data will be referred to as the ‘volunteered data’. The statistical analysis over volunteered data for first day of leaf predicts the trends in occurrence of spring.

However, there are limitations to the approaches described above that are necessary to be taken into consideration before performing any further analysis. The volunteered data provides a poor spatial extent and it varies due to irregular time span in the process of data collection. Although, the satellite data covers up for the drawbacks of poor spatial and temporal aspects of the volunteered data, spatial scales and data availability still remains as a matter of choice of sensors. The limitations can be removed by the third approach of building predictive models for start of spring.

A predictive model, the Spring Index model (SI model) predicts first day of leaf and bloom based on temperature data (Schwartz, 1997). The SI model is an example of the models that have been implemented on a large spatial and temporal scale. The original SI model was previously limited to temperate regions. In these regions, chilling temperature was an important factor in the predictive model. However, the chilling temperature was excluded to widen scope to the model to the sub-tropical areas. Hence, an extended version of SI model that is the SI-x, was developed considering maximum and minimum temperatures as input for the prediction model (Schwartz, Ault, & Betancourt, 2013). Further, the model was validated using the volunteered data. This research is based on the version of SI-x model.

Implementation of the SI-x on a continental scale needs storage and processing of large datasets. Currently, the SI-x predicts the first day of leaf using temperature data from the weather stations. These predictions are made for one location at a time on a standalone desktop. Therefore, with

limited processing power, the time required to process all the stations is enormous. Further, model results are correlated with volunteered data. These are characterised by a broad variety of formats (vector, raster) and structure. With the new phenology networks using web platforms for data collection, the data is in large quantity (Schwartz et al., 2012). The advancements in technology have increased the amount of data available and new technologies are required to fully take advantage of the collected information.

A current trend in GIS is to utilise high computing technologies for handling and processing data (Aysan et al., 2011). Yang et al. (2010) propose to “bring people, information and computational tools together” in GIS. The needs in GIS exists in computing power for processing large data volumes, data access for wider audience and low cost software. Xiaoqiang and Yuejin (2010) suggests cloud computing may contribute to improve on this state of affairs. Cloud computing is defined as “a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, application and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction” (NIST, 2011). Storing, processing and visualization tasks are achieved by using the computing resources that the ‘cloud’ provides (Hu et al., 2012). Companies such as IBM, Microsoft, Google, and Amazon, offer cloud computing services.

GIS applications have used cloud computing platforms in various contexts. Mattmann et al. (2013) used a cloud computing environment to pre-process satellite imagery for monitoring the glaciers, and integrate this data into regional climate models. The Hadoop cloud computing platform was used to extract location-based information from volunteered data based on the Map reduce framework (Gao et al., 2014). Further, Hansen et al. (2013) have used the Google Earth Engine (GEE) cloud computing platform to extend the computational power to process a number of Landsat imageries for development of global forest change maps. These examples set a base line on future works aiming to take advantage of cloud computing services.

Google Earth Engine is a cloud computing platform for processing satellite imagery and other observation data. It provides access to a large warehouse of satellite imagery and the computational power needed to analyse those images. GEE was chosen for the following reasons: a) From the beginning of research, GEE was the available option b) GEE has been used within GIS for geospatial analysis on various dataset. For example, Hansen et al. (2013) used GEE as a tool to conduct time series analysis of forest change for the entire world, from 1950 until present year. A task that required a dedicated data handling system and thousands of servers for data processing.

Climate change is a large scale phenomenon. Therefore, this phenomenon needs to be studied over large areas such as a continental or global scale. This ascertains that studies will involve large amounts of data and on equally large number of processes. Implementation of SI-x currently faces the problem of inadequate technology for processing over scales. Therefore, cloud computing is capable of filling the technological gap to enable the SI-x predict different the onset of spring over a large spatial and temporal scale. The result of this implementation can further help understand the occurrence of phenomenon of spring which maybe not evident from the previous implementations.

This research focuses on the deployment of the SI-x in a cloud computing platform by adapting a desktop application to a cloud computing platform. The ultimate aim is to use cloud computing resources to process and predict the onset of spring (the result of the SI-x model) for grid location over continental USA in a time-series. These results could then be further analysed to identify trends in the spatial and temporal context. Within this context, this study reports the initial efforts through which the SI model has been adapted to a cloud computing based implementation and provides examples of the outputs and possible correlation strategies with volunteered data.

1.2 RESEARCH OBJECTIVES AND QUESTIONS

The aim of this research is to explore the feasibility of a cloud computing platform for the computation of data intensive tasks of the prediction of the onset of spring. Cloud computing is used to implement SI-x and produce results at a continental scale.

1. To review cloud computing as computational and data-driven technology for prediction of onset of spring.
 - (a) What are the requirements for the implementation of the SI-x model in a cloud computing platform ?
 - (b) What services are provided by cloud platform relevant for the implementation?
2. To create a prototype of a cloud-based SI model computable on a continental scale.
 - (a) How to adapt the SI-x model on Google Earth Engine (GEE) platform for computation on continental scale?
 - (b) How do the SI-x model outputs vary across spatial-temporal dimensions when implemented using cloud?
 - (c) What are the benefits and challenges of using the GEE platform for SI model computation?
3. To compare SI model results with the observed day of first leaf and bloom.
 - (a) Which are the suitable methods for comparison between a gridded model output and point observation data?
 - (b) How does the spatial scale factor influence the correlation with the observed data?

1.3 THESIS OUTLINE

The remaining chapters explain in detail the outcomes of this research. Chapter 2, summarizes relevant related work on phenology, prediction models, and cloud computing. Chapter 3 gives the description of the datasets used in this study. Chapter 4 explains the conceptual workflow of Si-x model and its adaptation to the cloud based platform. The results and outcome related to the research are mentioned in Chapter 5. Detailed discussion of the results corresponding to the research objectives is given in Chapter 6. Chapter 7 answers the research questions and recommends future work for this study.

Chapter 2

Related work

2.1 STUDY OF ONSET OF SPRING

Phenology studies have used satellite imagery to identify seasonal changes in vegetation. Several sensors are used to study vegetation indices such as Normalized Difference Vegetation Index (NDVI), Enhanced Vegetation Index (EVI) and Leaf Area Index (LAI). For instance, Some studies used AVHRR sensor data to identify the start of spring (Schwartz, Reed, & White, 2002; A. White & Thomson, 1997). Others used MODIS imagery for the same (Li, Qu, & Hao, 2010; Zhang et al., 2003). The satellite studies provide data for large spatial coverage and regular temporal scale. These make it possible to perform global scale studies. Though satellite imagery may be collected regularly, post-processing the data can lead to irregular temporal data available for studies. However, there are other approaches to onset of spring.

Volunteered data also provide a data source for determining information about start of spring. Volunteered data is complex in nature. Volunteered data include observations regarding any phenomenon or event collected by volunteers or observer groups for research. Schwartz (1994) identified lot of missing data for various years. This was mainly due to non-continuous nature of collection of data. Therefore, the model was used to fill up missing information from the phenology volunteered data. Schwartz (1997) developed the first spring index model (SI model) to predict first day of leaf and bloom. The model calculates growing degree day i.e. the sum of hourly temperatures in daytime and chilly temperatures. A new model extended Spring index model (SI-x) was proposed to calculate the first day of leaf and bloom based on maximum and minimum temperatures (Schwartz, Ault, & Betancourt, 2013). SI-x model is a “suite of metrics” of sub-models based on three plant species namely, Lilac(*Syringa vulgaris* and *Syringa chinesis*), Honeysuckle(*Lonicera tatarica*) and Zabeli(*L.korolkowii*) (Schwartz, Ault, & Betancourt, 2013). Each sub-model is calibrated using the co-efficient of respective plant species. The SI-x is a regression based model using four predictor variables. Further, the predicted results from the model were validated using volunteered data.

Comparison of results between satellite and volunteered data or SI model with volunteered data are undertaken(H. Wang et al., 2014). Spring index model results have been correlated with ground observation data in previous studies (M. A. White et al., 2009). Previous studies have some common statistical methods used for correlation of data (Schwartz et al., 2002). In this study, descriptive statistics such as maximum, minimum, mean and standard deviation were used. Another commonly used statistical method is Pearson’s correlation. These methods are applicable to vector form. However, comparison of two different formats i.e. vector and a raster is rare. In this situation, either the vector needs to be converted to raster or from raster to vector.

2.2 TRENDS IN ONSET OF SPRING FROM LITERATURE

Various studies conclude occurrence of an early spring (Menzel, 2000). Similar trend has been identified from phenological studies in Europe (Ahas et al., 2002; Studer et al., 2005). Studies

from North America show some distinctive regional patterns in spring occurrence (Schwartz & Reiter, 2000). Schwartz, Hanes, and Liang (2013) observed anomalies in onset of spring especially in South-East of USA. According to the author, topography and strategic location could be reason for delayed spring. The observed trend in this region contradicts to the overall trend observed in the rest of USA. Parmesan and Yohe (2002) in their multi-specie phenological studies conclude an early arrival of spring. But, the authors argue whether it is just a co-incidence since, several factors other than climate affect biological changes at micro and macro levels. However, Sagarin (2001) contradicts the early spring especially in long term variability analysis.

Li et al. (2010) identified factors affecting onset of spring, in USA. The first factor is latitude, which shows a onset pattern from lower to higher latitudes with greatest change from 42°N and 44°N. The second factor, altitude, appeared when observations showed, an early occurrence of spring in coastal regions, compared with later occurrence in mountainous regions. Although, altitude as an influential factor remains a polemic topic (Parmesan, 2007). Studer et al. (2005) finds that temperature can be driving factor for spring occurrence in a temperate ecosystem, however for high altitude regions; there are several factors which need to be further studied in relation to determining the start of spring. T. Wang et al. (2014) found that the arrival of spring is affected by large variance in temperature at local level.

2.3 CLOUD COMPUTING

With recent advances in technology, volume and variety of structure and unstructured data increases rapidly. Various data mining and analysis techniques are being used to extract information from the data (Gao et al., 2014). At the same time, processing such data requires powerful hardwares and high performance computing technologies. The requirements can be addressed by state-of-the-art computing technologies e.g. grid, cluster and cloud computing. The technologies are based on distributed architecture and on same concept of interconnected network of servers for performing computations. However, the characteristic difference lies in their deployment architecture (Buyya et al., 2009). These technologies have seen applications in GIS.

Grid computing has applications in the field of GIS (Petcu et al., 2012). Foster et al. (2008) give a comparative study of cloud and grid computing. P. Wang et al. (2013) tested the performance of computing an image processing algorithm over grid computing, cloud computing and standalone desktop. The grid computing was based on MPI (Message Passing Interface) framework which is most commonly used in the gridded network of the servers. For cloud computing, the algorithm uses the Map Reduce framework which handled the task distribution and load balancing. The results based on handling complexity of the functions showed that the best performers were both grid computing and cloud computing compared to the standalone desktop application. Same was the case with handling input data. Among, grid and cloud computing, latter was most preferred. The reasons being ease of use for programming interface and data handling, load balancing and each process was not handled by the user.

In the past few years, GIS looks towards building a-state-of-art technology to perform geo-computations (Yang et al., 2010). The data are typical large datasets of satellite imageries, survey data, volunteered data, computed imageries from various algorithms etc. The basic processes of GIS are collecting, storing, processing and visualizing data. An infrastructure catering to these processes, as well as integrating the scientific community with the advancing technology is imperative. Calheiros and Ranjan (2011) developed a cloud computing architecture for scientific applications. The tool offers a cloud computing architecture as well as a platform for deploying of scientific applications.

The report by NIST (2011) overviews the concept of cloud computing. Some characteristic

features. The characteristic features of cloud computing are: scalability, flexibility and ease of use (Armbrust et al., 2010). The demand for cloud computing lead to further research and contributions from major technological companies like Google, Amazon, Microsoft and IBM.

