Universidad Politécnica de Madrid
Escuela Técnica Superior de Ingenieros de Telecomunicación

BYPASS FRAUD DETECTION APPLICATION BASED ON CALL DETAIL RECORD ANALYSIS

TRABAJO FIN DE MÁSTER

MinKyoung Kang

2010

Universidad Politécnica de Madrid
Escuela Técnica Superior de Ingenieros de Telecomunicación

Máster Universitario en
Ingeniería de Redes y Servicios Telemáticos

TRABAJO FIN DE MÁSTER

BYPASS FRAUD DETECTION APPLICATION BASED ON CALL DETAIL RECORD ANALYSIS

Autor
MinKyoung Kang

Director
Jose Carlos González

Departamento de Ingeniería de Sistemas Telemáticos

2012

Abstract

The development of the internet has offered us various ways to communicate with others in the distance. Starting from an email and messenger, now using Voice over Internet Protocol technology, people can even call ordinary telephones using the internet with lower charges. There are some people abusing this technology to bypass interconnection points where charges are made. Some people gain profit providing call services using the resources of telecom operators and it is defined as a bypass (interconnection) fraud. The bypass frauds become ever cleverer every day. There are two challenges: Firstly the detection system should handle enormous call detail record generated every day with high performance, especially in consideration of privacy infringement issue. Secondly detection algorithms should intelligently adjust itself to the changing environment real time to detect new types of frauds.

Thus a delayed time detection system which retrieves the stored call detail records from database and mines fraudulent clients using a clustering is invented. As a next step, to overcome some limitations of the delayed time system, the real time bypass (interconnection) fraud detection system is created.

The real time bypass (interconnection) fraud application developed in this paper resolves two challenges stated in the first paragraph. Since call detail record is processed in real time as a call occurs, there is neither need for data storage nor privacy infringement. The core of the detection system is growing neural gas (GNG) which is a flexible artificial neural network. The growing neural gas continuously adjusts locations of all cells in networks according to the input data. Anomalous data which are located anomalously far from the assigned cell is detected as outliers. The performances of two systems are compared with the penalty points and the result shows us that real time application has much better performance.

Abstract

El desarrollo de Internet nos ha ofrecido varias maneras de comunicarse con los demás en la distancia. A partir de un correo electrónico y mensajería,

ahora usando tecnología de Voz sobre Protocolo de Internet, la gente puede incluso llamar a teléfonos ordinarios el uso de Internet con menores cargas. Hay algunas personas que abusan de esta tecnología para evitar los puntos de interconexión donde los cargos se hacen. Algunas personas se aprovechan de los recursos de los operadores de telecomunicaciones proporcionando servicios de llamadas. Eso se define como un bypass (interconexión) fraude. Los fraudes se hacen más inteligentes cada día. Hay dos retos: en primer lugar el sistema de detección puede manejar un enorme call detail record (CDR) se genera cada día con un alto rendimiento, especialmente en la consideración de infracción tema de la privacidad. En segundo lugar los algoritmos de detección inteligente puede ajustarse con el tiempo cambiante entorno real para detectar nuevos tipos de fraudes.

Así, un sistema de detección en tiempo diferido, que recupera los registros de llamadas almacenados detalle desde bases de datos y agrupa datos usando clustering para clasificar grupos sospechosos. Como siguiente paso, para superar algunas limitaciones de los ensayos anteriores, la carretera de circunvalación en tiempo real (interconexión) la aplicación de fraude que se propone.

Un sistema de detección en tiempo real resuelve dos problemas señalados en el párrafo primero. Como call detail records se procesan en tiempo real, no hay ni necesidad de almacenamiento de datos ni la violación de la privacidad. El núcleo del sistema de detección es growing neural gas (GNG), que es una forma flexible de la red neuronal artificial. El GNG ajusta continuamente ubicaciones de todas las celdas en las redes de acuerdo con los datos. Datos anómalos que se encuentran anormalmente lejos de la celda asignada se detecta como valores atípicos. Sus actuaciones se miden con los puntos de penalización y el resultado nos muestra que la aplicación en tiempo real tiene un rendimiento mucho mejor.

General index

Contents

1 Introduction

1.1 Context

The telecommunication frauds are continued for more than 40 years and as technology for detections develops defrauders continuously develop new ways to commit frauds to avoid being detected. The telecommunication system has become more fragile as the public switched telephone network system is converted into IP based networks. The trustworthiness of telephone network is greatly impaired. Those changes opened the new possibility of frauds such as automatic telemarketing calls, distributed attacks, bypassing fee of service, and surreptitious use of other's account.

There are apparently some means to tackle those issues in trustworthiness though; such solutions possess some weaknesses such as the followings.

- The centralized system limits the performance as large amounts of data are generated and analyzed every day.
- Storage of logs and records of customers and screening of such data raises the privacy infringement issue.
- The cooperation among different domains of networks is very low which improve the fraud detection.

Thus it is required to invent monitoring applications which detect frauds in VoIP network systems overcoming such limitations. There might be some anonymization methods of the call history or a prompt detection process deleted once the appropriate actions are taken to analyze customers' records.

Considering such challenges and concerns in our development process, bypass fraud detection applications are introduced in two different versions. First the delayed time system which detects the frauds from the stored call detail records (CDR) and the second is the real time monitoring system which detects frauds instantly from the real time input data. Because the first version of the system requires the stored CDR data for detection, it is hard to avoid the privacy infringement. As the real time monitoring system process data using bloomfilters in real time, the original data is anonymized automatically and there is no need of storing CDR data that contain private information of customers.

Given that CDRs are provided by operators, the quality of the data depends directly on the logging processes. Furthermore, in principle data sources and processing devices (SIP proxies) within the operators and ISPs are trusted, they just provide information regarding what they observe traversing their networks, even when this information is contaminated (e.g. by spoofed customers, bots, etc).

When the customers want to call overseas their call is interconnected by the network that provides switched circuits between regional centers of the Public Switched Telephone Network. The charge for these calls is based on a revenue scheme that is shared by the source operator that charges for the call initialization and the destination operator that charges for its finalization. The source operator sends just one bill to the clients adding two fees. Nevertheless those billing scheme can be bypassed by using VoIP, since the voice packets that used to be transported by the PSTN make a detour through the internet. This is possible by digitalization and packetization of the voice. As long as the caller has broadband access, this is not a complicated process. When this happens, it is define that the bypass of interconnection network has occurred.

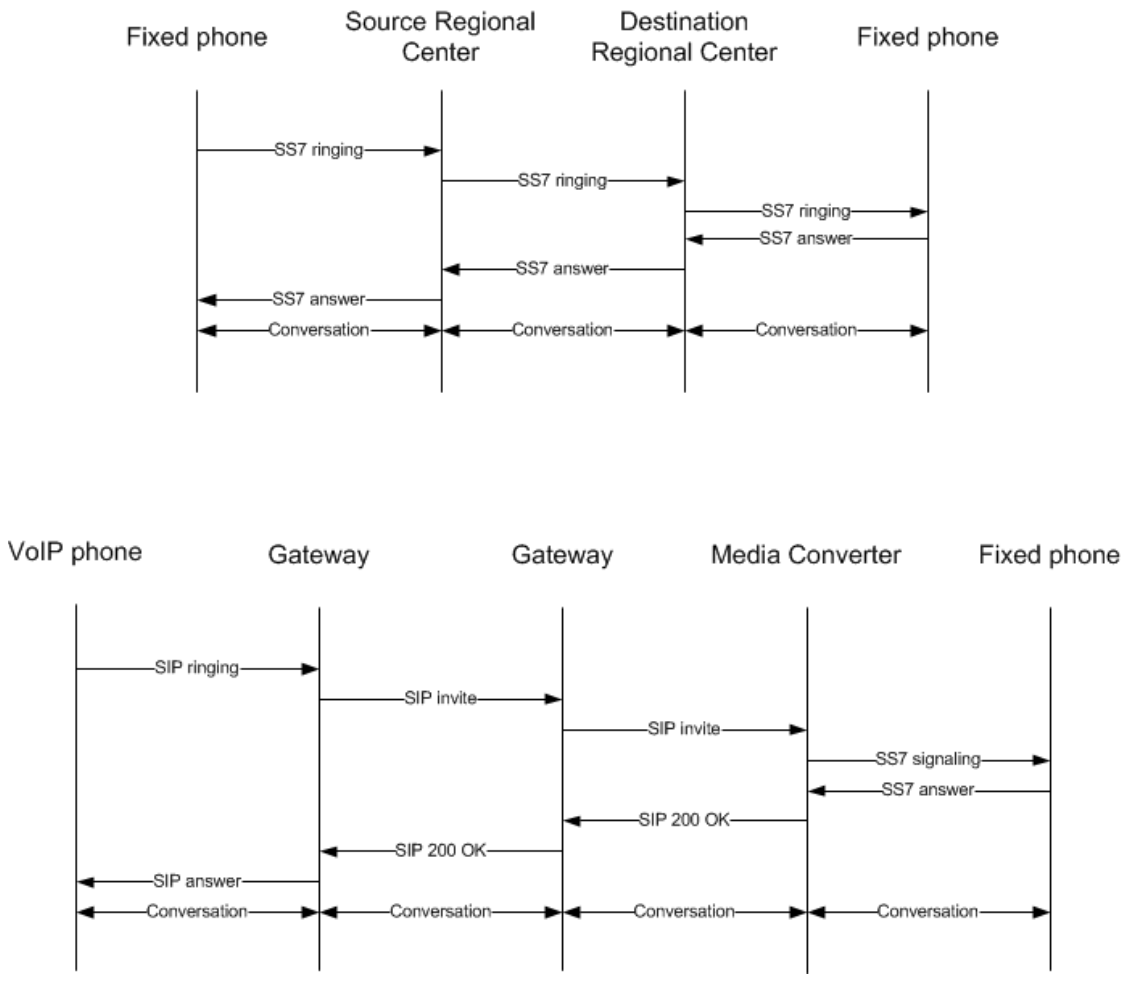


Figure 1: Sequence diagrams for traditional PSTN-based international calls (above) and VoIP-based international calls (below)

<Source: 257315-DEMONS D1.1 Scenario descriptions and requirements>

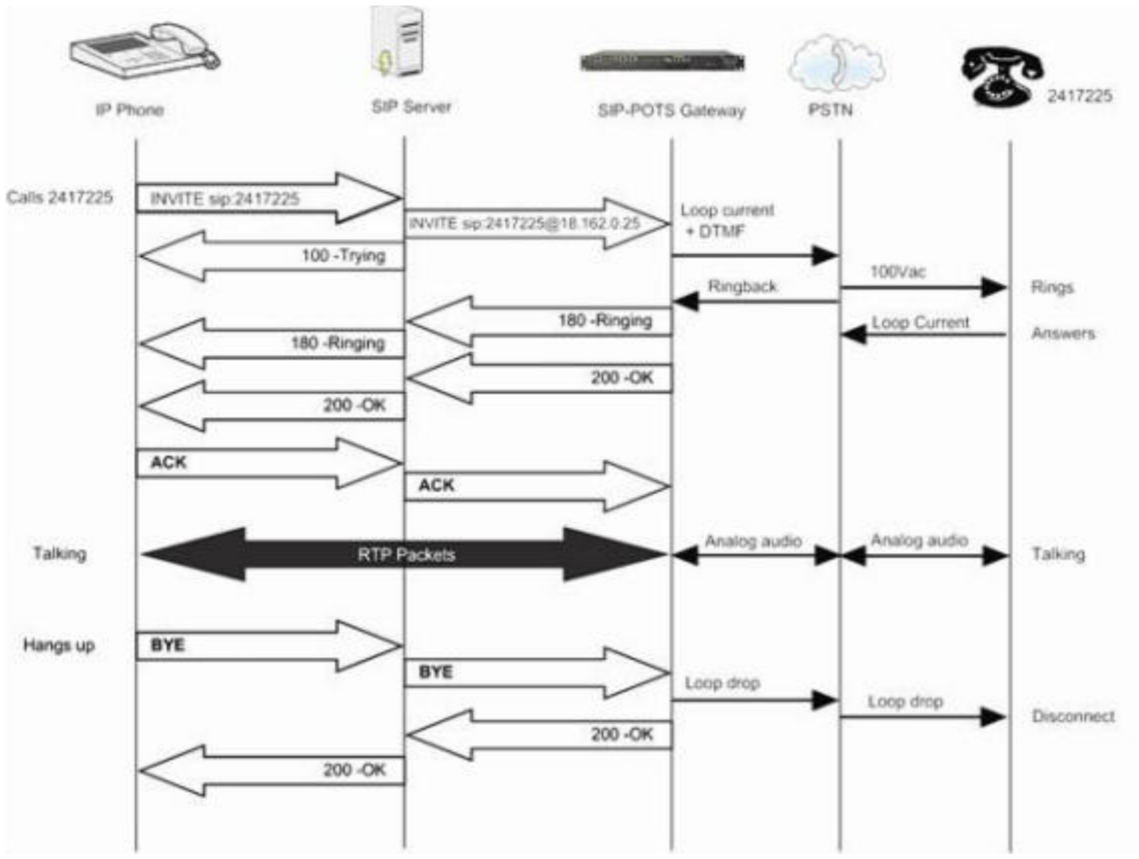


Figure 2: A SIP call set-up from an IP phone to the PSTN, using a SIP

<Source: http://www.axiaaudio.com/tech/voip/default.htm>

There are differences in the path of phone connection between traditional PSTN-based and VoIP-based international calls. In the VoIP-based system the regional centers and the interconnection points are replaced by several SIP gateways. As long as the customers have the internet access, most of the VoIP infrastructure is free of charge. With such advantage, most users are changing from the traditional PSTN based service into VoIP service.

With such changes, there is one complicated question as to what can be regarded as interconnection bypass frauds. If the VoIP calls are established by the people who own the VoIP equipment which includes software and hardware seeking for low fees of telephone service, by the law this is not regarded as illicit activity. Although traditional telephone providers are strongly against such acceptance, it is generally recognized as a legal service. However it is a question whether such acceptance can be applied to the third party which is provided with call services from the person who possesses the VoIP equipment with the fee. Depending on the region and detailed process of call establishment this process can be said illegal or legal.

The technical and legal facts related to interconnection frauds will be further explored in the next section.

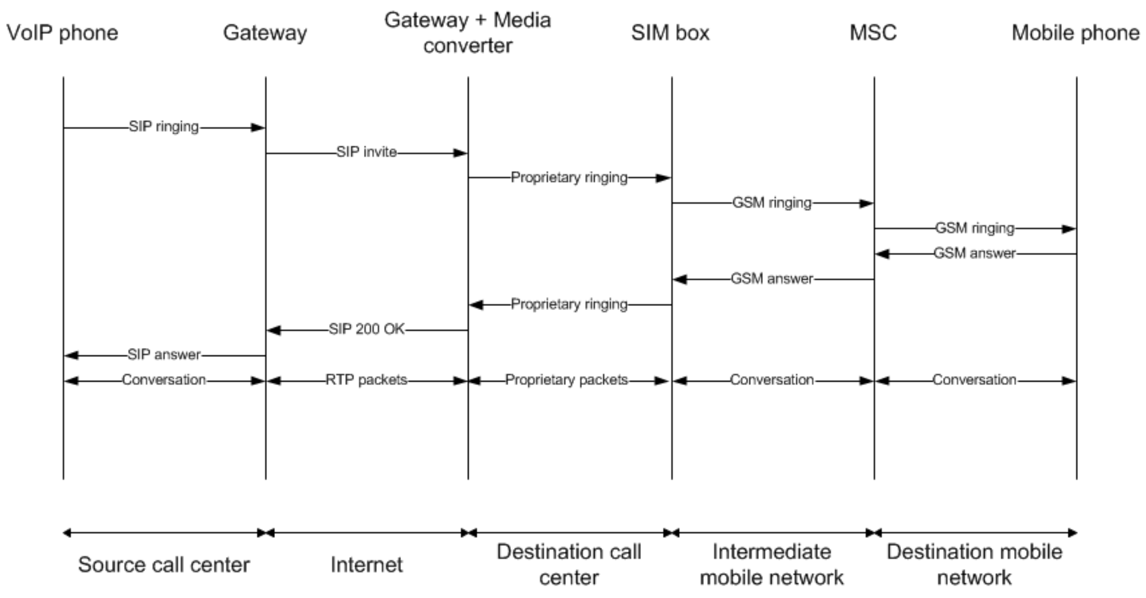


Figure 3: Common interconnection fraud related entities and transactions between them

<source: 257315-DEMONS D1.1 Scenario descriptions and requirements>

Calls can be realized between a caller and a recipient implementing the above mentioned VoIP architecture (a couple of gateways and the required media converters), plus a special network element, the SIM box, which exists at the destination call center. This system consists of a number of SIM cards in the box as re-entry points to telephony network. The function of SIM box is based on the algorithm: when an incoming call occurs, it de-packetizes voice data from TCP/IP packets and packetizes them again as GSM/UMTS packets. Those packets are sent through the SIM card that is available within the SIM box. If the destination of the call is the mobile device then the call should pass through the Mobile Switching Center (MSC), while the destination placed at the PSTN can be reached after the packets are converted into analog media. Other fraud architectures are possible as well, e.g. to use another GSM gateway at the specified source, the customers just have to call a unique number from where the calls are sent via VoIP to the destination; or to call to another Call Center at the destination (but this direction is infrequent).

There are several strategies designed to detect this fraud. The statistical algorithms are very efficient to identify a call center. The scenario described in figure 3 is about 100 outgoing VoIP calls that are established in a day. These calls are from the same SIP source gateway and arrive to the same SIP destination. Thus it is very possible that private branch exchange (PBX) software has been used at the source. If there is a case where several calls are realized in the same time period, and no incoming call history has been found one can say that private branch exchange has been used with very high probability. In the figure below, some more interesting features that characterize the fraudulent VoIP calls can be found. For example the calls that are established with the same periodical hiatus are suspicious in a sense that it is very possible that those calls are managed by machines.

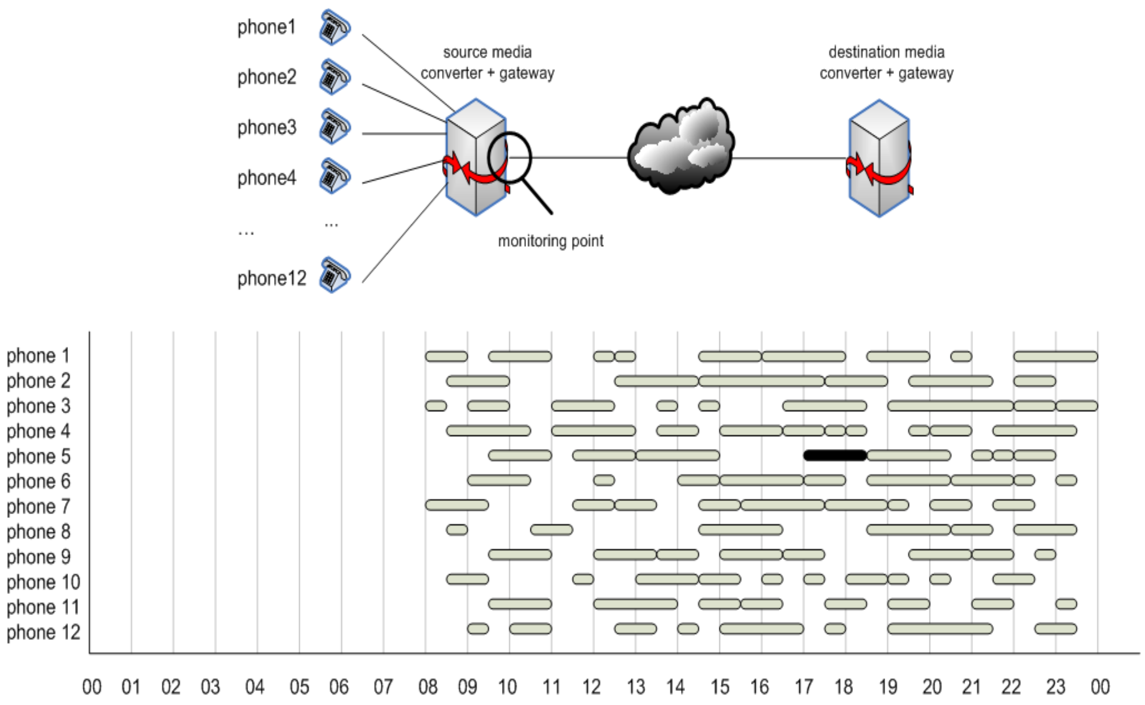


Figure 4: 100 voice calls done in a whole day using 12 phones and converted to 100 VoIP outgoing calls from the same source to the same destination; the black line represents a call using the phone number 5, starting at 5 p.m. and finishing at 6:30 p.m.

<source: 257315-DEMONS D1.1 Scenario descriptions and requirements >

There is an additional interesting analysis in the inter-domain correlation of VoIP calls which investigate which call centers are collaborating among them. Normally it is required to compare duration and start time of outgoing and incoming calls in several different domains.

In respect to each call, behavioral analysis can be implemented at the individual customer level at intra-domain. In this analysis, it is necessary to generate the normal behavior models that can define the customers who make the international calls. The information to build this model is as follows.

- Number of international calls per interval (a month, a week …).
- Timing features such as day of the week, flag for special date, time or duration.
- The destination of the international calls.
- The nationality of these destinations.

This analysis can be used to distinguish PSTN users that stop using international call services. Later the investigation can be done to screen destination networks to see if the customers are still receiving similar types of international calls (special dates, start times, and durations) especially from unknown SIM cards instead of the usual origins.

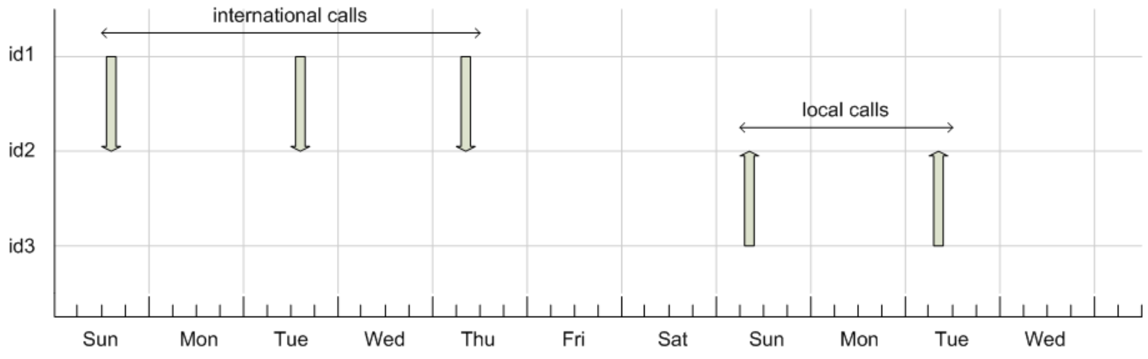


Figure 5: Periodic international calls from id1 to id2 that suddenly stop, and periodic local calls from id3 to id2 with a very similar timing pattern (days of the week and start times)

<source: 257315-DEMONS D1.1 Scenario descriptions and requirements>

This analysis can lead to the discovery of customers who stop to use habitual international calls. The result is not related to the profit generation or fraud detection.

