Indian Journal of Fundamental and Applied Life Sciences ISSN: 2231– 6345 (Online)
An Open Access, Online International Journal Available at www.cibtech.org/sp.ed/jls/2015/02/jls.htm
2015 Vol. 5 (S2), pp. 1467-1475/Amirabadi and Mohammadesmail
Research Article

THE UTILIZATION OF CLOUD COMPUTING FOR ANALYZING THE DATABASES OF VERY LARGER WEATHER MAPS USING THE LOAD DISTRIBUTION ASSISTANT ALGORITHM

Roghayyeh Masoumpour Amirabadi and *Sedigheh Mohammadesmail
Department of Library and Information Science, Science and Research Branch, Islamic Azad University, Tehran, Iran
**Author for Correspondence*

ABSTRACT
This study aims at investigating the cloud computing service and its applications in the field of storing and analyzing the very large weather images. These services are implemented through different cooperative and distributed agents. A region-oriented data structure is utilized to store and analyze the image and it stores and describes the image regions through the low-level descriptors. Different types of structural relationships are also considered among the regions. The specific aim of this operation is to make the data operations compatible to work in a distributed state. This divides an input weather image into several smaller sub-images to be separately stored and processed by different agents in system and thus it facilitates the parallel processing of very large weather images. The utilization of appropriate load balancer for distribution and allocation of agents with less workload is the key aspect in reducing the processing time for parallel operations.

Keywords: *Cloud Computing, Weather Maps, Load Distribution Assistant Algorithm*

INTRODUTION
A very large amount of multimedia data is produced worldwide every day. The meteorological institutions are producing the maps and images of very large data around the world. The techniques are required for storing and processing to manage these databases. In particular, the volume of data and difficulty of processing the very large images and multidimensional weather maps are the challenging problems. The ultimate goal of cloud computing service is to store and analyze very large databases. By this way, these services enable the users to access the storage resources and provide the additional virtual computing. In particular, they facilitate the analysis of very large images by providing the physical and computing resources. Nowadays, there are some types of very large images in some certain areas due to the exponential increase in the precision and quality of imaging equipment (Moghaddam and Asa, 2013). For instance, the "Blue Marble" project by NASA (The National Aeronautics and Space Administration) has obtained the images with the sizes of more than 1 GB (PNG file). Working with these images needs the large memory and processing resources. This management of databases is difficult and their sizes are constantly increasing (Iranpak *et al.* , 2012).
Our studied sample is focused on such these very large weather images with a large number of regions. This indicates that the storage and processing of image data requires the large computing resources and time. Paralleling and distributed processing of database are utilized to reduce the storage and processing requirements. The pilot model of cloud computing service is developed for storing and analyzing the images. These services store the relevant image and data. Their regions and relations are utilized as the basic entities for display and processing of an image.
Therefore, this system should be able to: 1) extract the image regions (and their relevant relations), 2) process, and 3) store them. The existing problem is that there are usually large numbers of these regions in an image and their relations can be fairly complex.
A data source is implemented for storing the very large images in current cloud computing services. A data structure for image analysis should easily and quickly provide the current information management. In our system, the information in the data structure is stored in a database. Furthermore, it is clear for service users whether they are operating a database (Yuen, 2010).

The parallelism can be utilized for large sets of images. In most cases, such this parallelism simply includes the classification of images into separate databases and this increases the performance by simultaneous parallel search of image in different databases. In addition to such this parallelism, we have created a segmentation algorithm for very large images and it divides an input image into several sub-images which can be stored in different databases and processed by different machines (in certain operations);then each sub-image can be individually evaluated as an independent image or as a part of a bigger image. Subsequently, the data structure is accepted for distributed processing. Our multi-agent system provides a clear uniform access to all components of sub-images (Knutson, 2007).

MATERIALS AND METHODS

The basic data structure for an image analysis system is an appropriate data structure for storing the information of an image. The structural relationships between the apparent entities in an image should be taken into account. Each region is characterized by its physical characteristics, including the color, shape, and location of applied descriptors. This display also makes it possible to determine multiple structural relationships among different regions of image.

