

# Trend Motif: A Graph Mining Approach for Analysis of Dynamic Complex Networks

Ruoming Jin, Scott McCallen

Department of Computer Science, Kent State University, Kent, OH, 44241  
{jin,smccalle}@cs.kent.edu

Eivind Almaas \*

Microbial Systems Group, Biosciences & Biotechnology Division,  
Lawrence Livermore National Laboratory, Livermore, CA 94551-0808  
almaas@llnl.gov

## Abstract

*Complex networks have been used successfully in scientific disciplines ranging from sociology to microbiology to describe systems of interacting units. Until recently, studies of complex networks have mainly focused on their network topology. However, in many real world applications, the edges and vertices have associated attributes that are frequently represented as vertex or edge weights. Furthermore, these weights are often not static, instead changing with time and forming a time series. Hence, to fully understand the dynamics of the complex network, we have to consider both network topology and related time series data.*

*In this work, we propose a motif mining approach to identify trend motifs for such purposes. Simply stated, a trend motif describes a recurring subgraph where each of its vertices or edges displays similar dynamics over a user-defined period. Given this, each trend motif occurrence can help reveal significant events in a complex system; frequent trend motifs may aid in uncovering dynamic rules of change for the system, and the distribution of trend motifs may characterize the global dynamics of the system. Here, we have developed efficient mining algorithms to extract trend motifs. Our experimental validation using three disparate empirical datasets, ranging from the stock market, world trade, to a protein interaction network, has demonstrated the efficiency and effectiveness of our approach.*

## 1. Introduction

The majority of recent studies of complex network have focused on characterizing the topology, or the change in topology, of complex networks [9, 2, 12]. However, in many real world applications, weights are often associated with the vertices or edges of the network. These weights are typically changing with time, thus forming a time series for each vertex and edge. Thus, knowledge of the network topology, paired with the time series data, provides

a comprehensive global picture of a dynamically changing system. Generally speaking, if each vertex of the network has a weight, we refer to it as a vertex-weighted network or graph, and if each edge of the network has a weight, we refer to it as an edge-weighted network or graph. Note that, a network can be both vertex-weighted and edge-weighted.

In the following, we will consider several systems that can naturally be represented as weighted networks, and where the system dynamics are captured in time series of weights.

**Financial Market:** In the financial market, companies interact with each other and form various relationships, typically including competitor, producer-consumer, ownership, etc. A complex network can be built to represent the interactions of all the companies in the financial market, where each company corresponds to a vertex, and the relationship between two companies corresponds to an edge. Each vertex (company) can be weighted by the corresponding time series of stock value. Since the price change of each stock is often correlated with or determined by the price changes of companies with which it has close relations, the network representation provides a framework to simultaneously analyze the dynamics of an entire financial market.

**Protein Interaction Network:** In the recent era of systems biology, new experimental approaches have been developed with the ability to rapidly measure thousands of molecular interactions. Among the most heralded are the so-called high-throughput techniques to characterize all pairs of proteins with the ability to physically interact. It has become customary to represent the resulting datasets as networks, where each vertex corresponds to a protein and two vertices are connected by an edge if the corresponding proteins can bind. In addition, the high-throughput microarray technology allow biologists to measure the distribution of gene products at different conditions and different time points. Thus, associating a time series from the microarray experiment for each protein provides a more comprehensive picture of the dynamically changing system

---

\*EA's work was performed under the auspices of DoE by University of California, LLNL under Contract W-7405-Eng-48, with support from the LDRD office

inside a cell.

While we have focused on the possibility that the network weights will change with time in response to a system's dynamic processes, the topology of the underlying network may change as well. However, for many systems the typical time scale for weight dynamics is significantly shorter than that of the changes in the network topology. Consequently, it is reasonable to consider the network as a static entity. We note that, while a network with time-varying weights contains significantly more information about the system, few methods have been devised that leverage this information. Specifically, scientists would like to know what are the basic rules that govern the evolution and changes of the complex system, and how two different dynamic systems can be compared.