Although the identification of the caller and the recipient is not the final goal of the analysis, it is evident that they are significant factors to decide on frauds. The analysis of their behavior will aid, in some ways, to detect the interconnection fraud.

On the other hand, there are some technological factors (required technical functionalities of the system are based on those factors). Examples of technological factors are gateways, routers and many other network elements, but those are secondary actors since those will only participate in the use case that can provide data.

To different telephony operators the information of the customers who suddenly stopped calling habitual international calls but still continue the calls with other means is a very significant clue for the fraud detection. The calls that continue to occur at similar time period might be established through illegal means. The destination operator can implement analysis on current incoming calls to detect calls that have similar traits to the model. To receive the information of models, the operators should exchange data about their suspicious clients so that destination operators can implement further analysis on calling patterns of their clients. The examples of information for model formation can be represented as follows:

- Number of international calls per interval (a month, a week …).
- Timing features such as day of the week, flag for special date, time or duration.
- The destination of the international calls.
- The nationality of these destinations.

This information is appropriately processed as a vector that contains essential information about the clients. Vectors can also be shared, such as the distribution of all the calls, which can be required by some correlation algorithms instead of raw vectors.

A similar analysis can be done with regard to the amount of calls. It is hard to share information of customers from different telecom operators and their call histories as the competition among operators increase.

It is notable that except for identifiers of recipients, such a package of information does not violate the privacy protection law of individuals. It is clear that much more intelligent exchanges would be those including the identifiers of the recipient. But there are two implied problems with this solution: screening of those identifiers is illegal even though it is an intra-domain task, and such information should be shared in an anonymized way.

Broadband providers also exchange a similar type of information which is not restricted to one customer but for several customers. In order to correlate the observed statistical anomalies within information, the features from different aspects of all the calls from a number of different users should be shared among different providers.

The fraudulent numbers are not very immediately detected in a sense that analysis requires a certain time period to accumulate data that make detection possible. Thus promptness is relatively less important factors for the application. However as a large amount of calls are established each day (about 400 millions occurrence of calls), the efficiency of the algorithm is an important factor in order to meet the real time requirements because the application should implement analysis on the all customers of mobile operators which amounts to millions in number but does not grow further dramatically any more. However, the potential VoIP customers can also be millions (not known a priori) and currently that number increases rapidly.

Except for the recipient identifiers, the type of information used for analysis does not violate clearly any privacy policy because nothing can be said about the identity of callers or recipients. Nevertheless, and for performance purposes, it would be better to share, at least, the anonymized identifiers of the recipients when trying to know if the usual destinations of a customer are still receiving international calls. However current anonymization mechanisms are demonstrated to be unsafe, and still require to be researched.

As already mentioned, the inclusion of the recipient identifiers in the package and feature vectors that are shared with other domains will be very useful for the performance: only the calls related to those recipients would be monitored at the destination mobile network, which implies a high reduction of processing time.

Now various legal aspects of the bypass fraud detection in different countries are introduced. In United Kingdom it is totally legal that any end users such as firms, individual, and etc buy, establish and utilize GSM gateways for their own purpose. On the other hand if the purpose is to provide service to third parties and to gain profits, it becomes illegal.

In Spain the law related to this issue is unclear because the usage of SIM boxes is legal when virtual mobile operators benefit VoIP networks to bypass the interconnection network; but on the contrary it is not mentioned as to the case when a Call Center obtains a gateway and use the SIM card of a licensed operator without any previous notice or agreement. There is one different thing that when the SIM cards came from third customers who resell their cards illegally. In any case, the legality of the each case is largely dependent on a valid contract between the Call Center and a telephony operator providing the SIM cards.

The detection technique is highly important for large telephone operators since it can be a great threat to their revenue every month. It is important for mobile operators as well because if their customers use the SIM cards in an incorrect way, this could lead to the violation of the contract either because customers are not utilizing SIM cards for their personal use or because they sold their SIM cards to the third party to gain illegal profits.

The incorrect use of the SIM cards could cause a contractual break by customers who acquired them; either because they are not using the SIM card for personal purposes or because they have resold the cards, which is strictly forbidden.

Another context that should be considered as the technical background of these applications is Blockmon system. The system is separated by blocks as its functional units. Functions are separated in blocks, because that scheme maximizes the reusability of each block when a block is repeatedly used. For example there are several different types of fraud detection applications within the Blockmon and detection applications usually write their output as a log in a text file. If we make one block that writes the relevant data in a file, this block can be reused every time it is required to record the detection results.

1.2 Objective

As introduced above, interconnection network connects regional centers of the public switched telephone networks and provides international call services. The service fee is charged in each connecting point of networks between centers. However nowadays with VoIP the voice is digitalized and packetized to be sent through the internet, bypassing the interconnection network and avoiding some charges of the service. If this is done by the owner of the VoIP equipment to reduce the cost, it is not considered as a fraud but if this service is provided to the third party to gain profit, it becomes a fraud. The distinction of the two is somewhat vague and varies depending on the laws of each country. Thus here we will only focus on the method to detect suspicious data and the legality of the clients' behavior will be judged in consideration of laws of each country.

The accumulation of such bypass (interconnection) frauds can lead to the great revenue loss of the major telecommunication operators. In addition as one client provides services to a number of customers, this causes the congestion in telephone network system and thus the quality of telephone service will be lowered. Thus a number of telecom operators are greatly interested in developing applications that can detect the illicit users in real time. From call detail records (CDR) which contain all clients' voice and SMS histories, the calling patterns of individuals can be tracked and analyzed to detect anomalous users which is suspicious of frauds.

The core purpose of the analysis is to detect the clients who provide illegal services to the third parties to gain profits abusing the resources of telecom operators. The enterprise and individuals that utilize the VoIP networks for telephone calls for their own use can be legally justified by the law. However as the VoIP telephone network can provide the functionally the same services with lower fees, people not only benefit this fact to pay lower fees for their own, but also try to gain profit establishing illegal business that provides cheaper telephone services to the third parties. These are possible with facts that traditional PSTN based telephone services are more expensive than the VoIP telephone services and ordinary customers generally cannot distinguish the difference between the two. Our goal is to detect such clients who provide illegal services.

As the performance of fraud detection systems is of a great concern, various trials have been made to modify the detection algorithms, models, and systems which allow us to increase the prediction accuracy and to maintain incorrect predictions lower than a certain threshold. The correct diagnosis of monitoring status can be improved by minimizing the false negative and false positive alarms.

The technical terms used in the above paragraph will be explained here. In this context, *false positive alarms* mean the percentage of legal transactions that are incorrectly classified as fraudulent when it is in fact not fraud. *Fraud catching rate* (or true positive rate or detection accuracy rate) is the percentage of fraudulent transactions that are correctly identified as fraudulent. *False Negative rate* is the portion of fraudulent transactions that are incorrectly classified as legitimate when it is in fact illegitimate.

The application receives call detail records of clients and extracts list of vectors which contain call history of each client. In the delayed time system these vectors are input into the conceptual web, one of clustering algorithms in WEKA for classification. The conceptual web algorithm will classify the clients to each cluster depending on their calling patterns. Clusters with few members and a high outgoing call volume will be suspected as frauds. On the other hand, in the real time system artificial neural network algorithms are used to figure out outlier points which are anomalously located farther from each centroid.

The clustering and artificial neural networks are used with the purpose of sorting out suspicious clients who have anomalous call detail records.

1.3 Structure of the document

This first part of the document is dedicated to learn about the general situation that necessitated the initiation of this project and the detection applications for the specific type of fraud called the bypass (interconnection) fraud. The structure and detailed processes for the application will be developed in the following chapters.

The second part of the document will be focused on the introduction of the state of the art of the fraud detection technology. Not many studies have been done for the bypass fraud detection system. Thus this section is not limited to the specific type of frauds. As the system functions as an application that detects frauds without known fraudulent patterns, the fraud detection system will be focused on outlier detections and unsupervised learning techniques especially that have relation with our interests.

The chapter three and four is about the specific implementation details of the bypass fraud detection applications in two different versions. Starting from the explanation of input data, all requirements, design, process and functions of fraud detection application is explained.

In the chapter five, using the penalty algorithm, rough semantic fraud detection standards are converted into numerical values. The penalty points are used to measure the performance of the detection accuracy. The results of two different versions of the fraud detection applications are compared. A system detects clients with higher penalty points can be said to have a better performance. The result shows us that the real time application has much better performance over the delayed time system.

2 State of the Art

The history of bypass (interconnection) frauds is based on Voice over IP services, and thus it has comparatively short history and it recently emerged as serious threats for telecom operators. Thus there is not much information on webs and not much research has been done for this specific type of frauds. Thus in this section the general state of the art of fraud detection systems will be introduced. In particular the focus lies on the survey of unsupervised machine learning algorithms that are used for the fraud detection.

In the first subsection of this chapter, fraud detection techniques especially based on data mining algorithms will be introduced such as clustering algorithms and artificial neural networks. The machine learning algorithms can be generally divided into two parts depending on the existence of the training data in which data are already labeled. If there is training data, supervised learning algorithm can be used and otherwise unsupervised learning algorithms such as clustering algorithms can be used. In this section, machine learning algorithms utilized for the fraud detection system, and the strength and weakness of each method will be introduced. The fraud detection applications are generally developed for three different usages such as credit card fraud, computer intrusion, and telecommunication fraud.

In the third subsection of this chapter, outlier detection technique will be introduced. Although it is the concept in different dimensions which include both supervised and unsupervised learning techniques, the focus will be made on statistical and mathematical perspectives to detect anomalous data without known fraudulent patterns. In this section several techniques that have been developed for the outlier detection will be introduced[1].

2.1 Machine learning algorithms

The data mining algorithm can be generally classified into two big categories called supervised and unsupervised learning algorithms. While supervised learning methods required pre-classified or labeled training data, unsupervised learning algorithm does not require such conditions. Although supervised learning methods show comparatively accurate detection performance, as it already has clear standards to decide on frauds, unsupervised learning methods are somewhat difficult to measure the performance. However it has a greater application opportunity, since it can function without previous detection histories, which are often hard to collect. For the bypass fraud detection, there is no previously classified fraud detection history data.

There is also one fraud detection method called a semi-supervised approach which is the mixture of supervised and unsupervised methods such as anomaly detection, Lee and Xiang (2001) suggested entropy, conditional entropy, relative conditional entropy, information gain, and information cost. However semi-supervised approach is not introduced and only unsupervised learning methods are considered in this section.

For unsupervised algorithms, Yamanishi et al (2004) detect the statistical outliers using the Hellinger and logarithmic scores for insurance[4]; Burge and Shawe-Taylor (2001) employed Hellinger score to determine the difference between short-term and long-term profiles for the telecommunications account. Bolton and Hand (2001) recommends the t-statistic as a score to compute the standardized distance of the target account with centroid of the peer group; and also to detect large spending changes within accounts.

2.1.1 Unsupervised Learning method

It is not easy to gain the real data for the research of fraud detection systen since those data are private information of individuals and they also contains confidential business information of the company. For the bypass fraud detection, we could not gain the information of the system. Some researchers make synthetic data for the analysis and the others seek for collaboration opportunity with the firm that has appropriate data. However our focus in here will be on unsupervised learning methods which do not require the training data.

For the antiterrorism, law enforcement and security purpose, the dominant analysis tools has been a link analysis and graph mining. However these tools are comparatively underevaluated in fraud detection research. A Netmap represents that the emergent group algorithm can be utilized to form groups of tightly connected data and how the visual analysis can lead to the detection of actual frauds from the insurance claim data[5]. In a paper[6], the visual telecommunication fraud detection system is described as a way to represent data flexibly with colors, position, size and other visual effects. This method is devised to combine human intuition with computation ability of machine, since sometimes adding some visual ability of human beings greatly increases the performance of detection. While machine only can distinguish subjects that are evidently distinguished with scientific proofs and sometimes make mistakes even in some obvious cases if the cases are not defined as an initial standard, humans can make decision on ambiguous cases and flexibly apply different measurements to make decisions.

Cortes et al [7] investigated the evolution of large dynamic graphs on a time basis for telecommunication fraud detection. Each graph consists of sub-graphs called Communities Of Interest (COI). The instability of using the current graph and storage of all graphs at all time steps can be overcome through the exponential weighted average approach that updates sub-graphs daily. Using the call quantity and duration to form COIs, mobile phone accounts are linked and consequently two distinctive features of fraudulent clients can be discovered.

Dorronsoro et al [8] invented a non-linear discriminant analysis algorithm which does not need a labeled training data. It functions to minimize the ratio of the determinants within or between class variances in weight projections. Credit card account's past transactions do not exist in history individually, so all transactions have to be separated into different geographical locations. Burge and Shawe-Taylor (2001) utilized a recurrent neural network to create short and long term statistical behavior description profiles. Hellinger distance is used to make comparison on two probability distributions and to give a suspicion score on telecommunications toll tickets.

Yamanishi et al [4] devised the unsupervised smartsifter algorithm which can deal with both categorical and continuous variables. It consequently detects statistical outliers based on Hellinger distance. Bolton and Hand [9] recommended Peer Group Analysis which can monitor inter-account behavior. It compares the weekly cumulative mean of a target account to other similar accounts (named as peer group) at subsequent time points. The distance metric/suspicion score is based on a t-statistic which defines the standardized distance from the centroid of the peer group. The invented Break Point Analysis is invented by them as well to screen intra-account behaviors over time. It can sense the sudden spending or rapid increase in weekly spending within a single account. Accounts are ranked by the t-test. The fixed-length moving transaction window contains twenty-four transactions: first twenty for training and next four for evaluation on credit card accounts.

2.2 Outlier detection

Outlier detection technique is an approach to find anomalous data which highly deviate from the expected data patterns. Such situation is well described in the figure on the next page. It has two sub-categories which are diagnosis and accommodation. The diagnosis approach detects the potential outlying points and excludes such points in the future learning and analysis process. On the other hand accommodation approach includes the outliers into the future processes and applies robust classification method to sort out anomalies. The robust classification set the boundary within data set so that it can distinguish outliers to normal data. Non-robust classifier methods produce skewed representation when outliers are left in and thus it is appropriate when there are only a few outliers in the data set. However non-robust classification methods are computationally less demanding than the robust methods.

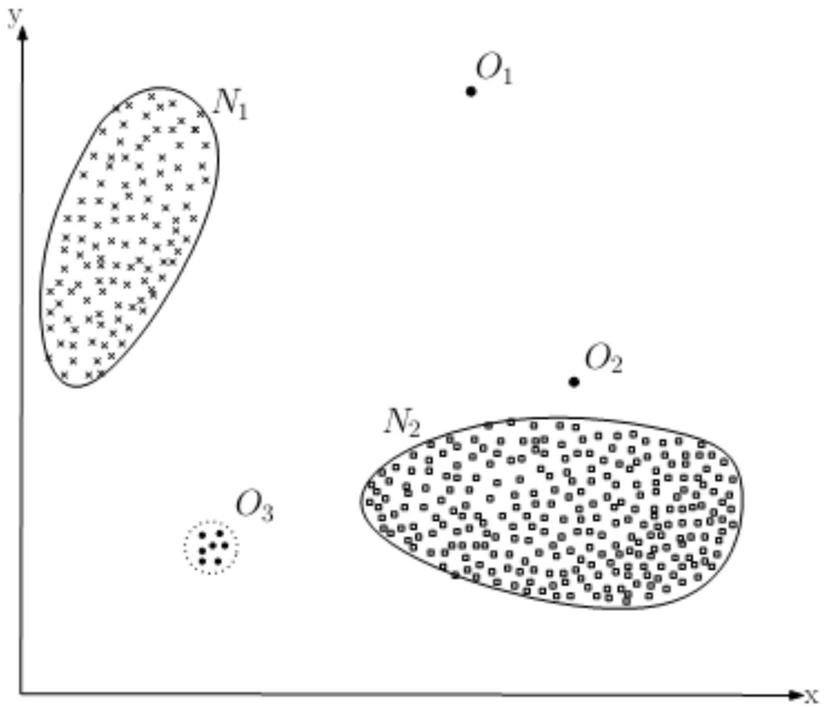


Figure 6 An example of outliers

<Source: A Survey of Outlier Detection Methodologies, Victoria J. Hodge and Jim Austin>

2.1.1 Statistical approach

It is the most rudimentary and initial approach taken to detect outliers in the system. In the early stage most of dataset in trials are one dimensional following both [10] and [11]. One of the principal examples of outlier detection in one dimensional data is from Grubbs' method (Extreme Studentized Deviate). He calculated Z values of all data to measure the deviation of each data from the acceptable range. This analysis is sensitive to the population of the data. As larger the population, the accuracy of the outlier detection increase.

Laurikkala et al [10] informally draw points in a box to sort out outliers relying on the eyes of humans. It is a similar technique as visual inspection and since the human perception is basis, more flexible detection for diverse types of data such as nominal, ordinal, and categorical data are all possible with this method.

One of the major challenges of statistical analysis will be the increasing dimensionality. More dimensions in data leads to more possibility to commit mistakes in detection, because the data points spread through a larger dimension making the density lighter. Several efficient statistical methods are able to focus on principal components and reduce the complexity owing to the greater dimensionality. However a number of techniques still cannot distinguish significant components from the unnecessary data using its algorithm. Thus preprocessing algorithms to preselect the salient attributes can additionally assist the detection process. It is also discovered that only a few attributes actually contribute to the anomalous deviation. Through the data preprocessing, one can sorts out significant attributes from not important data and reduce the quantity of information and complexity of data.

The proximity based analysis is very simple to implement because it does not take into account the distribution of data. There is not much complication to create a distribution model. However, the computational complexity of the model is proportional to the dimensionality and the size of the data, and this is the major disadvantage of this approach. There are many different ways to calculate the distance between instances, from the traditional Euclidean distance and Manhattan distance to Mahalanobis distance. Depending on the types of the data appropriate distance metrics can be selected. For example when there are correlations between attributes Mahalanobis distance is appropriate for the distance, although it is computationally more demanding because of the calculation of the data covariance. It is sometimes better to express the outliers within data.

$$\sqrt{\sum_{i=1}^{n}(\mathbf{x}_i - \mathbf{y}_i)^2}$$

Formula 1 Euclidean distance

$$d_1(\mathbf{p},\mathbf{q}) = \|\mathbf{p}-\mathbf{q}\|_1 = \sum_{i=1}^{n}|p_i - q_i|,$$

Formula 2 Mangattan distance

$$\sqrt{(\mathbf{x}-\mu)^T\mathbf{C}^{-1}(\mathbf{x}-\mu)}$$

Formula 3 Mahalanobis distance

Datta & Kibler [12] use a diagnostic prototype selection to store less data into a few seminal vectors. Skalak utilized both feature selection and prototype selection and thus large data is stored as a few lower dimensional prototype vectors. As outliers and noise are filtered out, the method becomes robust. However as prototyping filter out only significant data, it increases sparsity of the distribution, and the nearest neighbors become denser. Prototyping is quite similar to the k-means.

Dragon Research [13] and Nairac [14] used k-means for outlier detection. Dragon did experiment to detect news stories of new events in on-line. Each of the k clusters provides a local model of the data. The k clusters are represented by prototype vectors with the mean values of all clusters as attribute values. It is required to define the value of k, and such cluster will distinguish the outliers from normal data. Setting a value of k is a tricky problem and it is usually solved after empirical trials. For the classification purpose, data only computed k times with the formula below, so the computation complexity of the model is low and the classification can be done comparatively quickly.

$$\sum_{j=1}^{K}\sum_{n\in S_j}||\mathbf{x}^n - \mu_j||^2$$

Figure 7 k-means distance formula

The outlier can be detected based on its distance from the centroid. The threshold to decide on outliers is usually based on the distance of a vector from a prototype vector. The value of the threshold is chosen by the difference in the minimum and maximum values of a certain centroid.

The graph connectivity method is another proximity based approach introduced by Shekhar et al [15]. It is applied to traffic monitoring which topologically examines the connection of one node's neighborhoods rather than distance of each point from a centroid. A node is decided as outliers, if the number of connection it has is different to other nodes' general tendency.

The parameter method evaluates the data based on the predefined distribution. Thus there is no complexity in computation derived from the size of data since the base distribution is already set. One such approach is Minimum Volume Ellipsoid estimation (MVE) which generates the smallest permissible ellipsoid volume where the majority of the data distribution model exist (generally covering 50% of the data points).

A similar approach, Convex Peeling filters out any data point on the boundaries of the data distribution's convex hull [16]. Thus outliers are crossed out in the graph. In contrast MVE maintains all points and distinguish outliers by boundary. Both MVE and Convex Peeling are robust classifiers which set threshold around that fit boundaries to filter out outliers. Both methods make the dense convex hull and classify the data with a threshold for decisions, but they are only applicable with low dimensional data. On the other hand Principal Component Analysis (PCA) [17] is applicable to higher dimensional data as well. It figures out correlated attributes in the distribution of the data. PCA cannot outperform the complex boundaries by support vector machine or neural methods. It is assumed that the subspaces generated by the principal components are compact and thus this becomes a limitation on its applicability for sparse distributions. However, still PCA functions as an ideal pre-processor to reduce the dimensionality of data by selecting significant attributes for the outlier detection when the dimensionality is high. PCA can identify the principal component of greatest variance using its eigenvalue, because the size of the eigenvalue reflects the deviation of the data from the component vector. PCA holds the k principal components with greatest variance as a significant attributes, and discards all others which allows it to gain maximum information with minimum redundancy.

The regression techniques and PCA that are introduced previously are very simple and rudimentary analysis methods which can be implemented in many other practical cases. Tax & Duin [18] use Support Vector Machines (SVMs) and it generates class boundaries for supervised classification. SVMs use kernel functions to project data in higher dimensional plane and on that dimension, normal and outliers are distinguished.

Another major approach for outlier detection is an artificial neural network. It is generally non-parametric and model-based. It handles many different types of data changing the organization of nodes within networks as the traits and tendency of input data change. After training of nodes in networks, tests are made to label unclassified data. Although the high dimensionality decrease the performance of artificial neural networks, the impact of dimensionality is still less than statistical methods.

3 Delayed time detection

The application in delayed time retrieved the one day call detail record (CDR) from the data base to analyze the phone call history of clients. The call detail record is input into the clustering algorithm, called Cobweb (conceptual clustering algorithm), and clients are classified into several groups (clusters) depending on the calling patterns of each customer.

