*I.J. Information Technology and Computer Science,* **2017, 4, 48-62**
Published Online April 2017 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2017.04.07

# Mining Maximal Quasi Regular Patterns in Weighted Dynamic Networks

**Hardeo Kumar Thakur, Anand Gupta**
COE Division, NSIT, University of Delhi, India
E-mail: {hardeokumar, omaranand}@gmail.com

**Bhavuk Jain and Ambika**
IT Division, NSIT, University of Delhi, India
E-mail: {jainbhavuk630, ambikag44}@gmail.com

***Abstract***—Interactions appearing regularly in a network may be disturbed due to the presence of noise or random occurrence of events at some timestamps. Ignoring them may devoid us from having better understanding of the networks under consideration. Therefore, to solve this problem, researchers have attempted to find quasi/quasi-regular patterns in non-weighted dynamic networks. To the best of our knowledge, no work has been reported in mining such patterns in weighted dynamic networks. So, in this paper we present a novel method which mines maximal quasi regular patterns on structure (MQRPS) and maximal quasi regular patterns on weight (MQRPW) in weighted dynamic networks. Also, we have provided a relationship between MQRPW and MQRPS which facilitates in the running of the proposed method only once, even when both are required and thus leading to reduction in computation time. Further, the analysis of the patterns so obtained is done to gain a better insight into their nature using four parameters, viz. modularity, cliques, most commonly used centrality measures and intersection. Experiments on Enron-email and a synthetic dataset show that the proposed method with relationship and analysis is potentially useful to extract previously unknown vital information.

***Index Terms***—Dynamic weighted graph, evolving graph, regular graph, quasi regular graph, frequent graph.

## I. Introduction

Graph mining is an important topic in the field of Data Mining because most of the available data can be modelled using the graphs, in which the vertices represent entities and the edges represent relationship between them. We know that the amount of data is increasing at an exponential rate spanning across all the domains. But, at the same time, advances in data collection and storage capacity are coping up to make it possible to collect temporal graph (Dynamic Graph) dataset. Pattern recognition algorithms, earlier limited to temporal databases [1], now hold utmost importance for enhancing our understanding about local behavior in dynamic networks too broadly, till date, there are two main trends to analyze dynamic networks:

1. Mining of non-weighted, undirected dynamic networks.
2. Mining of attributed/weighted dynamic networks.

The former that involves the analysis of the overall structure of the graphs dates back to 2004 it has deal with the analysis of the World Wide Web [2]. It has primarily focused on temporal evolution of web graphs. Further, due to the need for mining patterns in frequent dynamic sub graphs, the first algorithm [3] for the same has been evolved which has been earlier limited to mining in the domain of Web Data. Thereafter, a method [4] has been brought forward which has revolved around mining partially periodic patterns in the structure of dynamic networks. However, Lahiri et al. have presented a method which is robust enough for mining patterns in attributed networks as well. Further, in order to extract specific information depending on the needs of the users, an algorithm [5] focusing on mining dense and isolated sub graphs by user-parameterized constraints has been developed in 2009. Thereafter, a novel method [6] for mining similar patterns in biological networks has been developed, which has also involved analyzing a sequence of graphs using two metrics: 'Prediction' and 'Coverage'. This has been followed by the development of an algorithm [7] for mining regular periodic patterns in dynamic networks. Subsequently, for obtaining greater insight into the evolving nature of dynamic networks, a method [8] for finding the most frequently changing components (MFCC) in evolving graphs has been presented.

Besides, the concept of attributed dynamic networks with the addition of vertex-weighted and edge-weighted networks also took shape in 2007. The central theme of the paper [9] has been on introduction and discovery of trend-motifs, which have targeted putative patterns of changes for a group of closely related entities. Afterwards, a method [10] to find dense homogenous sub graphs (having vertices which share a large set of attributes) has been developed for obtaining insight into the microscopic properties of attributed networks. A novel method [11] for graph clustering based on both structural and attribute similarities through a unified distance measure has been proposed in 2009. It has been followed with the evolution

 

of an efficient algorithm [12] for the NP-Hard problem of finding the highest-scoring temporal sub graph in a dynamic network. In 2013, a paper [13] focusing on mining the graph topology of a large attributed graph by finding irregularities among vertex descriptors and analyzing the resulting topological patterns using three measures has also been brought forward.

However, due to the ever increasing rate of data, the problem of noise/jitter has surfaced. Noise (a random occurrence of events) can be due to a variety of factors and causes a loss of important trends and patterns. The concept of jitter has been handled previously while mining of partial periodic patterns [4] and regular periodic patterns in the overall structure [7] in dynamic networks. However, noise can affect not only the presence and absence of relationships over a period but also the attributes of edges such as weight and direction. To express the importance of noise in such attributes better, let us take an example of any E-commerce network, which contains information about products that are bought together over time. For simplicity, let us assume that if the quantity of a product sold at a particular timestamp lies between 1 to 10, then it will be denoted by the character a and if it lies between 11 to 20, then it will be denoted by character b and so on. It is possible, that two products A and B over a period of 8 days are sold in quantities ranging from 1 to 10 and on the 9$^{th}$ day, due to problems in the website are not sold at all. Therefore, we represent the relationship between product A and B as, aaaaaaaa0. The pattern here would have been "aaa" if the products A and B would have been sold on the 9$^{th}$ day as well. This pattern (Quasi regular), though important, would have been lost if mined by the existing algorithms [14,7]. Our work tries to fill the gap that has been there due to the lack of focus on noisy aspect of dynamic datasets. The intention is to propose a framework which incorporates the presence of noise and proposes a more reliable framework.

Thus, as evident from the above example, due to the presence of noise, some important information is easily lost. The focus of this article has been to mine such noisy patterns. And such patterns when mined on structural aspect are known as Maximal Quasi Regular Pattern on structure (MQRPS) and when done on the weight aspect are known as Maximal Quasi Regular Pattern on Weight (MQRPW).

Also, we make use of an observation in order to increase the efficiency of our work. This observation has been presented as a claim in the article which is as follows:

An edge representing a relationship between two entities shall have the possibility of following a MQRPW if and only if it follows MQRPS.

Subsequently, a microscopic analysis of the evolution graphs formed from the patterns obtained has been done on the basis of 4 parameters which are explained in the subsequent subsections (Section5).

Summarizing, the main contributions in this paper are as follows: In Section-II, formal definition for mining Maximal Quasi Regular Patterns in dynamic networks has been presented. In Section-III, the proposed method for mining evolution graph on structure (MQRPS) or evolution graph on weight (MQRPW) is described. In Section-IV, a claim is introduced which explain how the proposed method can be restructured by incorporating the claim for saving our time while mining the aforementioned patterns. Further, complexity of the proposed method (Section-III) is assessed. In Section-V, intention of taking different parameters for the analysis of the obtained patterns is explained. In Section-VI, the method is evaluated on one real world dataset (Enron Email Dataset) and on a synthetically generated dataset. The synthetic dataset is taken to establish the relative importance of the proposed method and parameters in business networks. Further, we analyze the evolution graphs obtained on the basis of 4 parameters. It helps us to get an insight into the trends and habits. Finally, in Section-VII, discussion and concluding remarks are provided with future research possibilities.

## II. Notations and Definitions

In this section, we will define weighted dynamic network and the type of patterns that we are interested in within these network.

**Definition 1: Time series of graphs:** For a given sequence of T graphs $G = \{G_1, G_2,.....,G_T\}$ with $G_t = (V_t, E_t, W_t)$, where $1<=t<=T$, $V_t$ is the vertex-set, $E_t$ is the edge-set and $W_t$(weight on edges) is the weight-set of the graph at a timestamp '*t*'. We define G as a time series of graphs which can be transformed into weighted dynamic network.