### 1.1 Our approach

Our approach to analyze the dynamic complex network starts from local dynamics. It is based on the observation that the weight change of a vertex in a complex network is rarely an isolated event. They are often strongly correlated with, or possibly determined by, the changes occurring in its network neighbors. Similar observation can be made for the edges as well. For instance, in the stock market, the increase of the Intel stock price is likely correlated with the increase (or decrease) of AMD's stock, and both correlate with the stock price of PC producers such as HP and Dell. Similarly, in the protein interaction network, a biological process is very likely to result in the co-changes of several related proteins [14]. In other words, synchronized changes of weights over closely related vertices or edges can serve as a good indication of (local) dynamics or the evolution of a system.

The central theme of this paper is the introduction and discovery of *trend motifs*, which target putative patterns of changes for a group of closely related entities. Given a weighted (undirected) complex network, a group of such entities corresponds to a set of connected vertices. A possible pattern of change (*a trend motif occurrence*) is a set of connected vertices associated with a time span where the time series of each vertex displays a consistent *trend*. Here, we focus on two types of trends: the first corresponds to a steady increase in the time series, and the second corresponds to a steady decrease (see Section 2 for the formal definition). Consequently, a putative pattern is likely to correspond to a major event, or a sequence of events, occurring in the system. Therefore, extracting such patterns can help scientists identify such events, which often are hidden in large amounts of data.

Further, we define a *frequent trend motif* as a putative pattern which are over-represented in the complex network. Frequent trend motifs can help reveal the underlying mechanism governing the dynamics. For instance, a line subgraph with each vertex showing increase may correspond to a cascade in the system, and a clique subgraph with some vertices showing increase with others showing a decrease may indicate these changes are strongly correlated. Finally, we note that the distribution of trend motifs can be used

to categorize the dynamic networks, as we can expect that different types of networks will tend to have different types and distributions of such motifs [11].

## 2. Problem Definition

### 2.1. Trends and Trend Intervals

Given a graph  $G = (V, E)$  of  $N$  vertices  $V = \{v_1, v_2, \dots, v_N\}$  and a discrete time span  $[1, T]$ , the weight of vertex  $v_i$  is denoted as  $x_i(t)$ , for  $t \in [1, T]$ . Intuitively, we consider a trend as a subsequence of a time series that shows a consistent increase or decrease. Formally, we define an **increasing trend** as a subsequence  $[x_i(t_1), x_i(t_2), \dots, x_i(t_k)]$ , and  $t_j < t_{j+1}$ , of the time series  $x_i(t)$  with respect to two parameters  $\delta$  and  $\sigma$ , and it satisfies the following two conditions:

1. **Weight constraints:** for any time  $t_j$ ,  $x_i(t_{j+1}) - x_i(t_j) \geq \delta$ ,  $\delta > 0$ ;
2. **Step constraints:** for two time points in the subsequence,  $t_{j+1} - t_j \leq \sigma$ ,  $\sigma > 0$ .

Similarly, we can define the **decreasing trend**. If a subsequence satisfies one of these two definitions, either increasing or decreasing, it will simply be called a **trend**. We define a **maximal trend** as a trend that is not a subset of any other trend in the time series.

To facilitate our discussion, we define an interval  $[t_s, t_e]$  to be an **increasing trend interval** if it contains a trend  $[x_i(t_s), \dots, x_i(t_e)]$ , where  $t_s$  and  $t_e$  are the beginning and ending points of the trend, respectively. We use the notation  $[t_s, t_e] +$  to represent an increasing trend interval from start time  $t_s$  to end time  $t_e$ . Similarly, we can define the **decreasing trend interval** and denote it as  $[t_s, t_e] -$ . Thus, we define the maximal interval of trend as the longest time span in  $\{x_i(t)\}$  such that the values are consistently increasing or decreasing according the definition of a trend.

### 2.2. Trend Motif

Given the previous definitions, we can identify the trends which indicate the increasing and/or decreasing intervals for each of the vertices individually over the entire time series. A particularly interesting pattern, however, is observed when multiple trends occur simultaneously, and especially when they occur in nodes that are closely related through the network topology. To properly describe this phenomenon, we will formally introduce the concept of *trend motif occurrence*. Given the graph  $G = (V, E)$  and a subset of vertex  $V_s \subset V$ , let  $G(V_s)$  be the *induced subgraph* of  $V_s$  [3]. Mathematically, the induced subgraph of  $V_s$ ,  $G(V_s)$ , contains all the edges in  $E$  that have both ends in  $V_s$ .