3.1 Structure

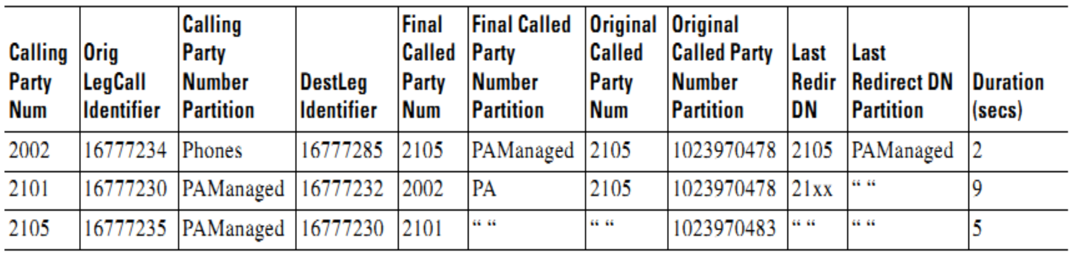
The system retrieves stored call detail record from the data base to process detailed analysis on the data in relevant time scale. In this application the duration will be set to one day. For a day about 30,000,000 lines for voice CDR (about 4 gigabyte size) and 10,000,000 lines for incoming voice CDR (about 1.5 gigabyte size) per each file.

Field	Description	Type
ID_CELDA_INI	ID of the cell where the call is initiated. Is a hexadecimal code. The last 8 characters correspond to the catalog.	Number(17)
TFNO_TEMM	Phone number of TEMM.	Number(10)
ID_CELDA_FIN	ID of the cell where the call terminates. Is a hexadecimal code. The last 8 characters correspond to the catalog.	Number(17)
TFNO_OTRO	Phone number of ANOTHER.	Number(10)
SENTIDO	Direction of the call (1 incoming 2 outgoing).	Integer
TIPO_LLAMADA	Call type used by the client: LOCAL, LD, LDI, LDM, and so on.	String
FECHA	Date on which the call occurs, in the format dd / mm / yyyy.	Date
HORA	Time that the call occurs, in hh: mm: ss.	String
DURACION	Call duration in seconds.	Integer
OP_ORIGEN	Operator of the paid call origin. Corresponds to the telephone operator of TEMMA. You may see calls with another operator other than TEMMA: this is because the line has been ported to	String

	TEMMA.	
OP_DESTINO	Operator of paid call destination. Corresponds to the OTHER phone operator.	String
COD_PAIS	Identifier of the country where the TFNO_OTRO is involved. If Mexico, the field is empty.	Number(7)
ID_TERMINACION	Identifier Code of call termination. It is possible that not all calls have it, in which case the field is empty.	Integer
IMEI	Code of terminal from the phone TEMMA.	Number(15)
ID_ROAMING	Roaming ID when the phone TEMMA sending or receiving calls is found. Possible values: • "NI" - Not reported. • "NA" - Not applicable. • "GE" - Generic. • "OT" - Other. • "NOROAMI" - No roaming. • "RMITRAR" - Roaming within the region. • "RMITERR" - Roaming interregional. • "RMNACIO" - National Romaing. • "RMITNCL" - International Roaming. • "RMMUNDI" - Global Roaming. • "RVISITA" - Roaming visitor.	String

Table 1 CDR description

<Source: Telefonica Mobil Mexico CDR description>



Calling Party Num	Orig LegCall Identifier	Calling Party Number Partition	DestLeg Identifier	Final Called Party Num	Final Called Party Number Partition	Original Called Party Num	Original Called Party Number Partition	Last Redir DN	Last Redirect DN Partition	Duration (secs)
2002	16777234	Phones	16777285	2105	PAManaged	2105	1023970478	2105	PAManaged	2
2101	16777230	PAManaged	16777232	2002	PA	2105	1023970478	21xx	" "	9
2105	16777235	PAManaged	16777230	2101	" "	" "	1023970483	" "	" "	5

Figure 8 CDR examples

<Source: Cisco Unified Communications Manager Call Detail Records Administration Guide>

The Call Detail Record (CDR) data of Voice and SMS services are the only necessary input data for this application and they are analyzed based on one day time period. (Note that for this algorithm raw Internet traffic is not required.) The CDR data which contain all information related to the service history are recorded line by line and information in each line is divided by the symbols such as comma and vertical bar. There are two types of CDR for incoming and outgoing calls and the description of CDR data is in table 1. The CDR data is a call history log of Telefónica mobile Mexico clients. (TEMM) Fields of call information are divided by '|' and each line represents either incoming or outgoing call occurred to Telefónica mobile Mexico's clients. There is no duplicate information about calls. To avoid duplication of call data, there is one additional CDR for the calls that is made between Telefónica customers. The call history among our customers is recorded in voice and incoming voice CDR file twice. In the incoming voice CDR file, the sentido field of this file is always '1' which means only incoming calls are recorded here. The field information is the same as the voice CDR.

However for our study only information as follows is necessary: The number that made the call, the number that received the call, the duration of the call, the time of the call, and the call type (SMS or voice). Thus there is a data preprocessing part where only necessary data is extracted from the total CDR data.

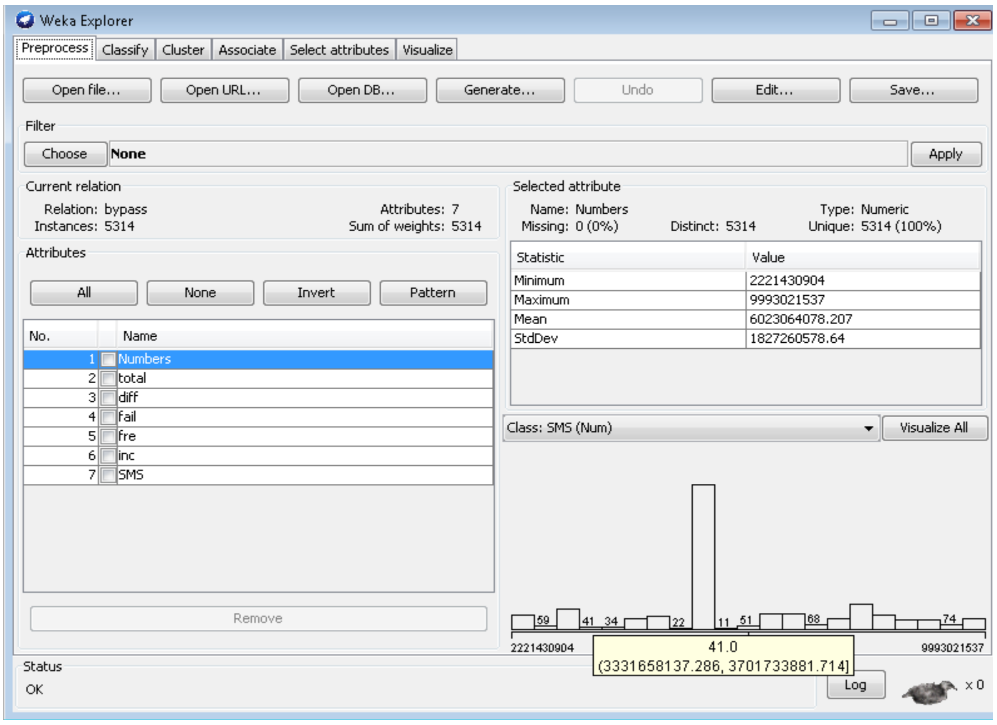
The program starts with reading CDR data file of voice and SMS. As CDR of Voice is read, calls that are made in irrelevant calling time period are filtered out and it selectively stores the information of our interests such as the number that made the call, the number that received the call, the duration of the call, the time of call, call type (SMS or voice) within the relevant time frame. The data are stored into files and are loaded into the MySQL database in sequence.

```
@relation 'iris'
@attribute 'sepallength' real
@attribute 'sepalwidth' real
@attribute 'petallength' real
@attribute 'petalwidth' real
@attribute 'class' {'Iris-setosa','Iris-versicolor','Iris-virginica'}
@data
5.1,3.5,1.4,0.2,'Iris-setosa'
4.9,3,1.4,0.2,'Iris-setosa'
4.7,3.2,1.3,0.2,'Iris-setosa'
4.6,3.1,1.5,0.2,'Iris-setosa'
5,3.6,1.4,0.2,'Iris-setosa'
5.4,3.9,1.7,0.4,'Iris-setosa'
4.6,3.4,1.4,0.3,'Iris-setosa'
5,3.4,1.5,0.2,'Iris-setosa'
4.4,2.9,1.4,0.2,'Iris-setosa'
4.9,3.1,1.5,0.1,'Iris-setosa'
5.4,3.7,1.5,0.2,'Iris-setosa'
4.8,3.4,1.6,0.2,'Iris-setosa'
4.8,3,1.4,0.1,'Iris-setosa'
4.3,3,1.1,0.1,'Iris-setosa'
5.8,4,1.2,0.2,'Iris-setosa'
5.7,4.4,1.5,0.4,'Iris-setosa'
5.4,3.9,1.3,0.4,'Iris-setosa'
5.1,3.5,1.4,0.3,'Iris-setosa'
5.7,3.8,1.7,0.3,'Iris-setosa'
5.1,3.8,1.5,0.3,'Iris-setosa'
5.4,3.4,1.7,0.2,'Iris-setosa'
5.1,3.7,1.5,0.4,'Iris-setosa'
4.6,3.6,1,0.2,'Iris-setosa'
5.1,3.3,1.7,0.5,'Iris-setosa'
4.8,3.4,1.9,0.2,'Iris-setosa'
5,3,1.6,0.2,'Iris-setosa'
5,3.4,1.6,0.4,'Iris-setosa'
5.2,3.5,1.5,0.2,'Iris-setosa'
5.2,3.4,1.4,0.2,'Iris-setosa'
4.7,3.2,1.6,0.2,'Iris-setosa'
4.8,3.1,1.6,0.2,'Iris-setosa'
  ....
```

Figure 9 an example of arff file

When reading the CDR of SMS, it only selects phone numbers and directly loads them into MySQL database. The phone numbers with SMS history are selected from database and are ordered by calling numbers to be joined to the SMS history data. Next all data is ordered by number and calling time to calculate the lapsed time between calls that are made by one client. The purpose is to measure the frequency of calls. When all data that include time between calls are again uploaded to MySQL database, a query is made to gain total number of calls, number of different phone numbers, number of calls that lasts less than 10 seconds, average lapsed time between consecutive calls. The processes in Java and in database are repeatedly done to manage a big quantity of data in the most efficient manner. The result of query is recalled to be written in the file in terms of vectors which are input data for WEKA (Data Mining

Software in Java). The vectors include the information about each phone number such as total number of calls, portion of calls made to different phone number, portion of calls that lasts less than 10 seconds, average time between each calls (in the unit of seconds), number of incoming calls and SMS. These vectors which are organized by each calling number are saved as arff (Attribute-Relation File Format) to be a test set for the neural network software, Weka. An example of input files for the Weka is shown in figure 8. The description of the process of clustering using Weka is shown in following figures.



Filter: Choose None Apply

Current relation
Relation: bypass
Instances: 5314
Attributes: 7
Sum of weights: 5314

Attributes

No.	Name
1	Numbers
2	total
3	diff
4	fail
5	fre
6	inc
7	SMS

Selected attribute
Name: Numbers
Missing: 0 (0%)
Distinct: 5314
Type: Numeric
Unique: 5314 (100%)

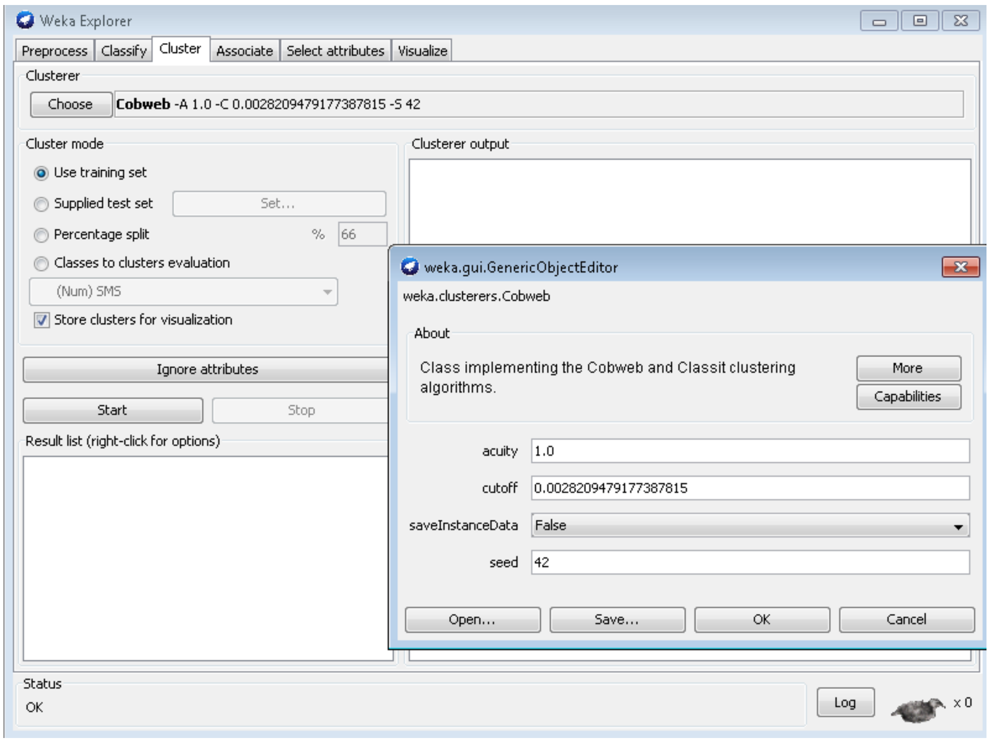
Statistic	Value
Minimum	2221430904
Maximum	9993021537
Mean	6023064078.207
StdDev	1827260578.64

Class: SMS (Num)

59 41 34 22 11 51 68 74
2221430904
41.0
(3331658137.286, 3701733881.714]
9993021537

Status
OK

Figure 10 A main window after choosing an input data



Weka Explorer

Preprocess | Classify | Cluster | Associate | Select attributes | Visualize

Clusterer

Choose **Cobweb** -A 1.0 -C 0.0028209479177387815 -S 42

Cluster mode: (●) Use training set; ( ) Supplied test set; ( ) Percentage split % 66; ( ) Classes to clusters evaluation; (Num) SMS; [✓] Store clusters for visualization

Clusterer output

weka.gui.GenericObjectEditor

weka.clusterers.Cobweb

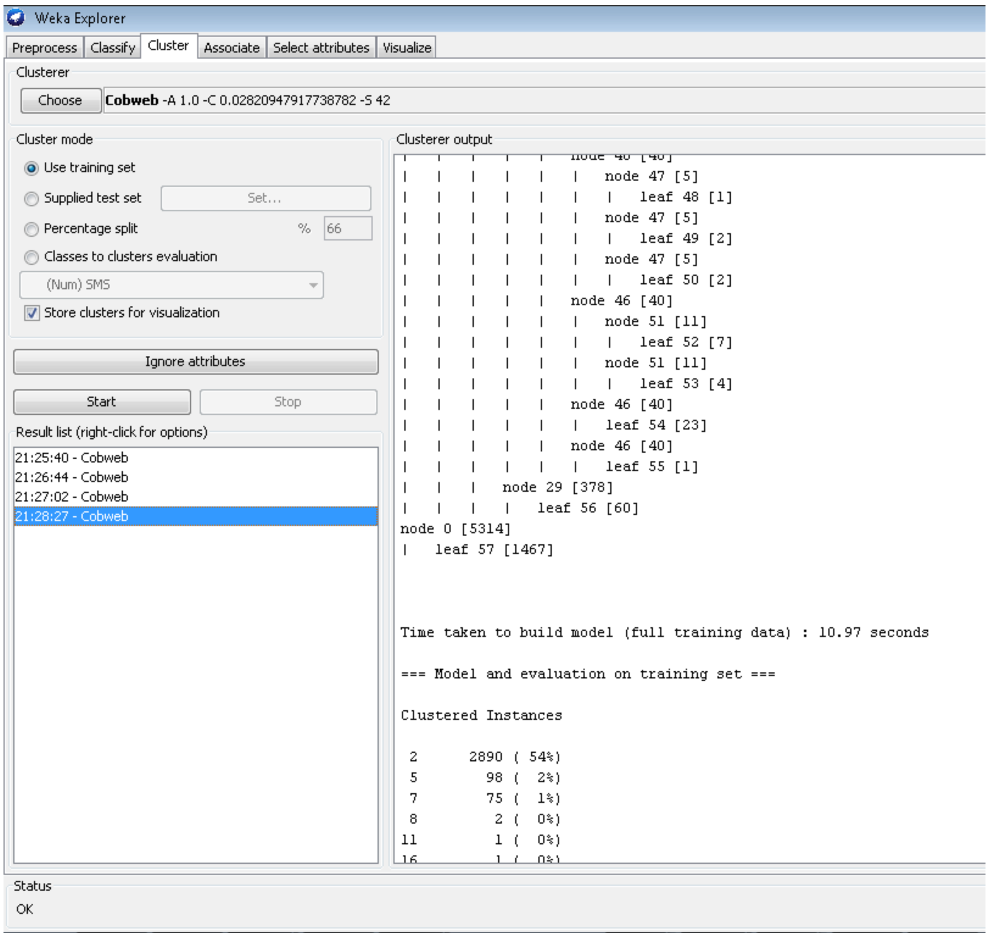
About

Class implementing the Cobweb and Classit clustering algorithms.

acuity	1.0
cutoff	0.0028209479177387815
saveInstanceData	False
seed	42

Status

OK



Weka Explorer

Preprocess | Classify | Cluster | Associate | Select attributes | Visualize

Clusterer

Choose **Cobweb** -A 1.0 -C 0.028209479177387 82 -S 42

Cluster mode: (●) Use training set; ( ) Supplied test set; ( ) Percentage split % 66; ( ) Classes to clusters evaluation; (Num) SMS; [✓] Store clusters for visualization

Result list (right-click for options)

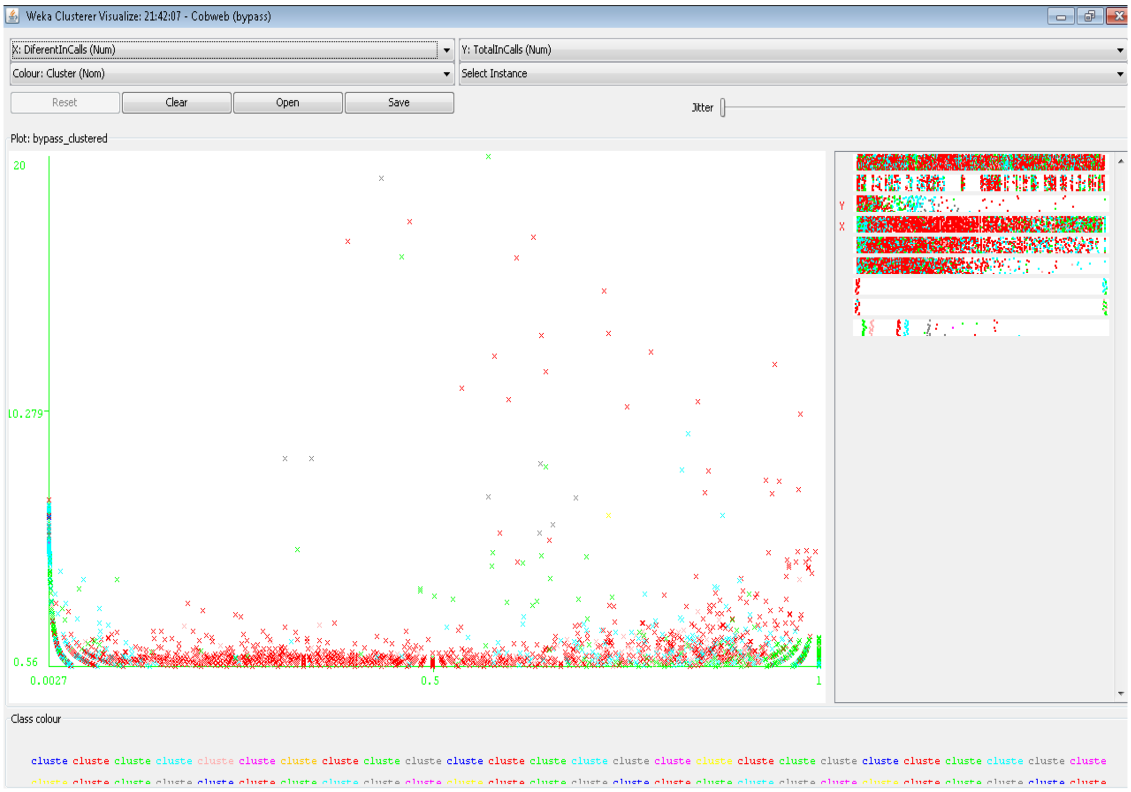
21:25:40 - Cobweb
21:26:44 - Cobweb
21:27:02 - Cobweb
21:28:27 - Cobweb

Clusterer output

```
|   |   |   |   |   node 46 [40]
|   |   |   |   |   |   node 47 [5]
|   |   |   |   |   |   |   leaf 48 [1]
|   |   |   |   |   |   node 47 [5]
|   |   |   |   |   |   |   leaf 49 [2]
|   |   |   |   |   |   node 47 [5]
|   |   |   |   |   |   |   leaf 50 [2]
|   |   |   |   |   node 46 [40]
|   |   |   |   |   |   node 51 [11]
|   |   |   |   |   |   |   leaf 52 [7]
|   |   |   |   |   |   node 51 [11]
|   |   |   |   |   |   |   leaf 53 [4]
|   |   |   |   |   node 46 [40]
|   |   |   |   |   |   leaf 54 [23]
|   |   |   |   |   node 46 [40]
|   |   |   |   |   |   leaf 55 [1]
|   |   |   node 29 [378]
|   |   |   |   leaf 56 [60]
node 0 [5314]
|   leaf 57 [1467]


Time taken to build model (full training data) : 10.97 seconds

=== Model and evaluation on training set ===

Clustered Instances

 2      2890 ( 54%)
 5        98 (  2%)
 7        75 (  1%)
 8         2 (  0%)
11         1 (  0%)
16         1 (  0%)
```