A diagram-based data structure is utilized in our system for storing the information of an input image. Each image region (essentially a connected set of pixels of images with the same subject value) is displayed as a graph node of data structure, and each region relationship, which connects a region to another region, indicates the arc of diagram.

The schema of relational database which contains the node description of regions (and their relationships) is developed and shown in Table 1.

Table 1: Classification of descriptors obtained for an image region

Color descriptors	Shape descriptors	Relationships
red level average	area	adjacent-to
green level average	major axis size	is-part-of
blue level average	minor axis size	related-to
red level mode	major axis orientation	disjoint-with
green level mode	minor axis orientation	
blue level mode	centroid	
red level variance	set of region points	
green level variance	set of border points	
blue level variance	average of centroid-border distance	
maximum red level value	variance of centroid-border distance	
maximum green level value	number of sides estimation	
maximum blue level value		
minimum red level value		
minimum green level value		
minimum blue level value		

(Source: Nurmi et al., 2009)

Segmentation Algorithm

The segmentation algorithm is utilized for distributing the sections of an image in different databases (Ghannadpour *et al.*, 2011). This topic itself is complex and important although we will focus on describing the cloud computing services and increased efficiency in this paper in order to investigate the limitations and relationships between the image separation process and sum of the diagram-based operations.

The segmentation algorithm divides the input image along its largest dimension (width or height) in order to obtain three sub-images of IC, IB and IA through the following algorithm:

FOR EACH region IN division Line DO

IF (NOT is Treated(region)) THEN
RegionSet = getSimilarRegions (region)
FOR EACH region_r IN regionSet DO
markAsTreated(region_r)
END FOR
regUnion = \begin{math}\cup\end{math} regionSet
// regUnion : union of regions in regionSet
I_A = I_A \begin{math}\setminus\end{math} regUnion
I_B = I_B \begin{math}\cup\end{math} regUnion
I_C = I_C \begin{math}\set minus\end{math} regUnion
END IF
END FOR

The division line is the middle line along the largest dimension of image (height or width). Obtaining the similar regions: The input region and the adjacent regions, which are similar, grow as far as a similar situation is not created. If all following relations are provided, two regions A and b will be similar:

$$\frac{|r_a - r_b|, |g_a - g_b|, |b_a - b_b|}{3} < th_{avg}$$

Note: (ra, ga, ba) and (rb, gb, bb) are three RGB values of regions a and b.

Three sub-images are obtained in this method. The central part of image (sub-image IB) is included in division processing and is ready for entry into the database. On the other hand, the left and right parts of input image (sub-images IC, IA) are the non-processed images and can be sent by other computers for operation. Some of the additional information and a part of central sub-image IB are sent along with the sub-image IC, IA. This information is utilized to describe the regional adjacencies which are among the adjacent sub-images (Burks *et al.*, 2000). It is a developed message passing interface to adapt the whole image to the information lagged in sub-images. Since the sub-images IC and IA have initially the information about the adjacent IB regions, IB should be informed of its external boundary when IC and IA are processed and stored (principally in remote database).

Finally, when the adaptation phase ends, the image becomes ready for analysis through the diagram operations consistent with the parallel data structure.

RESULTS AND DISCUSSION

Results

Chart Operations

The determined operations are based on the region-based data structure. We mainly utilize the morphology diagram-based operations. The initial diagram adjacencies, which include each adjacent region and its neighbor, are usually applied in this regard (Sadashiv& Kumar, 2011).

Implementing the Cloud Computing

The cloud computing has become a multi-agent implementation system. A schema of system is shown in Figure 1. First, a parallel and distributed system should determine how the operations are allocated to an agent. To minimize the implementation and response time, the implemented system utilizes a balancing method for distributing the operations among the agents. Each agent has a queue of local operations, but a central information agent. The studies are conducted on the advantages of using the load balancing algorithms for several agents systems, and parallel and grid systems (Lu, 2011).