**Definition 1 Trend Motif Occurrence:** Given a graph  $G$ , a *trend motif occurrence* of  $G$  is defined as the triple  $(V_s, [t_s, t_e], f)$  with  $(t_s < t_e)$ , where  $G(V_s)$  is a connected subgraph,  $f$  is a function  $f : V_s \rightarrow \{+, -\}$ , and  $[t_s, t_e] = [t_s^1, t_e^1] \cap [t_s^2, t_e^2] \cap \dots \cap [t_s^n, t_e^n]$ , where  $[t_s^i, t_e^i]$  is a maximal interval of trend for vertex  $v_i \in V_s$ , and  $n$  is the number of vertices in  $V_s$ .

Note that, if  $f(v_i) = +$ , the corresponding interval is increasing, otherwise  $f(v_i) = -$ . Basically, the function  $f$  labels each node of  $G(V_s)$ . We denote the labeled graph as  $G^f(V_s)$ . Additionally, we note that the interval  $[t_s, t_e]$  is the intersection of all maximal intervals of trend, and the intersection of the maximal intervals on  $[t_s, t_e]$  has to be nonempty. However, this intersection need not be a maximal trend interval on any of the vertices in  $V_s$ .

Based on the above definition, a very large number of trend motif occurrences may exist in a complex network for any time span. To reduce the number of motif occurrences, we introduce two parameters  $l$  and  $w$ , where  $l$  is the minimum interval length for a trend interval of each vertex in the motif occurrence and  $w$  is the minimal length for the intersection of the motif occurrence. We denote such a trend motif occurrence given  $l$  and  $w$  as  $(V_s, [t_s, t_e], f)(l, w)$ .

Finally, we introduce the concept of a *frequent trend motif*. Given two trend motif occurrences,  $(V_1, [t_s^1, t_e^1], f_1)$ , and  $(V_2, [t_s^2, t_e^2], f_2)$ ,  $V_1 \neq V_2$ , we refer to them as *equivalent* if their corresponding labeled induced subgraphs are isomorphic  $G^{f_1}(V_1) = G^{f_2}(V_2)$  [3]. In other words, there exists a one-to-one mapping between  $V_s$  and  $V'_s$ ,  $g : V_s \rightarrow V'_s$ , such that for any  $v_i, v_j \in V_s$ ,  $(v_i, v_j) \in E(G(V_s)) \Leftrightarrow (g(v_i), g(v_j)) \in E(G(V'_s))$ , and  $f_1(v_i) = f_2(g(v_i))$ . Here  $E(G(V_s))$  and  $E(G(V'_s))$  are the edge sets of the induced graph of  $G(V_s)$  and  $G(V'_s)$ , respectively.

**Definition 2 Frequent Trend Motif:** Given a support  $\theta$ , and two parameters  $l$  and  $w$ , if there are more than or equal to  $\theta$  distinct subset of vertices,  $V_1, \dots, V_t$ ,  $t \geq \theta$ , such that each set has at least a trend motif occurrence  $(V_i, [t_s^i, t_e^i], f_i)(l, w)$  being equivalent, then we refer to  $G_s^f(l, w, \theta)$  as a frequent trend motif, where  $G_s^f$  is a labeled subgraph that is isomorphic to  $G^{f_i}(V_i)$ ,  $1 \leq i \leq t$ .

Consequently, we can identify the following two related mining tasks.

1. **Extracting Trend Motif Occurrences:** Given two parameters  $l$  and  $w$ , we would like to find all the trend motif occurrences  $(V_s, [t_s, t_e], f)(l, w)$  in a graph  $G$ .
2. **Extracting Frequent Trend Motifs:** Given the support level  $\theta$  and the parameters  $l$  and  $w$ , we would like to find all the frequent trend motifs  $G_s^f(l, w, \theta)$ .

