

A Review of Motif Discovery Algorithms as the Main Units of the Complex Networks

Mohammad Ali Salari¹, Jamshid Tashk¹, Hossein Bobarshad^{*2}, Sahand Khakabimamaghani³

¹Department of Computer Science, Islamic Azad University, science and research branch, Saveh, Iran

²Faculty of New Sciences and Technologies, University of Tehran
Tehran, Iran

³School of Computing Science, Simon Fraser University, Burnaby, BC, Canada

Abstract

Today, Motif discovery has evolved into a comprehensive method in social, biological, technological and other types of complex networks in order to better understand their structure and function, providing significant information. As one of the structural characteristics of any network, motifs are small subnets (sub-graphs) within the input network that being observed more frequently than in randomized networks and have been called the building blocks of networks [2]. The main problem with recognition of large networks motifs is the increase exponentially in possible sub-graphs of the networks in terms of the motif size. Various algorithms have been already suggested to solve the problem with great computational complexity, requiring considerable execution time and consumption memory which have practically restrictions on the size of motifs. The present case study tries to show the importance and aims of the motif discovery as well as some strategies to solve the problems. Then, using a simple framework, the available algorithms are classified and their strengths and weaknesses are explored considering the experimental and laboratory data.

Keywords: Motif Discovery; Complex Networks; Frequency; Sub-graph

1. Introduction

Many studies have been carried out in recent years on such complex networks as social, computerized, and electronic, especially the biological ones including the metabolic network, gene regulatory network, and also the Protein-Protein interaction network to better understand the biological functions. Despite complex relations among the biological systems, networks provide for analyzing very different systems

within a single framework to achieve a better understanding about the system structure and function. Network properties are often examined based on the two criteria: General and Local features. The general properties help to have a holistic view regarding the network and depend on its structure. Of the main general criteria are degree distribution, the shortest path, and clustering coefficient [1]. While the local properties deal with network analysis at a micro level, evaluating small and local infrastructures which are called motifs. In 2002, Milo introduced motifs as the specific infrastructures with the possibility of repetition in network or even among different networks [2], suggesting a particular pattern for interactions between vertices of network and a special functioning in the network.

Fig. 1, for instance shows a triangle motif called Feed-forward loop. Being of the most frequent motifs, it may be found in many electronic and biological networks such as gene regulatory and neural networks responsible for controlling and transferring information [2, 3].

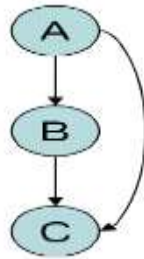


Figure 1. Schematic representation of a Feed-forward motif

Study on Motifs to understand the general principles of complex network structures, as well as a better understanding of network performance has attracted much attention. Research has shown the similar motifs are observed in networks with the same processes. Thus motifs can be used as a main component for network classification. In fact, motif discovery in networks is finding infrastructures in them with a statistical significance. From the view of graph theory, small sub-graphs are observed more frequently in comparison with the expected random networks [2]. Computationally, motif discovery in a given network is the main part of the studies undertaken. Normally, motif discovery algorithms consists of three major steps: 1) enumerating and searching the occurrences of sub-graphs of a given size in the network; 2) determining isomorphic sub-graphs as well as their classification in terms of various non-isomorphic classes; 3) determining the statistical significance of each classes by comparing their frequencies to those of an ensemble of random networks.

What follows in the present study is the aims including different algorithms and major aspects of motif discovery applied in the tools as well as introducing some new approaches not present in the past review papers; providing a framework for the motif discovery algorithm classification based on their applied methods; and also investigating the performances and their comparison against each other regarding the results from different studies.

2. The main features of motif discovery

Regarding motif finding tools there are some tasks to be considered when examining and developing these tools. They include way of deciding frequencies of motifs, determining sub-graph isomorphism, the methods of creating randomized networks, determining actuarial significance of the sub-graph's frequency and the strategies of identifying motifs with and without full enumeration.

When expanding an efficient motif discovery tool, all of the aforementioned elements should be considered. Therefore, each tool may utilize different varieties and combinations of all these variables which in the following sections will be discussed in detail.

2.1 Determining sub-graph frequency

The method of determining motif frequencies is a crucial part of each tool and algorithm. The concept beyond frequency is the number of matches of a motif in a network [4]. The frequency of a sub graph is strictly related to the kind of node and edge overlap that is allowed in counting the occurrences of the sub graph [5, 6], and this affects in a significant way both the results and the complexity of the counting.

Three different frequency concepts were discussed in Schreiber and Schwobbermeyer [5, 6], F1, F2 and F3. According to Schreiber and Schwobbermeyer F1 counts every match of this pattern. This concept gives a complete overview of all possible occurrences of a pattern even if elements of the target graph have to be used several times; indeed, it gives the possibility of arbitrary overlapping of nodes and edges between two sub-graphs. Frequency concepts F2 and F3 restrict the reuse of graph elements shared by different matches of a pattern therefore, F2 is only capable of node overlapping; and F3 does not allow any overlapping of nodes or edges. Figure 2, Figure 3, and Table 1 illustrate variations in sub-graph frequency based on different frequency concepts. The method of determining motif frequencies is of great importance due to the fact that motif frequency is used in the calculation of statistical elements namely, z-score and p-value [7]. Numerous tools use these statistical parameters to show whether or not motifs are statistically significant. It is necessary to note which frequency concept is used by which tool. The different restrictions upheld by each concept result in significant differences in the frequencies calculated by the different concepts [8]. In some networks the overlapping of edges and nodes may be an important aspect of motifs. However, sometimes it might be only relevant to find motifs that do not overlap at all. In this way, frequency concepts are of considerable importance when using and designing motif discovery tools [23].