Simultaneously, the number of network packets sent by transferring the operations is directly decreased by the local queue in agents. These local queues are rebalanced when an agent becomes very busy. The load balancing uses three main functionalities:

1- Advertising and discovering the services are applied by different agents in order to be put in a location where the other agents can access to them and maintain their own positions.

2- Prediction of performance is done for evaluating the new operations for a certain agent in order to balance the operation time for different agents. Each operation agent stores its system for any type of operation (division, storage, and operations of morphology). Using the operational parameters (e. g. the non-operated number of pixels or regions), the run time is predicted using a regression logarithmic function and the operation is assigned to an agent which has less estimated waiting processing time.
3- The queues of activities are utilized for planning purposes. When a new operation is allocated to an agent, the other activities may be implementing by this agent. Each agent puts the waiting activities in a local queue (Deelman *et al.*, 2008).

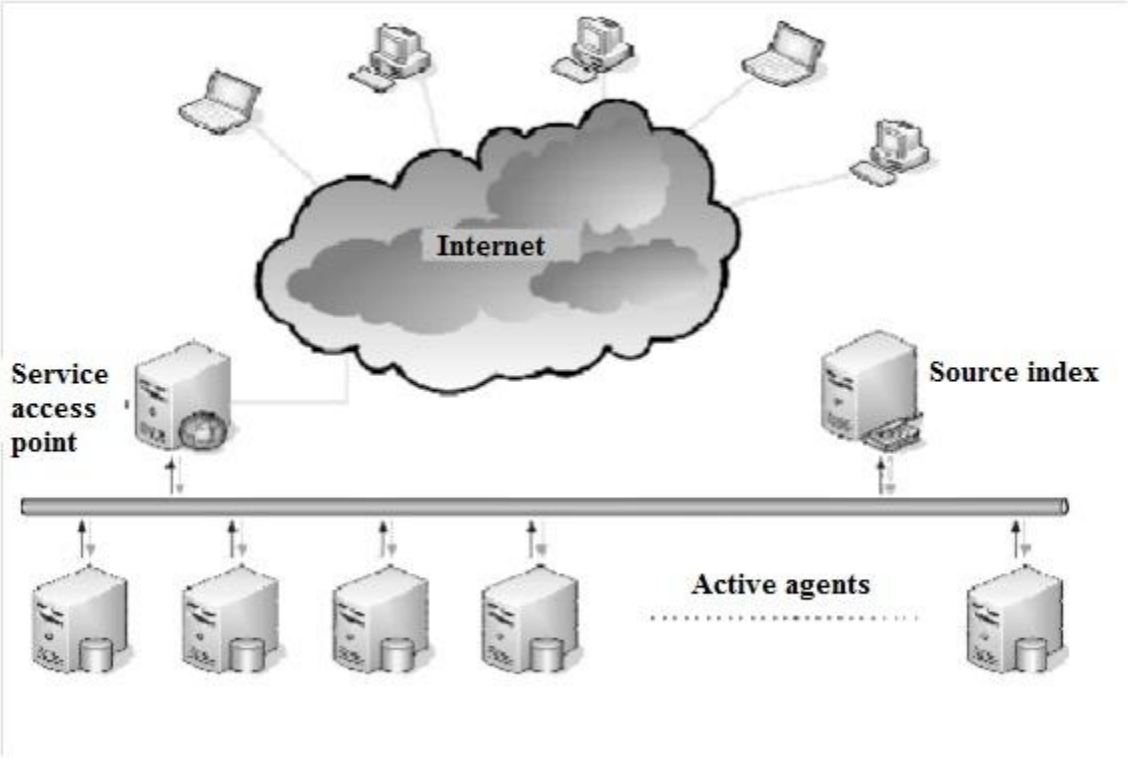

Figure 1: Schema of cloud computing service

Allocating the role of agents, which are present in a system, is an important issue in multi-agent systems. As shown in Figure 1, there are three agents in our system:

- The point of access agent to service is a unique agent in system. This agent is the entry point for end users. It suggests the procedures for storing and applying the cloud computing service operations.
- The source index agent is also a unique agent in system. This agent implements the system guide. This agent collects the information about the active agents in the system. The collected information includes the location of each agent, its current load, the estimation or performance prediction function and the image processing mode.
- Finally, the active agents play the main roles in system. Each active agent manages a relational database including the complete data structure described in data structure section. They have all functionalities for storage and analysis of images on themselves. Therefore, each active agent is able to:

1) Separate an input image (when it is very big), 2) extract the regions of image and descriptors, 3) The image data storage in database structure and, 4) do the certain chart-based filtering and operations on the images stored in the database (Beul *et al.*, 2010).
Furthermore, these agents have a message sending interface for two adapted data of separated images and sent sub-images obtained from the segmentation algorithm and it should be applied by other agents. The

active agents are also able to analyze their system load to calculate the estimated time of waiting implementation time which is already described. This data acquisition is vital for balancing the load and it will be described in the next section.

The implemented multi-agent system is developed using the Sun Java. Each active agent can be implemented in heterogeneous hardware with heterogeneous database management systems and in different operating systems. This system can be easily be updated with registering a new agent in source index. The number of agents is not limited in the system (Overpeck *et al.*, 2011).

Description of Load Balancing

In implemented system, the load balancing is initially utilized to divide and store the image. This is the first step in an image entry. Agent 1) It receives an image for processing, 2) It decides whether the image should be divided, 3) It divides the image if necessary (using the algorithm described in the section of segmentation algorithm) and calculates the regions and its node characterization, 4) It stores the regions and descriptors obtained in data structure, and finally 5) If the image is divided, it sends two non-operated sub-images to two agents (with the lowest pressure). Step 3) requires processing all pixels of sub-images. Therefore, our system utilizes a number of non-operated pixels of received sub-image to evaluate the waiting implementation time. It also makes it possible to calculate an estimation of factor loadings which are required in Step 5) to distribute the activities among the agents. The local evaluation is taken into consideration in all waiting activities in local line of agents (Andrew *et al.*, 2007).

The performed operations on different sub-images do not usually require the load balancing. This is because the operation is directly applied to the data structure where the sub-image is stored. The performance of operations is dependent on the number of regions allocated to each agent.

The schedule for each operation stores the start time, number of non-operated pixels, and the end time. The schedule agent utilizes those three parameters of previous implemented operation programs to evaluate the time of processing the input activities. A regression algorithm is applied in this regard. Each operation has its own time in time evaluation agent. When the local queue is changed, the scheduled program updates the information in.

In this method, the estimated time for processing the waiting activities in all active agents is stored in resource index agent. Therefore, when an active agent should submit a new operation (a sub-image which should be processed by another active agent) this requires the resource index agent that determines which agent is currently less applied. The agents utilize the web services as the interface of sending message among them (Chierichetti *et al.*, 2010).

Furthermore, the load monitoring application is another Java application in active agents. This application is responsible for storing the consumption of system resources for program of performed operation. This monitoring application should only store the programs of agent which have the executive activities. With respect to the way of implementing the model of Sun Java application, all Java applications in a machine are run in the real Java machine process. Therefore, this load monitoring should store the CPU usage for each application. Sun JMX Beans model is used for this purpose. Using the CPU usage rate as well as the total processing time of each operation, we are able to modify the evaluation performance when more than anoperation is running at the same (Weller *et al.*, 2006).

Balancing the Number of Regions

It is important to perform the homogeneous regional data distribution among the agents in order to reduce the processing time for performed operations on images. First, the total number of regions is not clear. As a result, the load balancing of segmentation algorithm is based on the non-operated pixels. Therefore, the number of pixels stored in different agents is almost homogeneous, but the number of regions is not necessarily like this. An optional step is utilized for re-balancing the number of regions in agents in order to increase the efficiency of operations. When the sub-images enter, the information about the destination agent and the number of each sub-image are sent and stored in source index agent. Once all sub-images are processed, the source index can calculate the total number of image regions and the ideal number of regions per agent. This creates a re-balancing strategy for transferring some areas to balance a number of regions among the agents (Samantha *et al.*, 2005).