**Definition 2: Weighted dynamic network:** For a given a time series of graphs G with T timestamps, a weighted dynamic network $G_d$ is defined as $G_d = \{V_d, E_d, W_d\}$, where $V_d = U_{t=1}^{T} V_t$, $E_d = U_{t=1}^{T} E_t$ and $W_d = U_{t=1}^{T} W_t$. So, $G_t = (V_t, E_t, W_t)$ and $1<=t<=T$, is a simple graph of interactions $E_t$ observed at a timestamp '*t*' among the set of uniquely labelled entities $V_t \subseteq V$ (union of all vertices) with a function $F_t$ mapping weights to edges: $F_t$: $E_t$ -> $W_t$. For example, Fig. 1(a) represents a dynamic weighted network across 9(T= 9) timestamps.

Here, we are interested in forming occurrence sequence and weight sequence of edges, which are defined next.

**Definition 3: Occurrence Sequence:** The occurrence sequence [7] of an edge e is a sequence of 1s and 0s of length T such that if 'e' appears at a timestamp t (t<= T), then the t$^{th}$ position of its occurrence sequence is 1, otherwise it is 0. Therefore, in Fig. 1(b), the occurrence sequence of edge AC is "110110111".

**Definition 4: Weight Sequence:** The weight sequence [15] of an edge e is a sequence of values of length T such that if an edge appears at timestamp t (t<=T), then the *t*$^{th}$

position of its weight sequence has the corresponding weight W, otherwise it is 0. Hence, in Fig.1(b), the weight sequence of edge AC is "ab0ab0abc".

Note that Occurrence Sequence is the set of $\{1 \mid 0\}^m$ and Weight Sequence is the set of $\{W \mid 0\}^m$, where W is the weight of an edge at that timestamp. We can now specify our concept of Summary Graph using Occurrence Sequence and Weight Sequence.

**Definition 5: Summary Graph:** For a given weighted dynamic network, the summary graph [17] Gs for the network G = { $G_1$,$G_2$,….,$G_T$ } is defined as a set of $G_s$ = {Vs, Es, Le}, where Vs = V ($V_1 \cup V_2 \cup \ldots \cup V_T$) , $E_s$ = E ($E_1 \cup E_2 \cup \ldots E_T$) , and $L_e$ is the set of labels of $E_s$ , which maps each edge *e* in $G_s$ with its corresponding occurrence sequence and weight sequence. Therefore, Fig. 1(b) represents the summary graph of Fig. 1(a).
This paper provides a method for mining MQRPS / MQRPW incorporating our notion of jitter, which is explained in the next definition.

**Definition 6: Jitter:** In the occurrence sequence, it might be possible that due to a random occurrence of an event, an interaction which is present might be recorded as being absent or vice-versa. The occurrence sequence of edge AC in Fig. 1(b) is "110110111". It is clear that the pattern, "110" will be missed by regular pattern miner [7]. This random occurrence of an event at 3rd position of pattern "110" is defined as jitter. Similarly, jitter can be present in weight sequence as well.
Here, we focus on mining patterns having length of jitter equal to 1.

**Definition 7: Maximal Quasi Regular Pattern(s) (MQRP):** The MQRP corresponding to occurrence/weight sequence for an edge E, is the substring(s) S, such that S = maxlength ($S_1$, $S_2$, …..$S_n$) where ($S_1$,$S_2$....$S_n$) are the substrings in the sequence that satisfy the condition of repeating consecutively minimum *th* (threshold) number of times and can have jitter. As an example, let the weight sequence be "abcabcabdmnopmnopmnoqwasdwasdwase" and the minimum threshold is 3, then the substrings satisfying criteria of minimum threshold are *(mnop,abc,wasd)* where *"mnop"* and *"wasd"* are considered as MQRPs (here, MQRP on weight).

**Definition 8: Maximal Quasi Regular Pattern(s) on Structure (MQRPS):** An edge *e* has MQRPS if it follows a MQRP in its occurrence sequence. In Fig. 1(b), the edge AC will have the MQRPS as "110".

**Definition 9: Maximal Quasi Regular Pattern(s) on Weight (MQRPW):** An edge *e* has MQRPW if it follows a MQRP in its weight sequence. In Fig. 1(b), the edge AC will have the MQRPW as "ab0".

Hence, if $L_{n_{(G_1,G_2,G_3.....G_n)}}$ represents the length of the occurrence sequence for an edge in the dynamic network, Pi corresponds to the *pattern* (MQRPS/MQRPW) in that edge, $L(P_i)$ is the length of the pattern and '*th*' is the threshold for which the pattern should appear continuously, then,

$$L(P_i) \leq \frac{L_{n_{(G_1,G_2,G_3.....G_n)}}}{th} \quad (1)$$

Since we have assumed the length of the jitter to be 1, then, the maximum percentage of noise in a pattern that can be captured by the proposed method is,

$$\alpha(Pi) \leq \frac{1}{L(P_i)} * 100 \quad (2)$$

Substituting the value of $L(P_{i)}$ from expression (1) into expression (2), we get,

$$\alpha(Pi) \leq \frac{100 * th}{L_{n_{(G_1,G_2,G_3.....G_n)}}} \quad (3)$$

From this expression, it can be inferred that the maximum percentage of noise in a pattern that can be captured is directly proportional to the threshold (th).

For example, if the threshold is 3 and the occurrence sequence for an edge over a time-period of 16 units is "1101101111010101", then the MQRPS will be "110" and "101". Since, the length of the patterns mined is equal to 3 and we have assumed the length of jitter to be 1, therefore the maximum percentage of noise is 33.33%.

Now, we state the concept of Evolution Graph.

**Definition 10: Evolution Graph(s)**: A sub graph G = (E, V) is an Evolution Graph on structure/weight, if it is connected and all its edges $E_i \in E$ follow the same MQRPS/MQRPW.

**Problem Definition:** Given a dynamic network G = ($G_1$, $G_2$, $G_3$, $G_4 \ldots G_{|n|}$) and threshold th, the method for discovering evolution graph(s) on structure (MQRPS) as well as on weight (MQRPW) and analyze such evolution graph(s) on different parameters (as described in section (V).

 

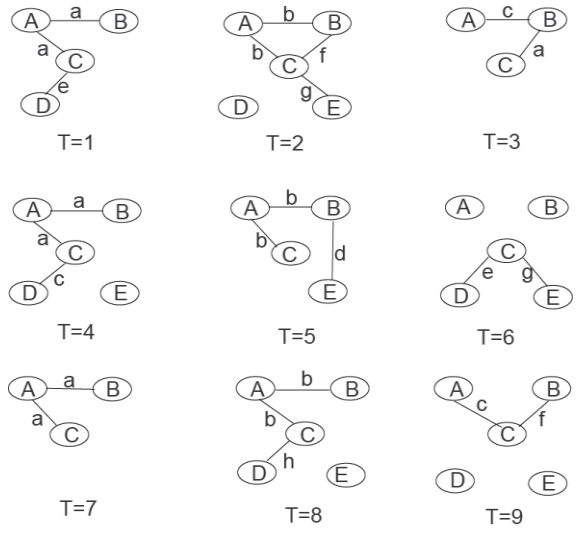


Fig.1(a) Weighted Dynamic Network

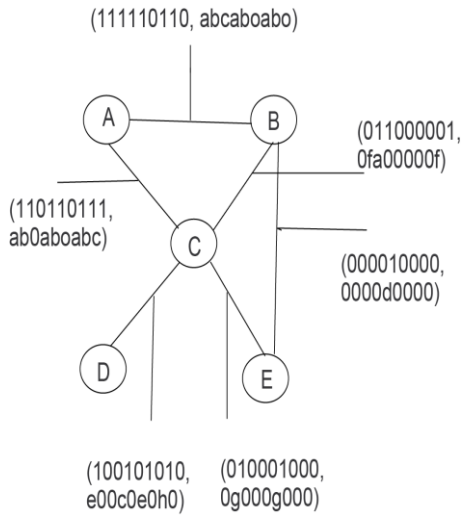


Fig.1(b) Summary Graph of Fig. 1(a).

