Large Scale Experimentation on Anomaly Detection Scalability and Performance

Yaya Sylla, Pierre Morizet-Mahoudeaux, Stephen Brobst

▶ **To cite this version:**

Yaya Sylla, Pierre Morizet-Mahoudeaux, Stephen Brobst. Large Scale Experimentation on Anomaly Detection Scalability and Performance. 20th International Conference on Artificial Intelligence (ICAI 2018), Jul 2018, Las Vegas, United States. pp.36-43. hal-02022807

HAL Id: hal-02022807
https://hal.science/hal-02022807

Submitted on 25 Jun 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Large Scale Experimentation on Anomaly Detection Scalability and Performance

Yaya Sylla[12] , Pierre Morizet-Mahoudeaux[1], Stephen Brobst[2]
[1]Sorbonne universités, UTC, UMR CNRS 7253 HEUDIASYC Compiègne, France
[2]Teradata Corporation Riyadh, Saudi Arabia & San Diego, USA
e-mail: yaya.sylla@utc.fr, pierre.morizet@utc.fr, stephen.brobst@teradata.com

Abstract— **This paper presents a study of various standard anomalies detection techniques in internet networks, using machine learning algorithms, classification algorithms, and graph techniques working on SQL/Mapreduce and SQL-Graph implementation platforms. This approach shows its efficiency for the study of large datasets. Firstly, several algorithmic approaches and possible implementations have been studied and tested by experiment to see how the SQL-MapReduce and SQL-Graph implementation can process large-scale data. Secondly the performance and scalability of the algorithms for large volumes of data have been compared to choose the most appropriate for typical anomaly detection. The application scope is very broad, such as spam detection, crime detection, mafia or terrorism community detection, network intrusion detection, malignant tumors detection in healthcare, fraud detection on banking transactions, identity theft detection etc. The main problem we will address on this paper is the processing performance on large scale data using an implementation of Massively Parallel Processing (MPP), SQL-MapReduce, and SQl-Graph.**

Keywords-component; Anomaly, Graph, Big Data, Classification, Machine Learning, Modularity, MapReduce

I. INTRODUCTION

Linking social networks data, spread across different heterogeneous data repositories, calls for addressing several challenging problems such as large-scale data management, algorithms optimization and parallelization, new knowledge representation paradigms for heterogeneous, redundant, non-certified or false information, association mechanisms, graph analysis for clustering and partitioning.

The major problems and challenges are performance and scalability. In this paper, we are demonstrating how the alternative that we are proposing, which relies on combining SQL and MapReduce, can answer the performance and scalability issues of the algorithms that are usually used for anomalies detections.

We begin by studying and comparing algorithms, methods and techniques used for the classification of data sets such as e-mails, SMS, personal data etc. according to rules corresponding to three groups: normal, likely to be abnormal and normal. Abnormality is characterized by abnormal values, rare events, or some objects that do not align with functional rules, exceptions, etc. Then we extend the application domain to community detection characterized by anomalies. In the case of communities, anomaly detection consists of cross-checking transactional data with descriptive data (enrichment of the knowledge of the network) corresponding to the communities involved in these transactions, to find typically abnormal behaviors. This approach can be considered, from the point of view of its principle, as representative of the problems associated with the anomaly detection in large networks. This study is focusing also on algorithms and techniques commonly used for anomaly detection and community detection. This will guide us to select appropriate algorithms and techniques that will be implemented and used for the experimentation phase to test efficient solutions to the main problems we mentioned in the beginning of this document.

To set up the experimentation phase and implement the experimental platform, we have studied the extent to which massive parallel and map reduce methods can improve the convergence speed of algorithms selected out of our survey of anomalies detection methods. We conclude that combining massively parallel processes, MapReduce and graphs and visualization techniques, drastically improves convergence performance as well as enhancing the representation of knowledge for interpreting results. In addition, the technique that we propose can be used to implement any of the studied algorithms.

The experimentation platform used for implementing and testing the performances of the algorithms is based on SQL-MapReduce and SQL-Graph combined with parallel processing. We choose appropriate datasets to highlight the impact of the parallel process approach using the MapReduce technique on each selected algorithm's performance.

The results are extended to the case of community anomalies detection by adding external information, corresponding to social networks simulation. Broadcast spam data are used of this initial experiment.

In the second phase of our study, to evaluate the performance of our clustering method, we carried out two different types of experiments. For the first one, we used synthetic networks [33] for which a pre-built community structure was well defined. In the latter one, we combined different graphs describing the structure of certain real-world networks [33].