The use of Best Fit Descending Algorithm is the basis for this transfer strategy which is extensively applied in problems of sorting and compressing the executable file (Zhao *et al.*, 2009).

Discussion

In this section, we provide some of the experimental results for increasing the performance in implemented service by distribution of image division and storage for several input images. The advantages of using several factors along with the load balancing are considered in the case of using a factor (and thus the lack of using the distributed processing). Afterwards, a morphological delay operation is utilized to demonstrate the benefits of parallel operations.

Table 2: Description of experimental images

Image id	Image size Piecewise-constant regions	Relations between regions	Size in MB
Image 1	1000x10001,961	6,904	3. 81
Image 2	1067x17494,095	12,752	5. 33
Image 3	1232x150742,421	248,794	5. 31
Image 4	1596x217821,451	150,078	9. 94
Image 5	5000x500067,723	367,022	95. 3
Image 6	16601x92019,037,812	48,818,032	610. 3

Six experimental images of American meteorological project by NASA are selected and described in Table 2. They have different sizes (ranging from 1000*1000 to 9201*16601 pixels) and different numbers of regions with fixed pieces for testing the performance of services for a range of image sizes.

As shown in figures 3 and 4, the number of initial regions does not directly depend on the number of image pixels (an image can be very large, while the number of regions with fixed pieces is relatively low at the same time). The system performance and load balancing, machinery and similar operating systems are utilized for better study.

The agents are installed on the computer monitors with QuadCore processor with 4 GB of RAM and a 64-bit Linux operating system. Each experimental image enters into our image analysis system, and the system divides it into several sub-images. The recent images are separately processed. The performance of division and storage stage is increased while using different numbers of factors described in Figure 2. The processing time for an agent is based on the percentage where the sign of 100% represents the processing time required for completion of operation only by an agent. For the cases where more than an agent is used, the processing time of an operation is the maximum processing time of all participating agents. As shown in Figure 2(a), the meantime of image division and storage in the data structure through two agents is about the half of (49. 78%) of total time for storage through only an agent. This time is reduced by 35. 50% through three agents Figure (2 b). It can be concluded that the load balancing is almost effective for storing the images because the time required for extracting the initial regions and total storage of an image is proportional to the number of pixels in an image. The number of pixels is known as a primary reason. Then the operations can be balanced among different agents due to the mentioned evaluation functions. On the other hand, two sub-images with the same numbers of pixels can have different numbers of regions. In this case, the required time for storage step should consider the sub-images with more regions and relations. For instance, this occurs in figure 3 like what presented in Table 3.

A diagram-based morphological delay is separately applied on each sub-image. Figures 2 (c) and (d) show the processing time required for applying the delay on the whole image using two and three agents compared to the use of only an agent; it is equal to 60. 14% and 46. 46% respectively on average. The target operations are regional-oriented in the system. Therefore, the distribution of number of regions among different factors (Table 3) is a factor which mainly determines the reduction of processing time. As mentioned before, all operations cannot be calculated in distributed states like this example. Some of

the operations cannot generally be consistent with the segmentation algorithm, while others have this ability.