Clearly, these mining tasks are different from traditional subgraph mining tasks [5, 7, 8]. In the subgraph mining, the label of each vertex is known, and the major task is to enumerate all the possible candidate subgraphs, counting their number of occurrences. Here, each motif occurrence is dynamically determined by the time series data. In addition, each induced subgraph may correspond to different types of trend motif occurrences, as each vertex may display different trends at different time points. If we label each vertex with either  $+$  (corresponding to increasing trend intervals) or  $-$  (corresponding to decreasing trend intervals), a vertex can have both labels. These considerations show that mining trend motifs is a challenging task.

### 3. Algorithms

#### 3.1 Extracting Maximal Trend Intervals

Consider we have a time series  $X(t)$ ,  $t \in [1, T]$  and two parameters  $\delta$  and  $\sigma$ , we would like to extract all the maximal trend intervals from  $X(t)$ . A simple attempt will be to extract all the maximal trends first and then generate intervals defined by the starting time point and the end time point of these maximal trends. However, this approach can be rather computationally expensive. In addition, that the maximal trend intervals are not necessarily the *the maximal intervals of trends*.

---

##### Algorithm 1 *ExtractTrendIntervals*( $\delta, \sigma, X$ )

---

```

1:  $Q \leftarrow \emptyset$  { sorted list holds the last  $\sigma$  elements seen }
2: for  $t = 1$  to  $|X|$  do
3:    $inc(t) \leftarrow \min\{inc(q) | X(q) + \delta \leq X(t), X(q) \in Q\}$ 
     {  $inc(t)$  is the earliest time that  $[inc(t), t]$  is an interval of
       increasing trend }
4:    $dec(t) \leftarrow \min\{inc(q) | X(q) \geq X(t) + \delta, X(q) \in Q\}$ 
     {  $dec(t)$  is the earliest time that  $[dec(t), t]$  is an interval of
       decreasing trend }
5:    $Q \leftarrow Q \cup \{X(t)\}$  { add to the queue }
6:   if  $|Q| > \sigma$  then
7:      $Q \leftarrow Q \setminus X(t - \sigma)$  { remove the earliest }
8:     if  $\forall X(q) \in Q, inc(q) > inc(t - \sigma)$  then
9:        $interval[+] \leftarrow interval[+] \cup \{[inc(t - \sigma), t - \sigma]\}$ 
10:    end if
11:    if  $\forall X(q) \in Q, dec(q) > dec(t - \sigma)$  then
12:       $interval[-] \leftarrow interval[-] \cup \{[dec(t - \sigma), t - \sigma]\}$ 
13:    end if
14:  end if
15: end for
16: return  $interval$ ;

```

---

Here, we introduce an algorithm with a linear time complexity to simultaneously extract all maximal intervals of both increasing and decreasing trends in one pass through a time series. The *ExtractTrendIntervals* algorithm is shown in Algorithm 1. The algorithm maintains a list  $Q$  that stores the last  $\sigma$  seen elements at any time point  $t$ , from the given time series  $X$ . We iteratively look at each of the  $n$  elements in  $X_i$  (The for loop at line 2). The key of this algorithm is for each time point  $t$ , we will derive two values,  $inc(t)$  and  $dec(t)$ , which correspond to the intervals of increasing trend and decreasing trend, respectively. Essentially,  $inc(t)$  is the earliest time point which can form an interval of increasing trend together with  $t$ . This is equivalent to say that  $[inc(t), t]$  is the longest interval which contains an increasing trend starting from  $inc(t)$  and end at the current time point  $t$ . This is achieved by appending  $X(t)$  to all the elements in  $Q$ , which satisfy the *weight increasing constraint* (Subsection 2.1) between  $X(t)$  and  $X(q)$ ,  $q \in Q$ . Among those satisfying the constraint, we will choose the one which has the earliest time point forming the interval of increasing trends (Line 3). The processing for  $dec(t)$  is similar (Line 4). Finally, we note that the computational complexity of this algorithm is  $|X|\sigma$ .

### 3.2 Algorithm for Trend Motif Occurrence Discovery

One of the major difficulties in enumerating all the trend motif occurrences is the massive search space which spans both the topology dimension and the time dimension: any subset of connected vertices (topology dimension) combining with an interval (time dimension) can be treated as a candidate of trend motif occurrence. However, only a small portion of these candidates will become the true occurrences.

