A Novel Single Pass Approach for Mining Obscure Periodic Pattern in Online Real Time Network Data Stream

Sonu Kumar
Assistant Professor
Department of Computer Science & Engineering
Roorkee College of Engineering,Roorkee
sonu3784@yahoo.co.in

Rimmy Chhabra
Department of Computer Science & Engineering
Roorkee College of Engineering,Roorkee
rimmychhabra1990@gmail.com

Abstract— **In this paper a novel approach has been presented for finding the obscure periodic patterns in single pass in network data stream based upon the idea of convolution. The mined information is used for online network management. The approach aims at capturing and analyzing the network data stream on a local information sphere, which has been divided into three phases: Phase 1 – Capturing network data stream, Phase 2 – Generation of the frequency distribution tables, Phase 3 – Analyzing the network data stream for obscure periodic patterns. The input string for analysis phase may be a single string or concatenation of many strings. However, the length influences the confidence with which we can mine the obscure periodic patterns. The results have been verified for different symbol periodicity, various pattern obtained and the corresponding confidence level, which were tested using the test cases with the string of 8, 40 and 80 characters.**

Keywords— obscure periodic patterns; convolution; network management.

I. INTRODUCTION

Data Stream Mining is the process of extracting knowledge structures [1] from continuous rapid data records. A data stream is an ordered sequence of instances that can be read only once or a small number of times using limited computing and storage capabilities. Examples of data streams include computer network traffic, phone conversations, ATM transactions, crowd sourcing and sensor data [5] etc. Mining hidden information accurately, reliably and predicting the future periodic patterns can help to counter the various issues pertain to network security, traffic management etc. Research in data stream mining has concentrated on discovering different types of patterns viz. sequential patterns, temporal patterns, periodic association rules, partial periodic patterns, surprising patterns etc. Existing periodic patterns mining algorithms either assume that the periodic rate is user-specified, or try to detect potential values for the period in a separate phase. The former assumption is a considerable disadvantage, especially in data streams where the period is not known a-priori. The latter approach results in a multi-pass algorithm, which on the other hand is to be avoided in online environments such as data streams.

Data stream [1] analysis is one of the most challenging problems of the recent times due to the fact of the computational limitations of the current availed microprocessor and buffering issues which limit the dynamic analysis of Data stream. This research stream becomes more fascinating by the fact that researcher around the globe are trying to reduce the complexity of the current availed algorithms to sustainable amount. Although not much success has been made.

Streaming data is ubiquitous and there is a real need to store, query and analyze [3] such rapid large volumes of data. Traditional data mining techniques are infeasible for analyzing this sort of data. Effective Knowledge Retrieval from Data Streams [4] is still a great challenge. The distinguishing trait setting data streams apart from disk-stored data is that streaming data usually exhibits time-changing data characteristics. As most decision making tasks rely on the up-to-dateness of their supporting data, the evolving nature of the data creates tremendous complexity for many mining algorithms. Thus, how to make mining algorithms more effective and efficient in view of changing data characteristics has become a major challenge in a wide range of application domains.

They assume that users either know the value of the period beforehand or are willing to try various period values until satisfactory periodic patterns emerge. Since the mining process must be executed repeatedly to obtain good results, this trial and-error scheme is clearly not efficient. Both approaches turn out to require multiple passes over the time series in order to output the periodic patterns themselves. However, real-time systems, which draw the attention of database researchers recently (e.g., as in data streams), cannot abide the time nor the storage needed for multiple passes over the data. Here we address the problem of mining periodic patterns in time series databases of unknown or obscure periods, hereafter referred to as obscure periodic patterns. We define the periodicity of the time series in terms of its symbols, and subsequently define the obscure periodic patterns where the period is a variable rather than an input parameter. We develop a convolution based application for mining the obscure periodic patterns in one pass. Our application mines the periodic patterns with unknown period in one pass.

On networks, data mining leads to two separate architectural areas which we call spheres. There are two types of information

spheres:

1. Local Information Sphere
2. Global Information Sphere

The local information sphere exists within each government agency and its goal is to perform online analysis of data and data mining of multiple high speed data streams with essentially unrestricted access to all he local data which is managed by the system.