Fig.1. Weighted Dynamic Network with Summary Graph.

## III. METHOD

In order to achieve the objective mentioned in the problem definition, a methodology has been worked out and is described in this section. The following will describe the input/output and each of the steps involved:

**Input:** A set of {G, s, th}, where G is the weighted dynamic network which is equal to $\{G_1, G_2 \ldots\ldots G_t\}$, *s* is the minimum number of times an edge should appear to be considered significant for pattern mining (3 as default value) and threshold '*th*' (3 as default value) is the minimum number of times a substring should appear consecutively in the occurrence/weight sequence to be considered as a valid pattern (MQRPS/MQRPW).

### *A. Construction of Summary graph:*

Summary graph is constructed corresponding to occurrence/weight sequence. For example, Fig. 1(b) represents the summary graph for Fig.1 (a).

### *B. Removal of infrequent edges:*

Remove those edges from the summary graph which appear at less than *s* timestamps in the dynamic network.

In Fig. 1(b), we consider the edges BE and CE as infrequent edges because the number of 1's in their occurrence sequence is less than 3 which results in Fig.2.

### *C. Searching for edges following a MQRPS/MQRPW:*

Algorithm is applied on each of the occurrence sequence/weight sequence in the summary graph (Fig. 2) to obtain the MQRPS/MQRPW for each edge. Applying the algorithm, we obtain the MQRPS for (AC, BC, CD, AB) in the summary graph in Fig. 2 which are ("110","00","01","110") respectively.

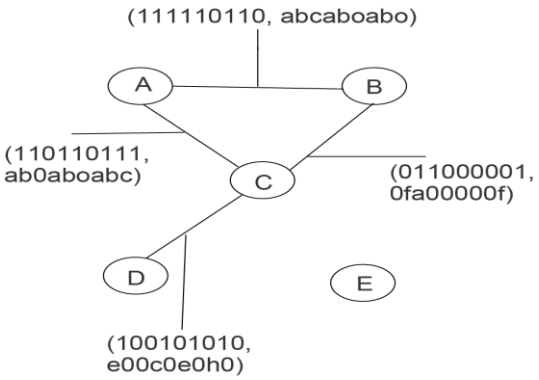


Fig.2. Summary graph after removing infrequent edges.

**Algorithm:**

**Function1:**

*Input: Sequence in which patterns have to be found, minimum threshold 'th'.*
*Output: Actual patterns in network (MQRPS/MQRPW)*

```
1. th = 3 (used if nothing mentioned)
2. len = floor(nchar(string)/th)
3. found:= FALSE
4. for sublen = len to 1 do
5.         for inlen = 0 to sublen do
6.             dif <- Take difference of sublen & inlen
7.             res <- Repeat "(\\2.\\3)" th-1 times
```

```
8.      pat <- Paste res and regular expression diff, (.{",inlen,"})
9.      test <- Match the substring available in pat with string using gregexpr and return all
        disjunct matches
10.     for r in test
11.         length = nchar(r)
12.         if ( length > 0 )
13.             for i = 1 to length
14.                 a <- substring of the original string to find the pattern
15.                 lenp <- nchar(a)
16.                 start <- r[i]
17.                 end <- r[i] + lenp*th - 1
18.                 b <- Pass the value of string,start,end,lenp to function2
19.                 patterns <- Write b in the list of patterns
20.                 found = TRUE
21.             end for
22.         end if
23.     end for
24.   end for
25.   if (found = TRUE)
26.       break
27. end for
```

**Function2:**

*Input: Substring with its starting point in the original string, threshold set by user, length of the pattern found by Function1.*

*Output: Actual pattern in a substring.*

```
1. s <- string
2. begin <- start
3. terminate <- end
4. length <- lenp
5. startchar <- Generate a sequence using `seq` command with input parameters as begin, terminate,
   length.
6. a <- Make a table showing the number of occurrences of each of the string of length in the original
   string s.
7. Return the pattern having the maximum value in a.
```

*D. Removal of edges having all 0's in their MQRPS/MQRPW:*

We have removed edges having all 0's in their MQRPS which are considered to be insignificant, and therefore, the edge BC is removed.

*E. Determination of Evolution Graphs on structure/on Weight:*

The edges with the same MQRPS/MQRPW are grouped to form evolution graphs on structure which should have at least two edges.

Since edges AC and AB follow the same MQRPS, both of them combine to form the evolution graph shown in Fig. 3.

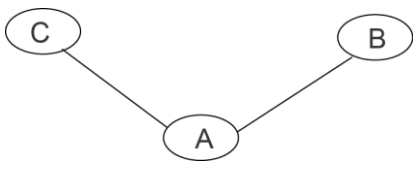


Fig.3. Evolution graph for MQRPS "110"

Similarly, in the same manner we can apply the proposed method for mining MQRPW as well. For instance, taking the same weighted network in fig. 1(a) and applying the steps (A-E), we will obtain evolution graph on weight as shown in Fig. 4.

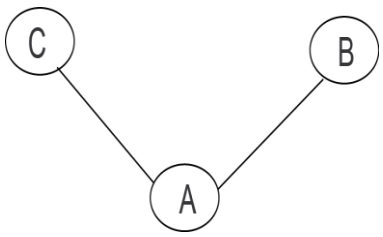


Fig.4. Evolution graph for MQRPW "ab0"

**Output: Evolution graph on structure (MQRPS)/on weight (MQRPW)**

Also, when MQRPS and MQRPW are required then instead of using the steps A-E of the method twice for-mining both, we can save our time by using the claim mentioned in the introduction.

In the next section, the claim is elaborated and used for

efficient mining of MQRPS as well as MQRPW.

## IV. Explanation of Claims

After mining MQRPW on edges, we matched these edges with those which follow MQRPS. We have observed that only a subset of the edges which follow MQRPS follow MQRPW. Therefore, we make the following claim.

### *4.1 Claim:*

An edge will have the possibility of following MQRPW if and only if it follows MQRPS.

*Proof:*

Let us prove it by contradiction. Let there be an edge linking the two vertices, having an occurrence sequence (string of '0's and '1's) and a weight sequence (string of characters /numerals) over a time period T. Let us assume that it follows MQRPW but not MQRPS. But, the presence of weight on an edge at a timestamp implies the presence of that edge at that particular timestamp as well, which is denoted by '1' in the occurrence sequence. Also, the flexibility of jitter = 1 holds in both the sequences. For understanding it better, let us take an example.

Consider minimum threshold (*th*=3) and any arbitrary weight sequence for an edge ***e*** over a time period (T = 12) to be *"abcdabcdab0d"*. Clearly, MQRPW for this edge ***e*** is *"abcd"*. Now, applying the concept that an edge will have a weight associated with it at a particular timestamp, if and only if it is present at that timestamp, it gives us the occurrence sequence of ***e*** as "11111111101". But, as can be seen, the occurrence sequence obtained follows MQRPS which is "1111".

Hence, if an edge follows MQRPW then it will definitely follow MQRPS, which proves our assumption being wrong. Therefore, claim is proved.

It follows from the intuition as well that if an edge follows MQRPS then only it has a chance of following MQRPW.