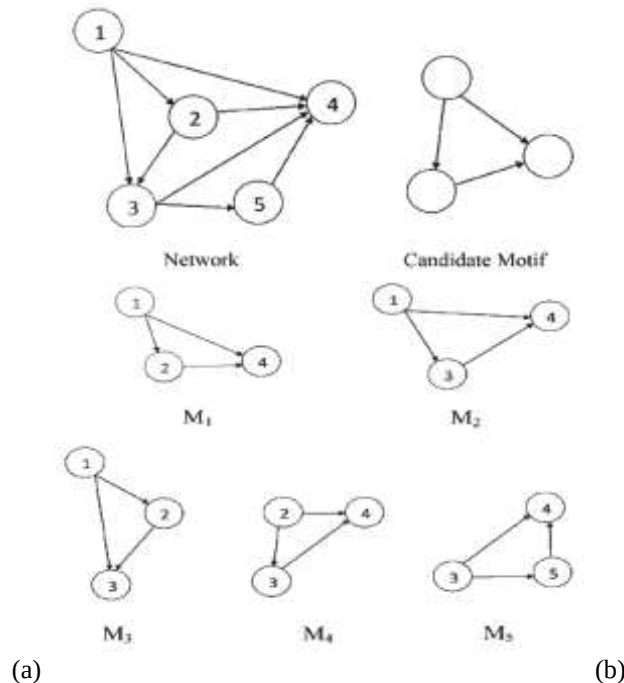


Figure 2. a) An example directed graph and a candidate motif. b) Instances of a candidate motif in the network given in Figure 2.(a) using frequency concept F1 [25].

2.2 Graph isomorphism

The graph isomorphism problem is the computational problem of figuring out if two finite graphs are isomorphic. The problem is neither known to be solvable in polynomial time nor NP-complete, and therefore might be in the computational complexity class NP. This issue involves determining if bisection occurs between the nodes of two graphs and that each corresponding node is adjacent to the same corresponding nodes [12]. Firstly two isomorphic graphs include equal number of nodes and edges and then same degrees for corresponding nodes. In the following, the world's fastest and well known method for isomorphism testing is NAUTY which is developed by Brendan D. McKay and that is used by several motif discovery tools.

NAUTY is able to produce a canonical label for the graphs to assist in isomorphism testing and if they are isomorphic, then their canonical label should be the same [12]. In order to forge a binary number as a canonical label for each graph, the given adjacency matrix of the graph should be concatenate row by row. By partitioning the graph into singleton sets it is possible to find automorphism group of vertices by the adjacency matrix of different orderings of vertices and checking if the matrix is the same. NAUTY then examines the automorphisms and computes a canonical label, which is the largest or smallest possible concatenated adjacency matrix [14, 23].

2.3 Statistical significance and motif detection

Network motif is described as a frequently and uniquely represented subgraph in a network and also to determine whether they are statistically significant, several structural uniqueness parameters are utilized [7]. If the network in question has a motif with a higher frequency than in a certain amount of random networks (threshold set by the user) then the motif is considered to be 'unique'. Moreover, statistical significance of motifs can be measured by calculating the z-score and p-value for a certain network [23].

Z-score: The Z_{score} is characterized as the difference of the frequency of this motif in the target network and its mean frequency in a sufficiently large set of randomized networks, divided by the standard deviation of the frequency values for the randomized networks [12]. The higher corresponds with the motif being more overrepresented and also Motifs are considered statistically overrepresented if the is greater than 2.0. For the assumed motif GP, this measure is defined as below:

Where N_{real} is the number which GP occurred in the input network, $\bar{N}$ is the mean number which GP occurred in the random networks and σ is the standard deviation.

P-value: The P represents the probability of a motif appearing in a randomized network an equal or greater number of times than in the network in question. For a reasonable calculation of the P at least 1000 randomized networks have to be considered. The lower means the more significant motif and motifs with a P less than 0.01 are regarded as statistical significant. All of these factors are important for setting standards to help distinguish between which subgraphs are overrepresented and which are not [23].

3. Classification of Motif discovery algorithms based on their applied strategies

In this section, we discuss several solution strategies for performing sub-graph census of the motif-finding problem include 'exact enumeration', 'sampling' and 'mapping'.

We intend to form a simple classification framework to display some representative motif discovery algorithms taking into account the strategies they utilize. In addition, some available algorithms for the motif discovery as well as the ones not analyzed so far will be reviewed in this section.

Subgraph census: Sub-graph census is the process of scanning the target network (node-by-node or edge-by-edge) along with enumerating all occurrences of all sub-graphs of a given size is called Sub-graph census. In so doing a pattern growth tree is usually used.

Full enumeration can be much time consuming and may require large amount of computational power. The size of the sub-graphs being searched for increases along with the possible isomorphic types. This makes exhaustive enumeration algorithms extremely time consuming and costly. This is because of the fact that finding the frequencies of each different isomorphic graph of all sizes in both the input network and the randomized networks is needed [23].