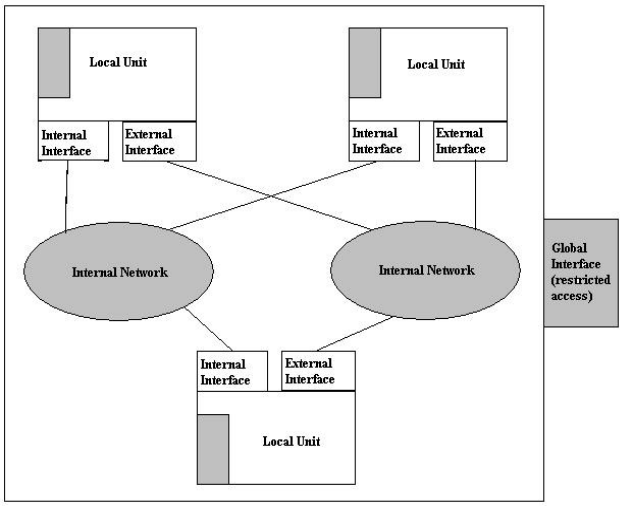

Fig. 1. Local Information Sphere

The global information sphere spans all participating government agencies, and mediates inter agency collaboration. It addresses the sometimes conflicting requirements of allowing the analysts from different agencies to effectively share information, hypothesis and evidence without violating applicable laws regarding privacy, civil rights etc.

In our project we develop a data mining application for the local information sphere. Whenever information is allowed to flow freely the information infrastructure should support unrestricted access to all data in most efficient way. This is the goal of the local information sphere which we consider. The basic components are local processing units. These units consist of one or more processors with large main memory and essentially unrestricted archival capabilities. We have not considered any specific topology. There is no requirement that the processing units be centrally located. In practice these units are more likely to be geographically distributed over the network

Our project aims at capturing and analyzing the network data stream on a local information sphere. We have approached the problem in a modular fashion. The entire project has been divided into three modules.

The approach intends to mine hidden Periodic patterns in a captured data stream. It is a network application through which we could

- Capture data streams.
- Analyze the data stream for hidden periodic patterns.
    - Develop frequency table for the patterns.

The software will be generic in use and could be used to monitor the incoming network traffic in any system and analyze it to further study the occurring periodic patterns. These patterns can be evaluated in further studies to gain useful information regarding security threats, network surveillance etc.

II. Problem Formulation

Effective Knowledge Retrieval from online real time Data Streams is still a great challenge due to rapid time-changing data characteristics. Thus, how to make mining algorithms more effective and efficient in view of changing data characteristics has become a major challenge in a wide range of application domains.

The various algorithms available for mining of periodic patterns in data stream are not effective because the computational requirements of applications based on such algorithms, are characterized by highly efficient processing of the input data stream in terms of both time and space. Furthermore, the task at hand is often to seek a small amount of useful information from the voluminous data so the objective is to trade the number of passes over the stored data with the amount of memory allocated and the time requirement for processing. Hence in order to prevent our course of development from being affected by above mentioned limitations of the available algorithms we have used a single pass algorithm.

The attention of the data mining research community very recently Indyk et. al.[9] have addressed this problem under the name periodic trends and have developed a $\Theta(n\log_2 n)$ time algorithm, where n is the length of the time series. Their notion of a periodic trend is the relaxed period of the entire time series, and their output is a set of candidate period values. In order to output the periodic patterns of the time series, a periodic patterns mining algorithm should be incorporated using each candidate period value, resulting in a multi-pass periodicity mining process. Specific to partial periodic patterns, Ma and Hellerstein [8] have developed a linear distance-based algorithm for discovering the potential periods regarding the symbols of the time series. However, their algorithm misses some valid periods since it only considers the adjacent inter-arrivals.

For example, consider a symbol that occurs in a time series in positions 0, 4, 5, 7, and 10. Although the underlying period should be 5, the algorithm only considers the periods 4, 1, 2, and 3. Should it be extended to include all possible inter-arrivals, the complexity of the algorithm of will increase to $\Theta(n^2)$.Yet, both algorithms require at least two passes over the time series in order to output the periodic patterns. Berberidis et al. [10] have solved the problem of the distance-based algorithms by developing a multi-pass algorithm for discovering the potential periods regarding the symbols of the time series, one symbol at a time. Their algorithm suffers from the need for incorporating a periodic patterns mining algorithm to output the periodic patterns of the time series.