### *4.2 Restructuring the method:*

Now, we add the following steps to the method proposed in Section3 for incorporating the claim. Here, we first mine MQRPS using the method (steps A-E) and then MQRPW using the steps as follows:

1) **Mapping of the edges following MQRPS to obtain their weight sequence for mining MQRPW:** Applying the Claim, only the weight sequences corresponding to the edges AC ("ab0ab0abc"), BC ("0fa00000f"), CD ("e00c0e0h0"), and AB ("abcab0ab0") are considered.
2) **Mining of MQRPW using the Algorithm:** MQRPW are obtained from the weight sequences using the algorithm mentioned in Section3. Therefore, for the edges AC, BC, CD, AB, the MQRPW are "ab0", "00", "0" and "ab0" respectively. Further, the patterns containing all 0's are removed and we are left with AC and AB.
3) **Determination of Evolution graphs on Weight:** All the edges following the same MQRPW are combined to form evolution graphs on weight.

Therefore, the edges AC and AB are combined to form the evolution graph (Fig. 6).

**Output:** Evolution graphs on structure (MQRPS) and evolution graphs on weight (MQRPW).

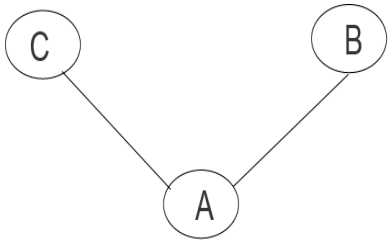


Fig.5. Evolution graph for MQRPS "110"

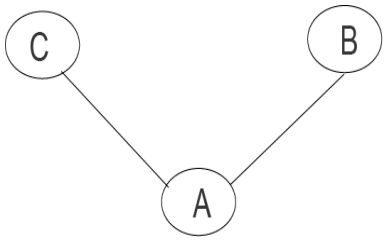


Fig.6. Evolution graph for MQRPW "ab0"

Now, we assess the complexity of the method proposed in Subsection 4.3.

### *4.3 Complexity analysis*

We try to analyze the computational complexity of the method step by step. Let there be T timestamps, and $E_1$,$E_2$, ....$E_t$ be the sets of edges in the corresponding timestamps of the dynamic network, respectively. Let the total, i.e. $m_{total} = |E_1|+|E_2|+ ....+|E_t|$ and $m_{real} = |E_1 \cup E_2 \cup,... \cup E_t|$, and $m_{qregular}$ be the total number of quasi-regular edges. Since the method and the steps followed for each quasi regular pattern (occurrence, weight) are same, we analyze the complexity for a quasi-regular pattern in general and then obtain the total complexity by using $m_{qregular}$ edges. In step A, a summary graph of the dynamic network is created by reading edges from different timestamps, and hence time complexity is $O(m_{total})$. In step B, each edge is checked if it is infrequent. According to the algorithm, in any particular iteration of *sublen*, the value of *inlen* varies from 0 to *sublen*, and a regular expression is formed which is matched with the original string. The *gregexpr* engine tries to match starting from the position 0,1,2,3 and so on till last position in the input string str. It will first match a fixed number of characters which is equal to the sum of (*sublen-inlen*) and (*inlen*) and then will try to match the 2nd and the 3rd capturing group exactly (th-1) times. If the match is successful, the result is inserted into the list named test. This process goes on until the end of string

*str*, and therefore, it performs, a total of *{th* [(sublen - inlen)+(inlen)]*length(str)}* number of computations. In the next for loop in line 10, the length of the match found is calculated, and then in the innermost for loop, starting from 1 to the length of the longest match, the actual pattern is found using the substring function, thus worst case being the maximum length of pattern which is *len = length (str) / th*. This leads to the overall complexity as *th*[(subleninlen)+( inlen)*length(str)* blength (str) / thc*. This is repeated a maximum of *len* times and thus the overall complexity is, *{th*[(sublen - inlen) + (inlen)*length(str)* [length (str) / th]2 }.* Thus the overall time complexity for the algorithm is O(length(str)$^3$) which is same as O(T). The process is repeated for all the $m_{real}$ edges, and so the time complexity of Step C is O($m_{real}$*$T^3$). In Step D, each quasi regular edge is visited only once, so the time complexity is O($m_{qregular}$). The process is again repeated for calculating MQRPW but with less input size. From the above analysis, we obtain the net time complexity as O($m_{total}$ + $m_{real}$*$T^3$).

## V. Parameters

The purpose of taking different parameters is to produce a better understanding of the nature of evolution graphs obtained by viewing it in different ways. We have used the following parameters for the analysis of the evolution graphs formed:

*1. Modularity*:

It is designed to measure the strength of division of a network into modules. Networks with high modularity have dense connections between the nodes within modules but sparse connections between nodes in different modules. The value of the modularity lies in the range [−1/2, 1). It is positive if the number of edges within groups exceeds the number expected on the basis of chance and its value has been calculated using "walk trap community detection" algorithm [16].

*2. Cliques:*

A clique [17] is a subset of vertices of a graph in which every two vertices are connected by an edge. Therefore, cliques can be used as a method for acknowledging structural characteristics of an entity or its behavior in a network.

*3. Intersection of two graphs:*

It outputs the common elements [18] between two graphs. A new graph, M, is formed by calculating intersection between the two graphs.

*4. Centrality measures:*

Centrality of nodes is an important factor to understand their embedding in a network. Centrality helps to identify the most important nodes in a graph, whereby this analysis is helpful for answering a variety of questions such as:

- Which two or three people must we select in order to pass a message to everyone in a network?
- Which node is most central or who is the most influential person(s) in a social network?
- Which are the key infrastructure nodes in the internet or urban networks?

There are various centrality measures for analyzing different aspects of structural position in a network. In this paper, we have considered most commonly used centrality measures. They are:

*A. Degree Centrality* [19]:

It is defined as the number of direct links/ties/connectivity's a node has. It can be used to analyze a person having the maximum number of connections.

*B. Closeness Centrality* [19]:

Closeness is based on the length of the average shortest path between a vertex and all vertices in the graph. Closeness score of nodes which connect with other nodes through many intermediate nodes is nearer to zero, whereas it is higher for nodes which are near the center of local cluster. It has been calculated using the formula given by Freeman. It helps in finding nodes which will be close to many nodes of any community within an evolution graph or which can communicate quickly with other nodes in a graph.

*C. Betweenness Centrality* [20]:

The betweenness centrality is equal to the number of shortest paths from all vertices to all others that pass through that node. A node lying on high number of paths between other nodes in a cluster of an evolution graph will have a very high betweenness centrality score. It has been calculated using the formula given by Brandes and it helps in identifying the nodes which are relatively more connected as compared to others.

*D. Eigenvector Centrality* [21]:

Eigenvector centrality is calculated by assessing how well connected a node is to the parts of the network with the greatest connectivity. Nodes with high eigenvector scores have many connections, and their connections have many connections and out to the end of the network. However, a node receiving many links does not necessarily have a high eigenvector centrality (it might be that all linkers have low or null eigenvector centrality). Moreover, a node with high eigenvector centrality is not necessarily highly linked (the node might have a few but important linkers). It has been calculated using the exact same formula given by Bonacich.

## VI. Experimental Analysis

We conduct the experiments in R on a system having 2.6 Ghz dual-core Intel core i5 processor with 8 GB of RAM and OS X Yosemite. To establish the robustness of the proposed method, we perform experiments on one real world dataset and one synthetic dataset.

 

*6.1 Datasets:*

***Enron E-mails dataset*:** The Enron email dataset is a publicly available dataset. It contains email sent to and by the employees of now defunct Enron Corporation in which the vertices stand for employees and edges stand for the emails sent/received. In this paper, we have used a cleaner version of dataset [22] which consists of emails within 28 months, from December 1999 to March 2002 and has 2352 vertices and 23536 edges. An additional weight has been assigned to each edge randomly which is represented by the character ' *a'* if the number of emails sent/received between 1 to 10, 'b' if the number of emails sent/received between 11 to 20 and so on.