The purpose of the former set of experiments is to assess the quality of the clustering produced by our approach in the context of a controlled environment in which the features of the networks taken into account are well-known. In particular, the pre-defined community structure is adopted as a ground

truth to evaluate the quality of the resulting partitions. This is done by adopting a measure, called normalized mutual information, inherited from the information theory [16], to determine the accuracy of the partitions with respect to the ground truth. The community detection algorithm uses the concept of shortest distance vectors algorithms, based on the shortest vector between members of the community sharing the same kind of transaction. This approach improves the overall system performance.

II. ANOMALY AND COMMUNITY DETECTION METHODS

Anomaly is an important topic in data analysis and can be defined as unexpected behaviors, outliers, rare events, deviant objects or peculiar objects [11], [13]. The detection of anomalies has been studied using several techniques and approaches like data mining, graph-based theories etc. [11], [12], [13], [14], [15], [17]. In many cases, anomaly detection is interesting if unexpected behaviors are detected very quickly to prevent against linked problems [11]. To solve anomaly detection problems, the methods and techniques used are highly dependent on anomaly types, such as fraud for example, as well as types of fraudsters [14], types of data [11], and the relationship between data formats [11]. Data formats are defined in 3 types: metric data, evolving data and multi-structured data [11]. In this paper, we considered only the multi-structured data formats including multiple data types like structured, semi-structures or unstructured

In his paper, Leonid Kalinichenk et al. [11] have presented a comprehensive review of techniques and methods for anomaly detection, one of which is machine learning, usually used in supervised, semi-supervised and unsupervised learning. Several researchers have done benchmarks of different supervised machine learning approaches to resolve anomaly detection problem by comparing the performance of the naïve Bayes, K-means and support vector machine [2], [17]. Naïve Bayes approach comes with a very good result [2], [17].

Community detection has been addressed by many recent researches and experimentations. The results have been applied on diverse disciplines such as internet, medicine, biology, sociology, to resolve complex network analysis problems. Community has several definitions. We define in this paper community as a group of nodes that are strongly interconnected in the network. There are several techniques and methods to resolve community detection problems, such as hierarchical clustering, modularity optimization, statistical inference, clique-based methods [22]. Suhas S Thorat [29] investigated several techniques in his survey on community detection and comes with a conclusion that the problem of maximizing the network modularity is NP complete.

In their study on community detection methods in social networks, Mehjabin Khatoon et al. [22] highlight the modularity method used as a fitness function to detect the overlapping community. This motivated us to target and study more modularity methods and techniques [5], [6], [7], [8], [9]. Resolving community detection problems by using modularity theories have been applied for different applications, such as, for example, social network analysis, protein network and pathway analysis, genomic analysis, cybercrime etc. [1], [5], [6], [7], [8], [9], [32], 11], [22]. The precursors on community detection on social network, Girvan and Newman [7], have proposed an algorithm used to find communities by discovering community structures based on the modularity Q. Vincent D. Blondel et al. [21] proposed another scalable method, called Louvain algorithm, to extract the community structure of large networks using modularity technique, and Michel Plantie and Michel Crampes showed by experimentation how Louvain algorithm is performing on large scale graph [21].

III. PARALLEL PROCESSING METHODS

In this section, we first address the classical massive parallel processing (MPP) technology for large dataset computation [22], [24]. Second, we focus on the MapReduce framework [24], and show the advantages of combining this programming approach with SQL for scalability performance on MPP architectures [22], [24], [25].

A. Massively Parallel Processing (MPP)

Parallel Database build on MPP using relational data model has proven its capabilities for achieving high performance and scalability with techniques like data distribution, data partitioning across nodes in a shared-nothing architecture [24], [25]. The techniques applied as data distribution, columnar, data partitioning provide efficient data analysis on very large-scale datasets. The most specialized MPP data warehouse vendors are for example Teradata Aster, Oracle, HP Vertica. There are also Open Source MPP solutions such as Postgres-XL and Greenplum Database. We will demonstrate how an MPP architecture based on a relational data model can handle parallelization by using SQL algorithm processing performance, and scalability by separating the data loading function from the data storage and query processing functions [25].

B. MapReduce

MapReduce is a programming framework for processing parallelizable problems over any large-scale (multi-petabyte) data (unstructured or structured) [26]. It combines two classes of functions: map and reduce, which are defined with respect to data structured in (key, value) pairs. Map takes a pair of data with a type and returns a list of key value pairs. It is applied in parallel to every item in the input dataset. The reduce function is applied in parallel to each group, which in turn produces a collection of values.

IV. DEVELOPMENT FRAMEWORK AND EXPERIMENTATION

We begin by selecting the solution to support the implementation of SQL and MapReduce on a distributed environment. Two solutions were preselected: Spark QL and Teradata Aster.