Cloud computing provides “Service models” and “Deployment models” (NIST, 2011). The cloud computing can be deployed in three ways: private, public or hybrid. In private cloud, a dedicated cloud architecture can be built with access to only a limited group. This form of deployment is usually in a closed institute where, data security is of highest priority. If the cloud platform is utilized and accessed by a number of users, it is called as public cloud. This type of model may be vulnerable to security challenges. The hybrid platform, the user can partly use the platform as a private and partly as a public. Further, the users can take advantage of cloud computing through service models. The three service models of cloud computing are named: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS).

1. Infrastructure as a Service (IaaS): This service provides the users with resources for storage and processing power. The users can take advantage of this on a pay per use basis depending on how much of the resources they consume for their task. They can deploy their own framework irrespective of the operative system. Amazon is one of the leading company providing infrastructures for business and scientific applications.
2. Platform as a Service (PaaS): This service provides a solution to develop application and deploy them on the cloud platform. The service provider usually provides users with an application framework and APIs supporting programming languages. Programmers and developers can write their own application. In this type of service, there is option for the users to just get an application development framework or infrastructure along with the framework. Some of the leading companies providing these types of services are Google and Microsoft. Google Earth Engine (GEE) is one example of platform which provides an infrastructure as well as APIs to develop application over the cloud.
3. Software as a Service (SaaS): In this type of service, the cloud service providers offers the complete package of software including the infrastructure and the developed application. The users take advantage of these customized services applications. Some examples of such services are Google Calendar which is available for free to the users with limited capability. The users have to buy upgrade to take advantage of the full capabilities.

2.4 GOOGLE EARTH ENGINE

Google Earth Engine (GEE) is a cloud computing platform providing GIS data as well as platform for performing analysis. This platform is dedicated to “Geospatial Analysis and Visualization”. GEE is the result of an initiative taken by Google Earth Outreach team to provide “Planetary-Scale Platform” for public benefit through non-profit organizations and scientific work (GEE, 2012). It provides access to a large warehouse of satellite imagery and the computational power needed to analyse those images. The storage memory can be counted in terra bytes and further more millions of terra bytes for processing. These services are made available through a vast network of infrastructure of servers.

A huge collection of GIS data is hosted in GEE’s network of servers. The data stored in GEE is provided through service called as data catalogue. The data catalogue consists of satellite imageries from Landsat, MODIS, and SRTM among others. These are global datasets from over 40 years of various missions. The data is collected, processed and available for further data analysis algorithms.

This is advantageous especially for scientific research work that requires readily available and pre-processed data. The uploading of the data is done majorly by the GEE developer team. However, in GEE users can upload their own datasets.

The Google Fusion tables can be used to upload data in GEE. The vector data in the form of points, lines and polygons are uploaded through a form in the Google Fusion table. This data is uploaded as a csv file. Once the data is uploaded as a fusion table, it is stored in the Google drive associated with the account. For the raster form of data, Google Fusion tables cannot be used. Until recently Google Map Engine was used for the same. But, since Map Engine was deprecated from beginning of January, other platforms need to be explored.

GEE as a PaaS, provides a development platform to adapt the algorithms or to run the analysis over the data available. APIs provide support for JavaScript and Python programming languages. The GEE exclusive libraries can be used for development or analysis programs. The applications developed on JavaScript API provide flexible integration with web applications as well service for computations. It is also possible to integrate web based applications with GEE for computation and processing of data. In Python API, along with earth engine libraries, the native python libraries can be used. The whole process of parallel computation is based on Map reduce framework.

2.5 MAP REDUCE CONCEPT

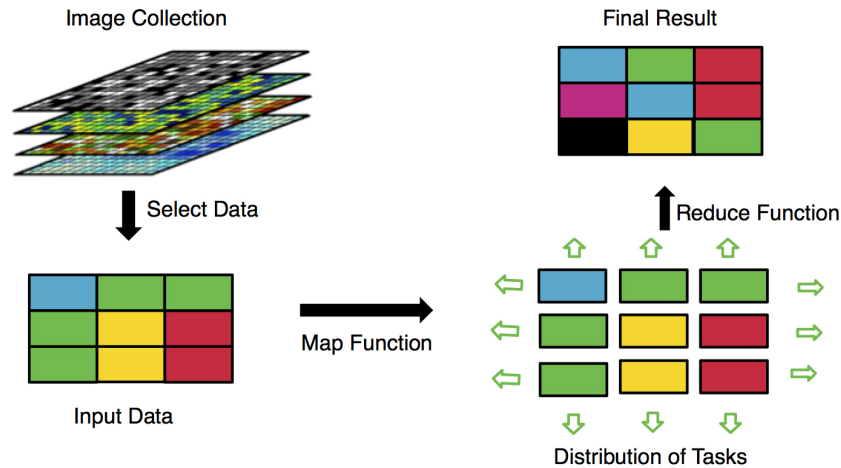


Figure 2.1: Concept of Map reduce framework

Map Reduce is a programming model designed for parallel computation in distributed architectures. Map Reduce was developed with an idea to breakdown a big tasks into smaller tasks (Licari, 2010). It basically consists of two sets of functions i.e. map and reduce, which are applied first to divide a processing work into several tasks, and then merge individual results. Advantages of using map-reduce are in the division of data, allocation and coordination of tasks between servers, and handling load balance.

Figure 2.1 gives a brief overview of the concept of map-reduce. Given a data source, the first map function divides data to be computed in several tasks. The master server handles the process to divide and further distributes the tasks to other servers. Results of map tasks are obtained, and forwarded to the respective reduce task. Tracking of each map and reduce task is based on a unique key assigned to each task by the master server. In case of failure of map functions, the master server reassigns the task, this makes sure all tasks are completed. Once, the results from reduce phase are

obtained, the master server brings together all the results and sends the output. The Map Reduce framework is dynamic in nature and based on simple theory to divide the tasks to reduce the load and obtain fast computations. This simplifies the large scale data processing tasks. Several scientific applications have used map reduce framework model (Castañe et al., 2012; Sehgal et al., 2011).

Chapter 3

Datasets

This chapter describes the temperature and volunteered Lilac datasets that are used in this research. Location based data was originally used in desktop based SI-x model. However, gridded temperature were used in the GEE based SI-x model. The following sections gives a description of these data and the pre-processing that was required to implement in GEE.

3.1 DATASETS FOR GEE-BASED SI-X MODEL

The GEE based SI-x model required two datasets that is the gridded Daymet temperature data and the historical volunteered Lilac data.

3.1.1 Daymet temperature data

A computed gridded form of temperature and longitudinal data is available for various studies. Daymet is a gridded dataset of daily maximum and minimum temperature. Figure 3.1 shows a sample screenshot of the Daymet data. The figure shows the average maximum temperature for USA and Mexico region. The Daymet dataset was computed from an algorithm that interpolated and extrapolated the weather data from few selected weather stations (P. E. Thornton & Running, 1999). The result of the algorithm is the computed data in the form of daily images starting from 1980 until 2013 at a spatial resolution of 1 x 1 km (P. Thornton et al., 2014). It includes information for length of the day (from sunrise to sunset), maximum and minimum temperature, water vapour, radiation, and precipitation in form of bands.

The computed dataset availability is limited to certain regions as provided from Daymet. From 1980, the dataset is available for USA and Mexico. In recent years, it also includes the southern part of Canada. However, the dataset is available only until 2010 for Mexico. These datasets can be downloaded in the form of tiles of 2 x 2 degree, further mosaicked to form raster images. Alternatively, the data can be downloaded pixelwise over a 1 x 1 km area by giving the information for the center of pixel location.

The Daymet dataset was ingested in GEE by Google developers. The developer team performed the process of downloading, pre-processing, and uploading of the final collection of daily imagery. The pre-processing time for research work was thus reduced. The availability of such a pre-processed data, is an advantage of GEE over other cloud computing platforms. However, it was observed that due to technical issues, sometimes a few imagery may not be accessible at times. Checks need to be performed on whether all imagery are accessible and ready for processing before starting any computations.

In cloud based SI-x implementation, Daymet data available in GEE was used. The maximum temperature, minimum temperature and day length was used in this implementation. Previously, the temperature data from weather stations were used while day length was computed for each day in SI-x.

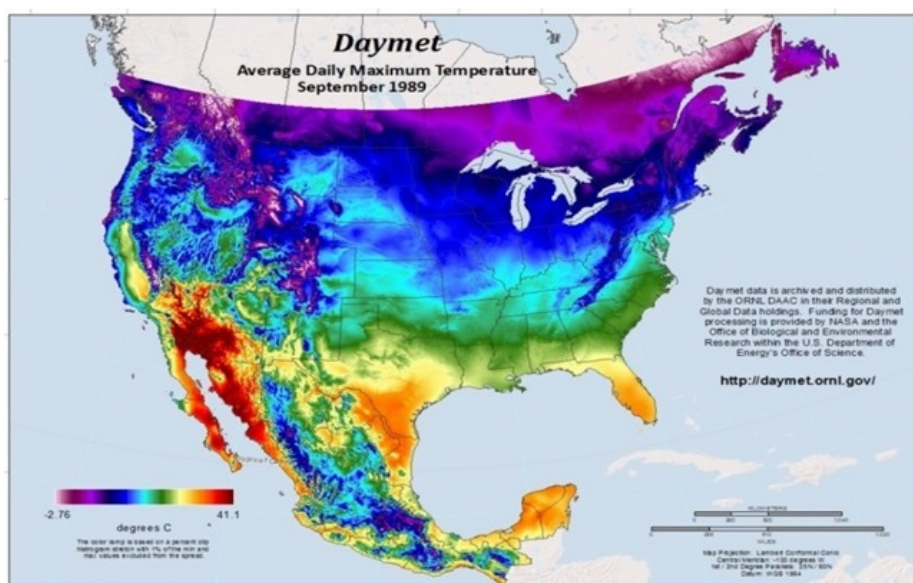


Figure 3.1: Sample of Daymet dataset of average daily maximum temperature for September,1989. Source: Retrieved from <http://daymet.ornl.gov>

3.1.2 Volunteered data

The phenology data collection in North America has varied over time and space. Starting from 1950 until 2003, the first leaf and bloom data for common lilac species, *Syringa vulgaris*, was collected at multiple locations in western USA and for cloned lilac, *Syringa chinensis*, from eastern USA. The other two species, Honeysuckle (*Lonicera tatarica*) and Zabeli (*L. Korolkowii*), are sub-models in SI-x. The former being limited to the eastern part of USA while latter is observed only in the west of USA. Detailed information on the evolving phenology network in the USA can be found in Schwartz et al. (2012) and Schwartz (1994). For years after 2003, new phenology networks were setup for collection of volunteered data over continental USA (Rosemartin et al., 2014).

The historical data for lilac species is available from 1956 up to 2003 (Schwartz & Caprio, 2003). The data is a list of station locations, and the observations of first leaf and bloom around these stations. It is available for two species of lilac namely *Syringa Vulgaris* and *Syringa Chinensis*. Besides, missing observations exist in this dataset, which restrains an effective data analysis.

3.2 PRE-PROCESSING OF DATA

In this study, the historical data from 1980-2003 was used for the purpose of correlation with SI-x model results from GEE. The GEE based SI-x only predicts the first day of leaf phenomenon for Lilac. Hence, the historical volunteered data for the first day of leaf of Lilac plant was used. The historical data downloaded was pre-processed before analysis in the GEE. Figure 3.2 shows the steps carried out in this process.

1. The first step was to download lilac observation data from NOAA. The data downloaded was in the form of two tables: one with station locations, and the other with lilac observations data. The data was selected for the years from 1980 to 2003 since the model results correspond to the same time period.