***Artificially generated E-commerce co-purchasing weighted network*:** We have generated a weighted dynamic network dataset containing information for 100 timestamps about the products that are bought together. There is an additional weight associated with such products which basically denotes the number of times they are bought together at a particular timestamp. The entire dataset consists of 100 different files where each file represents the dynamic network at a particular timestamp. Every file consists of three columns in which the first two columns denote the Product ID's whereas the third column denotes the weight. The weight is represented by a character *(a,b,c,d,e,f,g,h,I,j)* where *a* is used if the value of weight lies between 1 to 50, b is used if weight lies between 51 to 100 and so on. The dataset generated contains a total of 67,424 edges over a total of 5645 products.

*6.2 Observations and Result Analysis:*

1.) **Enron Email dataset:** In order to prove the usefulness of the proposed method, the experiments on Enron Email dataset has been conducted using a method [14] which ignores noise and compared the results with those obtained from the proposed method. The results obtained are summarized in Table 1.

Table 1. Comparison of the proposed method with a method in (Gupta et al., 2014)

| Measures | Regular pattern Miner (Doesn't handle noise) | Proposed method (Handles noise up to 1 position) |
|---|---|---|
| Number of Input edges(dynamic graph) | 23536 | 23536 |
| Number of edges following a pattern on structure | 1593 | 4807 |
| Number of MQRPS | 56 | 154 |
| Number of MQRPS followed by more than 3 edges | 33 | 101 |

Table 2. Result of the proposed method on Enron Email dataset

| | |
|---|---|
| Total number of input edges | 80237 |
| Total number of edges having presence at greater than *s (3)* timestamps. | 23536 |
| Total number of MQRPS | 154 |
| Total number of MQRPS followed by greater than 2 edges (Evolution graphs) | 101 |
| Total number of edges in evolution graphs of MQRPS | 4807 |
| Total number of MQRPW | 194 |
| Total number of MQRPW followed by greater than 2 edges (Evolution graphs) | 114 |
| Total number of edges in evolution graphs of MQRPW | 4322 |

It can be seen that in the Enron Email dataset, the patterns in as many as 3214 (4807-1593) edges would have been lost due to the presence of noise at just one position. The mining of these patterns and edges helps in the further analysis. The results of the proposed method obtained on Enron Email Dataset are summarized in Table 2.

Also, it can be seen that instead of taking 23536 edges for mining MQRPW, using claim only 4807 edges are considered. Therefore, the claim will be extremely helpful in dealing with big data as well. The running time for mining such patterns with and without the use of claim in Enron-email dataset is presented in Fig. 7.

It is clearly evident from Fig. 7 that the claim has been extremely beneficial to the present case as it has reduced the running time of mining MQRPW after mining MQRPS by a large amount.

 

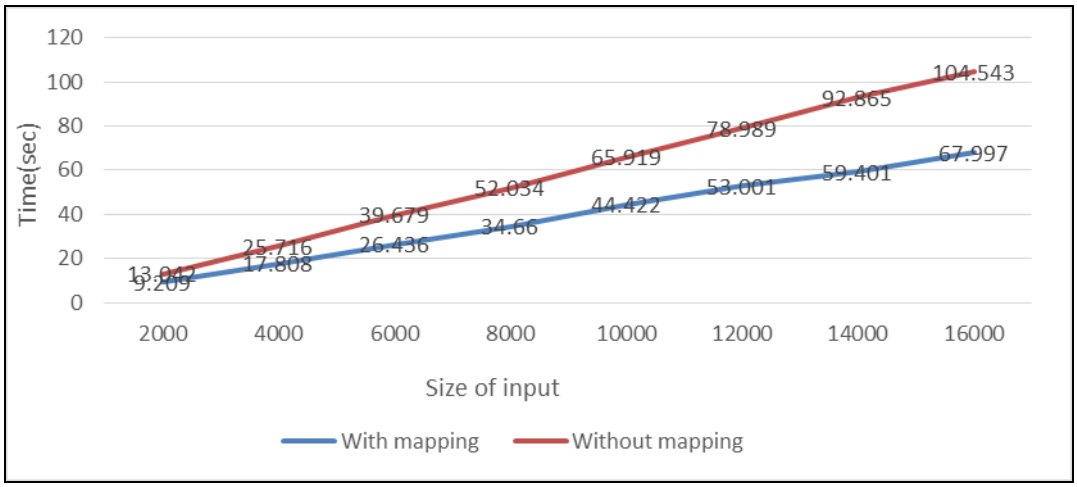


Fig.7. Running time analysis with and without claim

These evolution graphs (MQRPS) have been numbered from 1-101 for further analysis, which is as follows:

*A. Modularity:*

On calculating the modularity score of evolution graphs on structure for MQRPS (1-101) and plotting them, we have found an interesting relationship. It can be seen from the plot (Fig. 8) that almost all the patterns have their modularity score on the higher side (> 0.7). This leads to the conclusion, that random people who follow the same pattern have a very high tendency of forming communities with each other which might be due to their personal relationships, or it might be due to people belonging to the same department.

*B. Cliques:*

A clique refers to a group of entities which are all connected to each other. In this paper, we have on focused on largest cliques. A clique is largest if there is no other clique including more vertices. On plotting the size of the largest cliques against the evolution graph on structure for MQRPS (1-101), we get Fig. 9. It can be seen that the evolution graphs of some patterns have a largest clique of size 3. And therefore, it can be said that those 3 users (nodes which are part of the clique) demonstrate the possibility of a very high degree of a professional relationship.

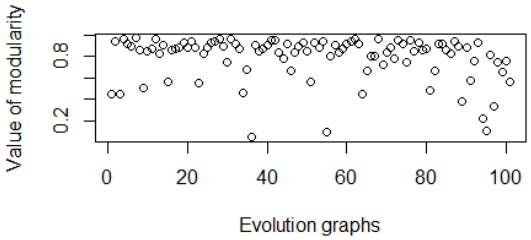


Fig.8. Plot of modularity against evolution graphs on structure

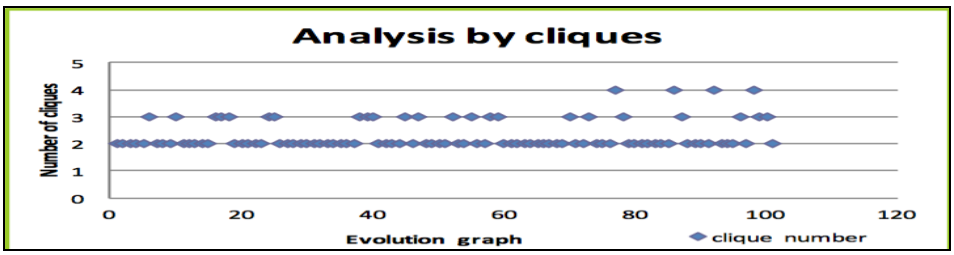


Fig.9. Number of cliques in evolution graphs on structure

*C. Centrality:*

Based on four different centrality measures, we try to discern most central node of a particular evolution graph on structure. The analysis involves extracting the number of nodes that have the maximum value for the centrality measures. The evolution graphs on structure when analyzed with the 4 centrality measures (degree, Eigen vector, closeness, betweenness) resulted in:

- Exactly same number of nodes having maximum value of the 4 measures.
- If the nodes are different, even then there is a single node which has maximum value for majority of the measures.

In other words, for example, we have found that in evolution graph on structure for MQRPS "1101", the nodes having the maximum degree appeared to be exactly same as the nodes that have the highest closeness and betweenness and are a subset of the nodes that have the highest value of Eigen vector in the entire evolution graph. However, there is a difference in 11 evolution graphs which provided different values and number of nodes possessing maximum values of the described

centrality measures.

So, if some information is to be passed on to a group of people, then it can be directly sent to the most central node(s) only. On plotting a graph of the number of nodes that have the highest value for the centrality measures in the evolution graphs on structure (number of edges > 100), we get the plot (Fig. 10).