Status

OK



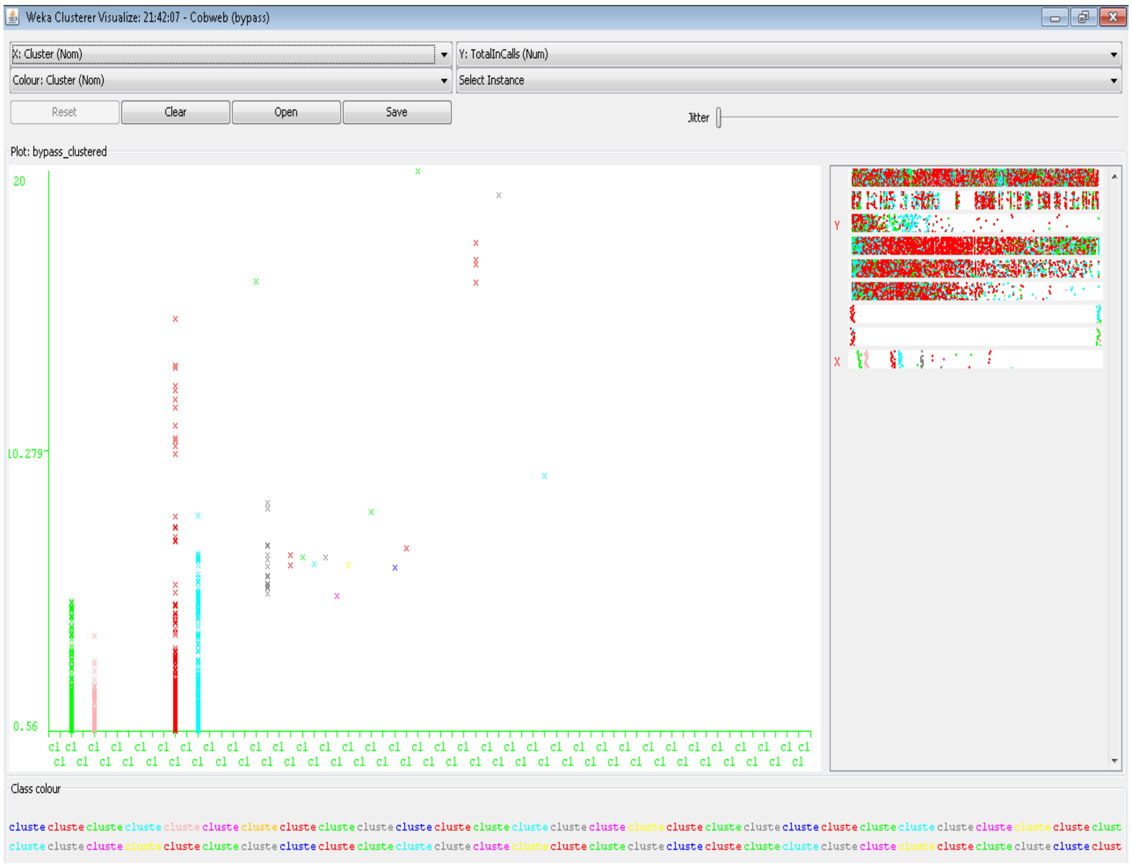


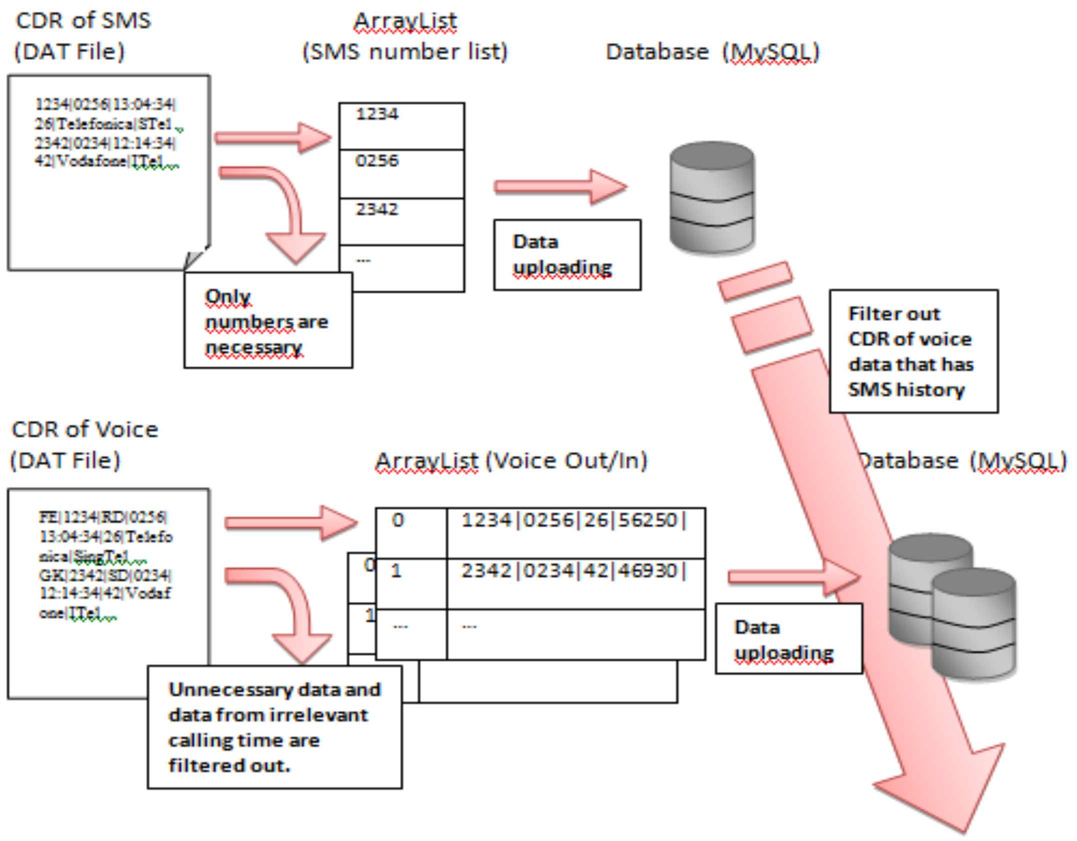
Figure 13 Graphical results of clustering

In this application, the conceptual clustering algorithm called Cobweb is used to group the call history vectors of clients according to their calling patterns. Among diverse clustering algorithm the conceptual clustering algorithm, Cobweb is selected since the algorithm is very effective to group clients who have similar calling behaviors. While most other clustering algorithms cluster data according to the Euclidean distance between data points and a cluster centroid, the Cobweb functions to minimize the standard deviation of all attributes in one cluster. Thus it is most effective to group members with their calling patterns to decide on suspicious groups. In a result of the application, clusters with few members are more likely to show suspicious behavior patterns.

This application classifies a large quantity of test set which approximately amounts to 5 million rows per day. Some traits and patterns about the frauds are known but still there are no definite standards to detect the fraud. The best method we can depend on in such situations is an unsupervised learning method, which does not require a training set for the classification. In particular, clustering algorithm is utilized in this application. A number of trials with different parameters and conditions have been carried and the clustering is executed with the best configuration setting to label the test set and consequently detect frauds from input data.

The data mining algorithms have the remarkable capacity to derive meaning from complicated data with imprecise standards through a learning process. There are two types of learning: one is supervised learning and the other is unsupervised learning. The classified training set is provided in supervised learning so that algorithm can learn it and predict a result of the future test set. On the other hand unsupervised learning does not have a training set and neural networks are required to infer basic functions or patterns within the data set. In Weka as one presses different tabs such as classify, and associate, diverse algorithms can be discovered and they can be easily applied to the data.

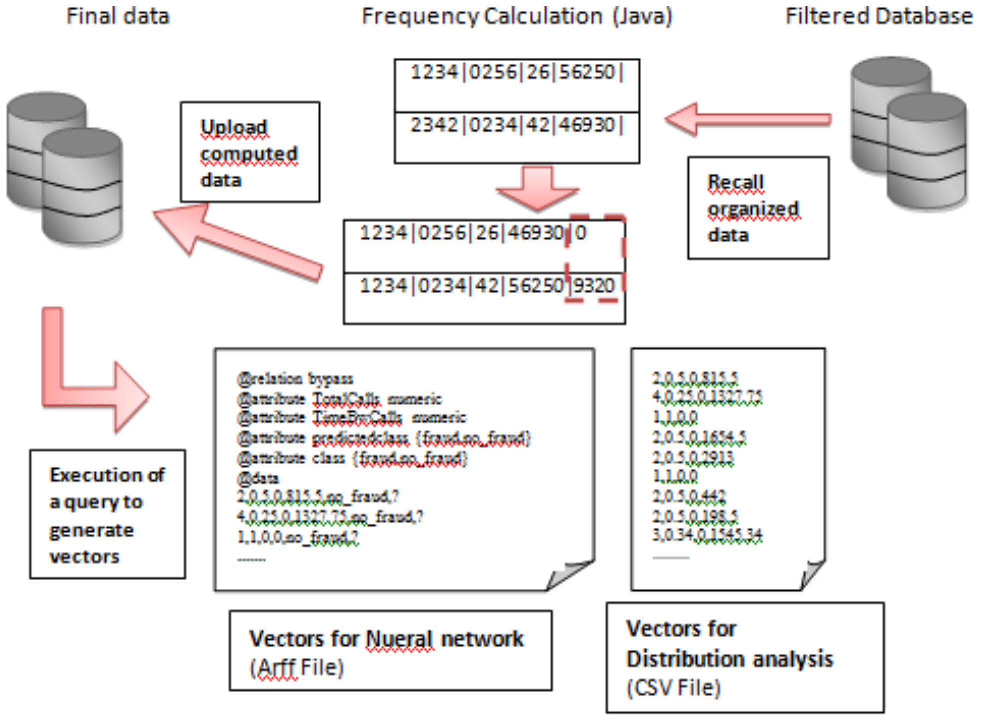
The following figures graphically represent the detection processes in the application process.



```
1234|0256|13:04:34|
26|Telefonica|STel
2342|0234|12:14:34|
42|Vodafone|ITel
```

```
FE|1234|RD|0256|
13:04:34|26|Telefo
nica|SingTel
GK|2342|SD|0234|
12:14:34|42|Vodaf
one|ITel
```

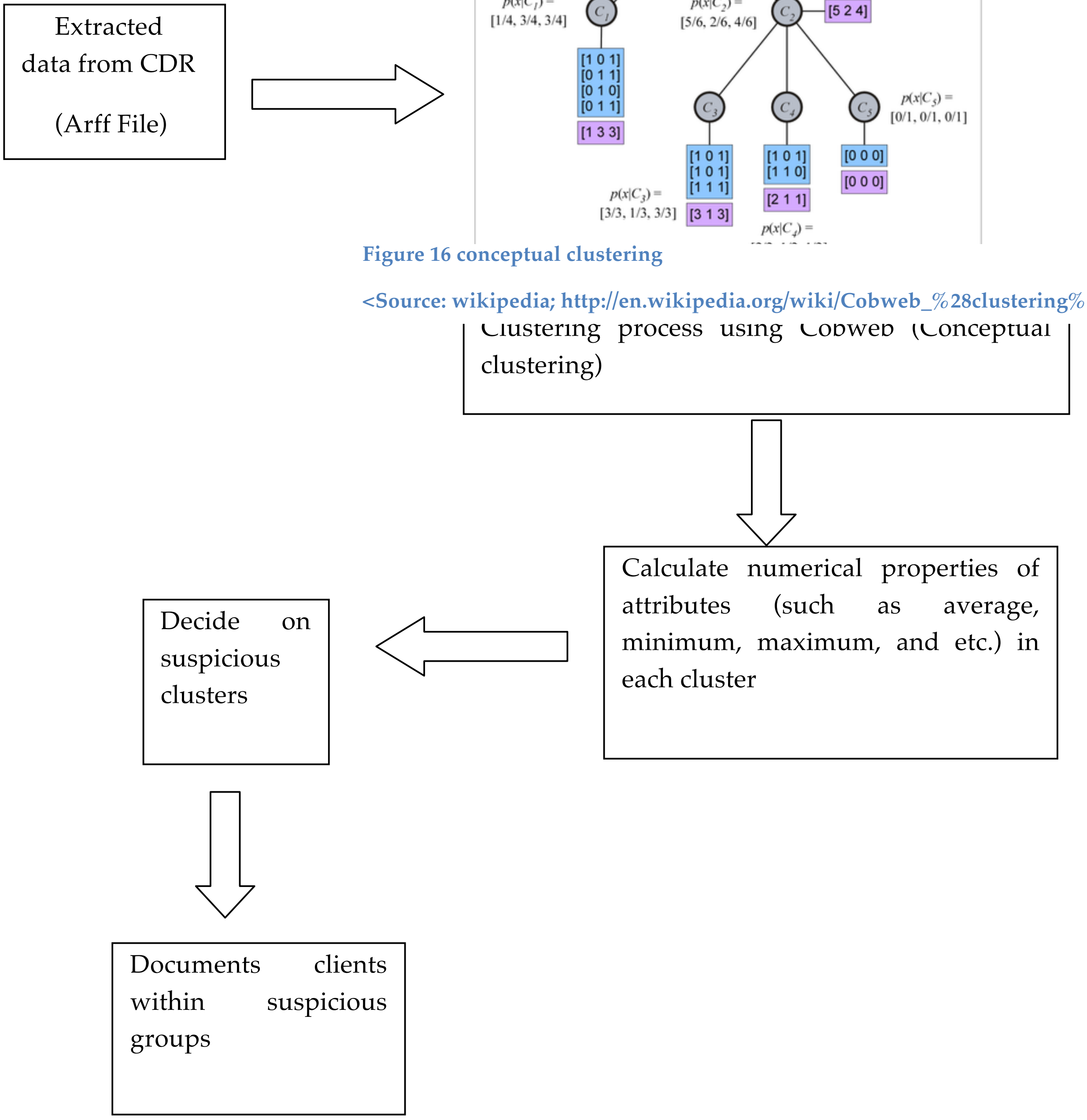
Figure 14 The first data extraction stage



```
@relation bypass
@attribute TotalCalls numeric
@attribute TimeBwCalls  numeric
@attribute predictedclass {fraud,no_fraud}
@attribute class {fraud,no_fraud}
@data
2,0.5,0,815.5,no_fraud,?
4,0.25,0,1327.75,no_fraud,?
1,1,0,0,no_fraud,?
-------
```

```
2,0.5,0,815.5
4,0.25,0,1327.75
1,1,0,0
2,0.5,0,1654.5
2,0.5,0,2913
1,1,0,0
2,0.5,0,442
2,0.5,0,198.5
3,0.34,0,1545.34
--------
```

Figure 15 the data processing and detection stage


Extracted data from CDR
(Arff File)
Clustering process using Cobweb (Conceptual clustering)
Calculate numerical properties of attributes (such as average, minimum, maximum, and etc.) in each cluster
Decide on suspicious clusters
Documents clients within suspicious groups

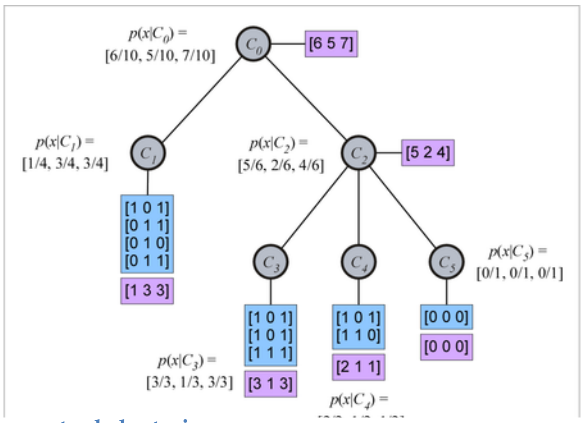
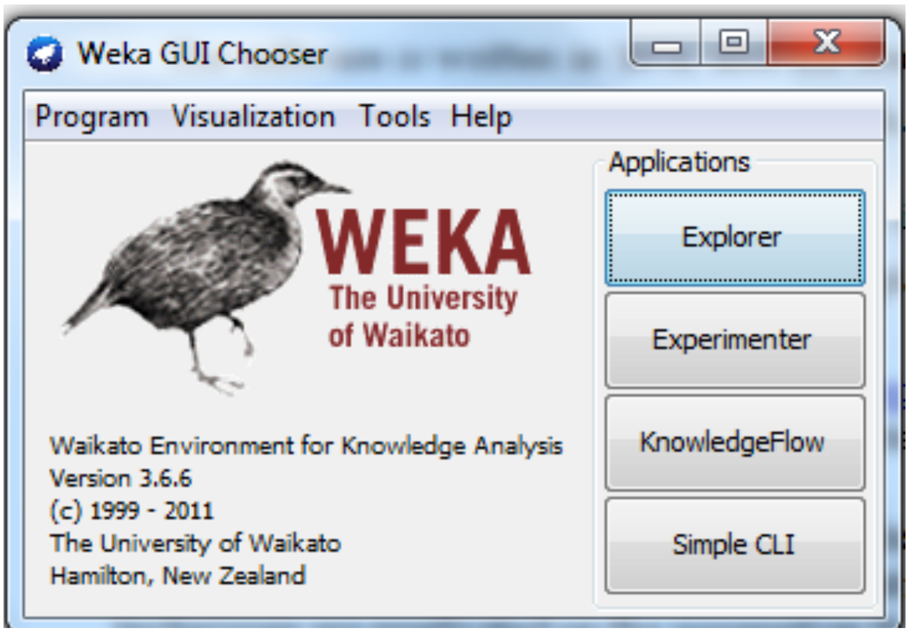


Figure 16 conceptual clustering

<Source: wikipedia; http://en.wikipedia.org/wiki/Cobweb_%28clustering%29 >

3.2 Used Technology

3.2.1 WEKA

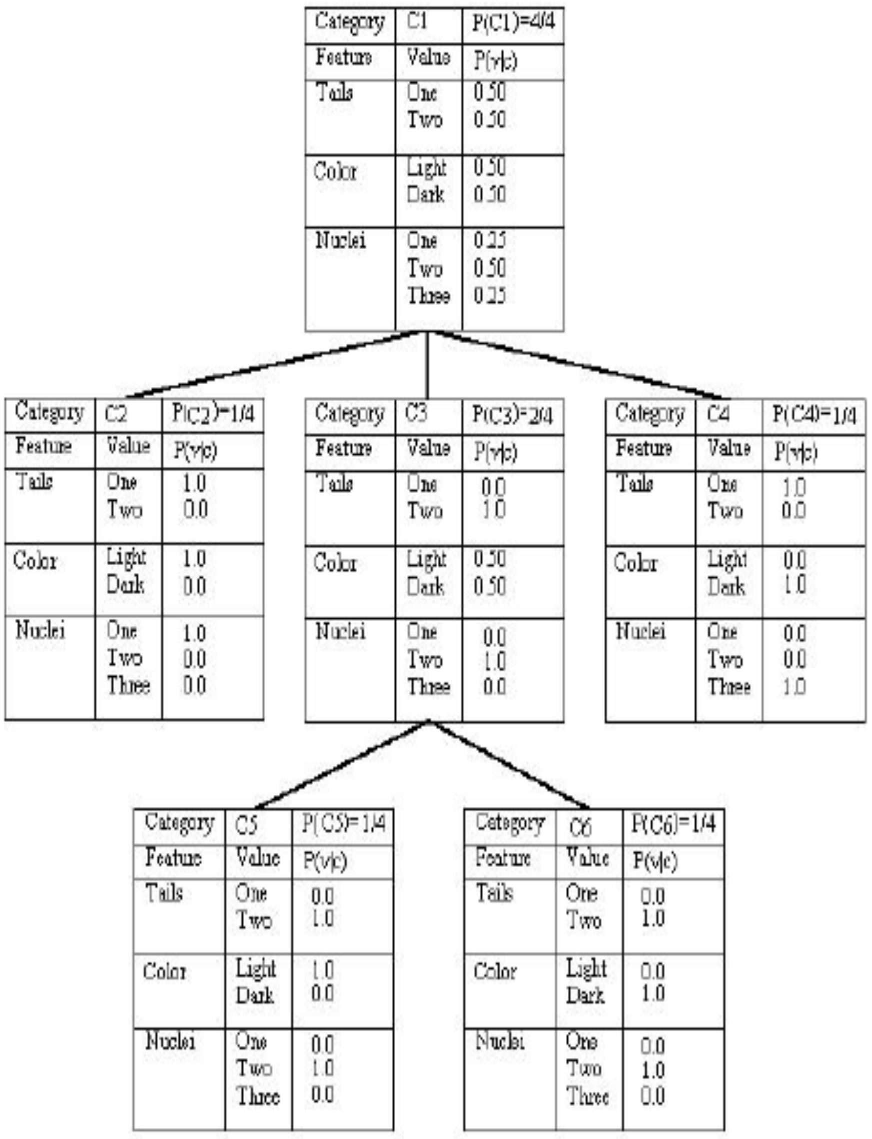


Waikato Environment for Knowledge Analysis
Version 3.6.6
(c) 1999 - 2011
The University of Waikato
Hamilton, New Zealand

Figure 17 Main GUI of Weka

Weka is software written in Java, which consists of a group of data mining algorithms. It is free software provided by the University of Waikato, New Zealand. It also supports additional visualization tools and algorithms for data analysis such as association, clustering and learning. Before the real analysis, the process of making appropriate input data is possible with data pre-processing tools. The tools provide various features such as filtering, removal, changing attributes, deleting empty data, and so on. With such algorithms and input data, it constructs models to predict the unknown values, to classify data, and to associate data. The result of such modeling can be presented either in text files or in graphs.

The software is written in Java, and the source code is open to public. Thus one can freely modify the code considering their needs. Weka does not process a large amount of data well. It depends on the complexity of the algorithm though; Weka in general consumes a lot of memory for the mining process. The allocation of greater memory for the Weka can be set using the command line [20].