Spark allows for the use of efficient SQL execution with machine-learning algorithms using the resource description framework (RDD). Teradata Aster is a parallel database that enables the storing and processing of structured, unstructured and multi-structured data. After several tests and looking for a flexible solution we decided to use Teradata Aster [28], which saved us time for implementation.

A. Experimentations: Materials and Settings

We conducted several experiments during our study. For the first one, we used a desktop running several virtual machines on VMware environment (the number of virtual machines will be presented in the Results sub-section). Then, we added external servers for social network simulation in the case of the spam data experience. The goal was to find a typical example of cross evidence of transactions for descriptive data (enrichment of the knowledge of the network) including the actors of these transactions to find typically abnormal behavior. From the point of view of its principle, this approach can be considered to represent problems related to anomaly detection in large networks. We will observe the scalability and the performance of the algorithms processing.

The second experiment was conducted on a physical cluster of computers.

B. Experimens

1) Initial Experiment Platform Configuration

The experiment hardware configuration is based on a HP EliteBook 8570p Notebook detailed in Table I.

TABLE I. EXPERIMENT PLATFORM

Processor	Intel® Core™ i7 Quad-Core
RAM	16 GB
Storage	256 GB SED SSD Internal 1 TB SSD External Drive
Operating System	openSUSE 64 bits
Virtualization	VMware Workstation 64 bits

2) Experimentation Hardware Configuration

To simulate parallel computing, we used a virtualization solution based on VMware with 4 virtual servers (1 Aster Queen node and 3 Aster Worker nodes.)

3) Extended Experimentation Platform Configuration

The extended experimentation platform is a physical cluster composed of several nodes with the configuration for each node shown in table II.

TABLE II. EXTENDED EXPERIMENTATION PLATFORM: 1 QUEEN NODE AND 4 WORKERS NODES

Processor	6-core CPU Intel® Xeon® 2.1GHz Processors
RAM	96 GB
Storage	12TB
Operating System	openSUSE 64 bits

4) Experimentation Algorithms and Functions

The experiments were conducted by using several algorithms and MapReduce implementation functions that are listed in the Table III.

TABLE III. ALGORITHMS AND FUNCTIONS

Algorithms /Functions	Description
PSTPARSERAFS	[29] The PSTParserAFS function parses Personal Storage Table (PST) files (which store email in Microsoft software such as Microsoft Outlook and Microsoft Exchange Client) directly from Aster File Store (AFS).
NaiveBayesText	[29] Classifier function used to generate a model from training data
NaiveBayesTextPredict	[29] Classifier function utilize Naïve Bayes model to predict.

C. Results

1) Dataset

To begin with a consistent data set to conduct our experiments we downloaded from a free community site [1]. a data set based on personal storage table (PST) files that contain mails of over 140 people.

2) Data Integration

PSTPARSERAFS is a MapReduce function in Aster to read PST files. To run the function, the first step is to load and store all PST files as Aster File Store (AFS). Thus, we used the ***PSTParserAFS*** SQL-MapReduce command to read from AFS and load all email data files in the Aster corresponding table.

In our experiment, we consider only the emails of the "Inbox" folder by using the EXCLUDE option to exclude Draft, deleted items, Notes and Sent Items folders. For our first experiment we are using a sample containing all PST information. A given user may contain up to 140000 lines of mails. Some information, such as empty columns, can be considered as noise, and are removed by SQL script to select only the information requested for our analysis and proceed data cleaning by removing all noise. We generalize the cleaning for all users to generate the final dataset that we will use for the next step of our experiment and test the classification algorithms.

3) Learning technique

Following our comparative study presented in section III, the Naïve Bayes technique appears to be one of the best technique for document classification, such as shown by Arun D Panicker who made a similar study [2],[19] In our example, the training data defined two categories Spam or Not Spam.

We have selected a representative sample of data (10% - 15%) by using a SQL script and manually generated the training data set that was used for the naïve Bayes algorithm.

We start by generating the model using the training set, then we apply the naïve Bayes text classifier algorithm to classify mails. Fundamentally, we used two mains functions to apply the model: the ***NaiveBayesText*** and the ***NaiveBayesTextPredict***.