III. Problem Definition

Assuming a sequence of n time-stamped feature values is collected in a time series database. For a given feature t, let t_i be the value of the feature at time-stamp i. The time series of feature t is represented as $T = t_0, t_1, \ldots, t_{n-1}$. For example, the feature in a time series database for power consumption might be the hourly power consumption rate of a certain customer, while the feature in a time series database for stock prices might be the final daily stock price of a specific company. If we discretize the time series feature values into nominal discrete levels and denote each level (e.g., high, medium, low, etc.) by a symbol (e.g., a, b, c, etc.), then the set of collected feature values can be denoted $\Sigma = \{a, b, c,\ldots\}$, where T is a string of length n over Σ. A time series database may also be a sequence of n time-stamped events drawn from a finite set of nominal event types, e.g., the event log in a computer network monitoring the various events that can occur. Each event type can be denoted by a symbol (e.g., a, b, c, etc.), and hence we can use the same notation above.

A. Pattern Definition

The pattern is a template, or model which can be used to make or to generate things or parts of a thing, especially if the things that are created have enough in common for the underlying pattern to be inferred, in which case the things are said to exhibit the pattern. The simplest patterns are based on repetition /periodicity; several copies of a single template are combined without modification. Pattern is a sequence of symbols that occur in network data streams and we try to mine hidden periodic patterns in these patterns. Periodic patterns can be classified on the basis of their period as follows:

- Patterns with known period.
- Patterns with unknown period or obscure period.

The patterns which have obscure period are also known as 'obscure periodic patterns'.

B. Symbol Periodicity

In a time series T, a symbol *s* is said to be periodic with a period *p* if *s* exists "almost" every *p* timestamps. For example, in the time series T = abcabbabcb, the symbol *b* is periodic with period 4 since *b* exists every four timestamps (in positions 1, 5 and 9). Moreover, the symbol a is periodic with period 3 since a exists almost every three timestamps (in positions 0, 3, and 6 but not 9). We define symbol periodicity as follows:

Let $\pi_{p,l}(T)$ denote the projection of a time series T according to a period p starting from position l; that is

$\pi_{p,l}(T) = t_l, t_{l+p}, \ldots, t_{l+(m-1)p}$,

where $l < p$, $m = (n - l)/p$, and n is the length of T.

For example, if T = abcabbabcb, then $\pi_{4,1}(T)$ = bbb, and $\pi_{3,0}(T)$ = aaab. Intuitively, the ratio of the number of occurrences of a symbol s in a certain projection $\pi_{p,l}(T)$ to the length of this projection indicates how often this symbol occurs every *p* time-stamps. However, this ratio is not quite accurate since it captures all the occurrences even the outliers. In the example above, the symbol *b* will be considered periodic with period 3 with a frequency of 1/4, which is not quite true. As another example, if for a certain T, $\pi_{p,l}(T)$ = abcbac, this means that the symbol changes every *p* time- stamp and so no symbol should be periodic with a period *p*. We remedy this problem by considering only the consecutive occurrences. A consecutive occurrence of a symbol s in a certain projection $\pi_{p,l}(T)$ indicates that the symbol s reappeared in *T* after *p* timestamps from the previous appearance, which means that p is a potential period for *s*. Let $F_2(s, T)$ denote the number of times the symbol *s* occurs in two consecutive positions in the time series *T*. For example, if T = abbaaabaa, then $F_2(a, T) = 3$ and $F_2(b, T) = 1$.

C. Obscure Periodic Patterns

The symbol periodicity not only determines the candidate periodic symbols, but also indicates corresponding periods and locates their corresponding positions. There are no pre-assumptions of the period value, and so obscure periodic patterns can be defined as follows:

If a time series *T* of length n contains a symbol *s* that is periodic with period *p* at position l with respect to an arbitrary periodicity threshold, then a periodic single-symbol pattern of length *p* is formed by putting the symbol *s* in position l and putting the "don't care" symbol § in all other positions. The support of such periodic single-symbol pattern is estimated by

$$\frac{F_2(3, \pi_{p,l}(T))}{\lceil (n-l)/p \rceil - 1}$$

For example, in the time series T = abcabbabcb, the pattern a§§ is a periodic single symbol pattern of length 3 with a support value of 2/3, and so is the single-symbol pattern §b§ with a support value of 1. However, we cannot deduce that the pattern ab§ is also periodic since we cannot estimate its support. The only thing we know for sure is that its support value will not exceed 2/3.