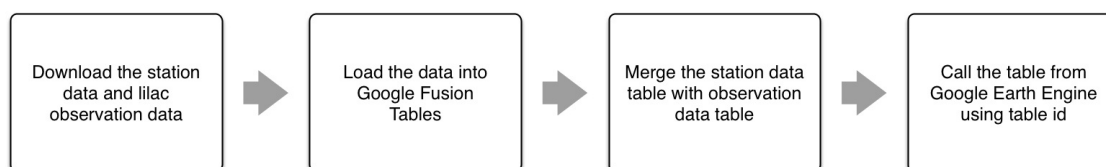


Figure 3.2: Steps involved in pre-processing of the historical Lilac data

Station ID	Year	Plant type	Date of First Leaf	Date of First Bloom	Station Name	State/Prov	Lat	Long	Elev
41010	1990	2	73	97	BOULEVARD 2	CA	32.4	-116.18	1097.56
41010	1991	2	53	100	BOULEVARD 2	CA	32.4	-116.18	1097.56
41010	1993	2	59	82	BOULEVARD 2	CA	32.4	-116.18	1097.56
41018	1981	2	103	130	BOWMAN DAM	CA	39.27	-120.4	1630.18
41018	1985	2	86	126	BOWMAN DAM	CA	39.27	-120.4	1630.18
41214	1989	2	79	999	BURNEY	CA	40.53	-121.4	957.32
41214	1991	2	130	140	BURNEY	CA	40.53	-121.4	957.32
41214	1993	2	69	999	BURNEY	CA	40.53	-121.4	957.32
41215	1980	2	60	112	BURNT RANCH 1 S	CA	40.48	-123.28	640.24
41215	1981	2	69	111	BURNT RANCH 1 S	CA	40.48	-123.28	640.24
41215	1982	2	79	121	BURNT RANCH 1 S	CA	40.48	-123.28	640.24
41614	1982	2	91	142	CEDARVILLE	CA	41.32	-120.1	1423.78
41653	1983	2	63	126	CHALLENGE RANGER STN	CA	39.29	-121.13	780.49
41653	1984	2	44	85	CHALLENGE RANGER STN	CA	39.29	-121.13	780.49
41653	1987	2	84	104	CHALLENGE RANGER STN	CA	39.29	-121.13	780.49
41653	1988	2	59	84	CHALLENGE RANGER STN	CA	39.29	-121.13	780.49

Figure 3.3: Merged table for station locations and observation data for Lilac in Google Fusion table.

2. The Google Fusion table was used to ingest the station location and observation data. The two data sets were uploaded into fusion tables as a comma separated value file (.csv). Each generated table had a unique primary key of station ID. The data can only be accessed in GEE when uploaded through the Google Fusion table.
3. The merge table function in Google Fusion table was used to join the two tables using station ID as the primary key. Figure 3.3 shows a snapshot of the merged fusion table. An unique table ID was assigned to this new table.
4. The table ID is used to call the merged table into the GEE. This data is loaded into the GEE as a feature collection. Figure 3.4 shows this uploaded data in GEE.

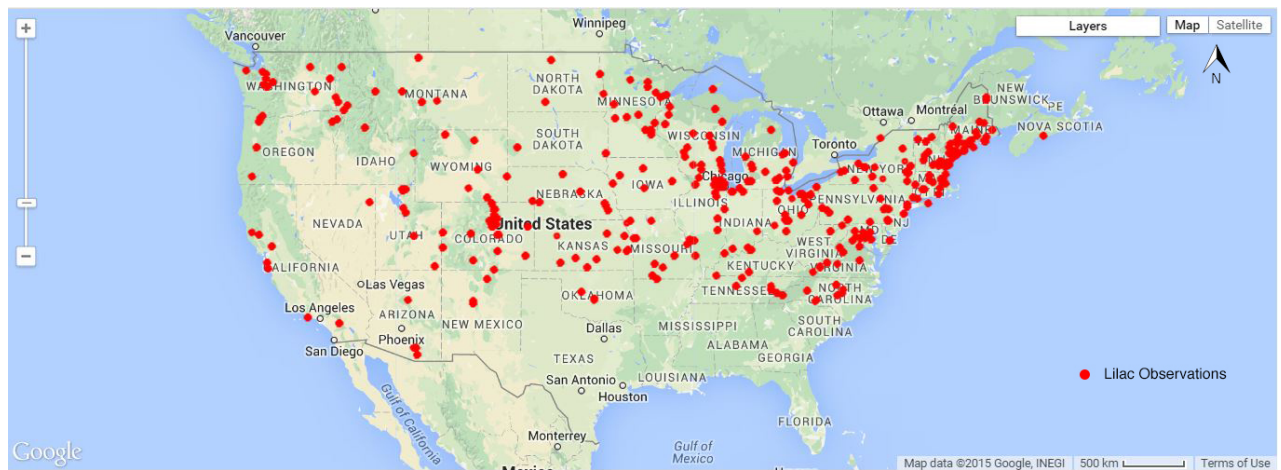


Figure 3.4: Visualisation of the observations for the first day of leaf of Lilac species from 1980-2003

Chapter 4

Adaptation of SI-x model to cloud computing

An assessment of the SI-x was performed to adapt the model in cloud computing platform. Further, a modified version of the original workflow of the desktop based SI-x model was designed to fit the cloud based platform of GEE. In addition, it should be noted that this adaption also involved the transition of SI-x model from prediction of the first day of leaf of Lilac species at every weather stations to a grid based predictions in the GEE. This chapter describes the existing and the modified workflows of SI-x model.

4.1 CONVENTIONAL SI-X MODEL DESIGN

Initially, the SI-x model was implemented in FORTRAN. Later, the same model was implemented using the MATLAB toolbox (Ault & Zurita-Milla, 2013). The input for the MATLAB implementation was weather station data. A .mat file was created consisting of the input data and fed into the MATLAB code. Each class file in MATLAB calculated the functions of SI-x model. These functions calculated the prediction of first day of leaf for all the years at the specific weather stations. The output of the SI-x model was plotted as a graph in MATLAB. The MATLAB workflow can be seen from Figure 4.1. The workflow represents the computational model of the MATLAB code implementation. Since, the model design dated back to the 1980s, the model was typically designed to implement on a traditional desktop. The first step was to identify the processes relating to the main functions upon which the model exists.

As seen in Figure 4.1, the main driver function handles, input, output and the main regression function of the SI-x model. The temperatures were converted from Celsius to Fahrenheit and 1D array of the input maximum and minimum temperature data was created. The data for all stations is further checked for continuous daily temperatures for each year. Missing temperature values are aggregated from the previous and next day. This input is passed to the regression function.

The regression function was implemented for all stations in time-series. The one dimensional array of maximum and minimum temperatures, day length and solar declination was used to calculate the growing degree hour (GDH). GDH is the sum of the hourly day temperatures. This regression function includes four predictor variables, MDS0, DDE2, DD57, AND SYNVAL. The count of the days passed since first day the year was calculated as MDS0. The variables DDE2 and DD57 included the GDH for last three days starting from second to third day of the year and fifth to seventh day of year respectively. The variable SYNVAL were the count of the synoptic events for the days when DDE2 was greater than 637.

These computations were based on the formulas as defined in the article by Ault and Zurita-Milla (2013). Static co-efficient variables were defined for each predictor variable based on the three plant species. However, just the lilac plant species were considered in this study. The first day of the leaf is then predicted when the summation of the four variables was greater than 1000.

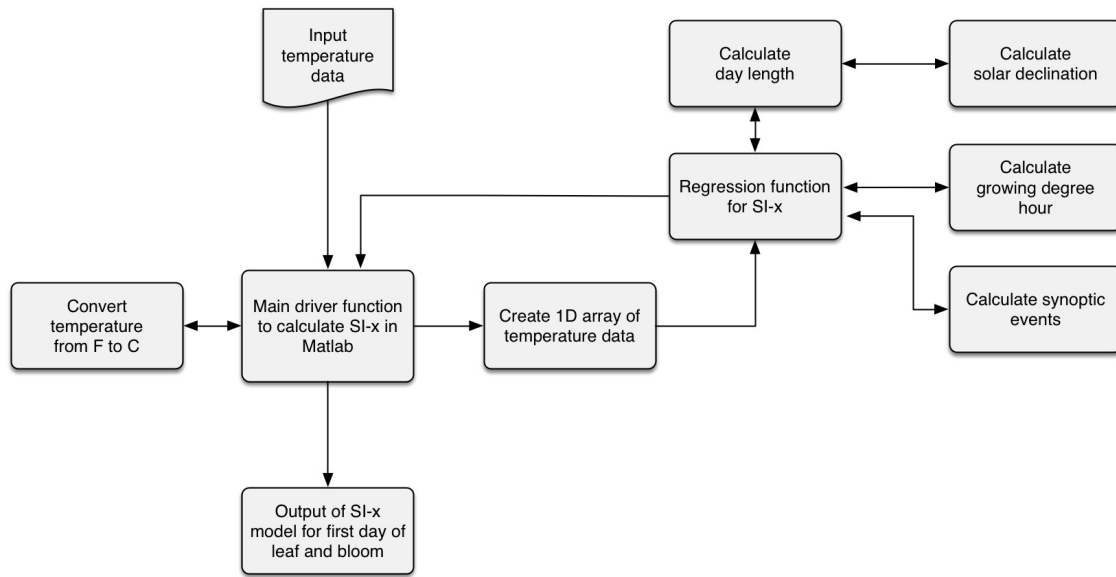


Figure 4.1: Block diagram of a typical workflow of the SI-x in MATLAB

4.2 CONCEPTUAL DESIGN FOR CLOUD BASED SI-X

The first approach to adapt the SI-x into cloud computing platform involved using the same workflow of SI-x model using Matlab toolbox. However, this led to running the SI-x-model partially on the user desktop and therefore, the complete capacity of the cloud computing platform was not being utilised. Hence, a new approach was devised, to design a new algorithm based on the parallel computing concept of cloud computing that is GEE.

The input to the cloud based SI-x model was set of the gridded daily temperature imageries. Each imagery consists of approximately 10 million grids. Therefore, in comparison to the previous SI-x implementation, the GEE-based SI-x model was to predict the first leaf over all grid locations simultaneously. The Map reduce framework was used to perform parallel computations over all the grids. The new algorithm was conceptualized by redesigning the computational method of the SI-x in cloud platform.

The basic script for the functions in the algorithm was developed using join and merge functions in an image collection. These functions were then passed through the map function over the image collection. This would distribute the tasks and perform the functions over each pixel of the image collection. All the functions were designed to avoid any iterations call that would call the data structures in a serialized form. The iteration function calls slow down the process and result in more time for computations, especially in a parallel computation. The code was further optimized to be able to process more than 24 images. The code was rewritten to avoid iterations and use other functions available in GEE. Though this was not the optimal solution, further optimization techniques and improvements to the code is possible, once more functions shall be available in GEE in near future.

Based on the pseudo code algorithm, a scripted code for the model was created using the functions provided by the GEE library in JavaScript API. The SI-x model in GEE processed up to 60 days of daily temperature images. These can be further scaled up to process 240 daily images in GEE by resolving the problems faced in processing more than 60 images.