NaiveBayesText is utilized for model generation from training data optained during the previous step and SQL statement is used to call the NaiveBayesText function on the training function

The generated data model is exported to use the ***NaiveBayesTextPredict*** and install it as a function in Aster and apply it on all datasets. Usally, traditional predictive models like Decision Trees or NaiveBayes use numerical and / or categorical variables. The advantages of using Aster is that

it can generate paths, patterns and all possible combinations of patterns. These combinations of events and patterns tend to be more predictive compared to a single value, e.g. number of times a customer had a higher usage in a month compared to a set of patterns that document the relationship between the usage and other activities. The sequence of events is important and that is not reflected in the variables used in traditional statistical models. This is one of our motivations to use this approach.The extract from the Predict function output is illustrated in figure 1.

doc_id	prediction	loglik_spam	loglik_nospam
<EJDC4KM02XH...	spam	-109.191788536396	-149.660261988262
<MKUMHUVLQ...	spam	-222.696632785808	-271.351298292432
<FQCKCXPE1JO...	spam	-632.396301204099	-834.652562646449
<A2XFXLMJPBL...	spam	-359.405622767388	-410.803708785791
<OQFJQZ0NKA1...	spam	-154.688353958987	-198.252759711495
<KQ4QOZU5RSB...	spam	-797.579712769266	-950.017807678344
<BHERPJG4EYG...	spam	-1189.85758726811	-1419.21076155231
<HYBJ4O4IJJM1I...	spam	-209.803291814276	-265.411263296477
<D4HWAWGUIV...	spam	-95.6858855713708	-100.57893004069
<BQUT0VXU131...	spam	-610.980718798318	-783.618850372208

Figure 1. NaiveBayes Predict Output.

We arrived with the experiement to generate a dataset table containing 110 000 rows. We have run the same script on different architecture and the result are mentioned below comparing 3 configurations:1 Worker VM, 2 Workers VM and 3 Workers VM (see figure 2.).

Step	SMP PC	2 Virtual node Aster	3 Virtual node Aster
PSTParserAFS	00:01:17	00:01:13	00:01:15
NaiveBayesText	00:53:05	00:12:41	00:09:39
NaiveBayesTextPredict	00:32:23	00:10:13	00:08:08

Figure 2. Performance on VM

For PSTParserAFS, the computation time is almost the same for all platforms but the computation time using 2 Workers and 3 Workers is better but not linear.

The first conclusion is that, by increasing the number of nodes, the execution time of NaiveBayes algoritms implemented on SQL-MR is performing well. This will be validated in the next experiment using physical nodes clusters with high data volumes.

4) Graph Analytic technique

a) In this experiment we will focus on the relation between email messages using graph classification techniques. The study presented in section II motivated our choice for modularity technique to identify community on the email dataset. We will compare the computation using single thread architecture using a PC single Virtual Machine in the same configuration as Aster Worker node, with MPP architecture (using Aster virtual workers with 1 worker node, 2 worker nodes). The modularity technique we decided to test and implement on Aster is Louvain [21].

The first phase of our experimentation is based on compiling and testing the original Louvain implementation in C++ that we downloaded from Sourceforge [30].

The process are:

- Step .1 Compile convert.cpp file
- Step .2 Conversion from a text format (each line contains a couple "src dest")
- Step .3 Compile community.cpp file
- Step .4 Computes communities and displays hierarchical tree

We didn't succeded with the Virtual Machine with the Configuration (1 CPU, 2 Core, 2 GB RAM) to execute the Algorithm with our initial dataset.

The second phase was to implement Louvain Modularity on Aster according to the workflow shown in Figure 3. Before that, we have used Aster SDK by adapting the source code to the Aster MapReduce framework.

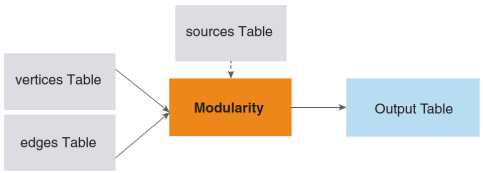

Figure 3. Modularity Implemented on MPP (SQL MR).

Why Louvain? Because we compared several algorithms of section II, and Louvain drew particular attention to its ability to analyze very large graphs. We used the example algorithm with, and without using our approach combining massive parallelization of processes and MapReduce functions. We didn't get any response after an hour and the Virtual Machine crashed. Then we decided to implement the algorithm using the Aster MapReduce framework. Aster already has the modularity MapReduce function. We decided to procced with the following test:

- Compare Louvain MR with Aster Modularity
- Compare execution time between 2 configurations on MPP

5) Results without MPP

The convergence is very slow and requires significant resources proportional to the number of nodes in the graph. For example, our experiment using Aster Modularity on 1 Aster Worker node didn't process the entire data set. To make it append on the same configuration, we had to use only 10% of the dataset.

The figure 8 below shows the data representation on a graph using ***GraphGen*** MapReduce function. This experimentation does not indeed help to conclude on emails containing spam or Not spam and we observed no apparent relationships because of the small sample size (10%). Also, we see several clusters grouping some senders reepresented by different color but without clear explanation

Figure 4. Modularity execution on SMP platform

6) *Results with MPP*

We used our approach based on SQL-MapReduce and the results are promising because we have seen a very rapid convergence using 110 000 rows and especially without any filter (fullfDataset). The table IV shows the difference between Louvain and Aster Modularity.

