

Cloud Computing and MapReduce for Reliability and Scalability of Ubiquitous Learning Systems

Samah H. Gad

Computer Science, Virginia Tech
samah@vt.edu

Abstract

Ubiquitous learning research is about seamlessly enabling learning through the use of sensors that gather data from the learner surroundings and adapt learning content accordingly. Nowadays, mobile devices play a big part of these systems due to their advanced capabilities, like communicating with other devices through different methods and standards, and the ability to connect other gadgets expanding functionality even further. On the other hand, sensors integrated with these systems are very advanced as well and sophisticated making it possible to gather tremendous amount of data. In this paper, a ubiquitous learning system is presented. The system adapts learning content based on applying an understanding degree detection algorithm on an input of brain signals collected from the learning student. The adapted learning contents are then sent back over to be displayed on a mobile device. The main focus of this paper is on how to provide the necessary reliability and scalability for such systems. There are a number of challenges that are associated with realizing these systems. I'm proposing in this paper that Cloud Computing and MapReduce are better approaches and solutions for these two problems. A proof of concept was implemented and evaluated to support my proposed solution. Evaluation showed the effectiveness of using cloud computing in such systems with an increasing number of users.

Keywords Ubiquitous learning, Cloud Computing, Mobile Learning, Wireless sensors, EEG, Signal processing.

1. Introduction

Ubiquitous Learning is an evolving field with various devices and sensors used in supporting these types of systems. To enable learning, typically, these systems usually gather data using sensors and present learning content based on collected data. A ubiquitous learning system that uses Electroencephalography (EEG) signals is presented in this paper. The EEG signals are gathered from a student using an EEG sensor attached to his or her head and sent to a centralized server through his or her mobile device. The EEG signals are represented as a large matrix of data (frequencies). Detecting the understanding degree is based on processing the brain signals and detecting the understanding degree of the student. Based on this computed degree, specific learning materials are displayed

to the student on the mobile device being used. Such systems demand highly available storage and processing power with ensured reliability and scalability.

Different uses of backend frameworks were proposed in literature to support processing sensory data in Ubiquitous systems. In research done by [9], a comparison between different distributed computing systems like clusters, grids, and clouds was presented. The comparison was done mainly to know which distributed architecture would be the best to support a specific e-learning system. Factors considered were the number of users, size of contents, location of users, growth rate of users, growth rate of content, number of concurrent users, and others. It was concluded that if an e-learning system is to be used by a number of organizations, as a web based system, the cloud should be used. This study also suggests using cloud computing architectures from the general perspective. It did not mention though the effect of devising a MapReduce model on reliability and scalability.

Examples on emerging ubiquitous learning systems are presented in [10] [8] [5]. The three systems have one common idea, gathering sensor data from the user or the user's surroundings, do some processing and adapt the output of the system, which is mostly learning content, based on this collected data. These systems highly demand reliability and scalability which are the two main issues focused on in this paper.

In the rest of this paper, the proposed solution is introduced in section 2. In section 3, the challenges faced while developing the proposed solution are presented. In section 4, the system infrastructure is discussed in details. The details then of the system implementation are in section 5. In section 6, an evaluation of the system is presented. Finally, in section 7, there is the conclusion and future work.

2. Problem Definition and Challenges

Using sophisticated and advanced sensors in ubiquitous systems often lead to very high-dimensional data. Processing such data demands high computational power and storage capabilities. The computational power of mobile devices (e.g. iPhone, iPad, Androids, and Slate) is better nowadays but it is still considered significantly low in trying to process enormous data like brain signals with high dimensionality. Brain signals should be temporally stored

in memory for some processing to be done against it. Mobile devices, with the limitation of limited system memory, cannot afford loading the data in memory for processing. The number of users for such systems is unpredictable but with typical learning systems that number is expected to be high. Limiting the storage and processing power still to one central server will result in having a single point of failure that can likely not handle an increasing demand that is highly possible to happen. In Figure 1, the middle part, between the EEG signal input and the output learning materials, is the adaptation algorithm that is needed to be executed in order to select and display the right learning content to the student.

Most ubiquitous learning systems focus on the affordance of everywhere and anytime yet they always force the student to use a specific platform (e.g. Android, iOS, or Windows Mobile). Giving students the freedom to use any platform they prefer or is available to them is not an option anymore in this technology era.