The algorithm was designed by dividing the model process into blocks and performing com-

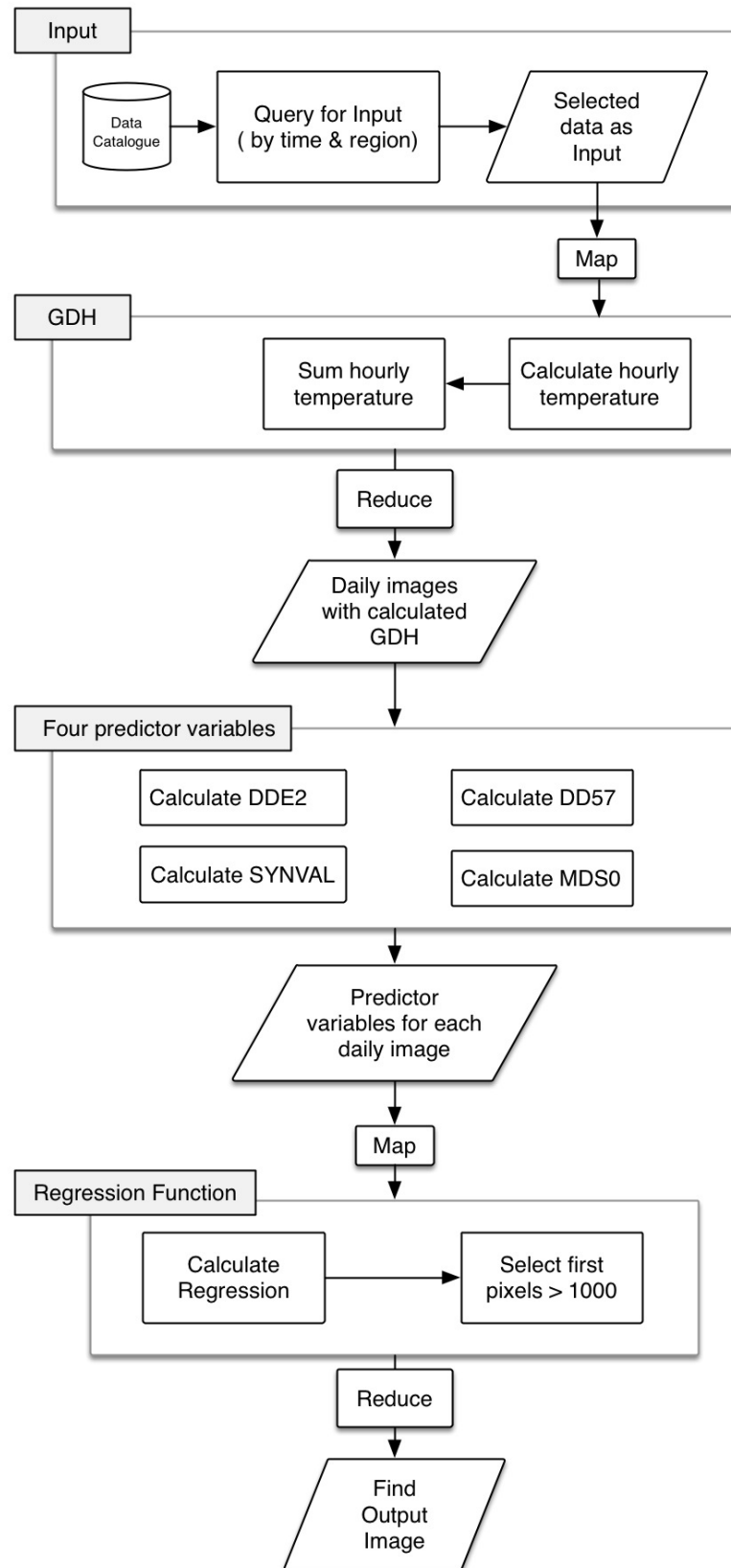


Figure 4.2: Workflow of algorithm to implement SI-x in cloud computing.

putations over the individual blocks. Figure 4.2 shows the new algorithm design, followed by a detailed description of each block.

1. Input data

The first step was to handle the input data. The input data to the model was a set of gridded imageries. A cloud based database was to be designed to deploy the data. In case of GEE, the data was readily available for use. The GEE included the libraries for data handling and filters for querying the dataset. The filter function was used to call the required datasets and for particular days, months and years. The raster images were provided to the users in the form of image collections in GEE by Google. Therefore, further operations were to be designed considering the functionality available for image collection. Unlike previous method to run the model per location, in this cloud computing platform, the this model was adapted to perform pixel based computation. The succeeding blocks were designed to perform further computations of the SI-x model in GEE. Refer algorithm 1 for detailed implementation.

Algorithm 1: Algorithm for selection of input data in GEE.

Input : Daymet data as earth engine's "ImageCollection" object from GEE Data catalogue.

Output: Filtered sub-collection of daymet data as ImageCollection object.

```

1 begin
2   Call Image collection  $\leftarrow$  'NASA/ORNL/DAYMET'
3   Filter collection to select year
4   Filter collections required for days of year
5 end

```

2. Calculate Growing Degree Hour

The parallel computations over every pixel were performed by map and reduce functionality in GEE. The map function was to define the tasks in form of arithmetic operations to be performed on each image in image collection. The reduce function was used to compile the results of the computations and give single output. This architecture was developed to reduce time required for high computational process especially with large scaled data. The concept of map reduce would divide the number of functions relatively into smaller tasks. Once these tasks are distributed and processed over a network of servers, the final output is obtained from the reducer function. Therefore, the time required for Map Reduce function to complete a process depended on the complexity of the function, scale of the input data, and the complexity of the data structure.

Since, the input data was raster imageries, band operations were mainly used for performing mathematical operations. The second block comprised of computation of the growing degree hour. The input collection from previous block has band information for day length and maximum and minimum temperatures. Unlike previous SI-x, day length from band information is used. These calculations perform band math on per image starting 1st January. The result of this function was input for the next process to calculate the four predictor variables of the regression function. From algorithm 2 the detailed steps for calculating GDH using GEE functions are seen.

3. Calculate four predictor variables

Algorithm 2: Algorithm for calculating GDH in GEE.**Input** : Filtered sub collection of Daymet data for map reduce function.**Output:** ImageCollection object of calculated growing degree hour.

```

1 begin
2   For every map function in image sub-collection
3     tmax image  $\leftarrow$  maximum temperature band image
4     tmin image  $\leftarrow$  minimum temperature band image
5     Convert tmax and tmin from Celsius to Fahrenheit
6     Difference  $\leftarrow$  tmax-tmin
7     Day length image  $\leftarrow$  convert seconds to hours
8     BaseT  $\leftarrow$  Base Temperature of 31
9     hourlyTemp  $\leftarrow$  calculate hourly temperature and subtract BaseT
10    For hourlyTemp < zero assign zero
11    Reducer to sum the bands
12 end

```

Algorithm 3: Algorithm for calculating the predictor variables in GEE.**Input** : Image collection for computed growing degree hour(GDH).**Output:** ImageCollection with four predictor variables per image.

```

1 begin
2   joinCollection  $\leftarrow$  For every join function in GDH collection
3   For each image, filter the images from last 2 days
4   Save the filtered images as property
5   DDE2  $\leftarrow$  For every map function on joinCollection
6   Merge and sum the filtered image
7   Set time property for 3rd Day
8   DD57  $\leftarrow$  For every map function on joinCollection
9   Reducer to merge and sum the filtered image
10  Set time property for 5th Day
11  collection  $\leftarrow$  For every map function on dde2
12  For every pixel value more than 637 set 1
13  eventColl  $\leftarrow$  For every join function in collection
14  For each image, filter the images from all previous days
15  Save the filtered images as property
16  SYNVAL  $\leftarrow$  For every map function eventColl
17  Reducer to merge and sum the filtered images
18  Imagecollection = empty list
19  MDS0  $\leftarrow$  For i= startdate : enddate
20  Create image of value i
21  Add to imagecollection list
22  Join the collection DDE2, DD57,SYNVAL and MDS0
23 end

```

The four predictor variables were calculated based on their mathematical formulas by accumulation of the growing degree hour temperatures for all the days in the image collection. The results of each prediction function were stored as bands for every image in the image collection.

As from algorithm 3, each process for calculating predictor functions was executed based on the timestamp of the images in the image collection. The predictor variables DDE2 and DD57 were calculated from 3rd and 5th Day respectively according to the mathematical formula. In GEE, at the time of implementation, it was not possible to add static values for non-calculated days. This was in contrast to the previous SI-x model, where certain values of predictor variables were assumed and statically added for missing values in the beginning of the year. Therefore, this step was skipped to get the preliminary results. Each of the processes were executed without change of the timestamp and hence, the result used a regression function starting from 6th January instead of 1st January as in the previous model.

4. Regression function

Algorithm 4: Algorithm for Regression function of SI-x model in GEE.

Input : Image collection with 4 predictor variables per daily image - PredCollection
Output: Final Image with predicted values for valid values and masked for non-predicted value pixels.

```

1 begin
2   For every map function in PredCollection
3     Add a day of year band
4     Sort the collection with last to first
5     For every map function
6       Reduce and sum the bands
7       Mask the pixels where sum > 1000
8       Create a mosaic of the collection
9 end

```

The algorithm 4 shows the final process of calculating the regression function. The results of the four predictor variables for each day and location were multiplied by its respective co-efficient for lilac plant species and the day when the sum > 1000 was predicted as the first day of leaf. As the final prediction result, the output is single image with each pixel value equivalent to the day of the year when this has occurred. In the image collection where this phenomenon has not occurred, the pixels are masked.

4.3 METHODOLOGY FOR CORRELATION

The volunteered data uploaded in Google Fusion tables was used for the correlation. The results obtained from the model were correlated with the volunteered data. For the purpose of correlation, volunteer data was reprocessed and uploaded to GEE as described in chapter 3 . The GEE lacked functionality to perform statistical analysis for the required correlation. Hence, an alternative approach included to overlay point volunteered data on the SI-x model results and to extract the data for further analysis. The `getregion()` functionality from GEE was used for extraction of data .

Chapter 5

Implementation of SI-x in GEE

This chapter describes the results in the process of implementation of the SI-x in the GEE platform. The GEE performs pixel-wise operations using Map Reduce function. As discussed earlier, the iteration operations resulted in slowing down of computation time and indirectly increasing time. Therefore, it is recommended to avoid operations that will require serializing the processing and instead using map and reducing functions as much as possible.

5.1 RESULTS FROM GEE BASED SI-X MODEL IMPLEMENTATION

Based on the new algorithm design, the SI-x model was implemented in GEE on a per year basis, starting from day 1 to day n of that year (in this case, $n=60$). The GEE performed a pixel wise computation over daily images up to n days. A typical result was a single image (per year), with each pixel value equivalent to the first day of leaf. The JavaScript API of GEE was used to visualize the results. The screenshots from figures below show predicted results of SI-x in GEE for continental USA. The implementation of the algorithm was able to process upto 62 days. Each pixel value is equivalent to first day of leaf.

A continental scale visualization in GEE of predicted results is seen from figures 5.1, 5.2, 5.3 and 5.4. The results from different years was the occurrence of spring at latitudinal level. As in Figure 5.1, the year 2000 shows spring arrival up to mid-latitudes up to 60th day. However, the spring arrival for 2005 and 2010 from Figure 5.2 and Figure 5.3 is limited to lower latitudes until the same time. Further, topography plays a role in spring occurrence pattern. For example, the spring is observed consistently along the west coast of USA compared to east coast for all the years. These comparisons were made on continental scale with representative areas. The following regional scale comparison for all years gave an overall trend to occurrence of spring.

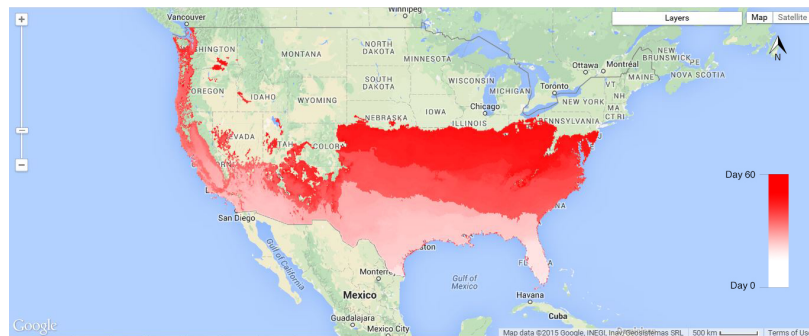


Figure 5.1: Screenshot from Google Earth Engine visualization panel. The map shows occurrence of spring for the year 2000 starting from Day 1. Each pixel refers to first day of leaf. The white colour indicates the early days of year while greener colours indicate later days of the year.