Kashtan et al developed a sampling method for subgraph counting which is a probabilistic algorithm [13]. This algorithm deals with estimating subgraph frequencies by sampling subgraphs. An advantage here is that it is less time consuming than full enumeration and also the runtime does not increase asymptotically as network size increases due to the features of the algorithm. Within the Kashtan's sampling algorithm larger networks than full enumeration algorithms are possible to be analyzed and larger motifs can be identified [23].

A random n -sized sub-graph is present in this sampling algorithm. An edge is picked randomly and its neighbors are all made into candidates to be the next edge. One of the candidates is picked at random while its neighbors are the new candidates. This process continues with one edge from all the neighbors being chosen randomly to be the next edge until a sub-graph of size n is created. All of the nodes from these edges and all the edges that connect these nodes forge the final sampled sub-graph [13]. An ordered set of $n-1$ edges needs to be picked for an n sized subgraph to be found. The probability of getting these ordered pairs is used to find the probability that an n sized subgraph will be sampled. From this and a few additional calculations the estimated subgraph concentrations are found [13, 23].

one of the main restrictions of Kashtan's method is that it has bias sampling [4]. In other words each sub-graph does not have a uniform probability of being sampled [15]. As a result, occurrences of a subgraph are not possible to be impartially estimated [4]. The algorithm tries to consider this by weighting each subgraph with a value of $1/(\text{probability of the subgraph being chosen})$ [15, 23].

Mapping: The mapping strategy of discovering subgraphs takes a size- k subgraph (candidate motif) and maps it onto the network in as many different places as it can. It stands against the view of enumerating all size- k sub-graphs in the network and then computing frequency of a candidate size- k motif. A mapping strategy ranks the nodes in the target network based on their (and their neighbors') degree properties. In the next step it searches for nodes in the input network that have similar features of nodes in the candidate motif [16, 18]. This strategy is suggested by Grochow and Kellis [16] and is used by MODA ('Motif-centric algorithms' section) as well. Mapping, when used with symmetry breaking, resolves isomorphism without explicit checking [25].

One of the shortcomings of this strategy is that when the size of the sub-graph gets bigger than 10, computations due to the number of possible variants become intractable [17]. Moreover, this process generates and computes frequency for sub-graphs that might not occur in the target graph which is a disadvantage [25].

A simple framework for describing a network motif finding algorithm based on its used strategies is discussed. Within this framework, some representative algorithms (Table 1) are dealt with. Note that these algorithms are selected not by their performance, but by the particular strategies they use. These algorithms fall under the category of two broad classes namely 'network-centric' and 'motif-centric' algorithms which are open to further classification depending on whether a sampling strategy is used [25].

Table1. Classification of network motif finding algorithms

	Network-Centric	Motif-Centric
Exact Search	MAVisto[9] NeMoFinder[10] Kavosh[12] G-trie[20] QuateXelero[21]	Grochow and Kellis[16,19]
Sampling	MFinder[13] FANMOD[11]	MODA[4]

3.1 Network-Centric algorithms

Main network is the starting point of the network-centric algorithms that all sub-graphs with size-k that occur in the target network are numerated by them. Network-centric algorithms have the benefit that sub-graphs that do not appear in the target network are never encountered. However, it needs to be mentioned that these algorithms fail in computing the frequency of a given sub-graph without doing subgraph census [25].

Mfinder: the first motif-mining tool, implements two kinds of motif finding algorithms: a full enumeration and a sampling method. Until 2004, the only exact counting method for NM detection was the brute-force one proposed by Milo et al. [2]. This algorithm was successful for discovering small motifs, but using this method for finding even size 5 or 6 motifs was not computationally feasible. Hence, a new approach to this problem was needed. Kashtan et al. [13] presented the first sampling NM discovery algorithm, which was based on ‘edge sampling’ throughout the network. This algorithm estimates concentrations of induced sub-graphs and can be utilized for motif discovery in directed or undirected networks [24].

MFinder fully enumerates subgraphs by starting with an edge. All motifs of different sizes are found that contain this edge [13]. Once a subset of nodes is found that is connected to the initial edge the subset is added to a hash table so it cannot be revisited [15]. When no more subgraphs can be identified the hash tables are cleared and the process begins again with a different edge. This is repeated until all edges have been used. Because a specific subgraph will be counted each time one of its edges is examined there is redundancy and number of times the subgraph will be counted is a multiple of its edge number [13]. Therefore, the count for a subgraph must be divided by the number of edges in the motif. Since MFinder looks at so many motifs and has redundancy it requires large amounts of memory. This causes the runtimes to be large and makes it hard for large motifs to be searched for [15]. Therefore, MFinder uses the biased sampling method that Kashtan et al developed [23].

[13] proposed a weighting scheme that assigns different weights to the different sub-graphs within network. The underlying principle of weight allocation is exploiting the information of the sampling probability for each sub-graph, that is, the probable sub-graphs will obtain comparatively less weights in comparison with the improbable sub-graphs; hence, the algorithm must calculate the sampling probability of each sub-graph that has been sampled. This weighting technique assists Mfinder to determine sub graph concentrations impartially [24].