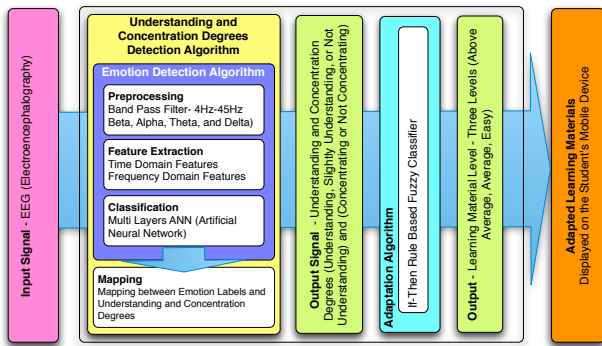


Figure 1. Brain Signal Based Adaptation Algorithm

3. Proposed Solution

In this paper, what I'm proposing is that Cloud Computing and MapReduce [3] [6] are effective solutions for processing limits, storage limits, reliability, and scalability problems. Ubiquitous Learning researchers will benefit from considering this approach as a replacement of other well know solutions in developing their applications . In the representative application presented, this will make the brain signals based adaptation algorithm unconstrained by the capabilities of the mobile devices that will be used to support the client side.

Adopting cloud computing, learning systems can be hosted on and offered in the form of Software as a Service (SaaS). The client side will just be responsible for collecting and passing over data to the cloud hosted application and displaying the learning contents sent back giving the freedom to develop applications for various platforms and for users to choose from any of the platforms to use. For the proof of concept, the client side was developed as a webpage but in the future, device native applications will be developed.

Research in Mobile Learning and Ubiquitous Learning focus on the aspects of everywhere and anytime and yet never mentions

platform issues. By storing the learning contents and processing the brain signals in cloud applications, this will be truly giving students the actual freedom to use the learning system anytime, anywhere, and on any platform.

Using Cloud Computing for processing and storage in such systems does not only afford platform independence but mainly it is about achieving the needed scalability of the system mitigating possible problems arising from loading a centralized server and having a single point of failure. Devising MapReduce is a way also to avoid having a single point failure in computation and in efficient computation of enormous highly-dimensional input data.

4. System Infrastructure

The infrastructure of the system is divided into two major levels: the Backend, and the Frontend.

At the front-end of the application, there will be a web interface or a mobile application which will give the flexibility for the student to choose any platform that he/she prefers. Through that client interface, brain signal data will be sent to the other levels of the system for storage and then after, processing with the decision being made remotely. This is considered as a preliminary step to test the functionality of the architecture. The proof of concept implementation done was in the form of a adaptive web page but in the future that will be replaced by a fully functional, from the learning content adaptation point view, application that can run across different platforms.

For storing the brain signals, the HDFS (Hadoop Distributed File System) was used as a distributed file system providing high throughput access to application data. The HDFS has a software infrastructure with an inherent capability to spread data across servers seamlessly elevating performance, scalability, availability and thus overall system reliability.

The backend of the system implemented was deployed on Amazon Web Service's elastic cloud (AWS EC2). A multi-node (two) node Hadoop cluster was used to handle parallel processing tasks. Each one of these nodes had a copy of the MapReduce Java program with one node orchestrating job scheduling. An Apache web server was responsible for the communication part done between Hadoop and the web interface. See Figure 2.

4.1 Mapper and Reducer

4.1.1 Mapper

Based on the concept of the two stage processing of MapReduce, the purpose of the Mapper is to extract features from each row in the matrix and emit the understanding degree of each row. The matrix represent the EEG signal recorded from a student. Each row in the matrix represents the signal coming from one electrode (there are 64 electrodes in the EEG sensor). The input to the Mapper is a Key, Value pair. The key is a (64*12000) matrix (tab separated values File). The output of the Mapper is a new (Key, Value) pair for each of the 64 rows. The Mapper emits 64 new key pairs. An example

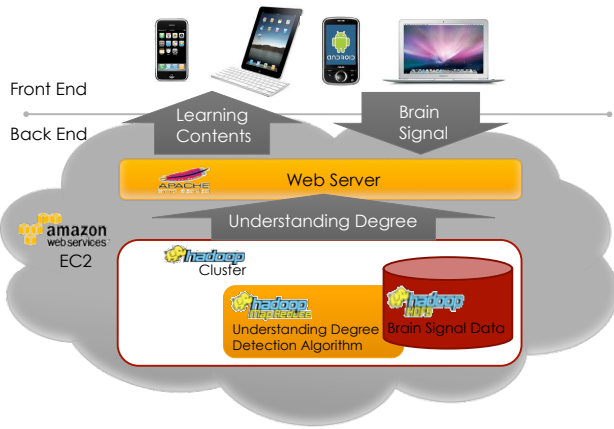


Figure 2. System Infrastructure

of the Mapper output will be something like (Understanding, 1), (Neutral, 1), or (Not Understanding, 1).

4.1.2 Reducer

For the next stage, reduction, the purpose of the reducer is to sum the (Key, Value) pairs emitted by the Mapper. The reducer will run for each unique key emitted by the Mapper and emit three (Key, Value) pairs. These (Key, Value) pairs represent the total aggregated number of Understanding, Neutral, Not Understanding keys. An example on that will be; (Understanding, 30), (Neutral, 10), or (Not Understanding, 24).