As is evident from Fig. 10, each centrality measure yields one node in the evolution graph having the maximum value for that measure. For example, in the evolution graph corresponding to pattern number 10, the node which has the maximum value of degree is same as the node that has the maximum value of betweenness, and is same as the node that has the maximum value of closeness, and finally, same as the one which has the maximum value of eigenvector. Such node (most central) might be the head of a department. Therefore, with the help of these 4 measures, we are able to narrow down on one single node which is most central for evolution graphs on structure having edges greater than 100.

The intersection parameter has not been used on the Enron Email Dataset since the available dataset is not large enough to divide it and obtain meaningful results.

Similarly, the application of the three parameters used above on the evolution graphs obtained from MQRPW in Enron Email Dataset can help in the analysis of user's habits in terms of the number of emails that they send/receive.

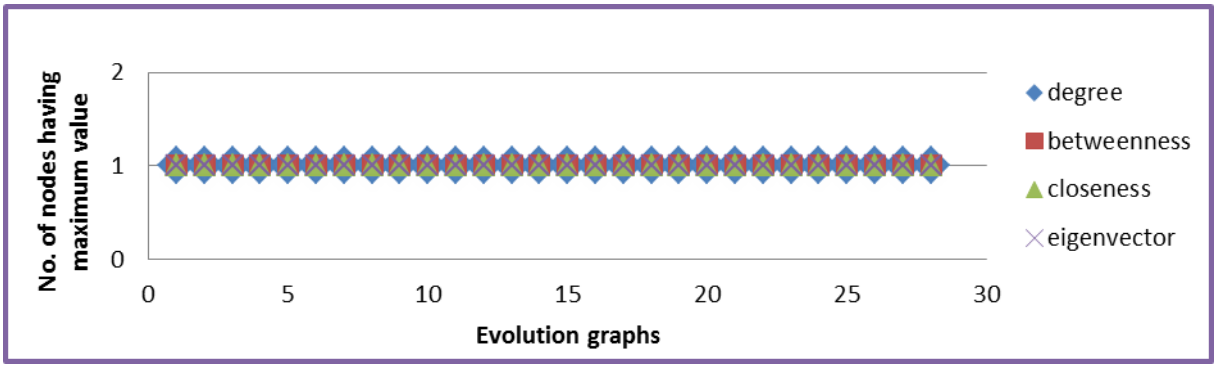


Fig.10. Number of nodes having maximum values for each centrality measure in each evolution graph on structure

Table 3. Result of the method on Synthetic dataset

| | |
|---|---|
| Total number of input edges | 67424 |
| Total number of edges having presence at least *s* (3) timestamps | 7046 |
| Total number of MQRPS | 536 |
| Total number of MQRPS followed by greater than 2 edges (Evolution graphs) | 145 |
| Total number of edges in evolution graphs of MQRPS | 5928 |
| Total number of MQRPW | 386 |
| Total number of MQRPW followed by greater than 2 edges (Evolution graphs) | 219 |
| Total number of edges in evolution graphs of MQRPW | 3090 |

**2.) E-commerce co-purchasing weighted network:** The results of the proposed method on the synthetic dataset described above are summarized in Table 3.

Each of the patterns (MQRPS) represents a sequence by which the products are bought together and they have been numbered from 1-145 for further analysis, which is as follows:

*A. Modularity:*

The value of modularity for some of the evolution graphs are summarized in Table 4. These evolution graphs correspond to the MQRPS which consist of 98.24% of all edges.

Table 4. Values of modularity in evolution graphs of MQRPS obtained

| MQRPS | Number of edges in evolution graph | Value of modularity |
|---|---|---|
| 11111111111111111111111111111111 | 3860 | 0.60 |
| 011111111101111111111 | 520 | 0.83 |
| 101111111110111111111 | 508 | 0.83 |
| 110111111111011111111 | 500 | 0.82 |
| 111011111111101111111 | 484 | 0.84 |
| 111101111111110111111 | 482 | 0.84 |
| 111110111111111011111 | 478 | 0.84 |
| 111111011111111101111 | 460 | 0.83 |

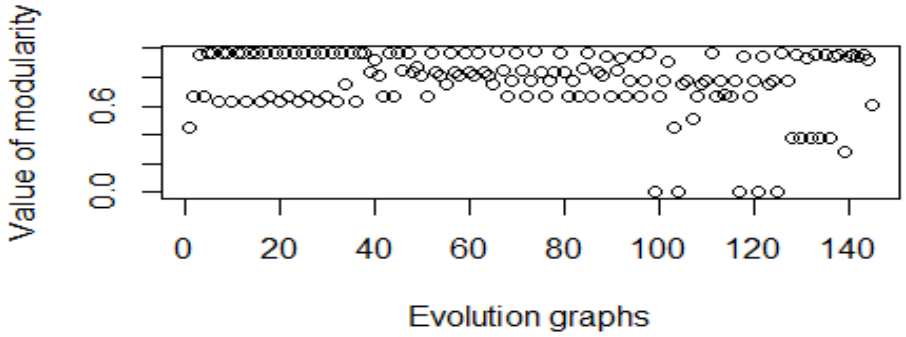


Fig.11. Plot of modularity against evolution graphs on structure

Further, we plot the value of modularity of each evolution graph for MQRPS (1-145) and obtain Fig. 11 where abscissa shows the pattern number for the synthetic graphs.

It can be inferred from the plot that almost all the evolution graphs have their value of modularity ranging from [0.5, 0.99] which shows the presence of high community structure in products following a particular pattern (MQRPS).

Now, as has been mentioned in the article [23], a high value of modularity depicts higher presence of communities and thereafter each community has the possibility of containing products belonging to a particular category/genre such as books, toys, music, etc.

It means that the product belonging to different categories that are sold by a particular pattern can be easily identified by an appropriate mapping of products with the respective product-id. This can help in setting up of offers and promotional schemes depending upon the patterns in which the products of different categories are being sold. This can help to boost profits.

Fig. 12 shows a sample evolution graph containing 65 edges for the MQRPS (110101010110). The different communities in the graph have been shown with the help of different colors and the value of modularity for this is 0.95.

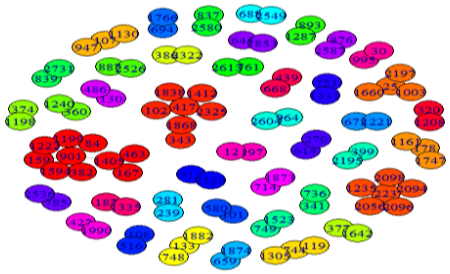

Fig.12. Communities in a MQRPS (110010101)

### *B. Cliques:*

For a particular evolution graph on structure, cliques help us to identify the items that are most frequently bought together. This can help in providing appropriate recommendations when user wishes to buy a particular product. Table 5 shows the size of the largest cliques obtained in the largest evolution graphs which contains about 83.3% of all the edges in the dataset.

Table 5. Cliques in evolution graphs of MQRPS

| Pattern in structure | Number of edges in evolution graph | Size of the largest clique |
|---|---|---|
| 11111111111111111111111111111111 | 3862 | 6 |
| 0111111111101111111111 | 520 | 3 |
| 1011111111110111111111 | 508 | 3 |
| 1101111111111011111111 | 500 | 3 |
| 1110111111111101111111 | 484 | 3 |

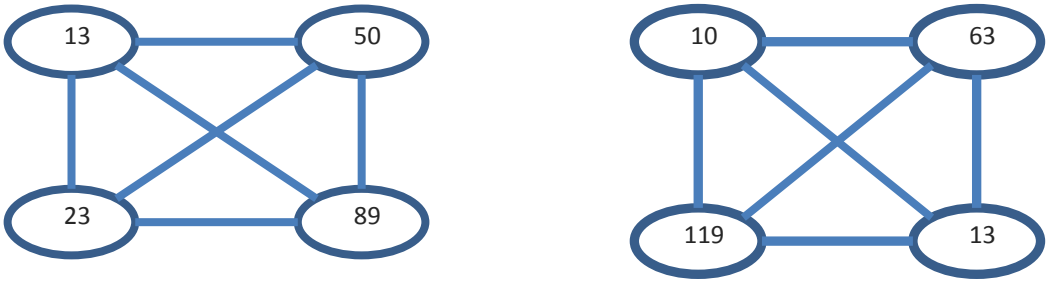


Fig.13. Largest Cliques in evolution graph on structure for MQRPS “11111111111111111111111111111111”

As an example, consider Fig. 13 which shows two of the largest cliques found in the evolution graph for MQRPS "1111111111111111111111111111111111" having 3862 edges.