ESU (Fanmod) : Wernicke [11] introduced an algorithm named *RAND-ESU* that provides a significant improvement over Mfinder.[13] This algorithm, which is based on the exact enumeration algorithm *ESU*, has been implemented as an application called *Fanmod*. [11] *RAND-ESU* is a NM discovery algorithm applicable for both directed and undirected networks, effectively exploits an unbiased node sampling throughout the network, and prevents overcounting sub-graphs more than once. It is able to identify motifs of sizes 3 – 8. Furthermore, *RAND-ESU* uses a novel analytical approach called *DIRECT* for determining sub-graph significance instead of using an ensemble of random networks as a Null-model. The *DIRECT* method estimates the sub-graph concentration without explicitly generating random networks [11, 24].

The full enumeration part of the *Fanmod* algorithm begins with one node and a list of possible vertices to which this node can be connected (i.e. the node's neighbors). Once a possible vertex is extended to it is removed from the list of possible extensions and its neighbors are added to the candidates that this vertex can be connected to next. Different combinations of possible extensions are chosen in order to form subgraphs of different sizes. Since the list of possible extensions is constantly changed, each subgraph is only enumerated once. Like Kavosh, *Fanmod* uses the NAUTY algorithm to test for graph isomorphism. [15, 23]

Fanmod's alternative method uses probabilistic sampling to reduce runtimes for identifying motifs. It uses randomized enumeration algorithm known as *RAND-ESU*. This sampling works by changing the full enumeration algorithm so that it randomly skips subgraphs. The *Fanmod* sampling algorithm chooses each size *k* subgraph with a certain probability [22]. This means that all subgraphs have the same probability of being sampled and all samples give different subgraphs. Because of the adjustments to the Kashtan et al algorithm, *Fanmod* is unbiased and results in all subgraphs having the same probability of being chosen [15, 23].

FPF (Mavisto) : Schreiber and Schwobbermeyer [6] proposed an algorithm named *flexible pattern finder (FPF)* for extracting frequent sub-graphs of an input network and implemented it in a system named *Mavisto*. [9] Their algorithm exploits the *downward closure* property which is applicable for frequency concepts F2 and F3. The downward closure property asserts that the frequency for sub-graphs decrease monotonically by increasing the size of sub-graphs; however, this property does not hold necessarily for frequency concept F1. FPF is based on a *pattern tree* consisting of nodes that represents different graphs (or patterns), where the parent of each node is a subgraph of its children nodes; in other words, the responding graph of each pattern tree's node is expanded by adding a new edge to the graph of its parent node [24].

At first, the FPF algorithm enumerates and maintains the information of all matches of a sub-graph located at the root of the pattern tree. Then, one-by-one it builds child nodes of the previous node in the pattern tree by adding one edge supported by a matching edge in the target graph, and tries to expand all of the previous information about matches to the new sub-graph (child node). In next step, it decides whether the frequency of the current pattern is lower than a predefined threshold or not. If it is lower and if downward closure holds, FPF can abandon that path and not traverse further in this part of the tree; as a result, unnecessary computation is avoided. This procedure is continued until there is no remaining path to traverse. The advantage of the algorithm is that it does not consider infrequent sub-graphs and tries to finish the enumeration process as soon as possible; therefore, it only spends time for promising nodes in the pattern tree and discards all other nodes [24].

NeMoFinder : This tool is a motif discovery algorithm used specifically to find motifs in PPI networks [10], and uses trees to partition the network in question. It uses concept F1. By allowing for arbitrary node and edge overlap it ensures uniqueness and is not downward closed. The required inputs for the algorithm

include maximum motif size, number of randomized networks, the target PPI network, frequency threshold, and uniqueness [7, 23].

NeMoFinder consists of three main steps. First, finding frequent size- n trees, then utilizing repeated size- n trees to divide the entire network into a collection of size- n graphs, finally, performing subgraph join operations to find frequent size- n sub-graphs.[10] In the first step, the algorithm detects all non-isomorphic size- n trees and mappings from a tree to the network. In the second step, the ranges of these mappings are employed to partition the network into size- n graphs. Up to this step, there is no distinction between *NeMoFinder* and an exact enumeration method. However, a large portion of non-isomorphic size- n graphs still remain. *NeMoFinder* exploits a heuristic to enumerate non-tree size- n graphs by the obtained information from the preceding steps. The main advantage of the algorithm is in the third step, which generates candidate sub-graphs from previously enumerated subgraphs. This generation of new size- n sub-graphs is done by joining each previous sub-graph with derivative subgraphs from itself called *cousin subgraphs*. These new sub-graphs contain one additional edge in comparison to the previous sub-graphs. However, there exist some problems in generating new subgraphs: There is no clear method to derive cousins from a graph, joining a sub-graph with its cousins leads to redundancy in generating particular subgraph more than once, and cousin determination is done by a canonical representation of the adjacency matrix which is not closed under join operation. *NeMoFinder* is an efficient network motif finding algorithm for motifs up to size 12 only for protein-protein interaction networks, which are presented as undirected graphs. And it is not able to work on directed networks which are so important in the field of complex and biological networks [24].