5. Implementation

For simplification and as a proof of concept the adaptation algorithm was simplified. Two time domain features were extracted for each row in the matrix and a rule-based fuzzy classifier was used on these features to classify them into three understanding levels. Each row in a single matrix will have its own understanding level. To decide what understanding level the matrix represent the rows are categorized into three categories (Understanding, Neutral, and Not Understanding). The one with the highest number (dominant) will be considered as the understanding level detected.

Two implementations for this simplified adaptation algorithm were done. The first one was a regular Java application that was written for baseline performance referencing in testing and evaluation. The second one was a Java application that was based on the MapReduce programming model running on the two-node Hadoop cluster.

For the second implementation, a web interface was developed to send the EEG data file and to display results, see Figure 3.

The Cloudera training image [1] was used in the beginning to test everything before moving it to Amazon EC2.

Before moving the application to Amazon EC2 two open source instances were chosen to set them up as two node cluster (master and a slave). The master and slave had the following building blocks; in the MapReduce layer the master had a job tracker and



Figure 3. Web Interface

task tracker and the slave had a task tracker. For the HDFS layer, the master had a name node and a data node and the slave had a data node only.

The configuration file of both were modified to set them as a master and a slave setup. The two chosen instances had the following specifications: Small Instance, 1.7 GB memory, 1 EC2 Compute Unit (1 virtual core with 1 EC2 Compute Unit), 160 GB instance storage, 32-bit platform I/O, and Performance: Moderate.

For communication between the backend and the front end, two PHP scripts were written. The first one was to upload the file of the EEG data to the HDFS. The second one was for accessing the output file of the reducer, comparing the three reduced keys sending the one with the higher value. The understanding degree with the highest value found reflects the actual understanding degree of the student.

6. Evaluation and Testing

For evaluating the system, a generated dataset was used. The dataset contained six different files that represented a recorded EEG signals. There were two files for each understanding degree. The file had tab separated values represented as a 64*12000. The 64 rows represented data coming from electrodes during 12000 epochs. Each file is around 7 MB.

Three different tests were done; functionality test, load test, and performance test.

6.1 Functionality Testing

Six matrices were used to test the service functionality. There were two for each understanding degree. From this test it appeared that the system is functionally operating correctly with 100% accuracy.

6.2 Load Testing

The load test was made in order to test the scalability and reliability of the system. To simulate system load, Neoload was used to record a user scenario for the service and simulate 10 virtual users. Amazon Web Services Monitoring was used to monitor the Average CPU usability for the master node while running the recorded

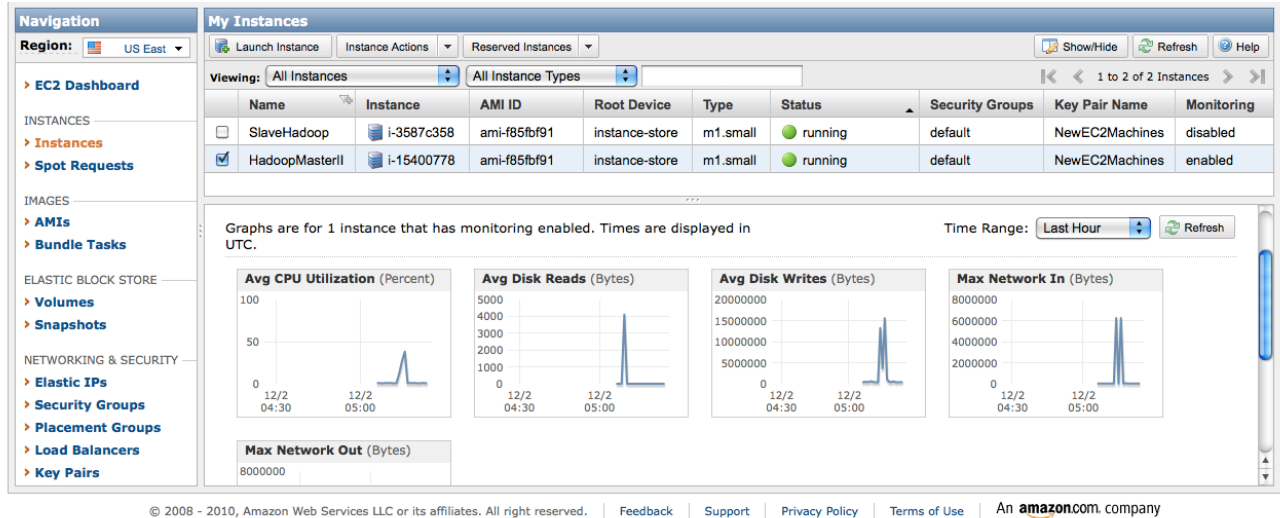


Figure 4. AWS Monitor - Average CPU Utilization

test scenario. After testing, as shown in Figure 6, for 10 simultaneous users it did not reach 50% of the CPU usage which is an indication for two good things; the system used less than 50% of its full CPU power for 10 users and that it is energy efficient.