In figure 5, we effectively observe the communities of spam servers (red nodes) or no-spam servers (blue nodes) and the relationships between mails. In the graph representation, we see 10 major SPAM clusters.

TABLE IV. LOUVAIN MODULARITY - ASTER MODULARITY

	Louvain Modularity	Aster Modularity
Data Acceptance	Only digit	Strings and digits
Performance with the same dataset size	53 Seconds	3 Minutes 24 Seconds
Input Data Structure	1 file: Source - Destination	2 tables -Vertices: Vertix_id -Edges: Source_id - Target_id
Output Data Structure	1 file: Source - Destination	1 table
Number of generated nodes	Bigger than Aster Modularity	Less than Louvain Modularity

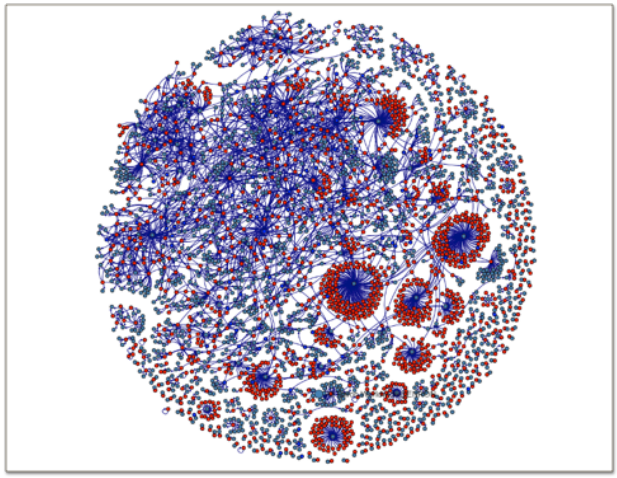

Figure 5. Modularity on Parrallel Architecture (With MPP SQL).

V. EXTENDED EXPERIMENTS

We exended the experiment using a bigger cluster (1 Queen + 4 Workers and SVM, Kmeans on Graph and Naïve bayes algortithms. Concerning the Kmeans experiment, the performance of kmeans on the Graph engine and the limitation of custom aggregators have been tested. We considered the following four variables in the performance test: the number of rows of dataset, the number of columns in dataset, the argument k, which represents the centroid number, and the iteration number. During our experiment, we fixed three of them and adjust the left one to compare the performance.

Test 1: rows = 10m Iteration number =10 with 10 columns

	10 columns	
	graph version	SQL-MR version
row = 10m iter = 10 k=10	33 sec	2 min, 42 sec
row = 10m iter = 10 k=100	41 sec	14 mins, 59 secs, 244 ms
row = 10m iter = 10 k=300	1 min, 7 sec	55 mins, 10 secs
row = 10m iter = 50 k=10	46 secs	11 mins, 26 secs
row = 40miter=10 k=10		

Test 2: rows = 10m Iteration number =10 with 100 columns

	100 columns	
	graph version	SQL-MR version
row = 10m iter = 10 k=10	1 min, 1 sec	3 mins, 59 secs
row = 10m iter = 10 k=100	2 min, 22 sec	35 min, 33 sec
row = 10m iter = 10 k=300	4 min, 25 sec	about 1hr 50mins (insert cost most of the time)
row = 10m iter = 50 k=10	2 mins, 8 secs	23 mins, 5 secs
row = 40miter=10 k=10	2 mins, 20 secs	9 mins, 44 secs,

Test 3: rows = 10m Iteration number =10 with 1000 columns

	1000 columns	
	graph version	SQL-MR version
row = 10m iter = 10 k=10	2 hrs, 20 mins, 31 secs	27min 40sec
row = 10m iter = 10 k=100	more than 3hrs (about 2hrs in iteration steps)	1 hrs, 52 mins, 44 sec
row = 10m iter = 10 k=300	3 hrs, 10 sec (about 2hrs in iteration steps)	about 3hrs 40mins (insert cost half of the time)
row = 10m iter = 50 k=10	12 hrs, 59 mins, 36 secs	1 hrs, 42 mins
row = 40miter=10 k=10		

We describe below the following observation during the experiment:

- When the dataset fits the memory, the graph engine iteration performance is very good. We have a great performance and Kmeans is scalable.
- When the dataset is spilled on a disk, the Sql-MapReduce version may have advantages in performance.
- When the dataset is large, the performance will not be affected much by the parameter k. The main reason is that most of the time consumed is dealing with the I/O tasks. The ratio of computation tasks is small, thus the execution time is not depending of the parameter.