Kavosh :Kavosh is a network motif discovery tool that uses trees to enable the detection of motifs. It can handle both directed and undirected networks [12]. There are four main parts of the Kavosh algorithm: (1) enumeration, (2) classification, (3) random graph generation, and (4) motif identification [12, 23]. Enumeration looks at the network in question and finds all subgraphs of given sizes (also preformed on random graphs). This is achieved by selecting one node and all the combinations of connections with the neighboring nodes via tree representation. The first level of the tree is the selected node, the second level consists of the neighbors of this node, the third level of the tree is made up of the neighbors of the previous neighbors, and so on. If a k sized graph is being searched for, all compositions of size $k-1$ are found. The 'revolving door algorithm' is used to go through all of the nodes at each level ascending from the bottom level and labeling each node as 'visited'. This ensures that no tree or subgraph is constructed more than once. The algorithm finds all of the combinations of the nodes including subgraphs with nodes in the same level (i.e. a subgraph size 3 can be made up of an initial node and two neighbors or an initial node, a neighbor, and a neighbor of a neighbor). After these motifs are found, the node is removed and a new node is used. This process is also carried out on the randomized networks to find the frequency and identify the motifs in the randomized cases for comparison. Constraints are placed on the construction of these trees (some explained above) so that each specific tree is only generated once. This avoids redundancy and extra computational time [23].

Classification involves placing the subgraphs found in the enumeration step into isomorphic classes. This is done using the NAUTY algorithm [12]. Random graphs are generated in Kavosh using the switching method. The frequencies of subgraphs in the input network are compared to frequencies in the random networks. Subgraphs are dubbed as motifs if frequencies are higher in the input network than they are in the random networks. Parameters often used include p -value, frequency level, and z scores. [12, 23]

G-Tries : In 2010, Pedro Ribeiro and Fernando Silva proposed a novel data structure for storing a collection of sub-graphs, called a *g-trie* [20]. This data structure, which is conceptually akin to a prefix tree, stores sub-graphs according to their structures and finds occurrences of each of these sub-graphs in a larger graph. One of the noticeable aspects of this data structure is that coming to the network motif discovery, the sub-graphs in the main network are needed to be evaluated. So, there is no need to find the

ones in random network which are not in the main network. This can be one of the time-consuming parts in the algorithms in which all sub-graphs in random networks are derived. A *g-trie* is a multiway tree that can store a collection of graphs. Each tree node contains information about a single graph vertex and its corresponding edges to ancestor nodes. A path from the root to a leaf corresponds to one single graph. Descendants of a *g-trie* node share a common sub-graph. Constructing a *g-trie* is well described in [20, 24]. After constructing a *g-trie*, the counting part takes place. The main idea in counting process is to backtrack by all possible sub-graphs, but at the same time do the isomorphism tests. This backtracking technique is essentially the same technique employed by other motif-centric approaches like *MODA* and *GK* algorithms. Taking advantage of common substructures in the sense that at a given time there is a partial isomorphic match for several different candidate sub-graphs [24].

QuateXelero: According to the available motif discovery algorithms, it seems necessary to finding an approach to decrease the need to call NAUTY isomorphism detection method, which is the most time-consuming step in many existing algorithms. To this end, in 2013 Khakabi et al. introduced an extremely fast motif detection algorithm called QuateXelero, which increase the performance of graph isomorphism problem [26]. QuateXelero has a Quaternary Tree data structure in the heart based on the well-known ESU (FANMOD) motif detection algorithm. A Quaternary tree is a rooted tree data structure and each internal node has at most four children. Accordingly, each internal node in the tree can have at most five neighbors, one of which is its parents and the others are its children. Each edge, connecting a parent to one of its children, can be labeled with a mark, which can be a number, character, or any other symbol. A labeled quaternary tree can be searched using a given string that consists of the same set of symbols used for labeling that tree. This searching initiates in the tree's root. In each step, one symbol is read from the input string and the current pointer, initially set as root, moves to the child of the current node, connecting edge of which corresponds to the symbol that is read recently from the input string. Because it is allowed to add nodes during the search, if one node in the path has no child for an input symbol, a child is added to the current node for that symbol and the current pointer moves to that child. Thus, this search continues until the input string is read completely.

This quaternary tree performs a partial classification for enumerated subgraphs in the proposed algorithm. This data structure, which is similar to G-Trie data structure in some aspects, is used before calling NAUTY and eliminates the need to use it most of the times. The results of the algorithm show that it has an acceptable performance in most of the cases.

3.2 Motif-Centric algorithms

Motif- centric algorithms have the ability to numerate frequency of all kinds of subgraphs in the main network. In this way they can directly verify whether the query sub-graph is a motif. To compute frequencies of all size-*k* graphs, a motif-centric algorithm needs to enumerate all possible size-*k* graphs. As network-centric algorithms, motif-centric algorithms can be further classified depending on whether sampling is used over the target network while counting subgraph frequencies [25].

GK: Grochow is a motif centric tool that is used for directed and undirected networks [16]. A specific target subgraph onto a global network is progressively mapped by the algorithm. In this way Grochow checks for isomorphism as it maps the query graph onto the network [15]. This wipes out the additional time and memory it would take to check for isomorphism and avoids full enumeration. The mapping algorithm goes through the query subgraph node by node in order to map the subgraph onto the network. A node will be specified and the tool will find all the 'candidate nodes', nodes in the network that have the same characteristics (i.e. same degree and neighbors with correct degrees). As the algorithm goes through each node in the query subgraph possible matches in the network are found while others that are

not entirely the same are removed once any inconsistency is found. This mapping ensures that only exact isomorphic subgraphs in the network are found [23].