In a time series T of length *n*, let $S_{p,l}$ be the set of all the symbols that are periodic with period *p* at position *l* with respect to an arbitrary periodicity threshold. Let S^p be the Cartesian product of all $S_{p,l}$ in an ascending order of l, that is

$$S^p = (S_{p,0} \cup \{\S\}) \times (S_{p,1} \cup \{\S\}) \times \ldots \times (S_{p,p-1} \cup \{\S\})$$

Every ordered pair $(s_0, s_1, \ldots, s_{p-1})$ that belongs to S^p corresponds to a candidate periodic pattern of the form $s_0 s_1 \ldots s_{p-1}$ where $s_i \in S_{p,I} \cup \{\S\}$.

For example, in the time series T = abcabbabcb, we have $S_{3,0} = \{a\}$, $S_{3,1} = \{b\}$, and $S_{3,2} = \{\}$, then the candidate periodic patterns are a§§, §b§, and ab§, ignoring the "don't care" pattern §§§.

IV. Methodology

In this work, we have studied and implemented a method for analyzing the data stream for mining obscure hidden periodic patterns in a single pass. In Mining Hidden Periodic Pattern in Network Data streams, we are implementing a single pass method for mining hidden periodic patterns in Network Data streams. These network data streams have been captured as packets and sent into a buffer. This buffer later is analyzed to discard the malicious packets and filter the packet information regarding source port, destination port, date, time, IP and Data. The IP is the source IP from which the packet has been originated. All the packets are captured via Network

Interface Card buffer. These packets can be captured from a predefined timestamps which will help us to accelerate or de-accelerate the packet capturing speed. All the filtered information is later stored in a database. The approach aims at capturing and analyzing the network data stream on a local information sphere, which has been divided into three phases.
Phase1: Capturing Network Data Stream
Phase2: Generation of the network trafficking graphs
Phase3: Analyzing the network data stream for obscure Periodic Patterns.

In the first phase the data stream is captured by reading the data buffer of the Network Interface Card. The data stream is read in the form of data packets. These data packets are then filtered, by removing the unnecessary fields. The data portion of these packets is extracted and stored in a data base. We have designed the data base for storing the data streams in MS Access.

In phase two, the captured data packets are used to generate the frequency distribution tables and graphs to represent the network traffic.

In the last phase these data streams are picked up from the data base and then analyzed for obscure periodic patterns. The analysis phase is given an input string which may be the single string or a concatenation of many strings. The length of the analyzed data string influences the confidence with which we can mine the obscure periodic patterns. This phase uses a special algorithm which is based upon the idea of convolution, for mining obscure periodic patterns in a single pass.