Category	C1	P(C1)=4/4
Feature	Value	P(v|c)
Tails	One	0.50
	Two	0.50
Color	Light	0.50
	Dark	0.50
Nuclei	One	0.25
	Two	0.50
	Three	0.25

Category	C2	P(C2)=1/4
Feature	Value	P(v|c)
Tails	One	1.0
	Two	0.0
Color	Light	1.0
	Dark	0.0
Nuclei	One	1.0
	Two	0.0
	Three	0.0

Category	C3	P(C3)=2/4
Feature	Value	P(v|c)
Tails	One	0.0
	Two	1.0
Color	Light	0.50
	Dark	0.50
Nuclei	One	0.0
	Two	1.0
	Three	0.0

Category	C4	P(C4)=1/4
Feature	Value	P(v|c)
Tails	One	1.0
	Two	0.0
Color	Light	0.0
	Dark	1.0
Nuclei	One	0.0
	Two	0.0
	Three	1.0

Category	C5	P(C5)=1/4
Feature	Value	P(v|c)
Tails	One	0.0
	Two	1.0
Color	Light	1.0
	Dark	0.0
Nuclei	One	0.0
	Two	1.0
	Three	0.0

Category	C6	P(C6)=1/4
Feature	Value	P(v|c)
Tails	One	0.0
	Two	1.0
Color	Light	0.0
	Dark	1.0
Nuclei	One	0.0
	Two	1.0
	Three	0.0

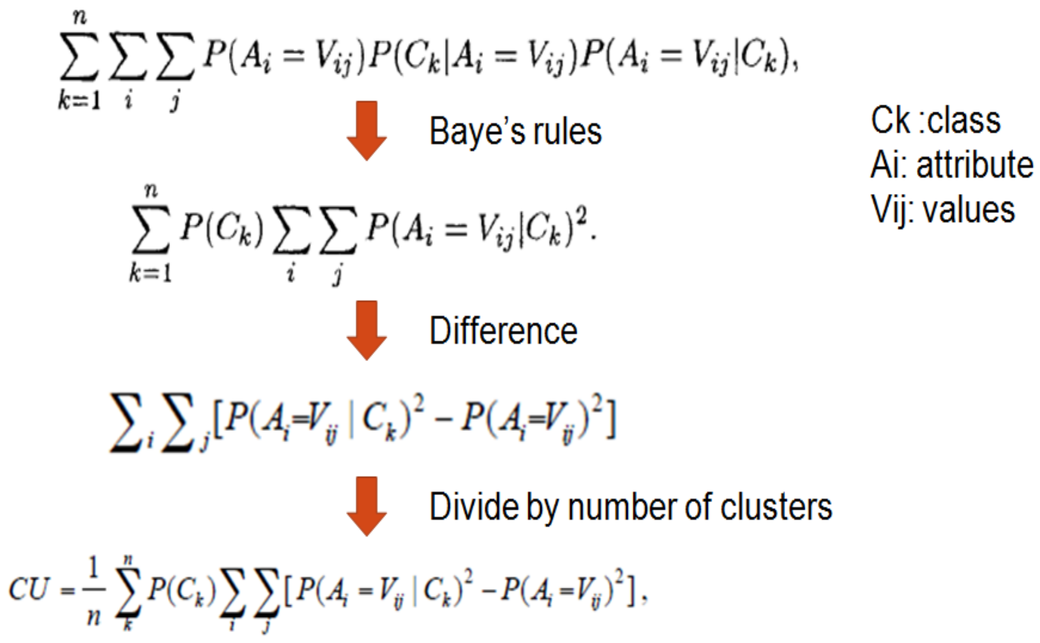
Figure 18 A tree for the conceptual clustering

<Source: Fisher, Douglas H. (1987). "Knowledge acquisition via incremental conceptual clustering". Machine Learning 2 (2): 139–172.>

In the application, the source code of Weka is modified to cluster call history of all clients. As the clustering algorithm, Cobweb, a conceptual clustering algorithm is utilized. Among different clustering algorithms COBWEB is selected, because it is incremental and is based on a unique algorithm to cluster data. COBWEB is devised to maximize the inference of data based on the observation from the previous data. The inference ability is measured by the category utility function. The category utility function is derived in the figure 16. The function is about the tradeoff of intra-class similarity and inter-class dissimilarity. The similarity is measured by the numerical calculation of possibility of inference of each individual attributes and the possibilities are added all together. The intra-class similarity can be represented as a function of

$P(A_i = V_{ij} \mid C_k)$ and it means a Predictability of *C* given *V*. Larger *P* means if class is *C*, *A* likely to be *V* and objects within a class should have similar attributes. The inter-class dissimilarity is defined with a function of *$P(C_k \mid A_i = V_{ij})$*, and it means predictiveness of *C* given *V*. Larger *P* means *A=V* and it also suggests that instance is a member of class *C* rather than some other classes. *A* is a stronger predictor of class *C* [23].

The conceptual clustering is an incremental algorithm and thus it starts with empty tree without any predefined structure. It builds a classification tree with leaves that represent instances and objects. The built nodes are probabilistic concepts (classes) that retain probabilities of being chosen as a certain attribute appears.



$$\sum_{k=1}^{n}\sum_{i}\sum_{j} P(A_i = V_{ij})P(C_k|A_i = V_{ij})P(A_i = V_{ij}|C_k),$$

$$\sum_{k=1}^{n} P(C_k)\sum_{i}\sum_{j} P(A_i = V_{ij}|C_k)^2.$$

$$\sum_i \sum_j [P(A_i{=}V_{ij} \mid C_k)^2 - P(A_i{=}V_{ij})^2]$$

$$CU = \frac{1}{n}\sum_{k}^{n} P(C_k)\sum_{i}\sum_{j}[P(A_i = V_{ij} \mid C_k)^2 - P(A_i = V_{ij})^2],$$

Figure 19 Basis of COBWEB algorithm

The algorithm of COBWEB develops as follows. Firstly an input data is added incrementally and the data is dropped down through the tree until a new node including the data is created. The probability of each node is updated. The category utility function is calculated at each level of the tree. Depending on the value of the category utility function a newly input data can be assigned to an existing class (inserting),to a newly created class node, to a class which is combination of two classes (merging), to a class divided into multiple classes (splitting).

It is also programmed to minimize the standard deviation of values within one node, and if the input data is not in numerical values, for example in nominal values, frequency of occurrence of nominal values is calculated and the data is classified into the node which contains the corresponding data with more frequency. Thus the clustering has effect of grouping data that shares the similar traits in each attributes. With this algorithm it is expected to classify the clients who show the similar calling patterns. The clustering methods that classify data using distance measures (most of clustering algorithms do so) cannot classify data in nominal values. Secondly, although the distance between two data points is not large, sometimes the real meanings of the data greatly differ. Because a conceptual clustering focuses on making inference within data, it more efficiently extracts the hidden behavioral patterns in calling histories.

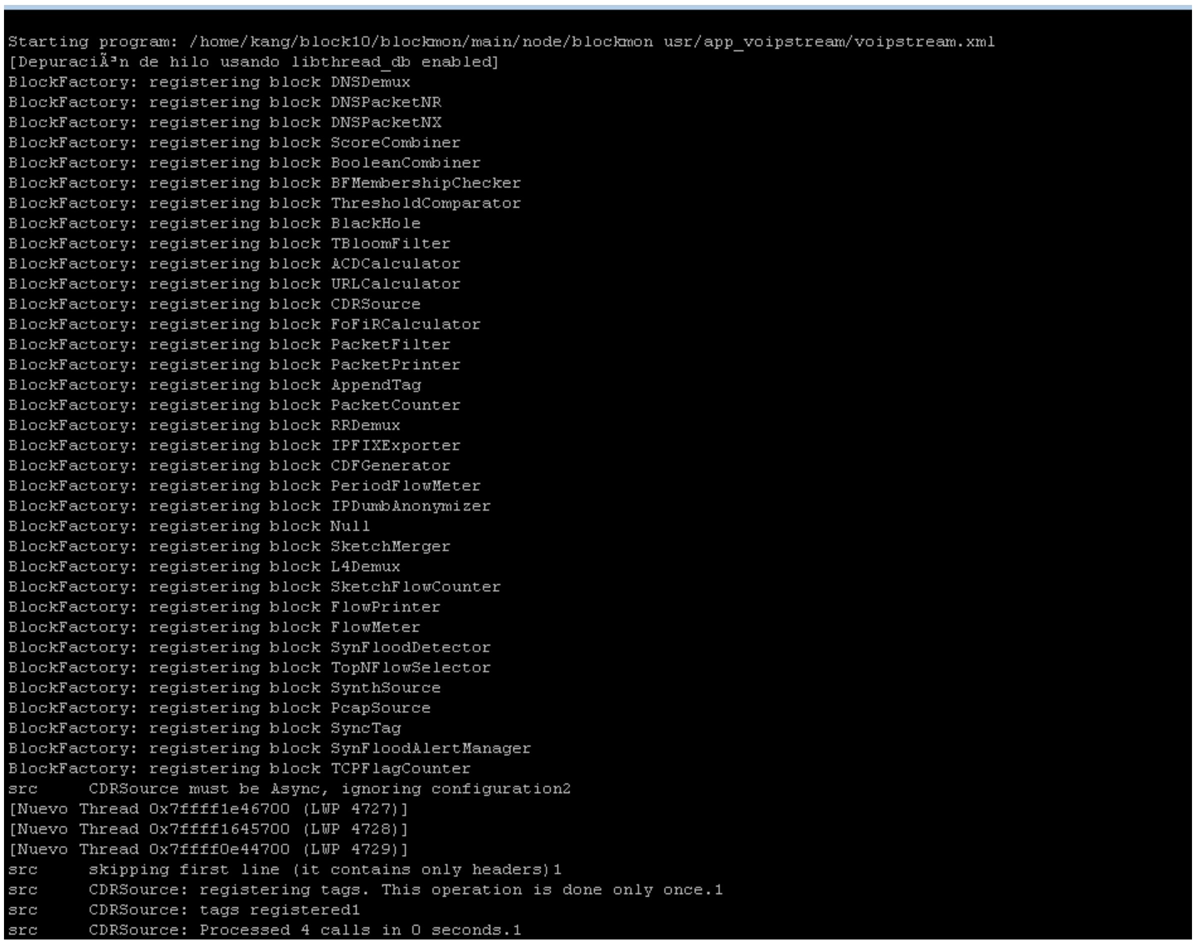
3.3 Stength and weakness

The delayed time system has its strength in a sense that it has sufficient information resource from the call detail records. As it directly retrieves the information about individuals' calling patterns, original data is easily approached and analyzed as one wants.

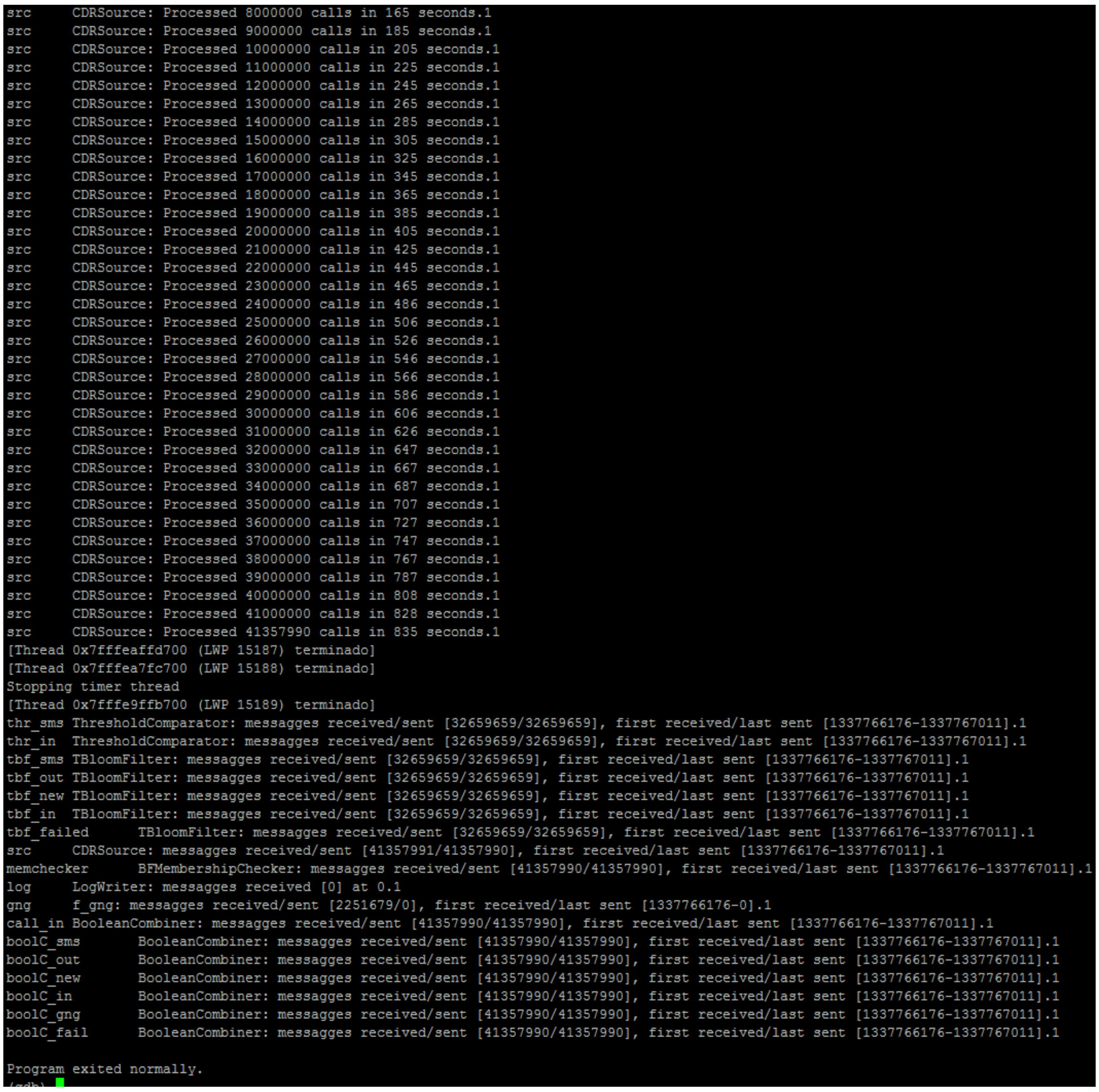
However as the privacy protection of clients has become more important, the storage of call history of individuals and further analysis of call history is very limited. Thus delayed time system cannot be used for official analysis purpose. In addition, there is comparatively larger time delay in detection. Thus real-time system with improved performance is introduced in the next chapter.

4 Real-time System

The real time system is invented to resolve the limitation suggested from the delayed time system. The real time system is based on the C++, Python and XML language which enable each function to work as blocks. It is run on the linux based machine using command-line. There are some screen captures, when the application is executed in linux. As the command line execute the blockmon with the target xml files, all basic blocks and special blocks indicated by xml files are registered and are ready to be used.

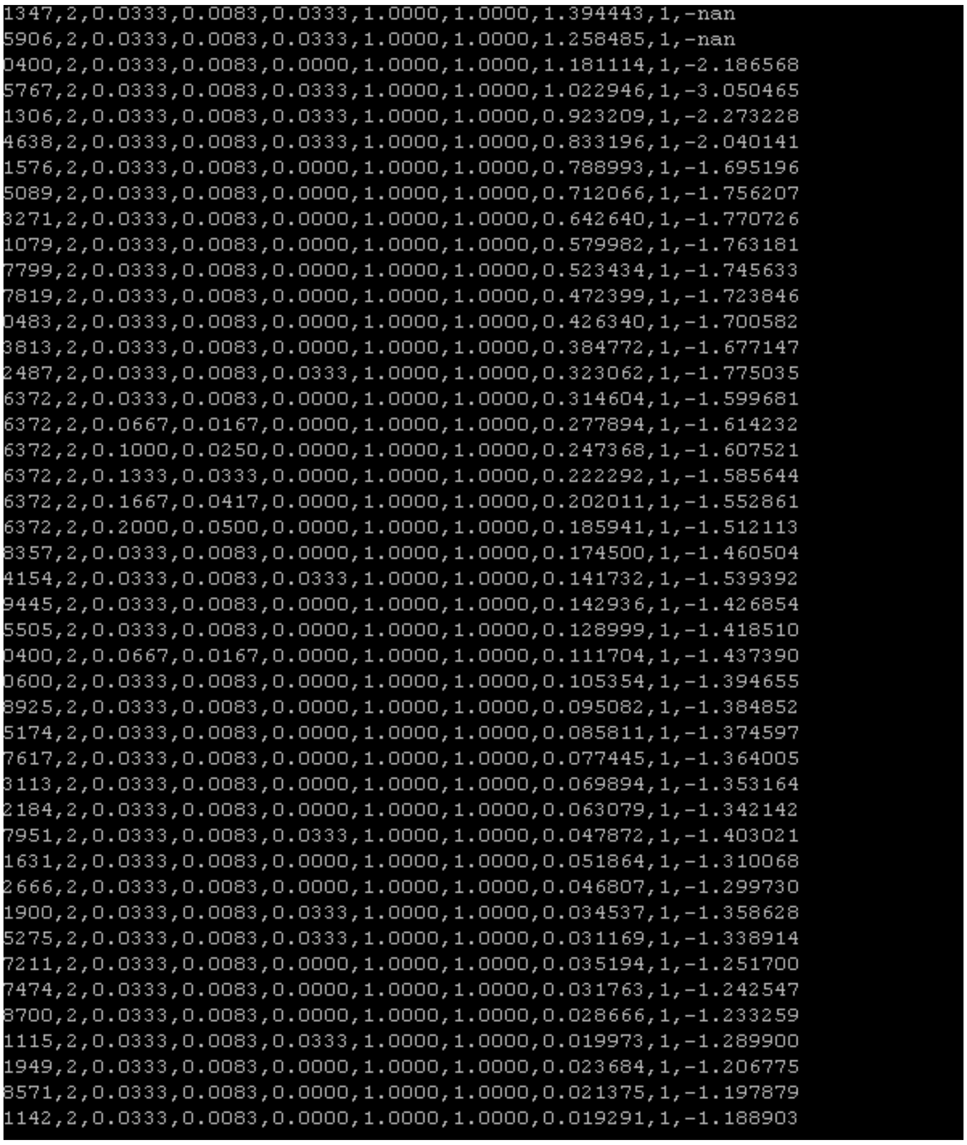


```
Starting program: /home/kang/block10/blockmon/main/node/blockmon usr/app_voipstream/voipstream.xml
[DepuraciÃ³n de hilo usando libthread_db enabled]
BlockFactory: registering block DNSDemux
BlockFactory: registering block DNSPacketNR
BlockFactory: registering block DNSPacketNX
BlockFactory: registering block ScoreCombiner
BlockFactory: registering block BooleanCombiner
BlockFactory: registering block BFMembershipChecker
BlockFactory: registering block ThresholdComparator
BlockFactory: registering block BlackHole
BlockFactory: registering block TBloomFilter
BlockFactory: registering block ACDCalculator
BlockFactory: registering block URLCalculator
BlockFactory: registering block CDRSource
BlockFactory: registering block FoFiRCalculator
BlockFactory: registering block PacketFilter
BlockFactory: registering block PacketPrinter
BlockFactory: registering block AppendTag
BlockFactory: registering block PacketCounter
BlockFactory: registering block RRDemux
BlockFactory: registering block IPFIXExporter
BlockFactory: registering block CDFGenerator
BlockFactory: registering block PeriodFlowMeter
BlockFactory: registering block IPDumbAnonymizer
BlockFactory: registering block Null
BlockFactory: registering block SketchMerger
BlockFactory: registering block L4Demux
BlockFactory: registering block SketchFlowCounter
BlockFactory: registering block FlowPrinter
BlockFactory: registering block FlowMeter
BlockFactory: registering block SynFloodDetector
BlockFactory: registering block TopNFlowSelector
BlockFactory: registering block SynthSource
BlockFactory: registering block PcapSource
BlockFactory: registering block SyncTag
BlockFactory: registering block SynFloodAlertManager
BlockFactory: registering block TCPFlagCounter
src     CDRSource must be Async, ignoring configuration2
[Nuevo Thread 0x7ffff1e46700 (LWP 4727)]
[Nuevo Thread 0x7ffff1645700 (LWP 4728)]
[Nuevo Thread 0x7ffff0e44700 (LWP 4729)]
src     skipping first line (it contains only headers)1
src     CDRSource: registering tags. This operation is done only once.1
src     CDRSource: tags registered1
src     CDRSource: Processed 4 calls in 0 seconds.1
```



```
src     CDRSource: Processed 8000000 calls in 165 seconds.1
src     CDRSource: Processed 9000000 calls in 185 seconds.1
src     CDRSource: Processed 10000000 calls in 205 seconds.1
src     CDRSource: Processed 11000000 calls in 225 seconds.1
src     CDRSource: Processed 12000000 calls in 245 seconds.1
src     CDRSource: Processed 13000000 calls in 265 seconds.1
src     CDRSource: Processed 14000000 calls in 285 seconds.1
src     CDRSource: Processed 15000000 calls in 305 seconds.1
src     CDRSource: Processed 16000000 calls in 325 seconds.1
src     CDRSource: Processed 17000000 calls in 345 seconds.1
src     CDRSource: Processed 18000000 calls in 365 seconds.1
src     CDRSource: Processed 19000000 calls in 385 seconds.1
src     CDRSource: Processed 20000000 calls in 405 seconds.1
src     CDRSource: Processed 21000000 calls in 425 seconds.1
src     CDRSource: Processed 22000000 calls in 445 seconds.1
src     CDRSource: Processed 23000000 calls in 465 seconds.1
src     CDRSource: Processed 24000000 calls in 486 seconds.1
src     CDRSource: Processed 25000000 calls in 506 seconds.1
src     CDRSource: Processed 26000000 calls in 526 seconds.1
src     CDRSource: Processed 27000000 calls in 546 seconds.1
src     CDRSource: Processed 28000000 calls in 566 seconds.1
src     CDRSource: Processed 29000000 calls in 586 seconds.1
src     CDRSource: Processed 30000000 calls in 606 seconds.1
src     CDRSource: Processed 31000000 calls in 626 seconds.1
src     CDRSource: Processed 32000000 calls in 647 seconds.1
src     CDRSource: Processed 33000000 calls in 667 seconds.1
src     CDRSource: Processed 34000000 calls in 687 seconds.1
src     CDRSource: Processed 35000000 calls in 707 seconds.1
src     CDRSource: Processed 36000000 calls in 727 seconds.1
src     CDRSource: Processed 37000000 calls in 747 seconds.1
src     CDRSource: Processed 38000000 calls in 767 seconds.1
src     CDRSource: Processed 39000000 calls in 787 seconds.1
src     CDRSource: Processed 40000000 calls in 808 seconds.1
src     CDRSource: Processed 41000000 calls in 828 seconds.1
src     CDRSource: Processed 41357990 calls in 835 seconds.1
[Thread 0x7fffeaffd700 (LWP 15187) terminado]
[Thread 0x7fffea7fc700 (LWP 15188) terminado]
Stopping timer thread
[Thread 0x7fffe9ffb700 (LWP 15189) terminado]
thr_sms ThresholdComparator: messagges received/sent [32659659/32659659], first received/last sent [1337766176-1337767011].1
thr_in  ThresholdComparator: messagges received/sent [32659659/32659659], first received/last sent [1337766176-1337767011].1
tbf_sms TBloomFilter: messagges received/sent [32659659/32659659], first received/last sent [1337766176-1337767011].1
tbf_out TBloomFilter: messagges received/sent [32659659/32659659], first received/last sent [1337766176-1337767011].1
tbf_new TBloomFilter: messagges received/sent [32659659/32659659], first received/last sent [1337766176-1337767011].1
tbf_in  TBloomFilter: messagges received/sent [32659659/32659659], first received/last sent [1337766176-1337767011].1
tbf_failed      TBloomFilter: messagges received/sent [32659659/32659659], first received/last sent [1337766176-1337767011].1
src     CDRSource: messagges received/sent [41357991/41357990], first received/last sent [1337766176-1337767011].1
memchecker      BFMembershipChecker: messagges received/sent [41357990/41357990], first received/last sent [1337766176-1337767011].1
log     LogWriter: messagges received [0] at 0.1
gng     f_gng: messagges received/sent [2251679/0], first received/last sent [1337766176-0].1
call_in BooleanCombiner: messagges received/sent [41357990/41357990], first received/last sent [1337766176-1337767011].1
boolC_sms       BooleanCombiner: messagges received/sent [41357990/41357990], first received/last sent [1337766176-1337767011].1
boolC_out       BooleanCombiner: messagges received/sent [41357990/41357990], first received/last sent [1337766176-1337767011].1
boolC_new       BooleanCombiner: messagges received/sent [41357990/41357990], first received/last sent [1337766176-1337767011].1
boolC_in        BooleanCombiner: messagges received/sent [41357990/41357990], first received/last sent [1337766176-1337767011].1
boolC_gng       BooleanCombiner: messagges received/sent [41357990/41357990], first received/last sent [1337766176-1337767011].1
boolC_fail      BooleanCombiner: messagges received/sent [41357990/41357990], first received/last sent [1337766176-1337767011].1

Program exited normally.
(gdb)
```