In order to efficiently discover these motif occurrences, we have to aggressively prune the search space. Here, we apply several techniques to reduce the search space. The first technique is based on the down-closure property: for any motif occurrence  $(V_s, [t_s, t_e], f)$ , any subset of connected vertices  $V'_s \subseteq V_s$  will correspond to a motif occurrence whose interval contains  $[t_s, t_e]$ . This will enable us to apply a depth-first search strategy to enumerate the motif occurrences from a single vertex to larger patterns. Secondly, we will enumerate all the motif occurrences which correspond to the same subset of vertices  $V_s$  and share the same labeling function  $f$  together. We refer to these motif occurrences as the same *type* of motif occurrences. This essentially enables us to enumerate the same type of motif occurrences in an efficient way.

Further, to reduce the cost of trend interval discovery, we extract all the maximal intervals of both increasing trends and decreasing trends for each vertex in the graph  $G$  using *ExtractTrendIntervals*. Then, for each vertex  $v$ , we record all the maximal intervals of increasing trends and decreasing trends (whose lengths are no less than  $l$ ) in  $v.interval[+]$  and  $v.interval[-]$ , respectively. Thus, we discover all the intervals of trends for each vertex only once. In addition, if a vertex does not have any interval, we remove them from the original graph  $G$ . This can help to reduce the search space.

---

#### Algorithm 2 *Build(Node v, Set N, Set E)*

---

```

1:  $N \leftarrow (N \cup Neighbor(v)) - E$  { $N$ : the set of vertices that
   can join to the occurrence;  $E$ : the set of vertices that are
   neighbors but cannot join to the occurrence;  $v$ : parent node;
    $Neighbor(v)$ : the vertices connect to  $v$ }
2: for each  $n \in N$  do
3:    $E \leftarrow E \cup \{n\}$ 
4:   for each  $k = \{+, -\}$  do
5:      $z \leftarrow Join(v.interval, n.interval[k], w)$  { $z$ : inter-
       vals of trends;  $w$ : intersection constraints}
6:     if  $z \neq \emptyset$  then
7:       create a new node  $v'$  for  $(n, z, k)$ 
8:       add  $v'$  to parent's ( $v$ ) children list
9:     end if
10:     $Build(v', N, E)$ 
11:  end for
12: end for

```

---

The key procedure in enumerating the trend motif occurrence is illustrated in the *Build* method (Algorithm 2),

which employs a depth-first search (DFS) strategy. All the occurrences are recorded in a tree structure. Each node of the tree corresponds to a vertex with certain trend, increasing (+) or decreasing (-). A path starting from the root to the given node  $v$  encodes one type of motif occurrence, and this node also records all the trend intervals of this type of motif occurrence in  $v.interval$ .

The *Build()* operation begins with a root node  $r$  that has no children, a set of neighbors  $N$  of the current motif occurrences and an excluded set  $E$  that records which vertices can no longer joined to the current occurrence. Both of the sets are initially empty ( $Build(r, \emptyset, \emptyset)$ ). In addition, we assume the root node  $r$  has all the vertex in  $G$  as its neighbors:  $Neighbor(v) = V(G)$ , and  $r.interval$  records only one interval  $[1, \infty]$ , suggesting it can intersect with any trend intervals without reducing their length.

For each time being invoked, the *Build()* procedure will find the new neighbors from the last vertex being added to the current motif occurrence (Line 1). Then, the algorithm iterates through the vertices in  $N$  and decides which of the remaining vertices can join with it (Line 2). For each vertex, we have to consider two cases, the increasing trend intervals and the decreasing trend intervals (Line 4). We compute the intersections of the intervals from the current motif occurrence with these new intervals (Implemented by *Join()* operation, which finds the common intervals of two sets of trend intervals). If a vertex with one type of trend intervals can join with current motif occurrence (the intersection set is not empty, Line 6), we will create a new node in the tree to record the vertex together with the trend intervals and we record this new node as a new child of the current motif occurrence (Line 7 – 8). Thus, a new type of motif occurrence is being discovered and stored. We will invoke *Build()* recursively to expand this new motif occurrence (Line 10). Note that in order to enumerate each motif occurrence only once, after we visit each vertex in the set  $N$ , we will add to the  $E$  list (Line 3). Therefore, this vertex will not be included in the motif occurrences which are being expanded later (Line 1).