On analyzing the dataset, we have obtained largest cliques having as many as 6 products in a particular pattern. From all these, we can infer that the products which form a clique, not only follow a particular pattern in which they are sold over time, but also are bought simultaneously with each other. This gives us much insight into people's habits.

*C. Intersection:*

We have divided the given dataset of 100 timestamps into 3 segments and use the proposed method on each of the 3 segments separately. Most of the patterns we obtain in the three segments are similar to each other, apart from the nodes following them. For example, a MQRPS, "111111" has a different set of nodes (i.e. different products) in each of the three segments. But, there are some nodes which are common across all the segments. These nodes have been found out by calculating the intersection of the evolution graphs of the same patterns (MQRPS) over the 3 segments. Through this, we are able to identify the set of products which are sold consistently with the same MQRPS over a time period. And therefore, this can be effectively utilized for predicting future trends and enhancing the profits over time. The results of intersection on some of the evolution graphs of a particular pattern over the three segments are described in Table 6 as follows:

Table 6. Intersection statistics on the three segments of E-commerce Dataset

| Pattern | Vertices in Part1 | Vertices in Part2 | Vertices in Part3 | Common vertices |
|---|---|---|---|---|
| 11111111111 | 4653 | 5073 | 5121 | 2672 |
| 1111111111 | 68 | 149 | 164 | 7 |
| 11111111 | 104 | 192 | 230 | 13 |
| 1111111 | 174 | 147 | 196 | 12 |
| 111111 | 179 | 166 | 202 | 14 |

*D. Centrality:*

We have analyzed the centrality on the basis of 4 parameters. In this case, it helps us to identify the product that is most frequently bought with all the other products over time. That product, in other words, will be the most central among the products that are sold with a similar pattern. As an example, consider the evolution graph in Fig. 14 for pattern (MQRPS) "10111111111".

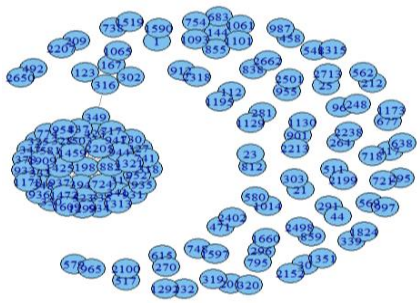

Fig.14. Evolution graph on structure for MQRPS "10111111111".

Here, the product represented by the node 198 is the most central. That is, it is most often bought with all the other products that are sold in a similar pattern (10111111111) and that there is a very high tendency that a person buying product numbered 198 will land up buying products linked to it directly. Therefore, the identification of such a product coupled with the information about the pattern that it follows while being sold, will help to increase the profits by offering, for example sales on the one who is most central.

On plotting the number of nodes having maximum values of the 4 centrality measures described in Section5 in the evolution graphs (having number of edges > 100), we have obtained the graph in Fig.15. wherein the abscissa shows the pattern number for the evolution graphs.

Different colors and symbols denote different centrality measures. For example, in the evolution graph for pattern numbered 9, there is only a single node having maximum value of eigenvector, betweenness and closeness, whereas, the number of nodes having maximum value of degree is 2. However, that single node which has the maximum value of the first three parameters has the maximum value of degree as well. And hence, we are able to narrow down on one single node having highest value for all parameters.

As evident from Fig.15, in almost all the evolution graphs, the parameters yield the same values for the number of nodes.

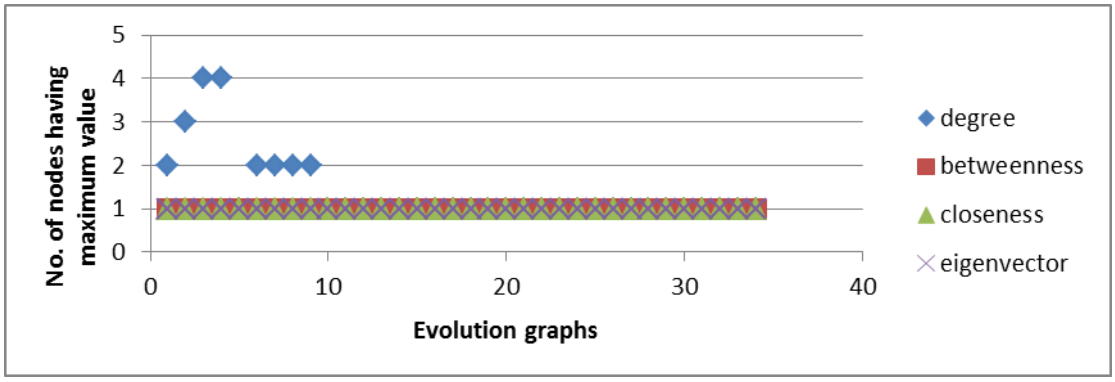


Fig.15. Number of nodes having maximum values for each centrality measure in some of evolution graphs obtained on structure

Similarly, the analysis of the evolution graphs obtained on the basis of weight can be analyzed using the above 4 parameters. Each of them hold a separate meaning and give insights into the quantities and amounts in which the products are sold. These insights can further help to boost the sales and profit from the products. Finally, the next section summarizes the paper through discussion and conclusion.

## VII. Discussion and Conclusion

### *7.1 Discussion*

We consider the evolving nature of the graphs and the fact that a pattern may start from any timestamp. So, in this paper, we present a method for finding informative patterns (MQRPS and MQRPW) in noisy dynamic networks. These patterns are extremely meaningful and provide us a better understanding of the complex networks under consideration. Then, the study of evolution graphs at microscopic level on the basis of four parameters has helped us to gain an insight into the nature of the patterns captured by the method. Each chosen parameter conveys its own meaning, helps us to view the evolution graphs in distinct way and thus makes unique conclusion(s). Modularity tells us about the high or low community structure in a particular evolution graph. Clique is used to extract that (those) community (communities) which is (are) strongly connected. Then, centrality parameter discerns most central node(s) which is (are) the part of some community. Intersection gives the edges (linking two nodes) which are consistently following the same pattern over a time period. Hence, these parameters are used to yield vital piece of information. There have been previous works [6] which have solved the problem of mining noisy patterns in dynamic networks, however, they focused on mining patterns only in the structural domain and without involving any comprehensive analysis. Whereas, the current article presents a holistic and a novel framework for mining noisy patterns across all the dimensions (structure, weight and direction) and also analyses the patterns obtained to derive real world insights. This kind of method is useful particularly to those organizations who deal with large amount of data. Though in the recent days, the data collection practices have become extremely accurate, however, there is always a possibility of noise creeping in which diminishes the purpose of the analysis and patterns obtained from such datasets. For an organization who recognizes how noise can be detrimental to their analysis, there exists two methods of action. Either, they can focus on removing noise from the data first and then conducting an analysis, or, they can instead strive to develop algorithms which take these noisy details into account. And, the framework presented in this paper focusses on the latter one.