Figure 21 Execution results



```
1347,2,0.0333,0.0083,0.0333,1.0000,1.0000,1.394443,1,-nan
5906,2,0.0333,0.0083,0.0333,1.0000,1.0000,1.258485,1,-nan
0400,2,0.0333,0.0083,0.0000,1.0000,1.0000,1.181114,1,-2.186568
5767,2,0.0333,0.0083,0.0333,1.0000,1.0000,1.022946,1,-3.050465
1306,2,0.0333,0.0083,0.0333,1.0000,1.0000,0.923209,1,-2.273228
4638,2,0.0333,0.0083,0.0333,1.0000,1.0000,0.833196,1,-2.040141
1576,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.788993,1,-1.695196
5089,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.712066,1,-1.756207
3271,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.642640,1,-1.770726
1079,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.579982,1,-1.763181
7799,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.523434,1,-1.745633
7819,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.472399,1,-1.723846
0483,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.426340,1,-1.700582
3813,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.384772,1,-1.677147
2487,2,0.0333,0.0083,0.0333,1.0000,1.0000,0.323062,1,-1.775035
6372,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.314604,1,-1.599681
6372,2,0.0667,0.0167,0.0000,1.0000,1.0000,0.277894,1,-1.614232
6372,2,0.1000,0.0250,0.0000,1.0000,1.0000,0.247368,1,-1.607521
6372,2,0.1333,0.0333,0.0000,1.0000,1.0000,0.222292,1,-1.585644
6372,2,0.1667,0.0417,0.0000,1.0000,1.0000,0.202011,1,-1.552861
6372,2,0.2000,0.0500,0.0000,1.0000,1.0000,0.185941,1,-1.512113
8357,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.174500,1,-1.460504
4154,2,0.0333,0.0083,0.0333,1.0000,1.0000,0.141732,1,-1.539392
9445,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.142936,1,-1.426854
5505,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.128999,1,-1.418510
0400,2,0.0667,0.0167,0.0000,1.0000,1.0000,0.111704,1,-1.437390
0600,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.105354,1,-1.394655
8925,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.095082,1,-1.384852
5174,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.085811,1,-1.374597
7617,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.077445,1,-1.364005
3113,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.069894,1,-1.353164
2184,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.063079,1,-1.342142
7951,2,0.0333,0.0083,0.0333,1.0000,1.0000,0.047872,1,-1.403021
1631,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.051864,1,-1.310068
2666,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.046807,1,-1.299730
1900,2,0.0333,0.0083,0.0333,1.0000,1.0000,0.034537,1,-1.358628
5275,2,0.0333,0.0083,0.0333,1.0000,1.0000,0.031169,1,-1.338914
7211,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.035194,1,-1.251700
7474,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.031763,1,-1.242547
8700,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.028666,1,-1.233259
1115,2,0.0333,0.0083,0.0333,1.0000,1.0000,0.019973,1,-1.289900
1949,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.023684,1,-1.206775
8571,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.021375,1,-1.197879
1142,2,0.0333,0.0083,0.0000,1.0000,1.0000,0.019291,1,-1.188903
```

Figure 22 Neural network result stored in txt file

4.1 Prototype in a nutshell

The bypass fraud detection system monitors call history in real time to detect fraudulent clients using growing neural network. The application consists of three major parts which are parser, processor, and detector.

The parser receives detailed information of each call and extracts only necessary features which should be input into the processor. The selected features are caller ID, callee ID, duration of the call, and type of call for voice record, and caller ID for SMS record.

The processor stores and updates the call activity of each client tracking their calling history. A pair of bloomfilter will check the occurrence of a call to a destination number in the call history of a client. Duration and type of call are also checked to trigger appropriate bloomfilters for counting. Five features which are total number of call, call to new destination, short call (less than 10 second duration), incoming call, and SMS are calculated during the process using five individual counting bloomfilters. Although time between calls is also an important value, it is not explicitly included in features, since as time between calls becomes smaller, the value of counting bloom filter increases faster. The counting bloom filters promptly store and fetch the call history of huge amount of clients in real time. At the end of the process the number of SMS and incoming calls are changed into binary number to reduce noise since a real numerical value is not significant for the fraud detection.

The detector consists of a growing neural gas which learns a topology of data generating new cells or deleting existing cells as necessary with only few predetermined parameters. Growing neural gas classifies data into existing cells based on Euclidean distance and the distance is also used as a standard to decide on frauds. The fraud decision is made based on normalized distance value with the assumption that large quantity of accumulated values would follow a normal distribution. If the normalized distance values are greater than 10, such data is filtered out from the network and alerts are made to notify frauds.

4.2 Pre-requisites and dependencies

The VoIP bypass fraud detection system requires CDR data formatted in specific order (already introduced in section 3.1) and separated by a delimiter ("|" for this application). The parser can only recognize all the necessary information if the input data is organized in above mentioned manners. Following data must be presented by CDR data to enable the detection process: caller ID, callee ID, duration of the call, and type of call.

Five on-demand Time-decaying Bloom filters are from [25] and it is implemented on an external library.

Growing neural gas algorithm is built on the fast artificial neural network (FANN) library. A slight modification has been made to measure the distance of data from the cell and to normalize all distance values measured by accumulating all data points.

On an as-needed basis, parameters of bloomfilters, and growing neural gas can be reconfigured by the users to create different types of topology in neural network or to set different threshold for the fraud detection.

4.3 Architecture and data

Every block in this application functions in sequence. Thus the output of one block is transferred to the next block as a trigger, messages, and input values.

First of all, CDR parser receives call detail records and decides whether a record is for call or SMS. If it is a call records, CDR parser extracts caller ID, callee ID, duration of the call, and type of call information. From a SMS record, only caller ID is extracted. Such extracted data are converted into boolean variables and directed into the boolean combiner. New destination checker is a pair of bloomfilter which receives caller ID and callee ID from the CDR parser to check whether a client is calling to a new destination or not. Failed call checker checks the duration of a call to mark a call that lasts less than 10 seconds. Call type checker checks whether a record is an incoming or outgoing call.

The outputs from checkers are stored in boolean variables and they directed into the boolean combiner. Boolean combiner combines Boolean variables that are transferred from previous blocks to trigger appropriate bloomfilters.

Bloomfilter enables prompt fetching and storing of call history of each client. It receives caller ID and Boolean variables from boolean combiner. Boolean variables trigger appropriate bloomfilters to record activities of each client as numerical values.

Binary converter receives SMS and incoming call values from bloomfilters to change quantitative values into binary values. If there is at least one occurrence of SMS or incoming calls, the variable is set to 1 and 0 otherwise.

Growing neural gas receives input features from individual bloomfilters and train the network with inputs. As a new call occurs, a vector containing information of a call is generated as an input to the growing neural gas. Growing neural gas classifies input vectors into a cell, which is nearest to the vector based on Euclidean distance. Next, it updates the location of cells to train network with input data. If the distance is farther than a certain threshold, it makes alert to notify the occurrence of fraud and such anomalous data do not make affect on the update of cell position.

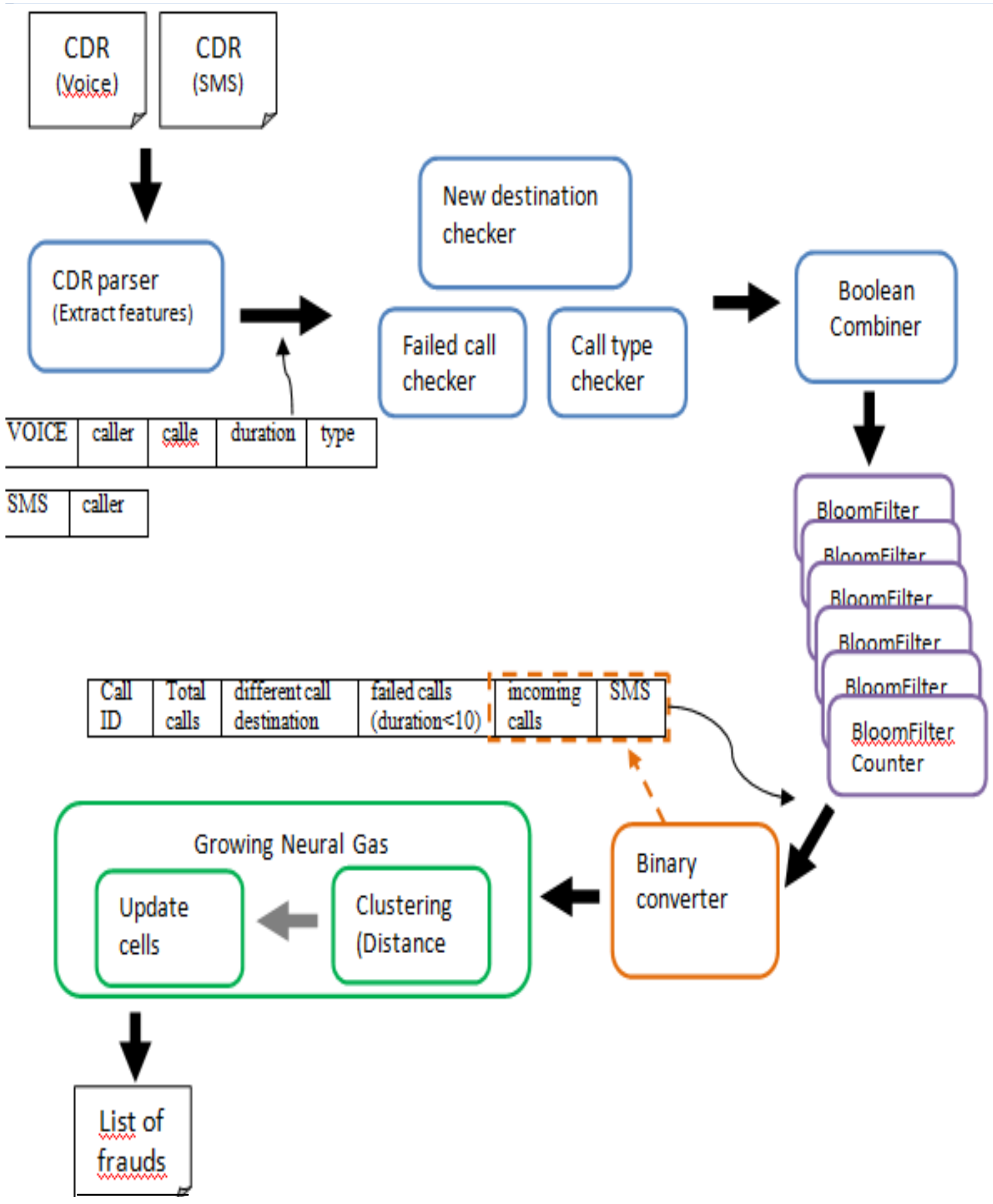


Figure 23 A block diagram of the application.

The whole process of the real time application is presented in Figure 20.

4.4 Used Technology

There are some notable technologies used in this application. Since the blockmon application works by blocks, each block takes part in different responsibilities in the application. Some blocks are based on several different technologies and those technologies will be introduced in this section. Firstly the core of the application is the detection algorithm, growing neural gas. Secondly Bloomfilters store and retrieve the call history data efficiently making the real time detection possible and avoiding privacy infringement issues. Lastly, the block design of all applications maximizes the reusability of code and collaboration among different projects.

4.4.1 Growing neural gas (GNG)

Created by Bernd Fritzke, growing neural gas (GNG) is an algorithm for incremental clustering based on unsupervised learning. Reflecting the position of input vectors, it incrementally creates a graph (network) which consists of nodes. The growing neural gas is more flexible version of the self organizing map (SOM), one of the major neural network algorithms. The basis of this technique is vector quantization. Vector quantization groups the vectors into each group represented by centroid. The position of centroid is constantly adjusted reflecting new input data with a certain scaling factor. GNG can adapt their centroids for the input distribution over time. GNG algorithm starts with two nodes and new input data is connected to the nearest node based on Euclidian distance. The nearest node is called winner node and its position should be adjusted to be closer to the input data. Thus the change in the input data is constantly reflected on the network of GNG.

If the current iteration is an integer multiple of a lamda, and the current node number has not been reached the maximum network size, a new node is inserted. To decide the position of a node inserted, a node with the largest error u and its neighbor with the largest error v are searched. The position of a new node is W_r, and the formula is as below. W_u and W_v represent the position of a node with the largest error and its neighbor with the largest error respectively. The edge between a new node and v is created.

$$\overline{w}_r \leftarrow \frac{(\overline{w}_u + \overline{w}_v)}{2}$$

The errors of nodes are also updated following the formulas below.

$$error_u \leftarrow \alpha \times error_u$$
$$error_v \leftarrow \alpha \times error_v$$
$$error_r \leftarrow error_u$$

Later the errors of all nodes are also decreased by the factor of β.

$$error_j \leftarrow error_j - \beta \times error_j$$

The parameters which include maximum network size, neighbor scale factors, and so on remain the same in time. Insertion of nodes ceases as standards that are set by users met or the network size reached its maximum limitation.

The insertion policy has some limitation in a sense that it disregards the necessity of a new node. The insertion is periodically made as a certain number of iteration is achieved. The ideal scheme for the insertion will be inserting node when the local or global mean errors reached a certain values.

When a new node is inserted according to the value of lamda, the performance of the algorithm varies greatly. If lamda is too low, the initial distribution of nodes will be in low quality as well. As a new node is frequently inserted, the local error will not be calculated properly and a node does not have time to distribute itself over the input space. Thus proper value for the lamda needs to be set with experiments. In this application, the value is configured based on the recommendation of FANN library [27].

Until now, basic algorithm for growing neural gas is introduced. There has been some modification in the algorithm of GNG, since the conventional library does not support the anomalous data detection function. Thus two important features are added to the program. In particular, the code that realizes the core update and detection processes are represented below.

```
FANN_EXTERNAL double* FANN_API fann_gng_neuron_rung(struct fann
* ann, struct fann_neuron *neuron, unsigned int epoch_cnt,
unsigned int max_epochs, FILE *conf, std::string s){

//initial parameter setting
      double runsum;
      double sec_min_dist = DBL_MAX;
      unsigned int i, j;
      struct fann_gng_neuron_private_data *priv;
      ann->gng_params->min_dist = FLT_MAX;
      priv=(struct fann_gng_neuron_private_data *) neuron-
>private_data;
      ann->gng_params->gng_closest_index = -1;
      ann->gng_params->gng_second_closest_index = -1;

//start to search the cells having the shortest and second
//shortest distance from the input data
      for ( i = 0; i < ann->gng_params->gng_num_cells; i++ )
      { runsum = 0;
        for (j = 0; j < ann->num_input; j++) {
          runsum += (priv->gng_cell_location[i][j] - neuron-
>inputs[j]) * (priv->gng_cell_location[i][j] - neuron->inputs
[j]);
```

```
          }
        if (runsum < ann->gng_params->min_dist){
          sec_min_dist = ann->gng_params->min_dist;
          ann->gng_params->min_dist = runsum;
          ann->gng_params->gng_second_closest_index = ann-
>gng_params->gng_closest_index;
          ann->gng_params->gng_closest_index = i;
        }
        else if (runsum > ann->gng_params->min_dist && runsum <
sec_min_dist)
        {
          sec_min_dist = runsum;
          ann->gng_params->gng_second_closest_index = i;
        }
      }
//it assigns distances and calculates the mean, standard
//deviation and Z value of all input data
      double temp=( (double)ann->gng_params->min_dist );
      double mean = ann->gng_params->dist/ann->gng_params->n;
      double sd = ann->gng_params->dist2/ann->gng_params->n -
      (mean * mean);double Z = (sqrt(temp)-mean)/sqrt(sd);

//write all values in the file so that one can track the
//detection process
     fprintf(conf, "%s,%d,%.4f,%.4f,%.4f,%.4f,%.4f,%f,%d,%f\n",
s.c_str(),(int)(priv->gng_num_cells),neuron->inputs[0],neuron-
>inputs[1],neuron->inputs[2],neuron->inputs[3],neuron-
>inputs[4],
        (double)ann->gng_params->min_dist,(int)ann->gng_params-
>gng_closest_index,Z);
        fflush(conf);


//If the Z value is greater than 150, then the call ID is listed
//in the suspicious client list

        if((ann->gng_params->gng_current_iteration > 10000)  &&
Z>100){
           bool exist=false;
             for(int k =0; k<100; k++){
                 if(strcmp( ann->gng_params->list[k], s.c_str())
==0){
                 exist=true;
                     memset(ann->gng_params->id,0,128);
                     break;
                 }
             }bool space=true;
          if(!exist){
           int t;
             for(t =0; t<100; t++){
                 if(strcmp( ann->gng_params->list[t], "0") ==0){
                 strcpy(ann->gng_params->list[t], s.c_str());
                 strcpy(ann->gng_params->id, s.c_str());
                 strcat(ann->gng_params->id, "\n");
                 space=false;
                     break;
```

```
                        }
                    }
                }
              else
                  memset(ann->gng_params->id,0,128);
          }else
                  memset(ann->gng_params->id,0,128);


          return &Z;
      }

//after finding the nearest neighbor (winner node) the position
//of a winner node and its neighbors are updated closer to the
//input data.


FANN_EXTERNAL void FANN_API fann_gng_neuron_update(struct fann
*ann, struct fann_neuron *neuron, FILE * file)
{
        //initial parameter setting
        unsigned int i, j, num_node_neighbors;
        unsigned int numcells = ann->gng_params->gng_num_cells;
        struct fann_gng_neuron_private_data *priv;
        int* node_neighbors;
        priv=(struct fann_gng_neuron_private_data *) neuron-
>private_data;
        ann->gng_params->gng_current_iteration++;

         double temp=( (double)ann->gng_params->min_dist );
        double mean = ann->gng_params->dist/ann->gng_params->n;
        double sd = ann->gng_params->dist2/ann->gng_params->n -
(mean * mean);
         double Z = (sqrt(temp)-mean)/sqrt(sd);

//if a certain condition is met the update is dismissed
// in this case when the number of iterations are over 1000, and
//Z value is more than 2. With the assumption that adjusting the
//networks for data that has Z values greater than 2 may
//train networks in a wrong direction.

         if((ann->gng_params->gng_current_iteration > 1000) && Z>2)
           return;


        ann->gng_params->dist2 += temp;
        ann->gng_params->dist +=sqrt(temp);
        ann->gng_params->n +=1;


        /*Move the winner node. */
        for (j = 0; j <ann->num_input; j++)
        {
             priv->gng_cell_location[ann->gng_params-
>gng_closest_index][j] = priv->gng_cell_location[ann-
>gng_params->gng_closest_index][j] +
          ann->gng_params->gng_winner_node_scaling_factor*
(neuron->inputs[j] - priv->gng_cell_location[ann->gng_params-
>gng_closest_index][j]);
        }

      /*Get the neighbors of the winner node and move them.*/
      num_node_neighbors = fann_get_number_neighbors_gng(ann,
ann->gng_params->gng_closest_index );
      if( (node_neighbors = (int*)
calloc(num_node_neighbors,sizeof(int)) ) == NULL)
      {
          return;
      }

      fann_get_neighbors_gng(ann, ann->gng_params-
>gng_closest_index, node_neighbors );

      for(i=0;i<num_node_neighbors;i++)
      {
        for (j = 0; j < ann->num_input; j++)
        {
          priv->gng_cell_location[node_neighbors[i]][j] = priv-
>gng_cell_location[node_neighbors[i]][j] +
            ann->gng_params->gng_neighbor_node_scaling_factor *
(neuron->inputs[j] - priv->gng_cell_location
[node_neighbors[i]][j]);
        }
      }

      /*
        Increment the edge age of all neighbors.
      */
      fann_increment_edges_of_neighbors( ann, ann->gng_params-
>gng_closest_index);
      fann_safe_free( node_neighbors );

      /* If closest and second closest are connected by an edge
set it to zero,
         If not,connect them with an edge. */
      ann->gng_params->gng_cell_edges[ann->gng_params-
>gng_closest_index][ann->gng_params-
>gng_second_closest_index]=0;
      ann->gng_params->gng_cell_edges[ann->gng_params-
>gng_second_closest_index][ann->gng_params-
>gng_closest_index]=0;

      /* Remove cell edges greater than age max */
      fann_remove_edges( ann );

      /* Remove cells with no edges */
      fann_remove_cell_with_no_edges( ann, neuron );

      /*       If current iteration is multiple if lambda and max
node count has not been meet, then insert a new node.
      */
      if( ((int)fmod((float)ann->gng_params-
>gng_current_iteration,(float)ann->gng_params-
>gng_iteration_of_node_insert) ==  0) && ( numcells < ann-
>gng_params->gng_max_nodes ) )
          fann_gng_insert_node( ann, neuron );


      /* Decrease all error-variables of all nodes */
      for (i = 0; i < ann->gng_params->gng_num_cells; i++)
      {
        priv->gng_cell_error[i] = priv->gng_cell_error[i] -
(ann->gng_params->gng_global_error_reduction_factor * priv-
>gng_cell_error[i]);
      }

      return;
}
```

<Source: This code is the modified version of the fann_gng.cpp file from fast artificial neural network library, a free open source neural network library; http://leenissen.dk/fann/wp/ >

First of all, in the modified code the distance of a data point from the nearest cell is calculated and guarded in two versions: the real distance, and square of the distance. There is also a variable that counts the number of input data. Thus with three values, we can calculate the mean, standard deviation and Z value of the data.

Secondly, in the update neuron method, the position of the new input data is always reflected into the position of the nearest cell and its neighbors. However the modification is made so that if a Z value of a distance is greater than 2, then the update of the position of nodes is not reflected. It means that if a certain data is anomalous, the algorithm does not consider it as a part of training data to set the right cell position for neurons in networks. This way more correct position of neurons can be set. The experiments are done to decide on right parameters and thresholds for the detection.