Grochow uses a method known as symmetry breaking to make sure that each subgraph is only mapped to once in order to decrease execution time and redundancy [16]. Graphs that are self-isomorphic are believed to have the same symmetries. Equivalent Nodes are nodes that can be mapped to one another. As a result, the nodes in a specific subgraph can be separated into equivalence classes. The Grochow algorithm ensures that mapping begins only from one representative of each equivalence class so that it is not possible to carry out multiple mappings beginning with equivalent nodes. Moreover, in order to avoid symmetry restrictions are added to the labeling of each vertex [15, 23].

MODA: MODA utilizes a pattern growth algorithm that takes in a query graph [4]. It takes advantage of information on the basis of previously found query graphs. Maintaining information about formerly found mappings helps to reduce computational time. It uses the concept of expansion trees, which are similar to pattern trees used in MAVisto, but applicable to the frequency concept F1. Then all minimally connected size-k trees of the root node are found, which is level 1, and finally an edge is added at each level until a complete graph is formed. The first level of the tree therefore represents the number of non-isomorphic trees. Each node of the expansion tree can be represented by an adjacency matrix consisting of 0 and 1. For undirected graphs, which are symmetric, only the numbers below the main diagonal are stored. Expansion trees are stored for every size k-graph. They are a static data structure which can be stored and retrieved and do not have to be found each time [4, 23].

The query graph from the first level of the expansion tree, which is composed of trees themselves, is taken from the mapping algorithm and is mapped onto the network. It holds onto their calculated frequencies. The frequencies at the second level of the expansion tree are dependent on the first level of the expansion tree, which are their parent nodes. MODA utilizes the symmetry-breaking conditions of the Grochow algorithm. It only uses the Grochow algorithm for the first level of the expansion tree. In order to find the frequencies of the all the next levels which are super graphs of the first level, all the information the algorithm finds about the first level are exploited. By exploiting information of formerly found mappings, MODA can be used to reduce computational costs. Additionally, MODA has a sampling method that can be used to decrease execution time with the sacrifice of accuracy [4, 23].

4. Experiments and Analysis

To compare the performance of existing motif discovery algorithms, some experimental results taken from literature have been extracted as can be seen in Fig. 3 and Tables 2 and 3. (Due to different numbers of generated random networks in different experiments, each figure and table should be examined separately [24].)

In this paper, different algorithms that aim to solve the motif discovery problem have been reviewed and given the carried out experiments, the comparison of a variety of motif discovery tools has been possible. Therefore, the sampling method increases the algorithm performance as long as it does not prevent from the complete search of the sample space. The sampling algorithms make it possible to find greater motifs in an acceptable time [23]. On the other hand, exact enumerating algorithms are needed for the problem solving accurately. In fact, the two methods are mutually codependent regarding the accuracy and runtime. For instance, the runtime of Fanmod (sampling algorithm) is less than that of exact algorithms, while such exact algorithms as Kavosh can be applied to measure greater subgraphs. Effectiveness of the algorithms Kavosh in comparison to other algorithms is due to the subgraph census strategy, using NAUTY to explore graph isomorphic. In addition, with an increase in the size of subgraphs and networks, runtime and possibility of the exact algorithms decreases considerably, in these cases motif-centric

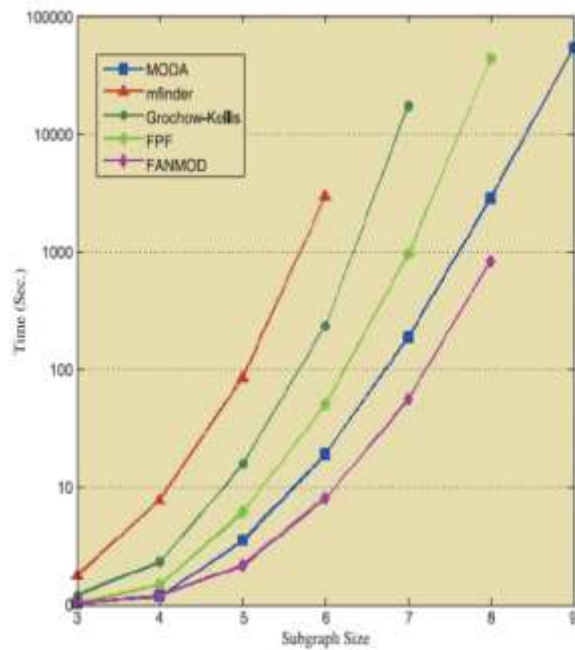


Figure 3. Runtimes of Grochow-Kellis, Mfinder, FANMOD, FPF and MODA for subgraphs from three nodes up to nine nodes[4].

Algorithms like MODA prevent from reduplication of subgraphs thanks to the mapping technique and symmetry breaking method, yielding an appropriate performance (Figure 3).

The Steady trend present in these experiments is due to the inability of MFinder and MAVisto to deal with subgraphs as large as the other tools. The MFinder stores all subgraphs in memory and considers the same subgraph many times, and also Mavisto due to its complex approach to counting the subgraphs are not so effective and can only find the ones with the size of 5 or less and waste a great deal of runtime. Fanmod is efficient in computation and tends to produce results due to counting every subgraphs just once and can identify motifs up to size 8, while other exact algorithms like Kavosh have the potentiality to find larger motifs and sometimes faster than Fanmod (table 2), mainly because of its use of NAUTY as an efficient subgraph enumeration strategy for isomorphism checking. Yet the algorithms NeMoFinder, Kavosh, MODA, and GK are suitable for the sizes greater than 8 and need greater runtimes.