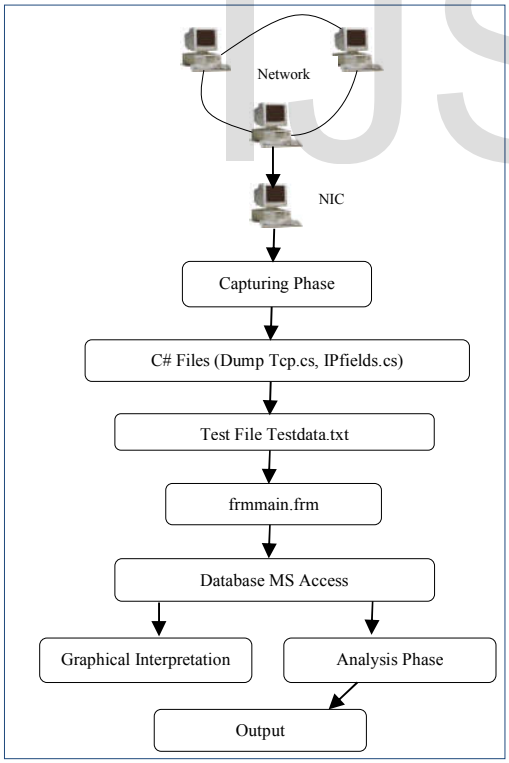

Fig. 2. Data Flow Diagram

First of all we have to capture the network data streams because that is the basic input. For capturing the data streams we require a computer on a network which can be based on any topology and of any size Packet capturing (or packet sniffing) is the process of collecting all packets of data that pass through a given network interface. Capturing network packets in our applications is a powerful capability which lets us write network monitoring, packet analyzers and security tools. To capture the data stream in the NIC card we use a network application written in C#. This application requires various libraries for its execution such as WinPcap Library and SharpPcap. The application works in a promiscuous mode i.e. it captures all the data packets that pass through the NIC card. The output generated by this application contains source IP, destination IP, port number, date and the data packet. The output is stored in a text file testdata.txt which acts as a buffer which temporarily stores the output generated by the application.

A. Mining Obscure Periodic Patterns

Assume first that the period p is known for a specific time series T. Then, the problem is reduced to mining the periodic patterns of length p. In other words, the problem is to detect the symbols that are periodic with period p. A way to solve this simpler problem is to shift the time series p positions, denoted as T (p), and compares this shifted version T (p) to the original version T. For example, if T = abcabbabcb, then shifting T three positions results in T (3) = §§§abcabba. Comparing T to T (3) results in four symbols match. If the symbols are mapped in a particular way, we can deduce that those four matches are actually two for the symbol a both at position 0, and two for the symbol b both at position 1.

Therefore, our proposed algorithm for mining obscure periodic patterns relies on two main ideas. The first is to obtain a mapping scheme for the symbols, which reveals, upon comparison, the symbols that match and their corresponding positions. Turning back to the original problem where the period is unknown, the second idea is to use the concept of convolution in order to shift and compare the time series for all possible values of the period. Hence, all the symbol periodicities can be detected in one pass. We describe those ideas in detail starting by defining the concept of convolution as it derives our mapping scheme.

CONVOLUTION METHOD

Convolution is a mathematical operator which takes two functions f and g and produces a third function that in a sense represents the amount of overlap between f and a reversed and translated version of g.

The convolution of f and g is written f*g. It is defined as the summation of the product of the two functions after one is reversed and shifted.

$$(f * g)(m) = \sum_{n} f(n)g(m-n)$$

B. Mapping Scheme

Let T = $t_0, t_1, \ldots, t_{n-1}$ be a time series of length n, where t_i's are symbols from a finite alphabet Σ of size σ. Let Φ be a mapping for the symbols of T such that $\Phi(T) = \Phi(t_0), \Phi(t_1), \ldots, \Phi(t_{n-1})$. Let

$$C^T = (\Phi(T)' \otimes \Phi(T))',$$

and C_i^T be the i^{th} component of C^T . The challenge to our one pass method is to obtain a mapping Φ of the symbols, which satisfies two conditions:

1. When the symbols match, this should contribute a non-zero value in the product $\Phi(t_j).\Phi(t_{i-j})$, otherwise it should contribute 0, and
2. The value of each component of C^T ,

$$C_i^T = \sum_{j=0}^{i} \Phi(t_j).\Phi(t_{i-j})$$

should identify the symbols that cause the occurrence of this value and their corresponding positions.

We map the symbols to the binary representation of increasing powers of two. For example, if the time series contains only the 3 symbols a, b, and c, then a possible mapping could be a : 001, b : 010, and c : 100, corresponding to power values of 0, 1, and 2, respectively. Hence, a time series of length n is converted to a binary vector of length σ.n. For example, let **T = acccabb**, then T is converted to the binary $\bar{T}$=0011001001000010100**10.** Adopting regular convolution, defined previously, results in a sequence C^T of length σ.n. Considering only the n positions 0, σ, 2σ, . . . , (n - 1)σ, which are the exact start positions of the symbols, gives back the sequence C^T . The latter derivation of C^T can be written as using the projection notation.

$$C^T = \pi_{\sigma,0}(C^{\bar{T}})$$

The Database entries are concatenated to form a combined string of various lengths. This length(s) of string is significant as they decide the domain of the patterns that will be predicted. After concatenation our method is applied to the combined string that uses Convolution method which enables us to predict patterns and the confidence for each of such patterns. Also, we can analyze the database with various graphs that are available in the software.