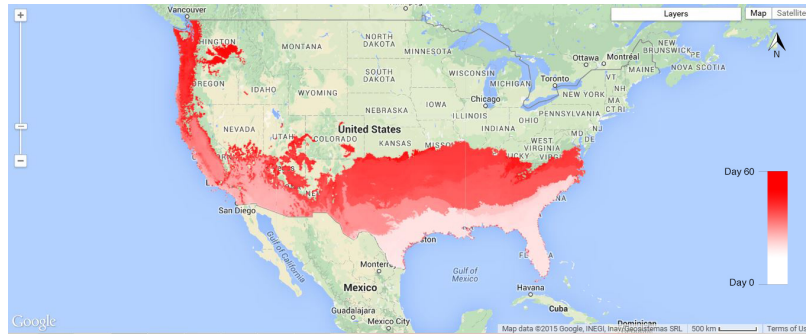


Figure 5.2: Screenshot from Google Earth Engine visualization panel. The map shows occurrence of spring for the year 2005 starting from Day 1. Each pixel refers to first day of leaf. The white color indicates the early days of year while greener colors indicate later days of the year.

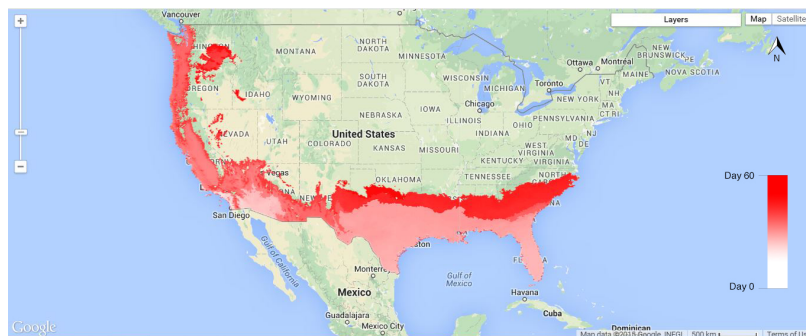


Figure 5.3: Screenshot from Google Earth Engine visualization panel. The map shows occurrence of spring for year 2010 starting from Day 1.

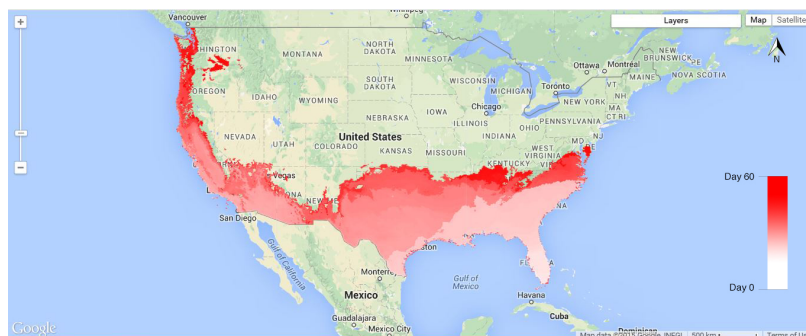


Figure 5.4: Screenshot from Google Earth Engine visualization panel. The map shows occurrence of spring for year 2013 starting from Day 1.

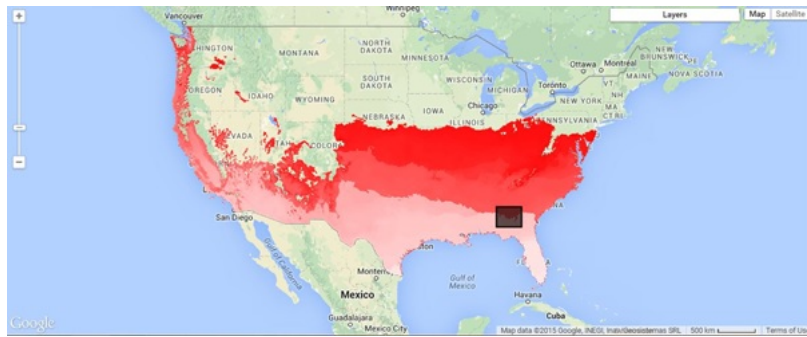


Figure 5.5: Screenshot for Year 2000 with the Area of Interest (AOI) for time-series analysis. The co-ordinates for polygon as ((-86,31), (-83,33)).

5.1.1 Results from time series data from area of interest

A region in the South Eastern USA was selected to study time-series occurrence of spring at a regional scale. An area of interest(AOI) was selected to identify trends in change in occurrence of spring as seen from Figure 5.5 . The time-series analysis was performed in GEE by running computation for each year. A polygon with coordinates - in the South-eastern part of USA. Once, the model was run for each year, the `reducer.mean()` function in GEE was used to calculate the mean first day of year in the AOI. The results were plotted as seen in graph from Figure 5.6. The graph shows an overall variability in mean value of first day of leaf. Some extreme occurrences are noted from the graph. For example, a relatively late onset of spring in 2004 is followed by a very early onset in 2005. The observation is in accordance with findings from literature of early spring observation all over USA. Similarly, an extreme peak rise in year 2010 shows occurrence of late spring. Hence, this year observed extended winters and lead to late arrival of spring. Post-2010, a steep consistent downward trend indicates an early spring pattern.

5.1.2 Results from workload vs. time analysis

The computational code of GEE based SI-x algorithm consisted of 4 blocks as described in chapter 4. Each block computed the functions leading to final regression function for one year. The code was run on daily images consisting of approximately 10 million pixels. Hence, each block in the algorithm was computed 10 million x number of images times for the prediction of the first day of leaf. To assess the processing time for each block, the code was run 5 times for a number of images. The processing time from 5 runs was averaged and plotted into a graph. The graph in Figure 5.7 shows average processing time of each block with different number of images. From the graph, a higher processing time is noted for more number of images. Nevertheless, block 1 and 2 are computed in a consistently less time as compared to block 3 and 4. Further, for images more than 55 the computations ran into timeout error. The reason for the same could be the complex functions executed in block 3 and 4. Therefore, the block 3 and 4 were further assessed in depth.

The further analysis of the algorithm involved checking the processing time for individual functions in block 3 and 4. The individual functions in the respective blocks were run 5 times and average time for each was calculated. The 5 iterations were for different number of images. A graph of average processing time for each function with respective number of images was plotted. The plotted graph is seen in figure 5.8. The functions from block 3 calculated DDE2, DD57, SYNVAL, and MDS0 while the block 4 computed the final image for the first day of leaf. The MDS0 was the least complex function since it created a new image collection. Therefore, it took less processing time compared to the of the functions. A peak in processing time was observed at

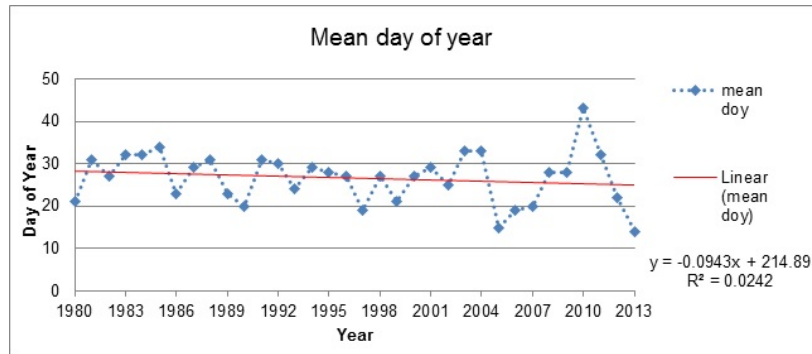


Figure 5.6: Graph of mean day of year values for each year starting from 1980 until 2013.

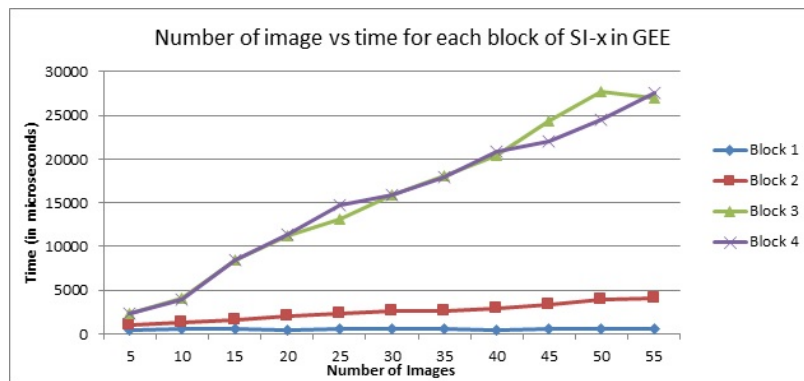


Figure 5.7: Graph for number of images v.s time for each block in SI-x in GEE.

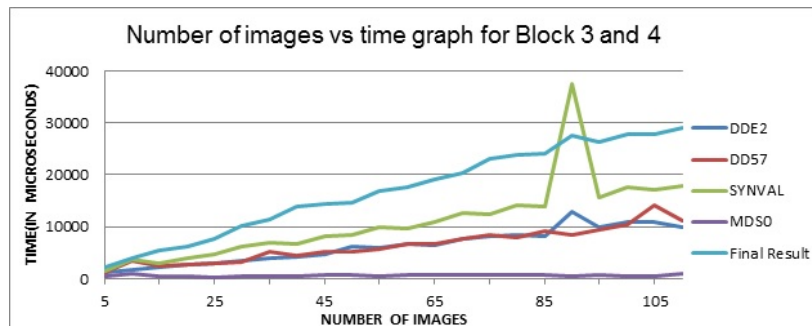


Figure 5.8: The above figure shows correlation between number of images and time taken to process the functions in Block 3 and 4 of SI-x in GEE.

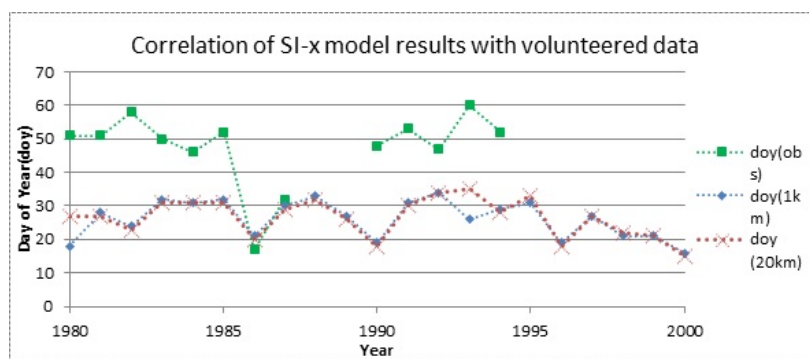


Figure 5.9: Correlation of volunteered data with SI-x model results from GEE. The SI-x model results were obtained at 1km and 20km scale.

85-95 numbers of images. This was repeated for 6 times to get similar result. Thus, it cannot be termed as an outlier.

5.2 RESULTS OF CORRELATION WITH VOLUNTEERED DATA AND ITS INTERPRETATION

The results of SI-x model in GEE were correlated with the volunteered data starting from 1980. The volunteered data uploaded through Google Fusion tables was used for the same. The aim was to correlate the model results for each year with the respective year of volunteered data. However, GEE currently has limited functions for performing analysis. Further, the SI-x model results for the complete predicted region could not be downloaded so that further analysis could be done. The other option was to manually insert the locations and fetch the data using GEE function for each location. Considering all limitations, the analysis was performed on a single station location. Since, all volunteered data was not available for continuous time period, the station with data for longest time period was selected. The selected station location was overlaid with the SI-x model results in GEE to get predicted values.

One station was selected for further analysis. The station selected for comparison was located at coordinates- (-100.07, 30.56) in Texas region. For the selected location, the data was available for *SyringaChinesis* species of the lilac plant, with some missing years of observations. The get region function in GEE was used to overlay the point location over the raster image and get the results. Since, the function did not give results for the complete feature collection, the get region function was implemented at individual location. To understand the difference in the SI-x model results obtained at different scales, the results from the model were obtained at 1km and 20km spatial resolution. The model result values at different scales were obtained after the model was processed. The results are plotted as in figure 5.9. The figure shows the observations from volunteered data with the model results at 1km and 20km scale. Between model results and the observation there seems to be a difference of about 15-20 days. This implies that the SI-x model in GEE predicted first day of leaf earlier than the observations from the volunteered data. However, the model results matched the pattern from observed data in the year 1986 and 1987. In the year 1982 and 1993, the model predicted an early spring however, the observation for first day of leaf was at latter day. The model results obtained at 1km and 20km spatial resolution vary by a slightly. But in the year 1980 and 1993 results vary. However, in rest of the years the predictions are over or under by a few units.