### 3.3 Algorithm for Frequent Trend Motif Discovery

Before we set up to introduce the algorithm to find all frequent trend motifs, we will visit the *frequency* concept first. In the original Definition 2, any subset of vertices whose induced subgraphs are isomorphic to each other will be counted towards the frequency of a motif. However, a lot of them may have significant overlaps. A slightly different approach will only consider non-overlapped occurrences [8]. Here, we will allow any two occurrences share at most one vertex [13]. In other words, no edge can be shared between two occurrences for a given trend motif. Note that such a frequency concept will allow us to use the down-closure property for the motif enumeration. Given this, the major challenge in finding frequent trend motif is how to utilize the motif occurrence tree and the down-closure property to speedup the mining process.

**Algorithm 3** *ExtractFrequentMotifs*(Root  $r$ , Support  $\theta$ )

---

```

1:  $C_1 \leftarrow \emptyset; R \leftarrow \emptyset; k \leftarrow 1$ 
2:  $\text{Count}(C_1, r)$  {count the first level}
3: while  $|C_k| \neq 0$  do
4:    $C_{k+1} \leftarrow \emptyset$ 
5:   for each  $c \in C_k$  do
6:      $c.\text{count} \leftarrow \text{max\_independent\_set}(c.\text{motifocc\_list})$ 
7:     if  $c.\text{count} \geq \theta$  then
8:        $R \leftarrow R \cup \{c\}$  {record the motif  $c$  in resulting set  $R$ }
9:     end if
10:    if  $c.\text{count} \geq \theta$  or  $k = 1$  then
11:      for each  $v \in c.\text{motifocc\_list}$  do
12:         $\text{Count}(C_{k+1}, v)$ 
13:      end for
14:    end if
15:  end for
16:   $k \leftarrow k + 1$ 
17: end while
18: return  $R$ 

```

---

**Procedure**  $\text{Count}(\text{Set } C, \text{Node } v)$

```

19: for each  $v' \in v.\text{children}$  do
20:   if  $v'.\text{interval} \neq \emptyset$  then
21:      $\text{code} \leftarrow \text{canonicalcode}(v')$ 
22:      $c \leftarrow \text{search}(C, \text{code})$  { $c$  is created if it does not exist}
23:      $c.\text{motifocc\_list} \leftarrow c.\text{motifocc\_list} \cup \{v'\}$ 
24:   end if
25: end for

```

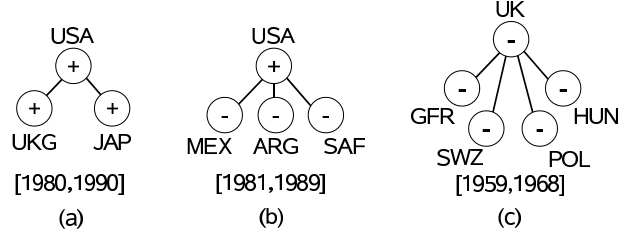
---

The *ExtractFrequentMotifs*() algorithm, shown in Algorithm 3, takes the root of the motif occurrence tree  $r$  and finds all of the motifs that appear at least  $\theta$  times. This is done in a level-wise fashion, similar to Apriori [1]. A key idea in this algorithm is to record each type of motif occurrence (corresponding to a node in the occurrence tree) when counting the frequency of each motif. This allows us to efficiently count the motif frequency for the next level without repeatedly accessing the same node many times.

#### 4 Experimental Results

Due to the space limitation, we only report our discover for the the global economics. Interested readers can look at the full paper [6] for the complete experimental results. This dataset is created from the publicly available Expanded Trade and GDP Data [4]. The data represents the yearly imports and exports, total trade and gross domestic product of 196 countries spanning the 52 years 1948-2000. The time series for each country is the proportion of its share in the global economy according to its gross domestic product(GDP) for that year. In other words, the time series for GDP-Norm is the normalized value of each individual annual GDP, divided by the total GDP for all countries during that year. The topology for the graph was created by comparing the yearly total trade for each country and its trade with each of the other countries. If the trade between country A and country B in any given year accounts for more than 10% of either country's total trade for that year, an edge is created between the the two countries.

