An Algorithm to Construct Network Motifs for Bioinformatics Study

Efendi Zaenudin, Ezra B. Wijaya†, Eskezeia Y. Dessie†, David Agustriawan, Jeffrey J.P. Tsai, Chien-Hung Huang*, Ka-Lok Ng*, Member, IAENG

Abstract—Biological network is formed by the interaction between the genetic elements. It is known that biological functions and processes can be understood in terms of network structures. To analyze a real network, one can decompose the network into smaller modules, so-called network motifs. Motif is a fundamental entity of real networks and it exhibits specific functions or properties. For a digraph, the number of possible patterns for N-node-motifs grows exponentially.

In this paper, we propose a systematic method which can generate all the allowed network motifs without bipartite, isolated and isomorphic motifs. We illustrated our approach by applying it to generate the complete sets of both 3-node-motifs and 4-node-motifs. It is expected that our algorithm can be applied to construct 5-node-motifs by parallelizing the algorithm.

Index Terms **-**Biological Networks; Digraphs; Network Motifs; Isomorphic Graphs; Permutation Matrices.

I. Introduction

Network motifs play an important role in various types of networks, including molecular networks, ecological networks, electrical networks etc [1-8]. For a digraph, after including the isolated motifs, bipartite motifs and isomorphic motifs,a total of 13, 199, and 9364 and 1,530,843 possible patterns can be defined for the 3-node, 4-node, 5-node and 6-node motifs, respectively [9-14].

The work of Chien-Hung Huang is supported by the grant of Ministry of Science and Technology of Taiwan (MOST) MOST 107-2221-E-150-038.

Declarations. The source of funding for publication is supported by MOST under grant MOST 107-2221-E-150-038.

The work of Efendi Zaenudin (EZ), Ezra B. Wijaya† (EBW), Eskezeia Y. Dessie† (EYD) and Dr. Ka-Lok Ng are supported by MOST under the grant of MOST 106-2221-E-468-017. Dr. Ka-Lok Ng work is supported by the MOST 107-2632-E-468-002, 107-2221-E-150-038 and grants from Asia University, 106-asia-06 and 106-asia-09. Dr. Jeffrey J. P. Tsai work is supported by the grant of MOST 107-2632-E-468-002. EZ, EBW and EYD are with the Department of Bioinformatics and Medical Engineering, Asia University. (e-mail:ezaenudin@mail.informatika.lipi.go.id, ezrafisioterapi@gmail.com, estu2003@gmail.com),they are supported by supported by MOST 107-2632-E-468-002. Both EBW† and EYD† equally contributed to this work. Efendi Zaenudin is with Research Center for Informatics, Indonesian Institute of Sciences, Bandung, Indonesia. David Agustriawan is with Department of Bioinformatics, Indonesian International Institute for life science, Indonesia(e-mail:david.agustriawan@i3l.ac.id)

*Corresponding author, Chien-Hung Huang is with the Department of Computer Science and Information Engineering, National Formosa University (e-mail: chhuang@nfu.edu.tw).*Corresponding author, Ka-Lok Ng is with the Department of Bioinformatics and Medical Engineering, Asia University & Department of Medical Research, China Medical University Hospital, China Medical University, Taiwan (e-mail: ppiddi@gmail.com).

II. Methods

Let $G = (V, E)$ be a motif composed of V vertices and E edges [15]. By analyzing the connectivity of each node one constructs an adjacency matrix, $\boldsymbol{A}$, to represent the motif. In the adjacency matrix a value of zero and one are assigned to represent non-interacting and direct interacting nodes, respectively. There is no self-interacting node. Therefore, the diagonal elements of $\boldsymbol{A}$ are zeros [16]. For N-node motifs, the possible number of edges range from $N-1$ to $2*C(N,2)$ edges.

Given the adjacency matrix, one can denote a motif by an integer. For example, the 3-node-motif named feed-forward-loop can be represented by the binary string 000100110, which is equivalent to '38' in decimal representation.

The complete set of the N-node motifs can be represented by 2^X adjacency matrices, where $X = 2*C\ (N,\ n)$ is the combinatorial factor for choosing $n = 2$ nodes from N nodes. The factor of two arises because we consider digraph.

For certain adjacency matrices, they could represent isolated motif, i.e. there is at least one disconnected node. For instance, let a, b, c denotes the nodes in a 3-node-motif. Isolate motif is a motif which one of the nodes doesn't connect to another node. In 3- and 4-node case, isolate and bipartite motifs are illustrated in Table I.

Isolated motif can be identified by studying the $\boldsymbol{A}$ matrix. Sums of the row and column elements of $\boldsymbol{A}$ define a matrix consists of in- and out-degree:

$$\left[In\ Degree = \sum_i \boldsymbol{A}_{ij}\ , Out\ Degree = \sum_j \boldsymbol{A}_{ij}\right] \qquad (1)$$

Columns one and two represent the in-degree and out-degree of a node, respectively. As an example in the first figure in Table I, node c is disconnected from nodes a and b. In the in- and out-degree matrix, both the third column and third row are zero.