5.3 FURTHER WORK

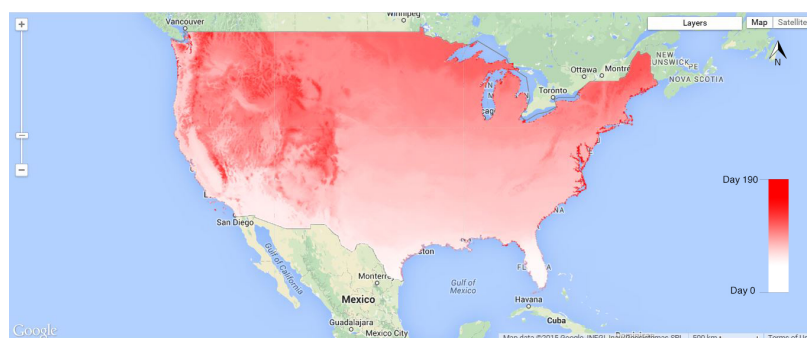


Figure 5.10: Implementation of SI-x model in GEE for 190 days for the year 2000 .

As mentioned earlier, the current implementation of SI-x model could run upto 60 days successfully. With further code optimization and trial and errors, it is certainly possible to run the model to process daily images upto 240 days and get the full results for complete spring cycle. However, even though desired, due to time and technical constraints at the time of writing the thesis, further explorations were not undertaken.

A similar ongoing research work undertaken by, Ms Emma Izquierdo-Verdiguier, researcher at ITC, builds on the adapted SI-x model and the GEE implementation discussed in this thesis. The existing code for GEE SI-x model was optimized further and could process the daily images upto 190 days successfully. Figure 5.10 shows a sample result of SI-x model (optimized) in GEE with results upto 190 days.

Later, the optimized code was available at the time of wrapping up of this thesis and could not be undertaken for further analysis due to time constraint.

Chapter 6

Discussion

A detailed discussion of results from chapters 4 and 5 is given in this chapter. Each section is further related to the research objectives as defined in Section 1.3.

6.1 CONCEPTUAL DESIGN OF CLOUD BASED SI-X

The approach to adapt SI-x on cloud computing was based on some initial assumptions. The first approach was to maintain the existing algorithm as is, while convert the input data format from gridded to 1D array format (input format for SI-x) in cloud computing platform. Whereas, the second approach was to redesign the model algorithm based on map reduce framework. The first approach implemented on Google Earth Engine used the platform only for data handling, while the processing would still be on the desktop. Hence, this was not a feasible option. Therefore, second approach, a more suitable since it leveraged the computational power and data management capability of cloud computing was chosen. The use of Map reduce framework to perform parallel computation made it possible to compute 10 x 365 million pixels for each year in a matter of seconds.

In cloud computing, the map reduce framework can be used for parallel computations. It consists of a master server handling distribution of tasks of the ‘map’ function and ‘reduce’ function to compile the results. The tasks are allocated to several interconnected servers performing computations. In case of any failure in computation, the master server reallocates a new task. Therefore, all the computations are eventually performed. The new algorithm was redesigned to adapt the SI-x to a cloud computing Map Reduce framework.

The conceptual design of SI-x for cloud computing as described in Section .. is generic in nature. The algorithm can be used to adapt the SI-x in other cloud computing platforms. The reason being map reduce framework is used in several cloud computing platforms. But, certain criteria need to be fulfilled to adapt SI-x to other cloud platforms. The first criterion is database infrastructure for storing and handling GIS data. The second criterion is the number of computing resources available along with development platform in high-level programming environment languages such as Python, JavaScript. Lastly, cost incurred as result of utilization of the resources. However, the design of algorithm and its characteristics are independent of the cloud platform in which it is implemented. Srirama et al. (2012) found from their experiments that same algorithm performs differently on two different cloud platforms. Several factors can affect the performance of the algorithm.

6.2 GOOGLE EARTH ENGINE IMPLEMENTATION

The benefits of GEE are availability of GIS data, easy to use programming environment and instant visualization of results. GEE is a cloud based platform developed specifically for geospatial analysis. The GEE offers a data source with pre-processed data as well as service for users to upload custom data. The task of ingestion and pre-processing of data is done by the GEE developers. This

saves a lot of time required for pre-processing raw data and building a database for any research work. It is also possible for the user to upload own customized dataset via fusion table (vector data) and Google Map Engine (for raster dataset; now deprecated). As a PaaS, a GEE programming environment is available for JavaScript and Python languages. To implement the code of algorithm in GEE, JavaScript API was used. It is an easy to learn high level programming language. The programming environment consists of a scripting panel, a visualization panel and a console. The results from the computation can be printed to adjoining console. However, one of the main advantages of GEE is real-time visualization of the raster or vector results on the visualization panel. The panel consists of default base maps for reference. The results from computations can be added to the base map and perform analysis. The inspector tool when clicked on the map, gives pixel information. The python environment is comparatively a recent development and hence under improvement.

However, there exist a few limitations in using GEE. GEE is an independent geo tool. The services are offered through its extensive library to perform geo computations. Since, this platform is under constant update; new functionalities are constantly being added or upgraded. One of the other problems in GEE is the iterate functionality. For example, the iterate function creates a bottleneck during computations. The reason is iterate function in GEE creates a serialized computational behaviour which is beyond the parallel computation feature. Also, the iterate function stores the results of each iteration which is in turn passed on to the next run. Therefore, this consumes more storage and computation memory resulting in high processing time. In other words, any functions outside map and reduce functions may not necessarily being computed in parallel.

Based on the conceptual design in chapter 4, a pseudo code algorithm was designed to adapt SI-x on GEE. The logic of pseudo code varies from original SI-x. The mathematical formulae of SI-x were used as reference to design the pseudo code. According to the formula, DDE2 predictor variable calculates from 3rd day of year and the values for day 1 and 2 are not accounted for. In the original SI-x, the DDE2 values for day 1 and 2 were calculated using growing degree hour (GDH) for 1st day. In ideal case, to calculate DDE2 for 1st day, GDH values from days of preceding year could be considered. However, in GEE the daily images are set together in image collections by their timestamps. These time-stamped images could not be manipulated unless a complete new image collection is created. As a result, the model starts computations from 6th January instead of 1st January as in original model.

The scripting in GEE involved a lot of complex functionality for various computations. A few tasks that can be scripted easily in the lower level language such as C++ seem complex. This limitation is mainly due to the method of structuring data in GEE and working of each function. For example, functions designed for image object and image collection object are different. In other words, the functions meant for image object cannot be used for an 'a set of images' called as an image collection. The map functionalities can be performed only on an image collection but not an image object. Therefore, understanding these subtle differences is the key to working with GEE libraries.

There is some functionality not currently available in GEE. The major drawback from GEE in this study was the exporting the computed results in combination with a lack of analytical functionality. The results obtained from computing prediction results for one year can be visualized in the JavaScript environment. In GEE, there is limitation on maximum number of pixels that can be downloaded at a time. Since, in the current work, each computed result for whole USA exceeded this maximum pixel limit, it resulted in insufficient memory error. The other option was batch download of image. However, this functionality is currently not available in JavaScript environment. The only possible option was to download a small region of polygon. In addition, there is no possibility to temporarily store the computed results that could be used further anal-

ysis. The current approach to save results is through Google drive or Google Map Engine (now deprecated). This was major setback for the conducting further analysis.

Currently, limited functions exist for performing analysis in GEE. The descriptive statistics functions are currently available. These were used in performing time-series comparison of model results over a region. Some spatial analysis functions were used to perform correlations of predicted results of the model and the observation data.

Further work was conducted to improve the performance of the current implementation of the SI-x code in GEE. Because of the implementation, the new code was able to process up to 190 days of daily images. Since, the optimized code was only available in last few days, it was not possible to conduct further analysis using the same data.

6.2.1 Discussion on the SI-x model

The pseudo code algorithm was implemented to compute results for up to 60 days in one year. These results were obtained after computation over 10 million pixels of each daily image. This proves that Google Earth Engine was able to leverage the equivalent computational power. For images more than 60, the script ran into a time-out error. This meant that computations are being performed at the back end but cannot be visualized in stipulated time. The time-out exception is always in place to keep a check the computation time on client side of the architecture. One solution can be to optimize the code to reduce the number of operations involved that was partially done through new implementation later on. However, saving the computed results needs more functionality in GEE.

Further, the regional time-series analysis shows variability in occurrence of spring. The region was particularly chosen from literature studies. The trend line over the years shows occurrence of early spring however, this change is not a very significant in this particular region. Further regional scale analysis is needed, to compare the phenomenon among various regions. However, while conducting such study, regional climatic, altitude and elevation play an important role.

6.3 CORRELATION OF POINT AND GRIDDED DATA

The scale factor plays an important role for correlation of point and gridded data. The SI-x model was designed with 1D array of input data and was further validated with observation data (point data). However, in this research the input data was computed temperature data of spatial resolution 1km. Therefore, comparison of gridded model results with point observation data may vary at different spatial resolution. This can be seen from the results in Section 5.3. The model results were obtained at native resolution of the input data at 1km and further at 20 km. This was done to compare variations in model results obtained from different aggregation scale levels. These were correlated with observation data. The results show minor variations in model results for most of the years for the two scales. However, some drastic changes in value were noted for one or two years. The summary of the above is that, when performing correlations between point and gridded data, the scale of grid plays an important factor.

Chapter 7

Conclusion and Recommendation

The research objective of the study was to review the cloud computing platform for adaptation of SI-x and in particular GEE. By implementing the SI-x on GEE, the model predicted the results for continental USA simultaneously in contrast to previous implementation of one station at a time. The computation results were obtained in less than a minute after processing more than 100 million pixels. Now, the processing is distributed and parallel over a host of servers and hence, the improved time of results. Further, the predicted results were analysed to find regional patterns and correlation with volunteered data.

7.1 CONCLUSION

Research objective 1:

The main requirement for implementing SI-x in cloud computing was to have functionalities to perform computations over raster input data. The original SI-x input data was location specific and input was in the form of 1D arrays. However, in current cloud based SI-x, the input data was raster dataset. Therefore, the requirement of cloud computing platform was to have functionalities for raster based operations and math algebra.

From the various cloud services, the research work needed IaaS and PaaS services for implementation of SI-x. The IaaS as a service provided data storage and management services for large datasets. Secondly, the programming environment to implement the model on cloud platform. The advantage of GEE was dataset required for the research was already available. The data store provides a set of dataset as well as possibility to upload custom datasets. These pre-processed datasets are also available for download and use for non-commercial purposes.

Research objective 2:

A new conceptual model was designed to adapt the model for GEE-based implementation. Further, implementation on GEE resulted in prediction results at continental scale for 33 years. A regional scale analysis showed slight trend of early spring. However, this case needs to be further studied in combination with other factors such as latitude, altitude and ecosystems. GEE was assessed for advantages and disadvantages.

Research objective 3:

For comparison of correlation of model results and volunteered data, the model results were obtained at different spatial scale for locations of volunteered data. The conclusions of the analysis, is that scale at which model results are obtained and further correlated needs further study.

7.2 RECOMMENDATIONS

The following recommendation for future work is proposed. The new implementation of the SI-x needs to be further tested. This has the potential to be extended for global scale. Further studies can be done to correlate the volunteered data with model results using spatial analysis for whole of USA at different spatial scales.