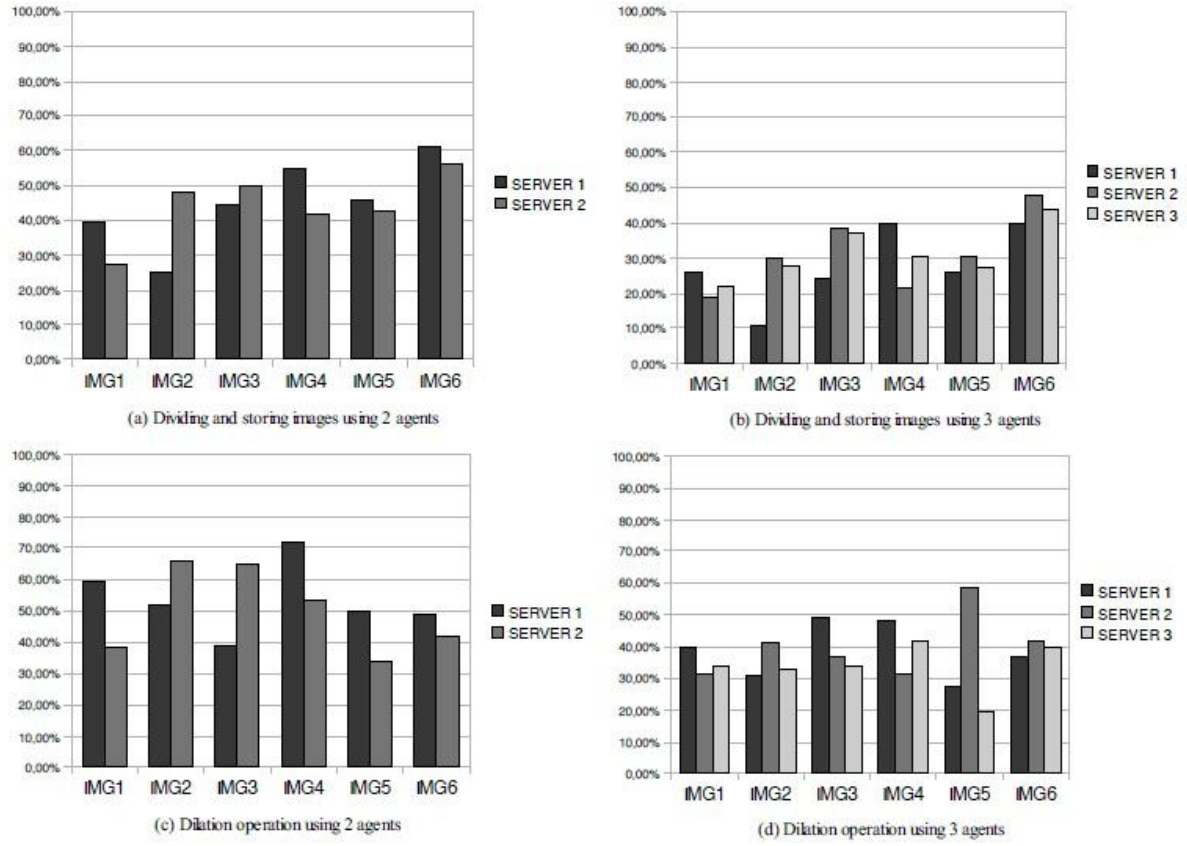

Figure 2: Time comparison with different numbers of agents

Table 3: Division of regions in different agents

Image id	One Agent	Two Agents	Three Agents
Image 1	[1,961]	[1,159; 802]	[872; 676; 413]
Image 2	[4,095]	[1,999; 2,196]	[1,202; 1,518; 1,375]
Image 3	[42,421]	[19,869; 22,552]	[16,949; 13,457; 12,015]
Image 4	[21,451]	[12,828; 8,623]	[8,630; 5,084; 7,737]
Image 5	[67,723]	[38,682; 29,041]	[20,121; 31,920; 15,682]
Image 6	[9,037,812]	[4,881,432; 4,156,380]	[2,884,214; 3,167,826; 2,985,772]

This paper provides the cloud computing services for image analysis and generally investigated the problems and challenges due to the processing the very large images. These services are implemented using the following parameters:

1) Region-oriented data structure for storing and processing the very large image data in distributed databases;

2) Collaborative multi-agent systems of parallelism and balance of workload are among the critical features to ensure the improved performance of our distributed system. The operations for managing and processing the structural data can directly operate on the database. In implementation of multi-agent

systems, each operating agent in heterogeneous hardware can be run with heterogeneous database management systems and on different operating systems.

REFERENCES

Andrew FW, Anthony JH and Andrew Ware J (2007). Two Supervised Neural Networks for Classification of Sedimentary Organic Matter. *Images from Palynological Preparations, Math Geology* **39**.