4.4.2 BloomFilter

A Bloomfilter is a simple data structure which is designed to be space-efficient based on probabilistic randomized features. It is mainly used for the membership queries which test whether a certain data is included in a group of data. Bloomfilters may commit false positive detection though (depending on the memory that is allocated to guard the data, the false positive rate becomes different), it efficiently guard and search the data to check their membership. There is no false negative error.

A Bloomfilter can represent a set of T={X_1, X_2, …, X_n} using an array of m bits which is set as 0 in the initial stage. It consists of k binary hash functions h_1, h_2, …, h_k which produce output in the range of 1, 2, …, m. There are two fundamental functions within bloomfilters. First is insert and the second is check. When you insert data, it is input to all k binary hash functions, and hash functions change the corresponding array bits from 0 to 1. When data is checked, data is input into all k binary hash functions again and this time we do not change the array bits but retrieve the values stored within bits. If all the retrieved values are 1 then input data is considered as being part of the group. If at least one of the retrieved values is 0 then input data is not a member of the group. The accuracy of the membership check performance is greatly dependent on the size of the array since as there are more bits, there is less possibility of the false positive detection.

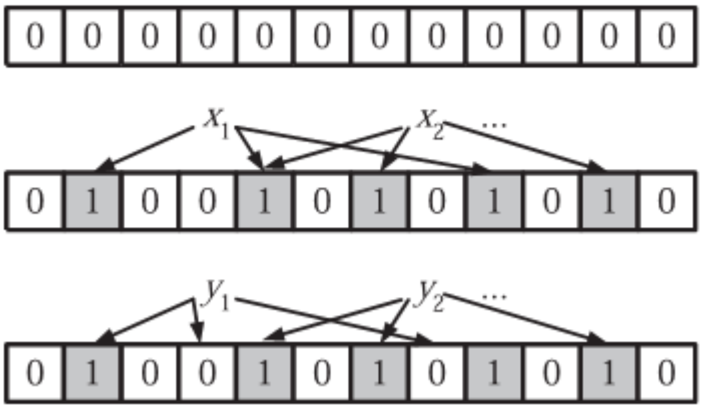


Figure 24 an example of a bloom filter. The filter starts an array filled with 0. As the input passed through the k hash functions, the corresponding output change 0 into 1 in the array of m bits. The third box shows how it functions to check the membership of the elements. In this example y1 is not part of members since not all values in array correspond to 1, while y2 is part of the group since all outputs are 1.

A counting bloomfilter is a generalization of normal bloomfilters. A bloomfilter has a limitation of usage when members of a set change over time. When one inserts data, the process is simple since it just changes the corresponding bits into 1 following the output of hash functions. However when one wants to delete the a certain data it is impossible to be done, since if the corresponding bits change into 0 from 1, it might affect the result of other input data that one did not intent to delete from the group. Thus to resolve this problem, a counting bloomfilter is devised. While bloomfilters utilize an array of bits, counting bloomilters have a bin of array to store larger amount of data. Each bin contains t bits and thus up to 2^t - 1 size of data can be guarded in each bin. In these bins, the counted values of bit's appearance are stored. If one wants to delete a certain element, it can be easily done by decreasing the counted value by 1. In counting bloomfilters, the proper counter size should be decided to avoid the overflow of counters.

When counting the number of occurrence of data, there are many cases when the time of occurrence of data is of great importance. As always recent occurrence is more important than later occurrences. Thus to weight those difference in importance, a time decayed bloom filter (TBF) is devised. A time decayed bloom filter decays the values of counts as time passed. Thus recent data has greater values than the old one. However there are some shortcomings in time decayed bloomfilters. Firstly there is a bias occurred when the result if retrieved in the random time. There might be limitation in performance since there is periodic decrement of values of bins at once. These problems cause researchers to create a new application called a countinous time decayed bloomfilter to store the membership information with a more sufficient manner [25].

4.4.3 Blockmon

The continuous growth of the internet posed us challenges especially in the security sector. The monitoring system of traffic and data is required to function more efficiently to handle enormous amount of traffic data that is generated each day. This landscape has imposed heavy burden to the monitoring system. A monitoring system needs to perform better and be extensible for unknown development in the future. Thus the concept of blockmon is introduced to provide more high performing, flexible and extensible services. Blockmon is inspired from three areas of work: modular networking, programmable measurement, and high-speed packet capture. It takes great benefits of recent multi-core hardware by splitting the load of each block into different cores in a computer as a user designs. There are some notable contributions that blockmon platform has. Firstly it is a new, flexible design for network measurement in a composable way. It analyzes data based on message-passing among blocks, associated with runtime via gates, and supporting re-composition at runtime. Secondly it is possible to apply this composition for the parallel measurement processing and full use of up to date multi-core hardware, which ensures high performance for line-rate measurement. Thirdly, wait-free rotating queue and new C++11 features minimize lock contention and allocation overhead. It consequently leads to the performance increase. Lastly, the integration of standards-based import and export allows us to extend an application across multiple nodes and easy import of data from other, non-Blockmon sources.

```
<composition id="voipstream" app_id="app_voipstream">
  <general>
      <clock type="wall" />
  </general>
  <install>
    <threadpool id="src_thread" num_threads="1" >
      <core number="0"/>
    </threadpool>
    <threadpool id="tbf_thread" num_threads="1" >
      <core number="2"/>
    </threadpool>
    <block id="src" type="CDRSource" sched_type="active"
threadpool="src_thread">
      <params>
        <source type="offline" name="cdrfile.log"/>
      </params>
    </block>
    <block id="memchecker" type="BFMembershipChecker">
      <params>
        <source_key name="src_dst"/>
        <result_key name="new_callee"/>
        <hashsize value="8"/>
        <filtersize value="20"/>
      </params>
    </block>
    <block id="tbf_ct24" type="TBloomFilter" sched_type="active"
threadpool="tbf_thread">
      <params>
        <id_key name="src_id"/>
        <count_key name="src_id"/>
        <result_key name="ct24"/>
        <add_key name="established"/>
        <timewindow value="86400"/>
        <basewindow value="360"/>
        <hashsize value="8"/>
        <filtersize value="20"/>
        <addvalue type="dynamic" value="duration"/>
      </params>
    </block>
    <block id="fofir" type="FoFiRCalculator" sched_type="active"
threadpool="metric_thread">
      <params>
        <result_key name="fofir"/>
        <basewindow value="360"/>
        <activation value="30"/>
        <threshold_zero lowerbound="5" upperbound="10"/>
        <threshold lowerbound="2" upperbound="10"/>
      </params>
    </block>
    <connection src_block="src" src_gate="source_out" dst_block="memchecker"
dst_gate="in_bfmcheck"/>
    <connection src_block="memchecker" src_gate="out_bfmcheck"
dst_block="boolcombiner" dst_gate="in_boolcombiner"/>
    <connection src_block="boolcombiner" src_gate="out_boolcombiner"
dst_block="tbf_rcr" dst_gate="in_tbf"/>
    <connection src_block="tbf_ct24" src_gate="out_tbf" dst_block="fofir"
dst_gate="in_fofir"/>
  </install>
</composition>
```

Figure 25 An exmaple of XML file for the composition

It will easily allow us to extend its functions across different applications and systems. Observing at the high-level, Blockmon consists of a group of entity called blocks which have individual processing action for example parsing CDR, classifying data into groups, and learning data to build artificial neural networks. These blocks communicate to each other by sending messages via gates. These messages are tagged by the tagging system which helps each block to distinguish specific message that a block wants to retrieve. [26]

A set of interconnected blocks which are gathered to complete the whole process that is designed to be done with blocks is called composition. Compositions are defined using XML files which express the list of blocks that will be used in the compositions and corresponding parameters of each block. The blockmon core and each block are implemented in code using C++, and the system is controlled by the python-based command line interface (CLI).

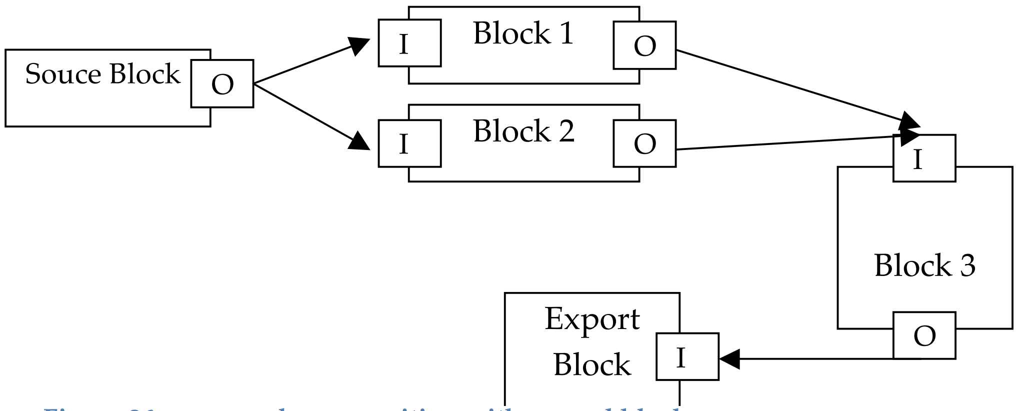


Figure 26 an example composition with several blocks

A block processes data individually based on different functional code. It can take part in diverse functionality such as packet filtering, monitoring, fraud detection, data importer and data writer.

All blocks are based on a common superclass. When new blocks are created, they just inherit from the superclass and at least two methods should be implemented: configure (receives XML messages corresponding to the blocks to get the handles for the parameters) and a receive_message method (receives messages from the previous blocks, so the data can flow through blocks as a composition represents). Instead of passively receiving the message that arrive, block can also be functions regularly with time by a thehandle_timer method. A source block is the start of the composition. It captures the input data and process only necessary information for the process. The source block always sends messages but never receives the message as it works as an input point. A connection point between blocks is called gate. Following the connection that is defined in the composition file, blocks are connected at configuration time via gates. Messages are sent by output gate and received by input gate. For the flexibility of the blockmon, there are two ways of message sending: active and passive message passing. In the direct message communication, a sent message directly recalls the receive_msg method of a receiving block. This process is fast but inflexible since the receiving blocks will run within the thread of directly invoked blocks. Thus the chain of direct invocation should be avoided. On the other hand in the indirect message passing, each block is individually scheduled in different thread pools and different CPU cores as well. This allows true parallel processing on the multi-core environment without experiencing any blocking or locking.

4.5 Stength and weakness

The strength of the application lies in several blocks such as bloomfilters, GNG, and in the general structure of the application, a block design. The great challenge faced in the delayed time system is that it can greatly infringe the privacy of customers as the application stores and analyzes the information of every customer. In addition calculating the frequency of data consumes lots of time and memory during the analysis process. On-demand time decaying bloomfilters are used to store data during the detection process and to resolve the private data storage issue since it does not guard original information.

Secondly, when the data are counted in the bloomfilters, it also takes into account the time between the occurrences. Thus the concept of frequency is already reflected on the result of the data. The complicated frequency calculation phase can be avoided with on-demand time decaying bloomfilters.

Finally, the data from the bloomfilters are more or less normalized and data preprocessing is not necessary. The raw call history data are usually very spreaded. Thus for the better detection result data preprocessing such as normalization is required. However data from bloomfilters are already reduced appropriately and thus the variation in data is not large

5 Conclusion

In this section analysis and comparison of the results of application will be presented. Although it is hard to judge the performance of each system, as there is no known list of fraudulent clients, there are some behaviors that can be a symptom of the frauds. To change the semantic definition into numerical values, the simple penalty counting algorithms which accumulate penalty points as clients show suspicious behaviors is devised. The greater penalty points mean the greater the possibility to be detected as frauds. The detection results of the two different versions of the fraud detection systems are compared to the penalty scores. Thus appropriate threshold can be set and the method that presents better performance is decided.

category	points	Condition
Total call	20	The number of total call > 90
	18	40 < The number of total call <= 90
	16	25 < The number of total call <= 40
	14	15 < The number of total call <= 25
Time between call	8	0 <= time between call < 60
	6	60 <=time between call < 120
	5	120<=time between call < 240
Number of calls to different destination	20	calls to different destination (portion) = 1
	18	0.9 <= calls to different destination < 1
	14	0.7 <= calls to different destination < 0.9
	8	0.5 <= calls to different destination < 0.7
Portion of failed calls	8	portion of failed call = 1
	6	0.9 <= portion of failed call < 1

Table 2 Penalty points

The analysis result of the delayed time version of the system calculated average, maximum and minimum values of all attributes for each clusters. It is assumed that a cluster retains the elements that have similar types of calling patterns. Among all clusters, clusters with the large number of total calls, the large number of different calls, and small number of members are decided as a fraudulent group. Usually clusters with higher means have smaller number of members as fraudulent clients are comparatively smaller than the normal clients.

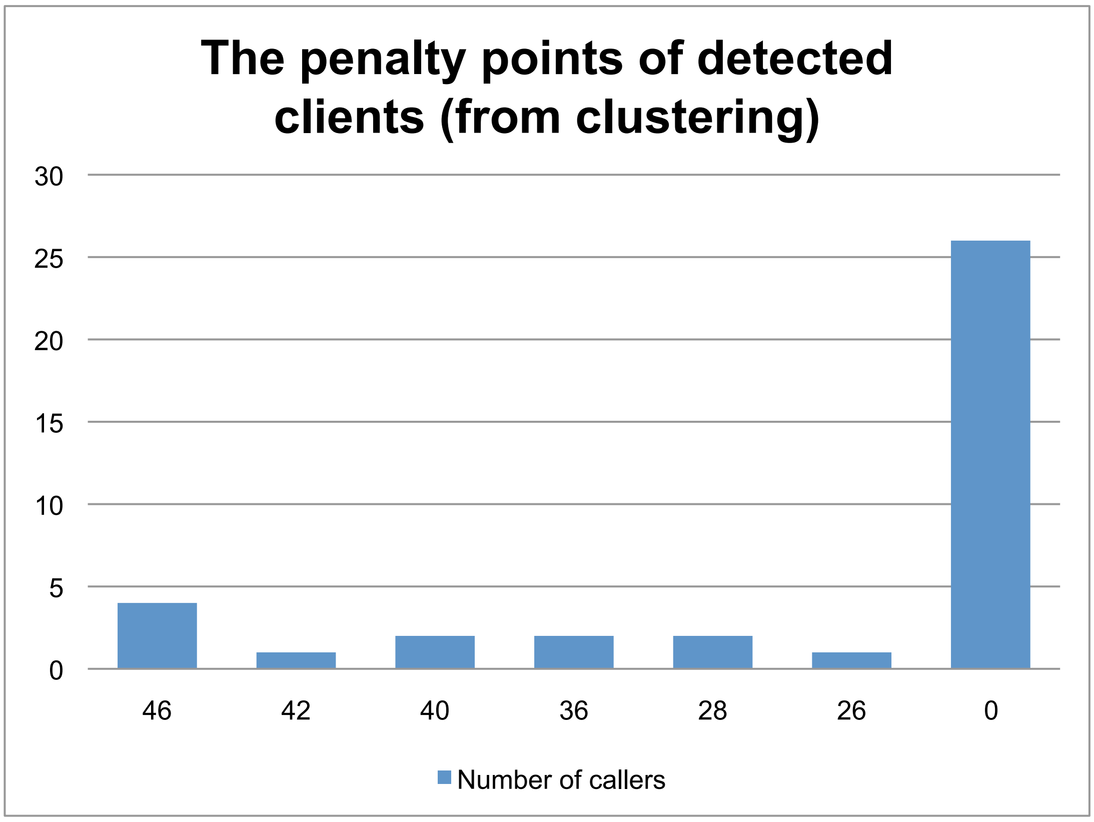
As the fraudulent groups are decided, the lists of numbers detected as suspicious in each system are compared. The comparison of the penalty points is also made according to the phone number of each client.

The numerical data for delayed time result and the real time result are different because the real time result is based on the data that is decayed in time within bloomfilters.

5.1 Detection result

5.1.1 Penalty algorithm result

For the privacy issues, phone numbers of all clients are deleted. All values are based on the real data of one day call detail record. There are phone number, Total outgoing calls, different number of destination, call less than 10 seconds, time between calls, and finally such information gives us a point according to the standards that are defined before.



5.1.2 Clustering result

The clustering is done with the Cobweb clustering algorithm in Weka software. The input attributes are phone number, the total number of outgoing calls in number, the portion of outgoing calls to different destinations, total number of failed outgoing calls (calls that last less than 10 seconds), the time between calls, and the existence of incoming calls and SMS in binary data.

The Cobweb is chosen, because of its unique clustering methods. While most clustering algorithms depend on distance measure to decide on similarities between data, the cobweb algorithm tries to maximize the utility function which increases the system's inference ability. Thus it tends to classify data which has similar types of values in each attribute and clients with similar calling patterns are grouped into the same cluster.

However there are two challenges to realize successful cobweb clustering. Firstly, clustering tools are very unpredictable and thus it is possible to face the output that was not wanted. Although the result is different from expected data, we do not know which parts should be modified to obtain the results that are wanted. Secondly, in each attribute such as the number of total call, the difference of minimum and maximum can reach more than 10,000 numerical differences, because of few anomalously large values. Such large difference often decreases the quality of the clustering results and if the numerical range of attributes are different, their importance in the classification process is different. Thus to prevent such confusion, normalization of input data is necessary. Indeed after normalization and filtering few anomalously large data, the classification result has a better quality.

The result of detection was not that reliable since the clusters could not be well-grouped reflecting the calling patterns of clients. Parameters are set as acuity is 1, cutoff is 0.015, and seed is 42. The first table is the analysis results of the penalty points of the suspicious groups. The average is very low and it shows that the detection system is not that effective. The second table is the analysis of clustering results. For each cluster number of instance, average, minimum, and maximum values are calculated.

Number of clients per penalty points	46	4
	42	1
	40	2
	36	2
	28	2
	26	1
	0	26
Number total		38
Average of penalty		12.1052632

Table 3 Clustering result analysis based on penalty points

		cluster3	cluster5	cluster7	cluster9	cluster10	cluster12	cluster14	cluster15	cluster17	cluster18	cluster20	cluster21
	Number	388	150	51	190	35	53	100	224	124	126	45	45
total number of call	max	33	41	48	1526	58	52	45	37	32	36	39	43
	min	30	38	46	54	55	49	42	34	30	33	37	40
	average	31.36856	39.30667	46.88235	123.2421	56.57143	50.32075	43.14	35.38839	30.90323	34.35714	37.84444	41.28889
different number of call	max	1	1	1	1	1	1	1	1	1	1	1	1
	min	0.003401	0.003367	0.00369	0.003344	0.002801	0.002114	0.003378	0.002801	0.003937	0.003436	0.005495	0.006993
	average	0.311669	0.353214	0.341125	0.329271	0.337375	0.422384	0.353809	0.297585	0.620785	0.570101	0.64868	0.64536
portion of failed call	max	1	1	1	1	0.767442	0.984375	0.98	1	0.98	1	0.970588	0.935484
	min	0	0	0	0	0	0	0	0	0	0	0	0
	average	0.293928	0.319957	0.282192	0.290767	0.261844	0.264077	0.311751	0.285464	0.198005	0.197846	0.233562	0.270475

		cluster22	cluster24	cluster26	cluster27	cluster28	cluster29	cluster32	cluster34	cluster35	cluster37	cluster40	cluster41
	Number	32	22	10	12	1	11	5	37	6	16	805	683
total number of call	max	47	50	57	69	16674	62	63	6560	77	53	32	36
	min	44	48	54	66	16674	58	63	65	75	51	30	33
	average	45.3125	48.95455	55.3	67.41667	16674	60	63	300.7297	76.16667	51.75	30.89068	34.30893
different number of call	max	1	1	1	0.981818	0.787281	0.97619	0.962264	1	0.970588	1	1	1
	min	0.002786	0.005682	0.033333	0.01	0.787281	0.004237	0.44	0.002985	0.498956	0.005291	0.009709	0.009569
	average	0.655174	0.625195	0.675898	0.734157	0.787281	0.40913	0.723834	0.62549	0.790553	0.693048	0.370797	0.341857
portion of failed call	max	0.771429	0.483871	0.533333	0.771429	0.596491	0.551724	0.275862	0.942857	0.558824	0.648438	1	0.971831
	min	0	0	0	0.007042	0.596491	0	0.142857	0	0.029412	0	0	0
	average	0.190403	0.179562	0.219226	0.153877	0.596491	0.191496	0.209429	0.238112	0.223291	0.233893	0.273289	0.26944

		cluster43	cluster45	cluster46	cluster48	cluster49	cluster50	cluster51	cluster53	cluster54	cluster56	cluster57	cluster58