References

- Ahas, R., Aasa, R., Menzel, a., Fedotova, V. G., & Scheifinger, H. (2002). Changes in European spring phenology. *International Journal of Climatology*, 22, 1727–1738. doi: 10.1002/joc.818
- Armbrust, M., Stoica, I., Zaharia, M., Fox, A., Griffith, R., Joseph, A. D., ... Rabkin, A. (2010). *A view of cloud computing*. doi: 10.1145/1721654.1721672
- Ault, T., & Zurita-Milla, R. (2013). *A matlab toolbox for calculating spring indices from daily meteorological data*.
- Aysan, A. I., Yigit, H., & Yilmaz, G. (2011, June). GIS applications in cloud computing platform and recent advances. In *Proceedings of 5th international conference on recent advances in space technologies - rast2011* (pp. 193–196). IEEE. Retrieved from <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5966820> doi: 10.1109/RAST.2011.5966820
- Buyya, R., Yeo, C. S., Venugopal, S., Broberg, J., & Brandic, I. (2009, June). Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, 25(6), 599–616. Retrieved from <http://linkinghub.elsevier.com/retrieve/pii/S0167739X08001957> doi: 10.1016/j.future.2008.12.001
- Calheiros, R., & Ranjan, R. (2011, January). CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice ...*, 41(1), 23–50. Retrieved from <http://doi.wiley.com/10.1002/spe.995><http://onlinelibrary.wiley.com/doi/10.1002/spe.995/full?cm=email-eng\&cs=if-2012\&cu=psj-13-54122\&cd=psj-13-54122-compsci-speccloudsim> doi: 10.1002/spe.995
- Castañe, G. G., Núñez, A., Filgueira, R., & Carretero, J. (2012). Dimensioning Scientific Computing Systems to Improve Performance of Map-Reduce based Applications. *Procedia Computer Science*, 9, 226–235. doi: 10.1016/j.procs.2012.04.024
- Cleland, E. E., Chuine, I., Menzel, A., Mooney, H. a., & Schwartz, M. D. (2007, July). Shifting plant phenology in response to global change. *Trends in ecology & evolution*, 22(7), 357–65. Retrieved from <http://www.ncbi.nlm.nih.gov/pubmed/17478009> doi: 10.1016/j.tree.2007.04.003
- Foster, I., Zhao, Y., Raicu, I., & Lu, S. (2008). Cloud computing and grid computing 360-degree compared. *Grid Computing Environments Workshop 2008 (GCE '08)*, 1–10. Retrieved from <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4738445> doi: 10.1109/GCE.2008.4738445
- Gao, S., Li, L., Li, W., Janowicz, K., & Zhang, Y. (2014, March). Constructing gazetteers from volunteered Big Geo-Data based on Hadoop. *Computers, Environment and Urban Systems*, 1–15. Retrieved from <http://linkinghub.elsevier.com/retrieve/pii/S0198971514000209> doi: 10.1016/j.compenvurbsys.2014.02.004
- GEE. (2012). Retrieved from <https://earthengine.google.org/#intro>
- Goodchild, M. F. (2007). *Citizens as sensors: The world of volunteered geography* (Vol. 69). Retrieved from <http://link.springer.com/article/10.1007/s10708-007-9111-y> doi: 10

- .1007/s10708-007-9111-y
- Hansen, M. C., Potapov, P. V., Moore, R., Hancher, M., Turubanova, S. a., Tyukavina, A., ... Townshend, J. R. G. (2013, November). High-resolution global maps of 21st-century forest cover change. *Science (New York, N.Y.)*, 342(6160), 850–3. Retrieved from <http://www.ncbi.nlm.nih.gov/pubmed/24233722> doi: 10.1126/science.1244693
- Hu, L., Yue, P., & Zhou, H. (2012, August). Geoprocessing in Google Cloud Computing: Case studies. *2012 First International Conference on Agro- Geoinformatics (Agro- Geoinformatics)*, 1–6. Retrieved from <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6311653> doi: 10.1109/Agro-Geoinformatics.2012.6311653
- Li, M., Qu, J. J., & Hao, X. (2010, September). Investigating phenological changes using MODIS vegetation indices in deciduous broadleaf forest over continental U.S. during 2000–2008. *Ecological Informatics*, 5(5), 410–417. Retrieved from <http://linkinghub.elsevier.com/retrieve/pii/S1574954110000543> doi: 10.1016/j.ecoinf.2010.04.002
- Licari, D. (2010). MapReduce. , 1–5.
- Mattmann, C. a., Waliser, D., Kim, J., Goodale, C., Hart, A., Ramirez, P., ... Hewitson, B. (2013, July). Cloud computing and virtualization within the regional climate model and evaluation system. *Earth Science Informatics*, 7(1), 1–12. Retrieved from <http://link.springer.com/10.1007/s12145-013-0126-2> doi: 10.1007/s12145-013-0126-2
- Menzel, A. (2000). Trends in phenological phases in Europe between 1951 and 1996. *International journal of biometeorology*, 44(August 1999), 76–81. doi: 10.1007/s004840000054
- NIST. (2011). *The NIST Definition of Cloud Computing* (Tech. Rep.). Author.
- Parmesan, C. (2007). Influences of species, latitudes and methodologies on estimates of phenological response to global warming. *Global Change Biology*, 13, 1860–1872.
- Parmesan, C., & Yohe, G. (2002). A globally coherent fingerprint of climate change impacts across natural systems. *Nature(London)*, 421, 37–42.
- Petcu, D., Panica, S., Frîncu, M., Neagul, M., Zaharie, D., Macariu, G., ... ÂtfeĂCnuĂĈ, T. (2012, November). Experiences in building a Grid-based platform to serve Earth observation training activities. *Computer Standards & Interfaces*, 34(6), 493–508. Retrieved from <http://linkinghub.elsevier.com/retrieve/pii/S0920548911001115> doi: 10.1016/j.csi.2011.10.010
- Rosemartin, A. H., Crimmins, T. M., a.F. Enquist, C., Gerst, K. L., Kellermann, J. L., Posthumus, E. E., ... Weltzin, J. F. (2014, May). Organizing phenological data resources to inform natural resource conservation. *Biological Conservation*, 173, 90–97. Retrieved from <http://linkinghub.elsevier.com/retrieve/pii/S0006320713002334> doi: 10.1016/j.biocon.2013.07.003
- Sagarin, R. (2001). False estimates of the advance of spring. *Nature*, 414(December), 600. doi: 10.1038/414600a
- Schwartz, M. (1994, March). Monitoring global change with phenology: The case of the spring green wave. *International Journal of Biometeorology*, 38(1), 18–22. Retrieved from <http://link.springer.com/10.1007/BF01241799> doi: 10.1007/BF01241799
- Schwartz, M. (1997). Spring Index Models: An approach to connecting Satellite and surface phenology. In H. Lieth & M. D. Schwartz (Eds.), *Phenology of seasonal climates* (pp. 23–38). Netherlands: Backhuys.
- Schwartz, M., Ault, T. R., & Betancourt, J. L. (2013, November). Spring onset variations and trends in the continental United States: past and regional assessment using temperature-based indices. *International Journal of Climatology*, 33(13), 2917–2922. Retrieved from <http://doi.wiley.com/10.1002/joc.3625> doi: 10.1002/joc.3625
- Schwartz, M., Betancourt, J. L., & Weltzin, J. F. (2012, August). From Caprio’s lilacs to the

- USA National Phenology Network. *Frontiers in Ecology and the Environment*, 10(6), 324–327. Retrieved from <http://www.esajournals.org/doi/abs/10.1890/110281> doi: 10.1890/110281
- Schwartz, M., & Caprio, J. (2003). North american first leaf and first bloom lilac phenology data. *IGBP PAGES/World Data Center for Paleoclimatology Data*, 2003–078.
- Schwartz, M., Hanes, J. M., & Liang, L. (2013, September). Separating temperature from other factors in phenological measurements. *International journal of biometeorology*. Retrieved from <http://www.ncbi.nlm.nih.gov/pubmed/23995622> doi: 10.1007/s00484-013-0723-2
- Schwartz, M., Reed, B. C., & White, M. a. (2002, November). Assessing satellite-derived start-of-season measures in the conterminous USA. *International Journal of Climatology*, 22(14), 1793–1805. Retrieved from <http://doi.wiley.com/10.1002/joc.819> doi: 10.1002/joc.819
- Schwartz, M., & Reiter, B. E. (2000, June). Changes in North American spring. *International Journal of Climatology*, 20(8), 929–932. Retrieved from [http://doi.wiley.com/10.1002/1097-0088\(20000630\)20:8<929::AID-JOC557>3.0.CO;2-5](http://doi.wiley.com/10.1002/1097-0088(20000630)20:8<929::AID-JOC557>3.0.CO;2-5) doi: 10.1002/1097-0088(20000630)20:8<929::AID-JOC557>3.0.CO;2-5
- Sehgal, S., Erdelyi, M., Merzky, A., & Jha, S. (2011). Understanding application-level interoperability: Scaling-out MapReduce over high-performance grids and clouds. *Future Generation Computer Systems*, 27(5), 590–599. Retrieved from <http://dx.doi.org/10.1016/j.future.2010.11.001> doi: 10.1016/j.future.2010.11.001
- Stöckli, R., & Vidale, P. L. (2004, September). European plant phenology and climate as seen in a 20-year AVHRR land-surface parameter dataset. *International Journal of Remote Sensing*, 25(17), 3303–3330. Retrieved from <http://www.tandfonline.com/doi/abs/10.1080/01431160310001618149> doi: 10.1080/01431160310001618149
- Studer, S., Appenzeller, C., & Defila, C. (2005, December). Inter-Annual Variability and Decadal Trends in Alpine Spring Phenology: A Multivariate Analysis Approach. *Climatic Change*, 73(3), 395–414. Retrieved from <http://link.springer.com/10.1007/s10584-005-6886-z> doi: 10.1007/s10584-005-6886-z
- Thornton, P., Thornton, M., Mayer, B., Wilhelmi, N., Wei, Y., Devarakonda, R., & Cook, R. (2014). *Daymet: Daily surface weather data on a 1-km grid for north america, version 2. data set. available on-line [http://daac.ornl.gov] from oak ridge national laboratory distributed active archive center, oak ridge, tennessee, usa. date accessed: 2014/05/20. temporal range: 1980/01/01 - 2013/12/31 (Tech. Rep.). Spatial range: N=53.0, S=10.0, E=-49.9, W=-133.5.*
- Thornton, P. E., & Running, S. W. (1999, March). An improved algorithm for estimating incident daily solar radiation from measurements of temperature, humidity, and precipitation. *Agricultural and Forest Meteorology*, 93(4), 211–228. Retrieved from <http://linkinghub.elsevier.com/retrieve/pii/S0168192398001269> doi: 10.1016/S0168-1923(98)00126-9
- Wang, H., Dai, J., & Ge, Q. (2014). Comparison of Satellite and Ground-Based Phenology in China's Temperate Monsoon Area. *Advances in Meteorology*, 2014, 1–10. Retrieved from <http://www.hindawi.com/journals/amete/2014/474876/> doi: 10.1155/2014/474876
- Wang, P., Wang, J., Chen, Y., & Ni, G. (2013, October). Rapid processing of remote sensing images based on cloud computing. *Future Generation Computer Systems*, 29(8), 1963–1968. Retrieved from <http://linkinghub.elsevier.com/retrieve/pii/S0167739X13001015> doi: 10.1016/j.future.2013.05.002
- Wang, T., Ottlé, C., Peng, S., Janssens, I. a., Lin, X., Poulter, B., ... Ciais, P. (2014, May). The influence of local spring temperature variance on temperature sensitivity of spring phenology.