VI. EXPERIMENT ON REAL CASES SCALABLE CLUSTERS

A. Experiment 1: Detecting Criminal Community

1) Analysis

With the results obtained from the experimentation phase, we used our approach on actual productions.

The main data sources for this project are Party dataset, Crimes dataset. The Party dataset contains about 155.000.000 records containing following information:

- Party Id,
- Party type (Citizen, Visitor, etc.)
- Party Information (First Name, Last Name, Birth Day, Parent information, etc.)

In the analysis, Party Data were coupled with criminal information to determine criminal communities according to some type of crime (drugs, rape, theft, murder, etc.). We obtained very good result that allowed us to identify for example:

- Relationship between Community (Family member) and relationship between each of them.
- Crime history data combined with Party to identify criminal community link with family cluster.
- Hidden networks within population
- Drug community linked with family relations.

2) Performance Results

The table V shows a better performance with 7 Aster nodes to process all the data. The customer has all his historical data on Hadoop (3 data nodes). It shows also how we can run the MapReduce function on Hadoop. It took much more time to adapt the algorithm.

TABLE V. ASTER – HADOOP SCALABILITY

Function	3 node Hadoop	3 node Aster	7 node Aster
Louvain Modularity	00:12:03	00:28:12	00:10:45

The Table V is described as follows:

- the first column is the function name,
- next is the time hh:mm:ss a 3 node (3 data nodes plus one name node) Hadoop cluster,
- next is a 3 node (3 workers plus one Queen) Aster cluster,
- next is a 7 node (7 workers plus one Queen) Aster cluster.

With this experiment, we observe that 8 Aster nodes are required to be more performant than 3 Hadoop cluster nodes [31]. This is explained by the fact that Aster has a fixed 6 virtual workers per node so 3 nodes is 18 way parallel but Hadoop using yarn [31] is configurable. It was tuned to use 42 map/reduce tasks and it easily beats Aster. Next, we ran it on a 7 node Aster cluster (6 virtual workers * 7 nodes = 42 ways parallel) and now it beats Hadoop. It also shows the importance of parallel thread on MapReduce scalability.

We observe that this process is highly CPU intensive and reads a complete file from start to finish. Thus, Aster's superior indexing and data access is not making a difference.

B. Experimentation 2: Customer Behavioral Analysis

1) Analysis

The Telecom Operator (monMOBIL) web site gives to their customers the ability to control and deal with their own mobile and landline numbers in *monMOBIL* by providing them with a lot of services and features that help them to do their operations easily, like view the bills and pay them, recharge prepaid numbers and transfer amounts between them, subscribe and unsubscribe in packages and services, monitor data usage, view referral-program points and redeem them with gifts, locate branches and WIFI spots in the country. New customers can view all *monMOBIL* products through *monMobil* store, check smartphone availability at *monMOBIL* branches, and a lot more features that save time and effort. The Big Data Analysis challenge is to use advanced analytic technique in distributed environment to:

- understand how customers use *monMobil* through all channels (excluding SMS and Twitter),
- assess the technical/commercial performance of monMobil and find areas of improvements,
- see if *monMobil* contributes to a better customer satisfaction.

2) Dataset

For the experiment, we collected a 3 months length data set shown in tab VI (for June, July and August 2017).

TABLE VI. DATA SET OF MOT CUSTOMERS

MOBILE SUBSCRIPTIONS	
Month	Count(*)
June	16 538 262
July	16 703 478
August	17 710 055

GLOBAL TRANSACTIONS	
Month	Count(*)
June	2 301 749
July	2 383 068
August	2 474 343

LL SUBSCRIPTIONS	
Month	Count(*)
June	15 804
July	17 136
August	15 304

GLOBAL VISITS	
Month	Count(*)
June	150 155 430
July	170 901 698
August	182 331 448

All monMobil data are merged sequentially into a single table in Aster platform as illustrated in the diagram below.

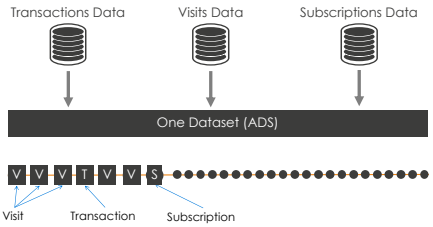

Figure 6. Merging of the monMobil data sets

We have used several technique like: **Sessionization**: to create visit sessions (no start/end timestamp nor login/logout timestamp available), **SQL Analysis**: to run a set of descriptive analyses, **Path analysis** to understand events paths throughout monMobil navigation and perform data preparation for SQL Analysis, **Attribution analysis** to weight events/channel contribution to subscription, **Association analysis** to find events/offers that are frequently paired with other events, **Tableau and Aster AppCenter** for data visualization

3) Results

Visit Analysis helped us to identify Customers likely to come on purpose with a "goal" in mind (Figures 7).