Beul S, Mennicken S, Ziefle M and Jakobs E (2010). What happens after calling the ambulance: Information, communication, and acceptance issues in a telemedical workflow. *In Proceedings of International ConfInfSoc i-Society '11, London, UK* 111–116 .

Burks TF, Shearer SA and Payne FA (2000). Classification of weed species using color texture features and discriminant analysis. *Transaction of the ASAE* **43**(2) 441-448.

Chierichetti F, Kumar R and Tomkins A (2010). Max-cover in mapreduce. *In Proceedings of the 19th International World Wide Web Conference (WWW'10)* 231–240.

Deelman E, Singh G, Livny M, Berriman B and Good J (2008). The Cost of Doing Science on the Cloud: The Montage Example. In SC, November.

Ghannadpour Seyed-Saeid, Hezarkhani Ardeshir, Mokhtari Ahmadreza and Fathianpour Nader (2011). Preparation of segmentation software of mixed statistical populations based on the probability plots, Master's Project, Faculty of Mining Engineering, Isfahan University of Technology.

Hassanipak AA and Sharafeddin M (2005). *Exploration Data Analysis* (Tehran university press) 30.

Iranpak Somayeh, Akbari-Fetidehi Mohammad-Kazem, Shah-Bahrami Asadollah and Sargolzaei-Javan Morteza (2012). Preparation of statistical data of Statistics Centerfor using in business intelligence systems, *The First National Conference on Information Technology and Computer Networks at Payame Noor University (PNU), Tabas, Payame Noor University of Tabas*.

Knutson TR, Sirutis JJ, Garner ST, Held IM and Tuleya RE (2007). Simulation of the Recent Multidecadal Increase of Atlantic Hurricane Activity Using an 18-km-Grid Regional Model. *Bulletin of the American Meteorological Society* **88**(10) 1549-1565.

Lu Sifei (2011). A Framework for Cloud-Based Large-Scale Data Analytics and Visualization: Case Study on Multiscale Climate Data. *In Third IEEE International Conference on Cloud Computing Technology and Science*.

Moghaddas Mohammad-Sadegh and Asa Barzin (2012). An introduction to the cloud computing: characteristics, needs, challenges. *The First National Conference on Innovation in Computer Engineering and Information Technology, Tonekabon, Shafagh Higher Education Institute*.

Nurmi D, Wolski R, Grzegorczyk C, Obertelli G, Soman S, Youseff L and Zagorodnov D (2009). The Eucalyptus Open-Source Cloud-Computing System. *In Proceedings of the 2009 9th IEEE/ACM international Symposium on Cluster Computing and the Grid (May 18 - 21), CCGRID, IEEE Computer Society*.

Overpeck JT, Meehl GA, Bony S and Easterling DR (2011). Climate data challenges in the 21st century. *Science* **331**(6018) 700–702.

Sadashiv Naidila Kumar and Dilip SM (2011). Cluster, Grid and Cloud Computing: A Detailed Comparison. *The 6th International Conference on Computer Science & Education*.

Samanta B, Ganguli R and Bandopadhyay S (2005). Comparing the predictive performance of neural networks with ordinary kriging in abauxite deposit. *Transactions of Institute of Mining and Metallurgy* **114** 129–139.

Sivadon Chaisiri (2012). Optimization of Resource Provisioning Cost in Cloud Computing. *IEEE Transactions on Services Computing* **5**(2).

Wehner MF, Bala G, Duffy P, Mirin AA and Romano R (2010). Towards Direct Simulation of Future Tropical Cyclone Statistics in a High-Resolution Global Atmospheric Model. *Advances in Meteorology*, Article ID 915303.

Weller AF, Harris AJ, Ware JA and Jarvis PS (2006). Determining the saliency of feature measurements obtained from images of sedimentary organic matter for use in its classification. *Computer Geoscience* **32**(9) 1357–1367.
Yuen M (2010). GENI in the Cloud," master's thesis, Dept. of Computer Science, Univ. of Victoria.
Zhao W, Ma H and He Q (2009). Parallel K-means clustering based on Map Reduce. *Cloud Computing* (Springer) **5931** 674–679.