- Global change biology*, 20(5), 1473–80. Retrieved from <http://www.ncbi.nlm.nih.gov/pubmed/24357518> doi: 10.1111/gcb.12509
- White, A., & Thomton, P. E. (1997). A continental responses phenology model climatic for monitoring variability vegetation to interannual. , 11(2), 217–234.
- White, M. A., de Beurs, K. M., Didan, K., Inouye, D. W., Richardson, A. D., Jensen, O. P., ... Lauenroth, W. K. (2009, October). Intercomparison, interpretation, and assessment of spring phenology in North America estimated from remote sensing for 1982–2006. *Global Change Biology*, 15(10), 2335–2359. doi: 10.1111/j.1365-2486.2009.01910.x
- Xiaoqiang, Y., & Yuejin, D. (2010, June). Exploration of cloud computing technologies for geographic information services. In *2010 18th international conference on geoinformatics* (pp. 1–5). IEEE. Retrieved from <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5567515> doi: 10.1109/GEOINFORMATICS.2010.5567515
- Yang, C., Raskin, R., Goodchild, M., & Gahegan, M. (2010, July). Geospatial Cyber-infrastructure: Past, present and future. *Computers, Environment and Urban Systems*, 34(4), 264–277. Retrieved from <http://linkinghub.elsevier.com/retrieve/pii/S0198971510000268> doi: 10.1016/j.compenvurbsys.2010.04.001
- Zhang, X., Friedl, M. a., Schaaf, C. B., Strahler, A. H., Hodges, J. C., Gao, F., ... Huete, A. (2003, March). Monitoring vegetation phenology using MODIS. *Remote Sensing of Environment*, 84(3), 471–475. Retrieved from <http://linkinghub.elsevier.com/retrieve/pii/S0034425702001359> doi: 10.1016/S0034-4257(02)00135-9

Appendix A

Appendix-A

Code for SI-x implemented in GEE

```
Code written by Riddhi Munde**/ Version Spring Index Model Script v5.0**/
var startdate = 0; var enddate = 60;

PART 1-SELECTION OF DAYMET INPUT DATA FOR A YEAR ***/
var collection = ee.ImageCollection("NASA/ORNL/DAYMET")
.filterDate('2000-01-01', '2000-07-31');
var doy_filter = ee.Filter.calendarRange(startdate, enddate, 'day_of_year');
var sub_collection = collection.filter(doy_filter);

/**Part 2-CALCULATE GROWING DEGREE HOUR TEMPERATURE(GDH)**/
//function to convert temperature
function convert_temp(image)
var f = image.multiply(1.8).add(32);
return f;
function GDH(im)
//convert temperature for tmax and tmin band
var tmaxband = im.expression("b('tmax') * 1.8 + 32");
//print(s);
var tminband = im.expression("b('tmin') * 1.8 + 32");
//print(w);
//create a blank image for further calculations on bands
var image = ee.Image().addBands(tmaxband).addBands(tminband);
//find temperature difference
var temp_diff = tmaxband.subtract(tminband);
image = image.addBands(temp_diff);
//select the daylength-convert second to hours
var daylen = im.select("dayl").divide(3600);
var ideal_dl = daylen.round();
image = image.addBands(daylen).addBands(ideal_dl);
//create band for hourly day temperature
var bs_temp = 31;
var t = ee.Image();
var temp = ee.Image();
var sunset = image.expression("b('tmax_1') * sin(3.1416/(b('dayl')+4)* b('dayl')) + b(2)");
image = image.addBands(sunset);

*Create hourly time stamp of the image*/
for (var i=1; i<24; i++)
temp = image.expression(
" (' + i + " == 1) ? b(2)
: (' + i + " <= (b(5)+1)) ? b('tmax_1') * sin(3.1416/(b(4)+4)*" + i + ") + b(2)
: b(6)-(b(6) - b(2))/(log(24 - b('dayl')))* log(" + i + ") ) ) "
//calculate growing degree hours by subtracting base temperature
temp = temp.subtract(bs_temp);
//if growing hours value less than zero,assign zero value
temp = temp.expression(" (b(0) < 0) ? 0 : b(0) ")
t = t.addBands(temp);
//sum the growing degree hours GDH
var gdh = t.reduce(ee.Reducer.sum());
return im.addBands(gdh);
/*Map the GDH fuction*/
var gdh_col = sub_collection.map(GDH);
//print('Collection with computed GDH',gdh_col);
/*****END OF PART 2*****/

/*PART 3 - CALCULATE THE FOUR PREDICTOR VARIABLES*/
/*PART 3.1 -Calculating 1st Variable -DDE2*/
//Select the result for summation of 3 images prior to the day
var temp_sum = (gdh_col.select('sum'));
//function to merge the 3 images
function add_dde2(image)
var sys_time = ee.Number(image.get("system:time_start"));
var list_collection = ee.ImageCollection.fromImages(image.get('match'));
var sum = ee.Image(list_collection.sum());
return sum.copyProperties(image, ['system:index', 'system:footprint']).set('system:time_start':
sys_time.add(172800000));
//select all images in the range
var join = ee.Join.saveAll('matchesKey': 'match', 'ordering': 'system:time_start');
//select images from 2 days prior
var prior_filter = ee.Filter.greaterThanOrEquals('leftField': 'system:time_start', 'rightField': 'system:time_start');
var filter = ee.Filter.and(ee.Filter.maxDifference('difference': 1000*60*60*24*2, 'leftField': 'system:time_start', 'rightField':
'system:time_start'), prior_filter);
//apply the join to the image collection
var join_result = ee.ImageCollection(join.apply(temp_sum, temp_sum, filter));
//call the function to sum 3 day temperature groups
```

```

var dde2 = join_result.map(add_dde2);
//print('accumulation of temperature',dde2);
/* Multiply DDE2 by its Co-efficient for LILAC */
var DDE2 = dde2.map(function(image)
return image.expression('b(0)*0.201').copyProperties(image,['system:index','system:time_start','system:footprint']));
);
//print('Variable 1- DDE2',DDE2);
/*END OF PART 2.1*/
/*PART 2.2 -Calculate 2nd Variable - DD57 */
//function to merge the 3 images
function add_dd57(image)
var sys_time = ee.Number(image.get('system:time_start'));
var list_collection = ee.ImageCollection.fromImages(image.get('match'));
var sum = ee.Image(list_collection.sum());
return sum.copyProperties(image,['system:index','system:footprint']).set('system:time_start':
sys_time.add(432000000));
var dd57 = join_result.map(add_dd57);
//print('accumulation of temperature',dd57);
/* Multiply the DD57 value by its Co-efficient for LILAC */
var DD57 = dd57.map(function(image)
return image.multiply(0.153).copyProperties(image,['system:index','system:time_start','system:footprint']));
);
//print('Variable 2- DD57',DD57);
/*END OF PART 2.2*/
/*PART 2.3 - Calculating 3rd Variable - SYNOP*/
//create a binary for all pixel value > 637
function check_637(image)
var output = image.expression('b(0) >= 637 ? 1 : 0');
return output.copyProperties(image,['system:index','system:time_start','system:footprint']);
//call the check_637 function to mask out the pixels with value greater than 637
var synEve = dde2.map(check_637);
//print('binary imagecollection',synEve);
//function to sum all images from Day 1 at every timestamp
function sumImage(image)
var list_collection = ee.ImageCollection.fromImages(image.get('match'));
var sum = (list_collection.sum());
return sum.mask(sum).copyProperties(image,['system:index','system:time_start','system:footprint']);
//select all images in the range
var join = ee.Join.saveAll('matchesKey': 'match','ordering': 'system:time_start');
//Filter to select all images from Day 1
var prior_filter = ee.Filter.greaterThanOrEquals('leftField': 'system:time_start','rightField': 'system:time_start');
for(var i = 0; i < enddate + 2; i++)
var filter =
ee.Filter.and(ee.Filter.maxDifference('difference': 1000*60*60*24*i, 'leftField': 'system:time_start', 'rightField':
'system:time_start'), prior_filter);
//apply the join to the image collection
var join_result = ee.ImageCollection(join.apply(synEve, synEve, filter));
//Perform summation of all images
var synval = ee.ImageCollection(join_result.map(sumImage));
//print('Variable 3 - Synval result',synval);
/* Multiply SYNOP with Co-efficient for LILAC */
var SYNOP = synval.map(function(image)
return image.float().multiply(13.878).copyProperties(image,['system:index','system:time_start','system:footprint']));
/*PART 2.4 Calculating 4th Variable - MDS0*/
//create custom image collection
var first_time = ee.Number(946684800000);
var blank = [];
for(var i = startdate; i <= enddate+2; i++)
var numb = ee.Number(86400000).multiply(i);
var p = ee.Image(i).set('system:time_start' : first_time.add(numb));
blank.push(p);
var mds0 = ee.ImageCollection.fromImages(blank);
/* Multiply mds0 value by its Co-efficient for LILAC */
var MDS0 = mds0.map(function(image)
return image.float().multiply(3.306).copyProperties(image,['system:time_start']));
//print('Variable 4 - Day since 1st January',MDS0);
/* PART 3.5 - JOIN THE IMAGE COLLECTIONS */
var MergeBands = function(element)
var i = ee.Image.cat(element.get('primary'), element.get('secondary'));
return i.float();
// Define the join and filter.
var join = ee.Join.inner();
var filter = ee.Filter.equals('system:time_start', null, 'system:time_start');
//apply the join to the image collections DDE2 AND DD57
var join1Result = join.apply(DDE2, DD57, filter);
// Merge the bands of both images to create a single image.
var DD2_DD57 = ee.ImageCollection(join1Result.map(MergeBands));
/* JOIN2 */
//join result of join1 to SYNOP
var join2Result = join.apply(SYNOP, MDS0, filter);
// Merge the bands of both images to create a single image.
var SYN_DSO = ee.ImageCollection(join2Result.map(MergeBands));
/* JOIN3 */
//join 1 and 2
var join3Result = join.apply(DD2_DD57, SYN_DSO, filter);
// Merge the bands of both images to create a single image.
var predCollection = ee.ImageCollection(join3Result.map(MergeBands));
//For above collection, b(0)-DDE2, b(1)-DD57, B(2)-SYNOP, B(3)-MDS0

/* PART 4 - REGRESSION FUNCTION */
// Add a Day of Year band.
var predCollectionWithDoY = predCollection.map(function(img)
var date = ee.Date(img.get('system:time_start'));
var doy = date.getRelative('day', 'year');

```

```
return img.addBands(ee.Image(doy).float().select([0], ['doy']));
//make first-to -last inversion of collection
var descCollection = predCollectionWithDoY.sort('system:time_start', false);
// Mask out values with band sum < 1k.
var predCollectionMasked = descCollection.map(function(img)
return img.mask(img.expression("b(0)+b(1)+b(2)+b(3) >= 1000"));
//print('predCollectionMasked',predCollectionMasked);
//create mosaic so last on top image
var mosai = predCollectionMasked.mosaic();
print(mosai,"final image");
//bounding region for USA
var usRegions = ee.FeatureCollection('ft:1fRY18cjsHzDgGijIS2nnpUU3v9JPDc2HNaR7Xk8');
var fImage = mosai.clip(usRegions);
//VISUALIZE THE FINAL RESULT IMAGE
Map.addLayer(fImage, 'bands': 'doy', 'min': 0, 'max': 60, 'palette': 'FFFFFF,0000FF', 'final Result Image');
/*****END OF PART 4*****/

/* PART 5- Mean value for a ploygon*/

var fImage = mosai.clip(usRegions)
var t = fImage.select("doy").clip(rect);
var ty = t.reduceRegion(ee.Reducer.mean(),rect,1000);
print(ty);
/*****END OF PART 5*****/

/*PART 6 Æ Get value at a point */
var fi = ee.ImageCollection.fromImages([mosai])
var points = ee.Geometry.Point(-100.07,30.56);
var res = fi.getRegion(points, 1000);
print(res);

/*****END OF PART 6*****/

/*PART 7 Æ Get time*/
function timeIt(label, o)
var start = Date.now();
print(label);
var result = o.getInfo();
print([' + (Date.now() - start) + '], result);

/*****END OF PART 7*****/
```