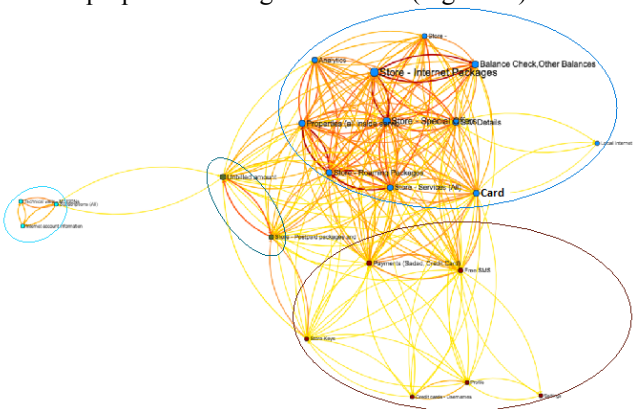

Figure 7. Page Association Analysis

Subscription Analysis, let us analyze offers association

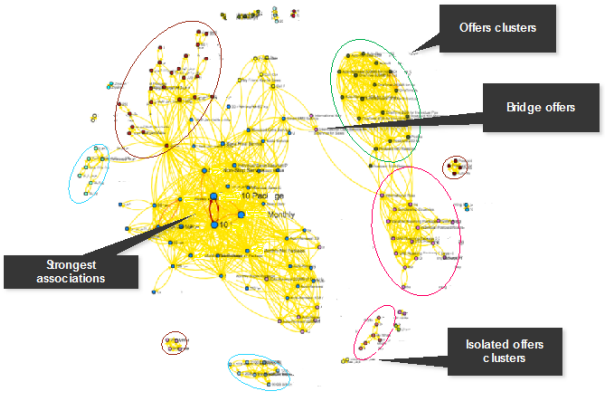

Figure 8. Page Association Analysis

Overall usage is increasing slowly but shows a gain in popularity. *monMobil* App channel ranks #1 in terms of popularity and commercial efficiency. Behaviors and preferences differ based on criteria such as customer nationality, line tenure and line type (adapt design, adapt offer push). Offers association/bridge offers could be considered to propose new bundles and drive customers to different kind of offers. Customer satisfaction has increased since *monMobil* was launched. The performance result table show the scalability of the methods using Mapreduce function on MPP architecture.

TABLE VII. SCALABILTY RESULT ON MPP ARCHITECTURE

Method	3 node Aster	6 node Aster
Visit Analysis	00:09:22	00:04:23
Offers association analysis	00:12:12	00:06:01

VII. CONCLUSION

In this paper, we have presented our motivations to study anomaly detection on large scale social networks for characterizing communities. The study addresses the problems of linking information spread over several heterogeneous networks, algorithms parallelization and optimization for network analysis, graph partitioning and clustering for structure extraction. The experiments using data mining algorithms and community detection on graph using the new approach provides an answer to anomaly detection using high data volume with very good computation performance.

We are continuing our experimentation combining deep learning technique and the distance metrics in vector-based approach by incorporate the same with CONCLUDE approach for improvising the outcome. Finally, we will compare the proposed advanced algorithm with the CONCLUDE approach to prove the efficacy of proposed scheme.

REFERENCES

[1] The Enron PST Data Set Cleansed of PII by Nuix and EDRM http://info.nuix.com/Enron.html

[2] "Question Classification using Machine Learning Approaches" International Journal of Computer Applications (0975 – 888) Volume 48– No.13, June 2012 Question

[3] N. Du, B. Wu, X. Pei, B. Wang, and L. Xu, "Community detection in large-scale social networks, ser. WebKDD/SNA-KDD '07. New York.

[4] Jing Wang ; Ioannis Ch. Paschalidis, "Botnet Detection Based on Anomaly and Community Detection", IEEE 29 February 2016

[5] Michelle Girvan, M. E. J. Newman, "Community structure in social and biological networks" Proc. Natl. Acad. Sci. USA 99, 7821-7826 (2002).

[6] Hemant Balakrishnan and Narsingh Deo, "Centrality based community discovery" School of Electrical Engineering and Computer Science University of Central Florida

[7] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices", Phys. Rev. E 74, 036104 (2006)

[8] Tianbao Yang1, Rong Jin2, Yun Chi3, Shenghuo Zhu3, "Combining Link and Content for Community Detection", KDD '09 Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining

[9] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos, "Netprobe: a fast and scalable system for fraud detection in online auction networks," in Proceedings of the 16th international conference on World Wide Web, ser.