In any data stream application on networks we cannot sate the patterns which are to be mined in advance. Nor can we state the periodicity with which the patterns are to be mined. Therefore we need an algorithm which mines all the hidden periodic patterns by considering all possible periodicities. This paper describes one novel method of mining obscure periodic patterns using CONVOLUTION as the computational base. This method uses a specific kind of mapping which is described in the MAPPING SCHEME as given below:

TABLE I. MAPPING TABLE FOR ALPHABETS ∑ = {A,B,C……..Z}

Symbol	Decimal Notation	Binary Representation
A	2^0	00000000000000000000000001
B	2^1	00000000000000000000000010
C	2^2	00000000000000000000000100
D	2^3	00000000000000000000001000
E	2^4	00000000000000000000010000
F	2^5	00000000000000000000100000
G	2^6	00000000000000000001000000
H	2^7	00000000000000000010000000
I	2^8	00000000000000000100000000
J	2^9	00000000000000001000000000
K	2^{10}	00000000000000010000000000
L	2^{11}	00000000000000100000000000
M	2^{12}	00000000000001000000000000
N	2^{13}	00000000000010000000000000
O	2^{14}	00000000000100000000000000
P	2^{15}	00000000001000000000000000
Q	2^{16}	00000000010000000000000000
R	2^{17}	00000000100000000000000000
S	2^{18}	00000001000000000000000000
T	2^{19}	00000010000000000000000000
U	2^{20}	00000100000000000000000000
V	2^{21}	00001000000000000000000000
W	2^{22}	00010000000000000000000000
X	2^{23}	00100000000000000000000000
Y	2^{24}	01000000000000000000000000
Z	2^{25}	10000000000000000000000000

The first condition is satisfied since the only way to obtain a value of 1 contributing to a component of C^T is that this 1 comes from the same symbol. For example, for T = acccabb, although $C_1^{\bar{T}} = 1$, this is not considered one of C^T components. However, $C_3^{\bar{T}} = 3$, and so $C_1^T = 3$, which corresponds to three matches when T is compared to $T^{(1)}$. Those matches are seen from the manual inspection of T to be two c's and one b. Nevertheless, it is not possible to determine those symbols only by examining the value of C_1^T, i.e., the second condition is not yet satisfied.

Therefore, we modify the convolution definition to be

$$(x \otimes y)_i = \sum_{j=0}^{i} 2^j x_i y_{i-j}$$

The reason for adding the coefficient 2^j is to get a different contribution for each match, rather than an unvarying contribution of 1. For example, when the new definition of convolution is used for the previous example, $C_1^T = 2^1 + 2^{11} + 2^{14} = 18434$. The block illustrates this calculation. The powers of 2 for this value are 1, 11, and 14. Examining those powers modulo 3, which is the size of the alphabet in this particular example, results in 1, 2 and 2, respectively, which correspond to the symbols b, c, and c, respectively.

T: 0 0 1 1 0 0 1 0 0 1 0 0 0 0 1 0 1 0 0 1 0
$\bar{T}^{(3)}$: 0 0 1 1 0 0 1 0 0 1 0 0 0 0 1 0 1 0
$C_3^T = C_1^T =$ $2^{14} + 2^{11} +$ 2^1

T: 0 0 1 1 0 0 1 0 0 1 0 0 0 0 1 0 1 0 0 1 0
$\bar{T}^{(12)}$: 0 0 1 1 0 0 1 0 0
$C_3^T = C_1^T =$ 2^6

Fig. 3. Calculation Block

Figure gives another example for C_4^T containing only one power of 2, which is 6, that corresponds to the symbol a since 6 mod 3 = 0 and 'a' was originally mapped to the binary representation of 2^0. This means that comparing T to $T^{(4)}$ results in only one match of the symbol a. Moreover, the power value of 6 reveals that the symbol a is at position 0 in $T^{(4)}$. Note that in the binary vector, the most significant bit is the leftmost one, whereas the most significant position of the time series T is the rightmost one. Therefore, not only can the power values reveal the number of matches of each symbol at each period, they also reveal their corresponding starting positions. This latter observation complies with the definition of symbol periodicity.