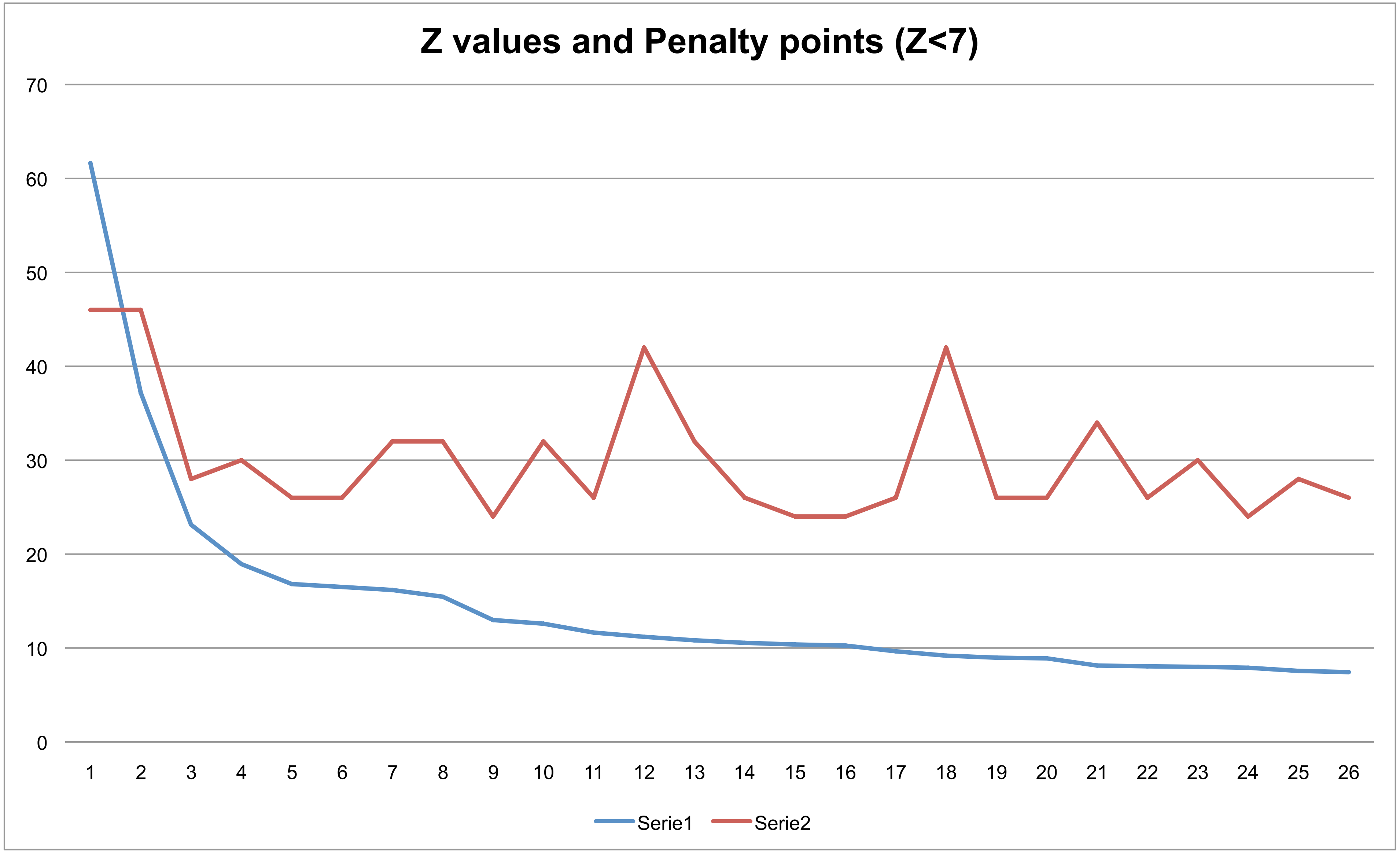
	Number	497	233	151	115	514	97	255	118	76	21	34	28
total number of call	max	40	47	52	56	1214	60	43	33	37	44	41	49
	min	37	44	49	53	48	57	41	30	34	42	38	46
	average	38.334	45.37768	50.33775	54.41739	125.358	58.59794	41.89412	31.33898	35.22368	42.61905	39.23529	47.82143
different number of call	max	1	1	0.995146	1	1	0.997992	0.966667	0.981132	1	1	0.976526	0.842105
	min	0.003484	0.012766	0.013514	0.015625	0.013699	0.010417	0.017241	0.013333	0.023256	0.033333	0.016393	0.026316
	average	0.354517	0.356103	0.340763	0.365732	0.370941	0.340451	0.338373	0.544445	0.478687	0.479064	0.52036	0.472514
portion of failed call	max	1	0.941176	0.969697	1	0.970588	0.969697	0.909091	0.866667	0.953488	0.766667	0.837838	0.763158
	min	0	0	0	0	0	0	0	0	0	0	0	0.022727
	average	0.2785	0.240398	0.240828	0.239837	0.272365	0.256465	0.256575	0.241843	0.240874	0.307962	0.244484	0.279316

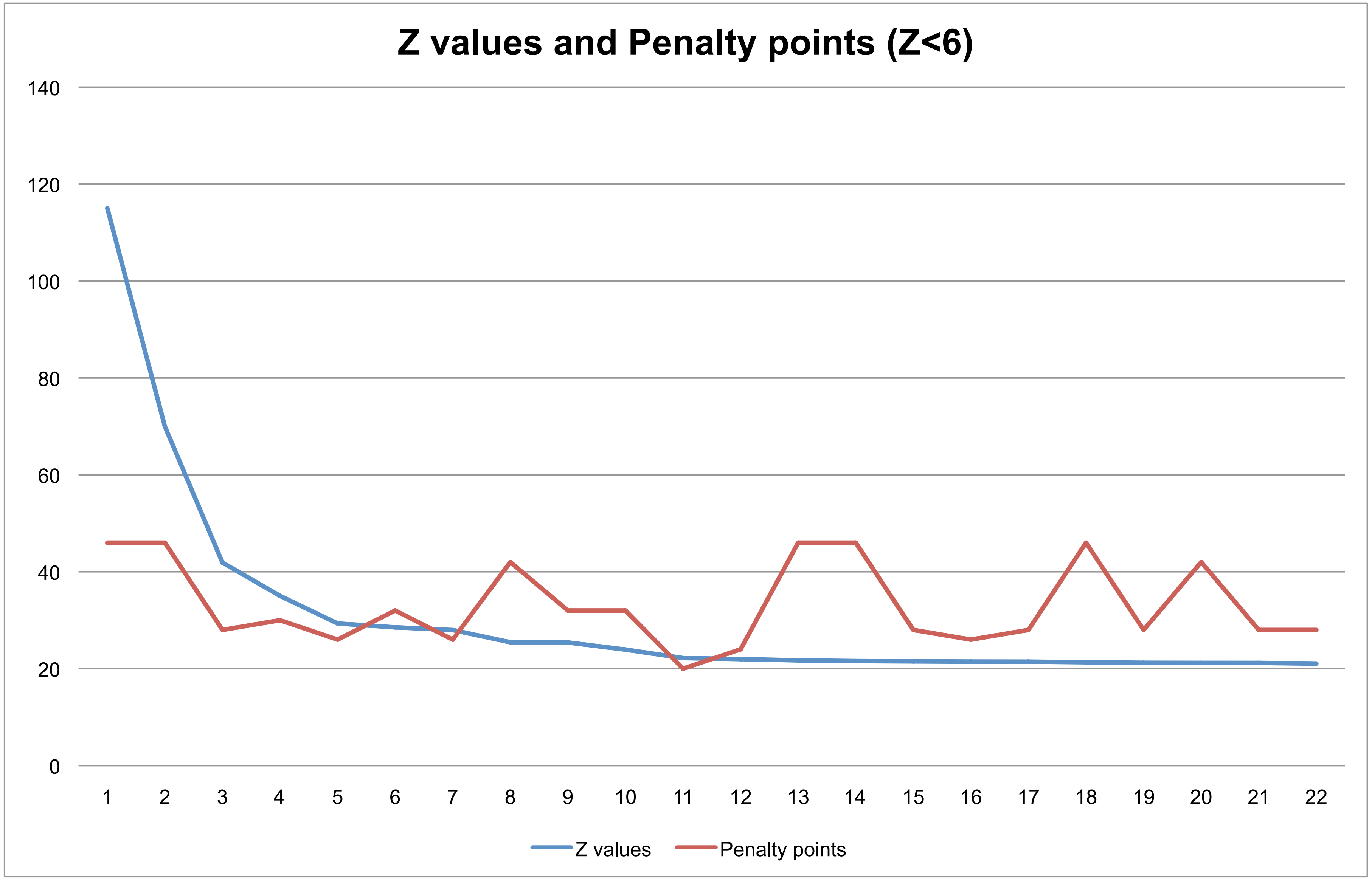
		cluster63	cluster64	cluster65	cluster66	cluster68	cluster69	cluster71
	Number	12	2	9	11	3	8	38
total number of call	max	54	55	59	62	64	67	693
	min	51	55	56	60	64	65	45
	average	52.5	55	57.44444	61.36364	64	66.125	162.0526
different number of call	max	0.931818	0.583333	0.983607	0.969697	0.939394	0.833333	0.941606
	min	0.027027	0.166667	0.128205	0.333333	0.1	0.030303	0.00641
	average	0.458719	0.375	0.482341	0.636454	0.550766	0.485455	0.597792
portion of failed call	max	0.909091	0.361111	0.645161	0.566667	0.6	0.9	0.422222
	min	0.014706	0.333333	0	0	0.030303	0.066667	0
	average	0.375133	0.347222	0.258241	0.23953	0.274617	0.313113	0.220418

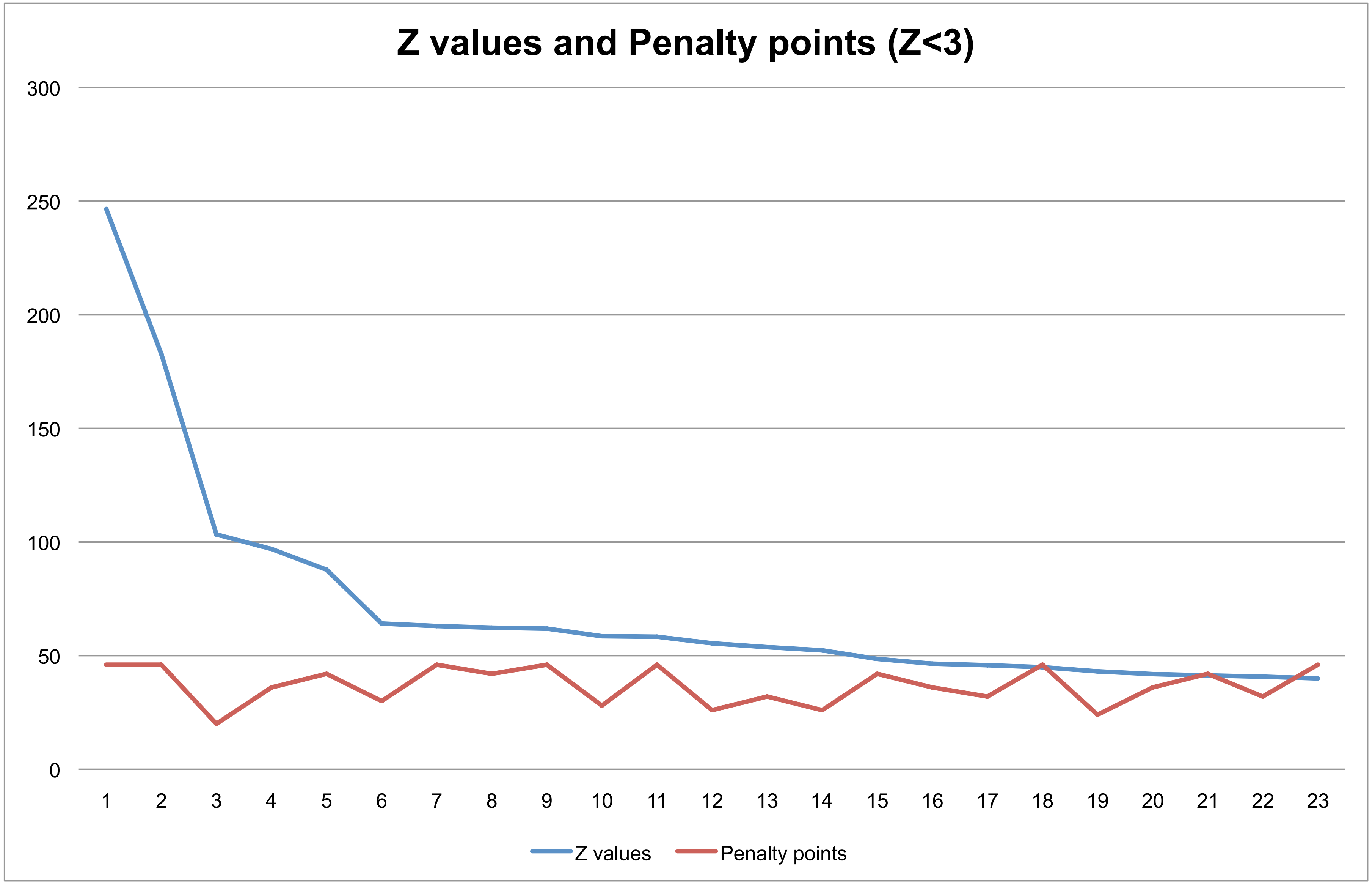
Figure 27 clustering result of CDR

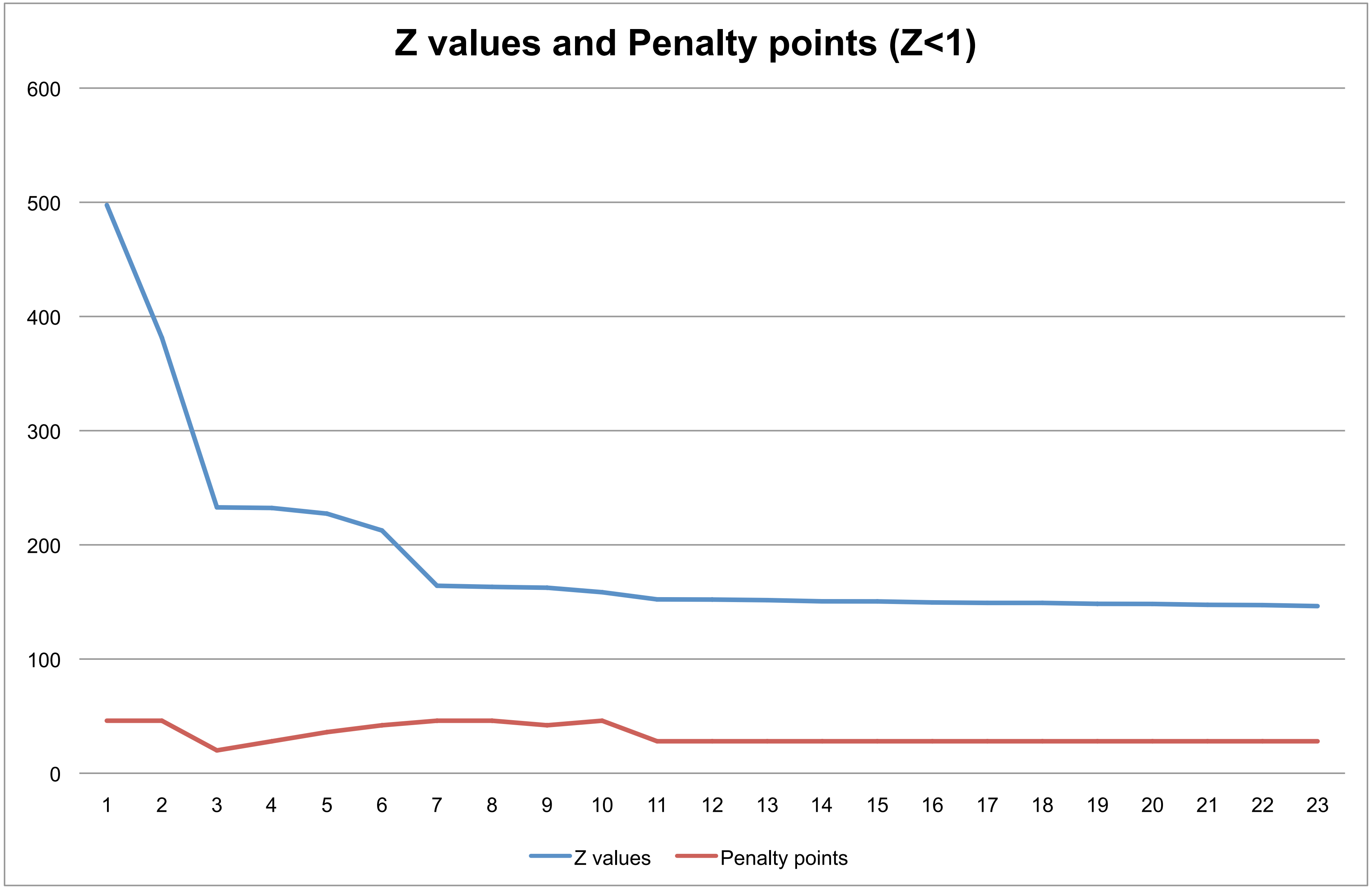
5.1.3 Real time system result

As already explained, real time system is based on the data that decays in time, because it intakes call detail record through bloomfilters and bloomfilters modify the real numerical data to guard the data efficiently. Thus the values of total number of calls, number of different calls and etc are different but they are from the same data from the same date. In here number is not revealed since it is related to privacy. Z values are calculated using the average and standard deviation of the whole distance values. It is notable that the average penalty points of the detection results are comparatively high, over 30 points. A number of trials have been made to decide appropriate values for Z. There are two types of threshold for the Z value. One is for the update of the neural network and the other is for the suspicious client detection. The former threshold is set as 2 and it means that every data point that has a Z value greater than 2 does not affect to the locations of nodes. The latter threshold is set as 150 and it means that every data point that has a Z value greater than 150 is likely to be a fraud and the call history of the relevant data will be written on a log. The values are selected after a number of empirical trials to have optimized results. The real time data analysis result is in tables below. Phone


Z values and Penalty points (Z<7)
70
60
50
40
30
20
10
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
Serie1
Serie2


Z values and Penalty points (Z<6)
140
120
100
80
60
40
20
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
Z values
Penalty points


Z values and Penalty points (Z<3)
300
250
200
150
100
50
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
Z values
Penalty points


Z values and Penalty points (Z<1)
600
500
400
300
200
100
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
Z values
Penalty points

6 Future Works

Bypass fraud (interconnection fraud) is a comparatively new type of frauds and not much research has been done to develop the refined detection system. Thus our systems are rudimentary trials to implement automatic detection system based on data mining algorithms.

There are two major issues developing these two versions of fraud detection systems. The first issue is measuring and advancing the performance of the system. Since we do not have previously known fraud patterns, it is hard to get grasp of the applications' performance. It is ideal if the known fraud detection list is provided to learn patterns of fraud and to ensure the standards of fraud detection. Although this document probably causes the privacy infringement and revelation of corporate confidential. However even though there are sufficiently accumulated the data, the detection mechanism that depends on the past fraud history limit the ability of the application. It cannot detect new types of frauds. With the assumption that frauds are rare and have different traits of calling patterns, devising efficient unsupervised outlier detection algorithm is the key to this challenge.

Secondly since the calling history of individuals implies a number of issues about their daily lives, social networks and behaviors, storing all data of customers and investigating such data in detail lead to the severe privacy infringement. However for more accurate analysis it is indispensable to store the call history of clients and implement in depth analysis on the data. To resolve this problematic situation some tools such as bloom filters which change the contents of data but store it in different formation for the later use. The monitoring system which processes the clients' information in real time is also very attractive option though; in any case storing the private information is not always avoidable. Thus anonymization or changing of personal information and data of calling history could be the ideal solution.

The development of more advanced algorithms for outlier detection and real time processing will be the future challenges to the detection system.

Reference

[1] Victoria J. Hodge and Jim Austin, "A Survey of Outlier Detection Methodologies," Artificial Intelligence Review, Volume 22 Issue 2, Pages 85 – 126, October 2004

[2] Richard O. Duda et al., *Pattern classification*, 2nd ed, Wiley-Interscience, 2000

[3] Jim Holmström, "Growing Neural Gas Experiments with GNG, GNG with Utility and Supervised GNG," Master's thesis, Computer Uppsala University, Olle Gällmo, 2002.

[4] Yamanishi, K., Takeuchi, J., et al, "On-Line Unsupervised Outlier Detection Using Finite Mixtures with Discounting Learning Algorithms," Data Mining and Knowledge Discovery 8: 275-300. 2004

[5] Netmap. Fraud and Crime Example Brochure. 2004

[6] Cox, K., Eick, S. & Wills, G. Visual Data Mining: Recognizing Telephone Calling Fraud. Data Mining and Knowledge Discovery 1: 225-231. 1997

[7] Cortes, C., Pregibon, D. & Volinsky, C. Communities of Interest. Proc. of IDA2001, 105-114. 2001

[8] Dorronsoro, J., Ginel, F., Sanchez, C. & Cruz, C. Neural Fraud Detection in Credit Card Operations. IEEE Transactions on Neural Networks 8 (4): 827-834. 1997

[9] Bolton, R. & Hand, D. Unsupervised Profiling Methods for Fraud Detection. Credit Scoring and Credit Control VII. 2001

[10] Laurikkala,J., Juhola,M.,and Kentala, E. ,'Informal Identification of Outliers in Medical Data'. In: Fifth International Workshop on Intelligent Data analysis in Medicine and Pharmacology IDAMAP-2000 Berlin, 22 August. 2000

[11] Rousseeuw, P. and Leroy, A., Ro bust Regression and outlier detection. Jonh Wiley&Sons., 3 edition.1996

[12] Datta, P. and Kibler, D., ‘Learning prototypical concept descriptions’. In: proceedings of the 12th international conference on Machine learning.pp.158–166, Morgan Kaufmann. 1995

[13] Allan, J., Carbonell, J., Doddington, G., Yamron, J., and Yang, Y., ‘Topic detection and tracking pilotstudy: Final report’. In: Proceedings of the DARPA, Broadcast News Transcriptions and Understanding Workshop. 1998

[14] Nairac,A., Townsend, N., Carr, R., King,S., Cowley,P., and Tarassenko, L., ‘A system for the analysis of jet system vibration data’. Integrated Computer Aided Engineering 6(1), 53–65. 1999

[15] Shekhar, S., Lu, C., and Zhang, P., ‘Detecting Graph-Based Spatial Outliers: Algorithms and Applications’. In: Proceedings of the Seventh ACMSIGKDD International Conference on Knowledge Discovery and Data Mining. 2001

[16] Rousseeuw, P. and Leroy,A., Ro bust Regression and Outlier Detection. John Wiley&Sons., 3edition. 1996

[17] Faloutsos,C.,Korn,F.,LabrinidisA.,KotidisY.,KaplunovichA.,andPe rkovicD., ‘Quantifiable Data Mining Using Principal Component Analysis’. Technical Report CS-TR-3754, Institute for Systems Research, University of Maryland, College Park, MD. 1997

[18] Tax, D. M. J., Ypma, A., and Duin, R. P. W., ‘Support Vector Data Description Applied to Machine Vibration Analysis’. In: Proceedings of ASCI’99, Heijen, Netherlands. 1999

[19] De Coste, D. and Levine, M. B., ‘Automated Event Detection in Space Instruments: A Case Study Using IPEX-2 Data and Support Vector Machines’. In proceedings of the SIE conference on astronomical telescopes and space, 2000

[20] WEKA Manual for Version 3-6-2, Remco R. Bouckaert, et al., University of Waikato, 2010.

[21] Kenji Yamanishi, Jun-ichi Takeuchi. “Discovering outlier filtering rules from unlabeled data: combining a supervised learner with an unsupervised learner,” Proceedings of the seventh ACM SIGKDD

international conference on Knowledge discovery and data mining, California, 2001, pp389-394

[22] Clifton Phua, Vincent Lee et al, A Comprehensive Survey of Data Mining-Based Accounting-Fraud Detection Research, Intelligent Computation Technology and Automation (ICICTA), 2010 International Conference on, vol 1, pp50 – 53

[23] Fisher, Douglas H.. "Knowledge acquisition via incremental conceptual clustering". Machine Learning, 1987, 2 (2): 139–172

[24] Fisher, Douglas H. "Improving inference through conceptual clustering". Proceedings of the 1987 AAAI Conferences. AAAI Conference. Seattle Washington. 1987, pp. 461–465.

[25] On-demand time-decaying bloom filters for telemarketer detection, Giuseppe Bianchi, Nico d'Heureuse, GermanySaverio Niccolini, ACM SIGCOMM Computer Communication Review archive, Volume 41 Issue 5, 2011

[26] Blockmon: A High-Performance Composable Network Traffic Measurement System, Andrea di Pietro, Felipe Huici, Nicola Bonelli, Brian Trammell, 2011

[27] Fast Artificial Neural Network Library is a free open source neural network library; http://leenissen.dk/fann/wp/