### *7.2 Conclusion*

- With the ever growing size of the data, noise finds its entry into it. Hence, the scope of this paper is to provide a method which mines Maximal Quasi Regular patterns on Structure and Maximal Quasi Regular Patterns on Weight in any weighted dynamic network dataset and the detailed examination of the evolution graphs formed. We convert the weighted dynamic network into a summary graph, obtain the occurrence sequence/weight sequence of the edges on the basis of its presence and absence and apply the proposed algorithm on the sequence for pattern discovery. MQRPS and MQRPW so obtained not only help us in making future predictions but also provide us with some crucial hidden information which might have been lost. Moreover, the addition of analysis helps to dig for more knowledge they (patterns) contain and thus, increases our understanding of the network. Hence, summarizing, the paper deals with the following questions: Why is it important to consider the presence of noise?
- Has noise been handled before? If yes, how it is different from the current direction of research?
- What's the use of patterns that have been mined?
- How an analysis on the patterns obtained can be used for the benefit of real world organizations?

To demonstrate the same, we have evaluated the proposed method on two datasets followed by the analysis. This has provided beneficial inferences. Further, by restructuring the method, we can mine MQRPS and MQRPW in a very less time. Therefore, it is clearly evident that it is able to deal with large scale networks to

 

extract the beneficial knowledge hidden in them.

Dynamic graph mining is relatively a new research topic and owes to a large number of applications. Currently, we have focused on finding patterns having noise at one position and analyzing them. In the current work, we have focused on patterns having noise only at single positons. However, there exists a possibility that noise might be present at more than a single position. Also, there is also a possibility of the presence of a relationship between the parameters that have been chosen for analysis. This relationship can help us to predict the values of some variables, given some other values. These are some of the future research areas that we intend to work in future.

## REFERENCES


[1] Manjeet Samoliya, Akhilesh Tiwari, "On the use of Rough Set Theory for Mining Periodic Frequent Patterns", International Journal of Information Technology and Computer Sciences (IJITCS), Vol. 8, No. 7, pp.53-60, 2016. DOI: 10.5815/ijitcs.2016.07.08

[2] P. K. Desikan and J. Srivastava, "Mining Temporally Changing Web Usage Graphs", In Proc. of the 6th International Workshop on Knowledge Discovery on the Web (WEBKDD'04), 2004, pp. 1-17.

[3] K. M. Borgwardt, H. P. Kriegel, and P. Wackersreuther, "Pattern Mining in Frequent Dynamic subgraphs", In Proc. of the 6th International Conference on Data Mining (ICDM '06) , 2006, pp. 818-822.

[4] M.Lahiri, Y. Tanya and B.Wolf , "Mining Periodic Behavior in Dynamic Social Networks", In Proc. of the 8th IEEE International Conference on Data Mining(ICDM'08), 2008, pp.373-382.

[5] C. Robardet, "Constraint-based Pattern Mining in Dynamic Graphs", In Proc. of the 9th IEEE International Conference on Data Mining(ICDM'09), 2009, pp. 950-955.

[6] C.H.You, L.B.Holder and D.J.Cook, "Learning patterns in the dynamics of biological networks", In Proc. of the 15th International Conference on Knowledge Discovery and Data Mining (KDD'09), 2009, pp. 977-986 .

[7] G.Qin, L.Gao,J.Yang and J.Li "Evolution Pattern Discovery in Dynamic Networks", In proc. of the International conference on signal Processing , Communication and Computing(ICSPCC 2011) 2011, pp.933-938.

[8] Y. Yang, J. X. Yu, H. Gao, J. Pei and J. Li, "Mining most frequently changing component in evolving graphs", In Journal of World Wide Web, Volume 17 Issue 3, PP. 351-376, May 2014.

[9] R. Jin, S. McCallen, and E. Almaas, "Trend Motif: A Graph Mining Approach for Analysis of Dynamic Complex Networks", In Proc. of the 7th International Conference on Data Mining (ICDM '07), 2007, pp. 541-546.

[10] F. Moser, R. Colak, A. Rafiey and M. Ester, "Mining Cohesive Patterns from Graphs with Feature Vectors", In Proc. of the International Conference on Data Mining (SDM '09), 2009, PP. 593-604.

[11] Y. Zhou, H. Cheng, and J. X. Yu, "Graph Clustering Based on Structural/Attribute Similarities", In Journal of the VLDB Endowment, Volume 2 Issue 1, PP. 718-729, August 2009.

[12] P. Bogdanov, M. Mongiovi and A. Singh, "Mining Heavy Subgraphs in Time-Evolving Networks" In Proc. of the International Conference on Data Mining (IEEE ICDM 2011), 2011, pp. 81-90.

[13] A. Prado, M. Plantevit, C. Robardet and J. Boulicaut, "Mining Graph Topological Patterns: Finding Co ariations among Vertex Descriptors", In Journal of the IEEE Trans. Knowl. Data Eng., Volume 25, Number 1, pp. 2090 - 2104 January 2013.

[14] A. Gupta, H. K. Thakur and P. Kishore, "Mining Regular Patterns In Weighted Directed Networks", In Proc. of 13th International Conference Of Information Technology (ICIT '14), 2014, pp. 215-220.

[15] A. Gupta and H. K. Thakur., "A novel method to determine regular pattern in edge labelled dynamic graphs," In Proc. of 7th International Conference on Data Mining and Warehousing, (ICDMW '13), 2013, PP. 39-47.

[16] P. Pons and M. Latapy, "Computing Communities in Large Networks Using Random Walks", In Journal of Graph Algorithms and Applications, Volume 10, Number 2, PP. 191-218 , 2006.

[17] D. Eppstein, M. Loffler, and D. Strash, "Listing all maximal cliques in sparse graphs in near-optimal time", In Proc. Of 21st International Symposium on Algorithms and Computation (ISAAC '10), 2010, PP. 403-414.

[18] R. D. Luce and A. D. Perry, "A method of matrix analysis of group structure", In Journal of Psychometrika, Volume 14, issue , PP. 95-116, 1949.

[19] L. C. Freeman, "Centrality in Social Networks Conceptual Clarification", In Journal of Social Networks, Volume 1, issue 3, PP. 215-239, 1979.

[20] U. Brandes, "A Faster Algorithm for Betweenness Centrality", In The Journal of Mathematical Sociology, Volume 25, issue 2, PP. 163-177, 2001.

[21] P. Bonacich, "Power and Centrality: A Family of Measures", In American Journal of Sociology, Volume 92, issue 5, PP. 1170-1182, 1987.

[22] L. Tang, H. Liu, J. Zhang and Z. Nazeri, "Community Evolution in Dynamic Multi-Mode Networks", In Proc. of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, 2008, PP.677-685.

[23] A. Clauset, M. E. J. Newman and C. Moore, "Finding community structure in very large networks", In Journal Physical Review E, Vol. 70, Number 6. Aug. 2004.


## Authors' Profiles

**Hardeo Kumar Thakur** is a Teaching cum research Fellow in Netaji Subhas Institute of Technology (NSIT). NSIT is affiliated to Delhi University, India. He is pursuing his Ph.D. in the area of dynamic graph mining.

**Dr. Anand Gupta** is an Associate Professor in Netaji Subhas Institute of Technology (NSIT). NSIT is affiliated to Delhi University, India. His research interest include Data Mining, database system, Image Processing and Information Retrieval.

 

**Bhavuk Jain** is an undergraduate student in Netaji Subhas Institute of Technology (NSIT). NSIT is affiliated to Delhi University, India. He has keen interest in data mining.

**Ambika** is an undergraduate student in Netaji Subhas Institute of Technology (NSIT). NSIT is affiliated to Delhi University, India. She has keen interest in data mining.

**How to cite this paper:** Hardeo Kumar Thakur, Anand Gupta, Bhavuk Jain, Ambika,"Mining Maximal Quasi Regular Patterns in Weighted Dynamic Networks", International Journal of Information Technology and Computer Science(IJITCS), Vol.9, No.4, pp.48-62, 2017. DOI: 10.5815/ijitcs.2017.04.07

 