6.3 Performance Testing

A standard Java implementation of the adaptation algorithm was run locally on a single machine to compute the baseline performance. For comparison, another Java MapReduce implementation of the same algorithm was run on a cluster of two machines.

First test:

The standard Java implementation was running on a machine with the following specification:

- Platform: Mac OS X
- Processor: 32-bit , 2.4 GHz Intel Core 2 Duo
- Memory: 4 GB

Second test:

The two machines that were running a cluster had the following specifications:

- Platform: Linux Ubuntu
- Processor: 32-bit , 1 EC2 Compute Unit (1 virtual core with 1 EC2 Compute Unit)
- Memory: 1.7 GB memory

From the first test the baseline performance was 1 sec (computation only). The second test took 7 sec (computation only). At a small scale, as expected, the test showed that running the standard Java code locally will be faster than using the MapReduce on a cluster based architecture.

Writing a traditional Java program (without MapReduce framework) and running it locally yielded significantly faster response.

With the overhead imposed by having a distributed file system (HDFS) and distributed processing (starting and running job trackers and task trackers), in contrast to the JVM (Java Virtual Machine) running a traditional Java program, apparently there is a constant time factor in place. However on large scale, as proved by MapReduce Benchmarks [7] [2] [4] that correlates well with the problem in hand (file operations, string operations, arrays, and comparisons), MapReduce proved to be more efficient.

7. Conclusion and Future Work

A Ubiquitous Learning system that used the cloud processing and storage power was designed, prototyped, and evaluated. Cloud Computing and MapReduce proved to be a solution for some of the problems in ubiquitous and mobile learning.

The MapReduce programming model is suitable for implementing signal processing algorithms (in this case EEG signal processing) and it aligns well with the inherent characteristics of such problems.

The cloud is not worth using if there is a small (less than 1000) number of user of the Ubiquitous learning system. Using cloud computing for processing and storage in highly loaded systems enabled scalability of the system without worrying about the load on a central server.

There are still some challenges remaining even after using cloud computing and MapReduce. Latency is one of them, given that data is sent to the cloud to get processed. Another challenge is the mobile device battery life. Given the communication needed with the backend and the sensors, that might drain the battery significantly faster. Screen size can also considered a limitation. Even if the mobile device gave afforded anytime, anywhere, any platform freedom, still it has a limited screen size compared to desktop computers.

In the future, a complete implementation for the adaptation algorithm and a re-evaluation of the system will be done. Also, a platform independent web application and device native applications will be developed on the client side. And finally, testing the service with actual brain sensors would also be done as future work.

References

- [1] Cloudera training image. Website, 2010. <http://www.cloudera.com/downloads/>.
- [2] Gridmix. Website, 2010. <http://Hadoop.apache.org/mapreduce/docs/current/gridmix.html>.
- [3] Hadoop mapreduce. Website, 2010. http://Hadoop.apache.org/mapreduce/docs/current/mapred_tutorial.html.
- [4] NNBench. <http://netlikon.de/docs/javadoc-Hadoop/branch-0.5/org/apache/Hadoop/examples/NNBench.html>, 2010.
- [5] O. Boyinbode and A. Bagula. An adaptive and personalized ubiquitous learning middleware support for handicapped learners. In *Eighth International Conference on Information Technology: New Generations (ITNG)*, pages 632–637, april 2011.
- [6] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. *OSDI'04: Sixth Symposium on Operating System Design and Implementation*, December 2004.
- [7] Kiyoung Kim, Kyungho Jeon, Hyuck Han, Shin gyu Kim, Hyungsoo Jung, and H.Y. Yeom. Mrbench: A benchmark for mapreduce framework. In *14th IEEE International Conference on Parallel and Distributed Systems, ICPADS '08.*, pages 11–18, 2008.
- [8] Yushun Li, Hui Guo, Ge Gao, Ronghuai Huang, and Xiaochun Cheng. Ubiquitous e-learning system for dynamic mini-courseware assemblying and delivering to mobile terminals. In *Fifth International Joint Conference on INC, IMS and IDC, NCM '09.*, pages 1081–1086, aug. 2009.
- [9] H.K. Mehta, M. Chandwani, and P. Kanungo. Towards development of a distributed e-learning ecosystem. In *International Conference on Technology for Education (T4E)*, pages 68–71, 2010.
- [10] S.C.B. Nelaturu, R. Kambham, N.J. Karna, R. Parupalli, and K. Mandula. Building intelligent campus environment utilizing ubiquitous learning. In *International Conference on Technology for Education (T4E)*, pages 230–231, july 2010.