[10] Michel Planti, Michel Crampes, "Survey on Social Community Detection"

[11] Leonid Kalinichenko, Ivan Shanin, Ilia Taraban, "Methods for Anomaly Detection: A Survey" CEUR Workshop Proceedings (2014)

[12] Supervised Machine Learning: A Review of Classification Techniques

[13] Shikha Agrawal, Jitendra Agrawal "Department Survey on Anomaly Detection using Data Mining Techniques", 19th International Conference on Knowledge Based and Intelligent Information and Engineering Systems, Procedia Computer Science 60 (2015) 708 – 713

[14] C. Phua, V. Lee, K. Smith, and R. Gayler, "A comprehensive survey of data mining-based fraud detection research," 09 2010. [Online]. Available: http://arxiv.org/abs/1009.6119

[15] W. Eberle and L. Holder, "Anomaly detection in data represented as graphs," Intell. Data Anal., vol. 11, no. 6, pp. 663–689, Dec. 2007. [Online]. Available: http://dl.acm.org/citation.cfm?id=1368018.1368024

[16] Pasquale De Meo, Emilio Ferrarax, Giacomo Fiumara, Alessandro Provetti, "Generalized Louvain method for community detection in large networks", ISDA '11: Proceedings of the 11th International Conference on Intelligent Systems Design and Applications, 2011

[17] Fragkiskos D. Malliarosa, Michalis Vazirgiannisa, "Clustering and Community Detection in Directed Networks: A Survey

[18] Fragkiskos" arXiv:1308.0971v1 [cs.SI] 5 Aug 2013 Abstract

[19] Pierre-François Marteau, Gildas Ménier, Leopold Ekamby "Apport de la prise en compte du contexte structurel dans les modèles bayésiens de classification de documents semistructurés "RNTI - E – « Fouille de Données Complexes » ISBN: 2-85428-702-9, Cépaduès ed.

[20] Michel Plantie, Michel Crampes "Survey on Social Community Detection", HAL Id: hal-00804234, https://hal.archives-ouvertes.fr/hal00804234 Submitted on 25 Mar 2013 HAL

[21] Vincent D. Blondel, Jean-Loup Guillaume1, Renaud Lambiotte and Etienne Lefebvre1, "Fast unfolding of communities in large networks", https://arxiv.org/abs/0803.0476v2 J. Stat. Mech. (2008) P10008

[22] Mehjabin Khatoon, W. Aisha Banu, "A Survey on Community Detection Methods in Social Networks", I.J. Education and Management Engineering, 2015, 1, 8-18

[23] Sai Wu, Feng Li, Sharad, Beng Chin Ooi, "Query Optimization for Massively Parallel Data Processing", ISBN: 9781450309769, 2011

[24] Joseph O. Chan, "An Architecture for Big Data Analytics", Communications of the IIMA: Vol. 13: Iss. 2, Article 1. Available at: http://scholarworks.lib.csusb.edu/ciima/vol13/iss2/1.

[25] Shivnath Babu, Herodotos Herodotou, "Massively Parallel Databases and MapReduce Systems", Foundations and TrendsR in Databases Vol. 5, No. 1 (2012) 1–104 c 2013

[26] Jeffrey Dean and Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Communications of the ACM, 51(1):107–113, 2008.

[27] Eric Friedman, Peter Pawlowski, John Cieslewicz, "SQL/MapReduce: A practical approach to self-describing, polymorphic, and parallelizable user-defined functions", Proceedings of the VLDB Endowment, VLDB '09 August 24-28, 2009, Lyon, France.

[28] Teradata Corporation, "Teradata Aster Analytics Foundation User Guide", Release 6.2, B700-1021-621K, November 2016

[29] Suhas S Thorat, Sharmila M Shinde, "A Survey on Community Detection", International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064, 2013

[30] Vincent Blondel, Jean-Loup Guillaume, Renaud Lambiotte, Etienne Lefevre, "Louvain method: Finding communities in large networks https://sourceforge.net/projects/louvain/

[31] Hadoop architecture, https://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html

[32] Rodrigo F.Berriel, Franco Schmidt, RossiAlberto, F.de Souza, Thiago Oliveira-Santos, "Automatic large-scale data acquisition via crowdsourcing for crosswalk classification: A deep learning approach", Computers & Graphics 68, August 2017.

[33] Satrio Baskoro Yudhoatmojo, Muhammad Arvin Samuar, "Community Detection On Citation Network Of DBLP Data Sample Set Using LinkRank Algorithm", Procedia Computer Science, Volume 124, 2017, Pages 29-37