Table.2 Runtimes of Mfinder, FANMOD, Mavisto and Kavosh for sub-graphs from three nodes up to ten nodes on three different networks [12].

Networks↓	Algorithms↓ Size →	3	4	5	6	7	8	9	10
Escherichia coli	Kavosh	0.30	1.84	14.91	141.98	1374.0	13173.7	121110.3	1120560.1
	FANMOD	0.81	2.53	15.71	132.24	1205.9	9256.6	-	-

Electronic	Mavisto	135 32	-	-	-	-	-	-	-
	Mfinder	31.0	297	23671	-	-	-	-	-
	Kavosh	0.08	0.36	8.02	11.39	77.22	422.6	2823.7	18037.5
	FANMOD	0.53	1.06	4.34	24.24	160	967.99	-	-
Social	Mavisto	210. 0	1727	-	-	-	-	-	-
	Mfinder	7	14	109.8	2020.2	-	-	-	-
	Kavosh	0.04	0.23	1.63	10.48	69.43	415.66	2594.19	14611.23
	FANMOD	0.46	0.84	3.07	17.63	117.43	845.93	-	-
	Mavisto	393	1492	-	-	-	-	-	-
	Mfinder	12	49	798	181077	-	-	-	-

Table 3. Experimental Result For QuateXelero(QX) vs. Kavosh and QX vs. G-tries[21]

Network	Size	Subgraphs	Classes	Subgraphs/ Classes	Processing Times		Total Time	
					Kavosh	QX	ESU+G-Tries	QX
Yeast	5	2508149	174	14414.65	23.4	0.5	37.85	10.51
Yeast	6	32883898	888	37031.42	438.5	8.9	651.07	190.20
Yeast	7	416284878	4809	86563.71	14056.2	166.4	14494.60	3611.77
Yeast	8	5184710063	27003	192004.97	224497	2609.5	-	-
Yeast	9	64730339589	156025	414871.59	-	53852.1	915907.47	125962.31
Electronic	5	19675	49	401.53	0.13	0	2.13	1.28
Electronic	6	97038	199	487.63	0.8	0.08	8.49	5.59
Electronic	7	495274	907	546.06	5.9	0.3	45.81	31.29
Electronic	8	2572125	4333	593.61	38.7	1.9	-	-
Electronic	9	13512688	20692	653.04	278.2	11.9	79.18	16.11
Electronic	10	71614362	96483	742.25	2614.2	71.2	550.76	90.76
E.coli	5	80724	590	136.82	0.48	0.05	15.63	5.51
E.coli	6	558080	3884	143.69	4.3	0.3	104.07	33.65
E.coli	7	4019781	23587	170.42	45.3	2.8	822.42	274.45
E.coli	8	29294103	136569	214.50	410.7	23.6	-	-
E.coli	9	212782282	768121	277.02	4000	190.7	1223.27	264.96
Social	5	10599	773	13.71	0.11	0.06	3.55	1.31
Social	6	52156	5062	10.30	0.82	0.36	22.74	9.00
Social	7	254674	30217	8.43	5.4	2.6	154.91	72.78

Social	8	1224376	165958	7.38	33.3	16.3	-	-
Social	9	5764767	854023	6.75	220.3	96.22	119.34	83.37

10, 100, 100, and 100 random networks were used respectively for Yeast, Electronic, E.coli, and Social networks.

Given the results in Fig. 3, the algorithm QX is much faster than Kavosh in all cases; the amount of this superiority depends on the network size and structure, motif size, and the variety of its non-isomorphic classes that lead to reducing the need to call NAUTY. The results suggest the better processing time than Kavosh and G-tries, the only drawback of the QX algorithm is the considerable amount of memory that is used to construct the quaternary tree for larger motif sizes and for networks containing larger number of non-isomorphic subgraph classes.

5. Conclusions

The study of networks helps to understand how systems and organisms work as a whole. Network motifs are biologically remarkable and also they are the building blocks of networks [2]. In this way the identification of the motifs become extremely significant in the researches for the understanding of networks.

some aspects of handling the network motif discovery problem in research include— handling graph isomorphism, generating random graphs, computing sub-graph frequencies in the network (by sampling or exact enumeration) and There are some factors that make network motif discovery more computationally complex which are also necessary parts of the network motif discovery process and in understanding networks. These issues include finding large motifs within the network, generating and searching a large number of random networks. While a few algorithms exist in the literature with experimental results accessible on selected target networks, the good and bad parts of each tool are imperative to observe so that algorithmic shortcomings can be avoided in future tools while effective viewpoints can be capitalized on. Network motif finding has proved to be a very complex task. Although many tools, methods and algorithms have been made for discovering network motifs, further improvements and new developments are a necessity in order to increase motif discovery capabilities [23].