Formally, let S0, S1, …..Sσ -1 are the symbols of the alphabet of a time series T of length n. Assume that each symbol s k is mapped to the σ-bit binary representation of 2 k to form $\bar{T}$. The sequence $C^{\bar{T}}$ is computed such that

$$C_i^T = \sum_{j=0}^{i} 2^j \bar{t}^j . \bar{t}_{i-j}$$

for i = 0, 1, . . . , σ_{n-1}. Thus,

$$C^T = \pi_{\sigma,0} C^{\bar{T}}$$

Assume that C_p^T is a non-zero component of C^T.
Let w_p denote the set of powers of 2 contained in c^T_p , i.e.,

$$Wp = \{w_{p,1}, w_{p,2}, \ldots\}$$

Where

$$C_P^T = \sum_h 2^{W_{P,h}}$$

and let

$W_{p,k} = \{ w_{p,h}: w_{p,h} \in W_p \Lambda w_{p,h} \bmod \sigma = k\}$.

As shown in the previous example, the cardinality of each $W_{p,k}$ represents the number of matches of the symbol s_k when T is compared to $T^{(p)}$. Moreover, let

$W_{p,k,l} = \{ w_{p,h}: w_{p,h} \in W_{p,k} \Lambda (n-p-1-\lfloor w_{p,h}/\sigma \rfloor) \bmod p = l\}$.

Revisiting the definition of symbol periodicity, we observe that the cardinality of each $W_{p,k,l}$ is equal to the desired value of $F_2(s_k, \pi_{p,l}(T))$. Working out the example where T = abcabbabcb, n = 10, and = 3, let s_0, s_1, s_2 = a, b, c, respectively. Then, for p = 3, W_3 = {18, 16, 9, 7}, $W_{3,0}$ = {18, 9}, $W_{3,0,0}$ = {18, 9}this implies $F_2(a, \pi_{3,0}(T)) = 2$ which conforms to the results obtained previously.

As another example, if T = cabccbacd where n = 9, = 4, and s_0, s_1, s_2 ,s_3 = a, b, c, d, respectively, then for p = 4, W_4 = {18, 6}, $W_{4,2}$ = {18, 6}, $W_{4,2,0}$ = {18} $F_2(c, \pi_{4,0}(T)) = 1$, and $W_{4,2,3}$ = {6} $F_2(c, \pi_{4,3}(T)) = 1$ which are correct since $\pi_{4,0}(T)$ = ccd and $\pi_{4,3}(T)$ = cc.

Input: A time series T = T_0, T_1,…….,T_{n-1} and the periodicity threshold.

Output: Candidate periodic patterns for T (a pattern and its support value)

Algorithm:

1. Select an arbitrary ordering for the symbols S_0,S_1, ….$S_{\sigma-1}$
2. From T, obtain a binary vector $\bar{T}$ by replacing every symbol S_k by the bit binary representation of 2^k.
3. Compute the sequence $C^{\bar{T}}$ where
   $C_i^T = \sum_{j=0}^{i} 2^j \bar{t}^j . \bar{t}_{i-j}$
   and consequently $C^T = \pi_{\sigma,0}(C^T)$
4. For p=1,2,……,n/2,
   (a) Compute the set W_P = {W_{P-1}, W_{P-2},……} where
   $$C_P^T = \sum_h 2^{W_{P,h}}$$
   (b) For k=0, 1,….., $\sigma - 1$, and l = 0,1,….,p-1, compute the sets $W_{p,k}$ and $W_{p,k,l}$ as defined, and consequently the values of $F_2(S_k, \pi_{p,l}(T))$,
   (c) If we have, $\frac{F_2(S_k.\pi_{p,l}(T))}{\lceil (n-l)/p \rceil - 1} \geq \varphi$ then p is a candidate period for the symbol S_k at position l.
   (d) Form the set of candidate periodic patterns according to Definition 3, calculate the support of each as defined, and output those whose support values are not less than the periodicity threshold.

Fig. 4. Mining Algorithm

Now we use an application written in Visual Basic, Frmmain.frm which carries out all the analysis on the captured data stream. This application initially accesses the text file testdata.txt and checks whether the fields captured by the C# code are in required format or not, it also trims off the header from the data packet and puts the data into MS Access database, each packet stored in the database has eight characters. The database acts as a buffer and works in FIFO fashion. The data stored in the database is again accessed by the application written in VB for:

- Graphical Representation
- Analysis of data stored in the database

Analysis of data stored in the database: For this we retrieve the single packet or a group of packets from database, this act as an input for our analysis application. The domain of the symbols in the packet is the uppercase alphabets (A-Z), now we map each alphabet to a 26 bit binary representation stored in the mapping table, for e.g. A will be 20 which will be 00000000000000000000000001, similarly rests of the characters are mapped. Then all these binary representation are concatenated and convolution method is applied on the binary concatenated string, after application of convolution we get to know the position and period of the patterns. Then we

calculate the support for the patterns. The final output will have set of all the candidate periodic patterns.