**Output and Performance** In the experiments, all trends were found with either  $\sigma = 2$  or  $\sigma = 3$  as the maximum



**Figure 1. Example Motif Occurrences**

time step, since a series that increases or decreases by  $\delta$  at least every two or three steps can reasonably be considered as moving consistently. Additionally, the maximum depth was constant at six, ensuring that we would enumerate all occurrences of motifs that contain up to 6 vertices. In Table 1, we can see the results of the experiments at different support levels. Given different parameters  $\sigma$ ,  $\delta$ ,  $l$  and  $w$ , we first show the total number of maximal intervals of increasing trends  $I+$  and decreasing trends  $I-$ . We also show the number of vertices which have intervals of both increasing and decreasing trends, denoted as  $|N+, -|$ , and only have intervals of increasing trends, decreasing trends and none, denoted as  $|N+|$ ,  $|N-|$ , and  $|None|$ . Then, we vary the support level from high to low, and report the total number of frequent trend motifs at each support level (*Count*) and the running time *Time*. Clearly, as the support level is reduced, more motifs are being discovered and the running time is increasing.

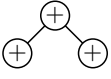
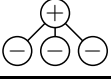
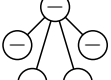
**Significant Trend Motifs** Here, we show several representative examples from our experimental results, and list them in Tables 2. Besides providing their frequency (*count*) in the corresponding datasets, we also compare them with randomized networks. Since our datasets combine both network topology and time series data, we will construct three types of randomized networks. The first type of randomization, referred to as *RS*, shuffles the time series data for each vertex and the underlying network topology remains the same. The second type, referred to as *RN*, shuffles the edges and labels (corresponding trends) among the vertices while preserving the degree distribution of each vertex [10], and the time series data remains the same. Finally, the third type of randomization, referred to as *RS/RN*, is a combination of the first two. We build 200 randomized networks for each type of randomization, and compute the average and standard deviation of frequencies for each trend motif in the 200 networks. Finally, we compute the Z-score for the significance of each motif as compared to the specific type of randomization.

The GDP-Norm dataset contains very interesting motifs. In the GDP-Norm motifs shown in Table 2, we see a very distinct dependence relationship among the countries. Very few motifs were found where all vertices were well connected, leading to the notion that the country with the highest degree can greatly affect its dependent neighbors. This would be further validated when we look at the specific trend motif occurrences.

**Table 1. GDP-Norm**

$\delta = 0.00014, \sigma = 2, l = 10, w = 8$					
$I+$	$I-$	$ N+, - $	$ N+ $	$ N- $	$ None $
48	79	18	24	48	106
Support	60	40	15	6	1
Count	66	193	301	405	1055
Time	0.01s	0.07s	1.01s	50.27s	128.7s
$\delta = 0.0002, \sigma = 3, l = 15, w = 10$					
$I+$	$I-$	$ N+, - $	$ N+ $	$ N- $	$ None $
21	69	6	14	59	117
Support	40	20	10	3	1
Count	65	134	154	202	322
Time	0.01s	0.02s	0.11s	3.38s	9.10s

**Table 2. GDP-Norm Motifs**

$\delta = 0.00014, \sigma = 2, l = 10, w = 8$				
Motif	RS	RN	RS/RN	
	Count: 15			
	$\mu \pm \sigma$ :	.12 $\pm$ .68	2.30 $\pm$ 2.55	.01 $\pm$ .16
	Z score:	21.9	4.99	96.6
	Count: 7			
	$\mu \pm \sigma$ :	0 $\pm$ 0	.65 $\pm$ 1.49	0 $\pm$ 0
	Z score:	7	4.26	7
	Count: 7			
	$\mu \pm \sigma$ :	0 $\pm$ 0	1.82 $\pm$ 2.44	0 $\pm$ 0
	Z score:	7	2.12	7