On the one hand, designing a parallel and robust algorithm that could simultaneously analyze different sections of the network for a distributed computation platform, afterwards reducing the running time and empowering our reach to larger motifs and networks can be a suitable technique to enhance these algorithms. On the other hand, another improvement could be made by reducing unnecessary computations over the random graphs. Moreover, intelligently search of respective networks for possible biological relevant motifs that have been identified as significant sub-graphs from experimental runs and literature review could be another helpful area of research [23].

6. Acknowledgements

Hereby, we gratefully acknowledge all the valuable comments made by reviewers and professors that provided helpful and organized information. In addition, we would like to thank Mr. Khakabi and the Bioinformatics' Center, University of Tehran for all their support and cooperation to make the present study possible.

References

- [1] J. A. Bondy, U. S. R. Murty, (1982). Graph Theory with Applications, 5 ed., Elsevier Science Publishing Co., New York.

- [2] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, (2002). "Network motifs: simple building blocks of complex networks," *Science's STKE*, vol. 298, p. 824.
- [3] Ratushny AV, Ramsey SA, Roda O, et al. Control of transcriptional variability by overlapping feed-forward regulatory motifs. *BiophysJ* 2008;95:3715–23.
- [4] Omid, S., Schreiber, F., Masoudi-Nejad, A.: MODA: An efficient algorithm for network motif discovery in biological networks. *Genes Genet. Syst.* 84, 385 – 395 (2009).
- [5] Kuramochi M, Karypis G. Finding frequent patterns in a large sparse graph. In: *SIAM International Conference on Data Mining (SDM'04)* 2004. SIAM, Lake Buena Vista, Florida, USA. pp. 243–71.
- [6] Schreiber F, Schwobbermeyer H. Frequency concepts and pattern detection for the analysis of motifs in networks. *Trans Comput Syst Biol* III(2005);LNBI 3737:89–104.
- [7] Ciriello, G., Guerra C.: A review on models and algorithms for motif discovery in protein-protein interaction networks. *Briefings in Functional Genomics and Proteomics* Advance Access. (2008)
- [8] Schwobbermeyer, H.: Network Motifs. In Junker, B., Schreiber, F. (eds.) *Analysis of Biological Networks*. Pp. 85 – 111. NJ: John Wiley & Sons, Inc (2008) .
- [9] Schreiber F, Schwobbermeyer H (2005). "MAVisto: a tool for the exploration of network motifs". *Bioinformatics* **21** (17): 3572–3574.
- [10] J. Chen, W. Hsu, M. L. Lee, and S. K. Ng, (2006). "NeMoFinder: Dissecting genome-wide protein-protein interactions with meso-scale network motifs," pp. 106-115.
- [11] Wernicke S, Rasche F (2006). FANMOD: a tool for fast network motif detection. *Bioinformatics* **22**: 1152–1153.
- [12] Kashani, Z., Ahrabian, H., Elahi, E., Nowzari-Dalini, A., Ansari, E., Asadi, S., Mohammadi, S., Schreiber, F., Masoudi-Nejad, A.: Kavosh: a new algorithm for finding network motifs. *BMC Bioinf.* 10:318 (2009).
- [13] N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon, "Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs," *Bioinformatics*, vol. 20 ,pp. 1746-1758, (2004).
- [14] Fortin, S.: *The Graph Isomorphism Problem*. University of Alberta: Dept of Computing Science, Alberta (1996)
- [15] Ribeiro, P., Silva, F., Kaiser, M.: Strategies for network motifs discovery. *IEEE International Conference*. 81-86 (2009)
- [16] Grochow JA, Kellis M. Network motif discovery using sub-graph enumeration and symmetry-breaking. *Res Comp Mol Biol* (2007);4456:92–106.
- [17] Sloane NJA, Plouffe S. Number of graphs on n unlabeled nodes. (A000088, Formerly M1253 N0479). *The Online Encyclopedia of IntegerSequences* 2010. <http://oeis.org> (15 January 2011, date last accessed).
- [18] Ullmann JR. An algorithm for subgraph isomorphism. *JACM* 1976;23:31–42.
- [19] Grochow JA (2006). "On the structure and evolution of protein interaction networks" Thesis M. Eng., Massachusetts Institute of Technology, Dept. of Electrical Engineering and Computer Science.
- [20] Ribeiro P, Silva F (2010). G-Tries: an efficient data structure for discovering network motifs. 25th ACM Symposium on Applied Computing – Bioinformatics and Computational Systems Biology Track, Sierre, Switzerland
- [21] Sahand Khakabimamaghani, Iman Sharafuddin, Norbert Dichter, Ina Koch and Ali Masoudi-Nejad (2013) QuateXelero: an accelerated exact network motif detection algorithm. *PLOS ONE*. 18;8(7):e68073.
- [22] Wernicke, S.: A Faster Algorithm for Detecting Network Motifs. In Casadio, R., Myers, G. (eds.) *Algorithms in Bioinformatics: 5th international workshop*. pp. 165 – 176. Springer (2005)
- [23] Wong EA, Baur B (2010) On network tools for network motif finding: a survey study. Online.

- [24] A. Masoudi-Nejad, F. Schreiber, and Z. Kashani, "Building blocks of biological networks: A review on major network motif discovery algorithms, " *IET systems biology*, vol. 6, no. 5, pp. 164-174, 2012.
- [25] E. Wong, B. Baur, S. Quader and C.-H. Huang, "Biological Network Motif Detection: Principles and Practice", in Oxford University Press, (2011)