Graphical representation: Here we generate various graphs like IP v/s frequency, date v/s frequency (for particular IP) and IP v/s frequency (for particular date) based on the data captured.

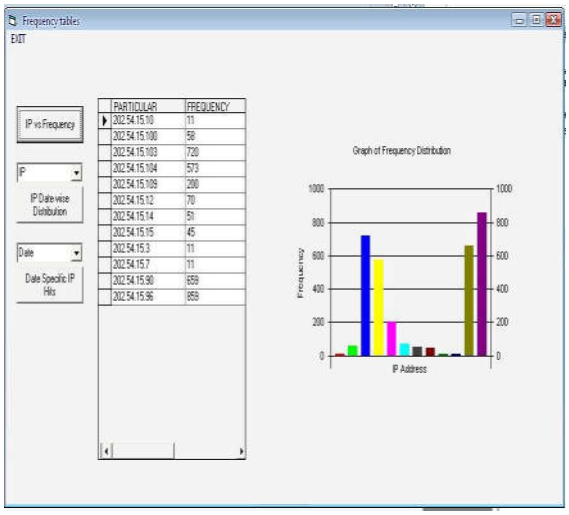

Fig. 5. Frequency of Packets against IP Addresses

V. Conclusions

A single pass method has been presented that mines hidden periodic patterns [7] in Network Data streams [1] with unknown or obscure periods. The mined information is used for online Network Surveillance. The approach aims at capturing and analyzing the network data stream on a local information sphere, which has been divided into three phases. The data streams are picked up from the data base and then analyzed for finding the obscure periodic patterns in single pass using a novel algorithm based upon the idea of convolution. The input string for analysis phase may be a single string or concatenation of many strings. However, the length influences the confidence with which we can mine the obscure periodic patterns. The results have been verified for different symbol periodicity, various pattern [6] obtained and the corresponding confidence level, which were tested using the test cases with the string of 8, 40 and 80 characters.

References

[1] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and Issues in Data Stream Systems", in Proceedings of PODS, 2002.

[2] Gama J., "Knowledge Discovery from Data Streams", CRC Press, 2010

[3] Gaber, M, M., Zaslavsky, A., and Krishnaswamy, S., Mining ,"Data Streams: A Review, in ACM SIGMOD Record", Vol. 34, No. 1, June 2005, ISSN: 0163-5808.

[4] Ganguly A. Gama J. Omitaomu o., Gaber M. M., Vatsavi R. R. (Eds., "Knowledge Discovery from Sensor Data", CRC Press, 2008

[5] Gama J. And Gaber M. M. (Rds.), "Learning from Data Streams: Processing Techniques in Sensor Networks", Springer Verlag 2007.

[6] S. Papadimitriou, J. Sun and C. Faloutsos, "Streaming pattern discovery in multiple timeseries" in Proc. 31st International Conference on Very large Data Bases 2005, pp. 697-708.

[7] Wei Cheng, Haoshan Shi, Xipeng Yin, and Dong Li, "An Elitism Strategy Based Genetic Algorithm for Streaming Pattern Discovery in Wireless Sensor Networks", IEEE Communications Letters, vol. 15, no. 4, April 2011, pp. 419-421.

[8] S. Ma and J. Hell Hellerstein "Mining Pertially Periodic Event Patterns With Unknown Periods," Proc. 17th IEEE Int "1 Conf. Data Eng., Apr 2001".

[9] P. Indyk, Journal of ACM, Vol 53, No. 3, May 2006, pp. 307-323.

[10] C.Berberidis,W.Aref,M.Atallah,I.Vlahavas,andA.Elmagarmid,"Multiple andPartialPeriodicityMininginTimeSeriesDatabases,"Proc.15thEuropean Conf.ArtificialIntelligence, July2002.