**Interesting Trend Motif Occurrences** In Figure 1 we show some interesting trend motif occurrences that were found in GDP-Norm dataset. The first motif (a), displays the partnership between the United States (USA), United Kingdom (UK), and Japan (JAP) during the 1980's which shows significant market share growth for all three countries. In (b), however, we see that countries that depended on the United States (USA), such as Mexico (MEX), Argentina (ARG), and South Africa (SAF), were losing global market share during that same period. We believe this displays a shift in the global economic structure. Finally, in (c), we note that several regional patterns also developed as motifs. Here we see a trend where the United Kingdom (UK) is decreasing, while the European countries that depend on it, such as Germany (GFR), Switzerland (SWZ), Poland (POL), and Hungary (HUN), are also decreasing during the 60's. Another interesting fact is that major motif occurrences found in GDP-Norm were occurring on approximately the 1955-1965 time span, and then again in the 1980 to 1990 time span. We believe that these two distinct time-based patterns can be due to the reconstruction efforts and emerging countries after World War II and then again during the waning years of the Cold War. Both eras marked major changes in the global economy and are portrayed through our identified motifs.

We are convinced that these motifs not only are statistically significant, but they identify key characteristics about the underlying dynamics of these complex systems. Clearly, the GDP-Norm motifs display highly correlated

subgraphs that show the major shifts in global economics.

## 5. Conclusions

In this paper, we have developed a data mining approach, making it possible to analyze evolving weighted complex networks. A list of new concepts and new algorithms enable the analysis from individual vertex (trend discovery), to a group of correlated vertices (trend motif occurrence), and to the common patterns of change (frequent trend motif) in a dynamic complex network. The detailed experimental study on three real datasets have demonstrated the significance of these patterns in uncovering significant events in the dynamic system, and to understand their characteristics. We hope our methodology will open a new avenue in applying motif mining to analyze the dynamics of complex systems.

## References

- [1] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, 1994.
- [2] Lars Backstrom, Daniel P. Huttenlocher, Jon M. Kleinberg, and Xiangyang Lan. Group formation in large social networks: membership, growth, and evolution. In *KDD*, pages 44–54, 2006.
- [3] Reinhard Diestel. *Graph Theory*. Springer-Verlag, 2000.
- [4] Kristian S. Gleditsch. Expanded trade and gdp data,. *J. Conf. Res.*, 46:712–724, 2002.
- [5] Akihiro Inokuchi, Takashi Washio, and Hiroshi Motoda. Complete mining of frequent patterns from graphs: Mining graph data. *Mach. Learn.*, 50(3):321–354, 2003.
- [6] Ruoming Jin, Scott McCallen, and Eivind Almaas. Trend motif: A graph mining approach for analysis of dynamic complex networks. Technical Report TR-KSU-CS-2007-05, Kent State University, 2007.
- [7] Michihiro Kuramochi and George Karypis. Frequent subgraph discovery. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*, pages 313–320, 2001.
- [8] Michihiro Kuramochi and George Karypis. Finding frequent patterns in a large sparse graph. In *SDM*, 2004.
- [9] Jure Leskovec, Jon M. Kleinberg, and Christos Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *KDD*, pages 177–187, 2005.
- [10] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594):824827, October 2002.
- [11] Ron Milo, Shalev Itzkovitz, Nadav Kashtan, Reuven Levitt, Shai Shen-Orr, Inbal Ayzenshtat, Michal Sheffer, and Uri Alon. Superfamilies of evolved and designed networks. *Science*, 303:1538 – 1542, 2004.
- [12] Gergely Palla, Albert-Laszlo Barabasi, and Tamas Vicsek. Quantifying social group evolution. *Nature*, 446(7136):664–667, April 2007.
- [13] F. Schreiber and H. Schwabermeyer. Towards motif detection in networks: frequency concepts and flexible search. In *Proc. Intl. Wsh. Network Tools and Applications in Biology (NETTAB'04)*, pages 91–102., 2004.
- [14] Paul T. Spellman, Gavin Sherlock, Michael Q. Zhang, Vishwanath R. Iyer, Kirk Anders, Michael B. Eisen, Patrick O. Brown, David Botstein, and Bruce Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast *saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell*, 9:3273–3297.