For 4-node-motifs, another type of isolated motif is the directed bipartite motif [14, 17]. A bipartite motif is a motif with nodes a and c connected, and nodes b and d connected only. For instance, the decimal representation of the motif is depicted in the fifth row of Table I, i.e. 8452, which is a bipartite motif. To identify whether the motif is a bipartite or not, we treat the pattern as an undirected graph. In other words, we are symmetries the matrix $\boldsymbol{A}$, and sum up all the matrix elements. If the sum divided by 2 is less than the minimum of edges, i.e. three edges for 4-node motifs, then we removed this bipartite motif and the associated adjacency matrix.

ISBN: 978-988-14048-5-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE I
ISOLATED MOTIF AND BIPARTITE MOTIFS

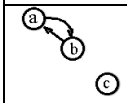
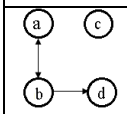
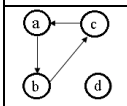
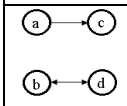
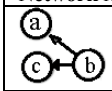
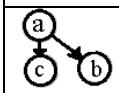
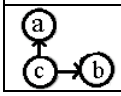
Network Motif	Node	Edges	Adjacency Matrix	Decimal Number	In & Out degree matrix
	3	2	$\begin{bmatrix}0&1&0\\1&0&0\\0&0&0\end{bmatrix}$	160	$\begin{bmatrix}1&1\\1&1\\0&0\end{bmatrix}$
	4	3	$\begin{bmatrix}0&1&0&0\\1&0&0&1\\0&0&0&0\\0&0&0&0\end{bmatrix}$	18688	$\begin{bmatrix}1&1\\2&1\\0&0\\0&1\end{bmatrix}$
	4	3	$\begin{bmatrix}0&1&0&0\\0&0&1&0\\1&0&0&0\\0&0&0&0\end{bmatrix}$	17024	$\begin{bmatrix}1&1\\1&1\\1&1\\0&0\end{bmatrix}$
	4	3	$\begin{bmatrix}0&0&1&0\\0&0&0&1\\0&0&0&0\\0&1&0&0\end{bmatrix}$	8452	$\begin{bmatrix}1&0\\1&1\\0&1\\1&1\end{bmatrix}$

Table II is an illustration of three different isomorphic structures that compose of three nodes and two edges. The network motifs are associated with decimal numbers 40, 192, and 6, and these motifs are considered as isomorphic graphs. Further analysis shows that adjacency matrices associated with isomorphic graphs are related by permutation matrix multiplication.

TABLE II
ISOMORPHIC NETWORK MOTIFS COMPOSE OF THREE NODES AND TWO EDGES

Network Motif	Adjacency Matrix	Decimal Number
	$\begin{bmatrix}0&0&0\\1&0&1\\0&0&0\end{bmatrix}$	40
	$\begin{bmatrix}0&1&1\\0&0&0\\0&0&0\end{bmatrix}$	192
	$\begin{bmatrix}0&0&0\\0&0&0\\1&1&0\end{bmatrix}$	6

An isolated motif is a motif composes of isolated nodes. For 4-node-motifs, another type of isolated motif is the bipartite motif. For instance, let *a, b, c* and *d* denote the nodes. A bipartite motif is a motif with nodes *a* and *b* connected, and nodes *c* and *d* connected only. Both isolated motifs and bipartite motifs were removed in this study.

Motifs with isomorphic [18, 19] structures are related by matrix multiplication. For 3-node motifs, the permutations are given by six permutation matrices including the identity matrix. In general, there are ***N***! permutation matrices, it is because each row and column of the permutation matrix consists of only one "1". Let P_k denotes the permutation matrices where *k* equals 0 to 5 [20], the six matrices are:

$$P_0 = \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}, P_1 = \begin{bmatrix}1&0&0\\0&0&1\\0&1&0\end{bmatrix}, P_2 = \begin{bmatrix}0&1&0\\1&0&0\\0&0&1\end{bmatrix}, P_3 = \begin{bmatrix}0&0&1\\1&0&0\\0&1&0\end{bmatrix}, P_4 = \begin{bmatrix}0&1&0\\0&0&1\\1&0&0\end{bmatrix},$$

$$P_5 = \begin{bmatrix}0&0&1\\0&1&0\\1&0&0\end{bmatrix} \quad (2)$$

The first matrix denoted by P_0 is the identity matrix. Some of the permutation matrices are symmetric, such as, P_0, P_1, P_2, andP_5; some are asymmetric (trace = 0).The inverse of a permutation matrix is again a permutation matrix, i.e.$P^{-1} = P^T$.

Left multiplication of adjacent matrix, P_LA resulted in row interchange, whereas right multiplication of A, i.e. AP_R, resulted in column interchange. It is known that matrices multiplication satisfies the association law, i.e. $P_L(AP_R) = (P_LA)P_R$. If $P_L(AP_R) = X$, then motif A and X are motifs with isomorphic structures. Since the indices L and R run from 0 to 5, there are a total of 36 row and column interchanging operations.

One can consider two consecutive permutation matrices multiplications, i.e. P_yP_x; however, a product of permutation matrices is also a permutation matrix; such as $P_1P_2 = P_4$ and $P_5P_2 = P_3$. Hence, there is no need to consider higher order multiplications. For 4-node-motifs and 5-node-motifs, there are 24 and 120 permutation matrices associated with them respectively. Fig 1 is the workflow of this algorithm.

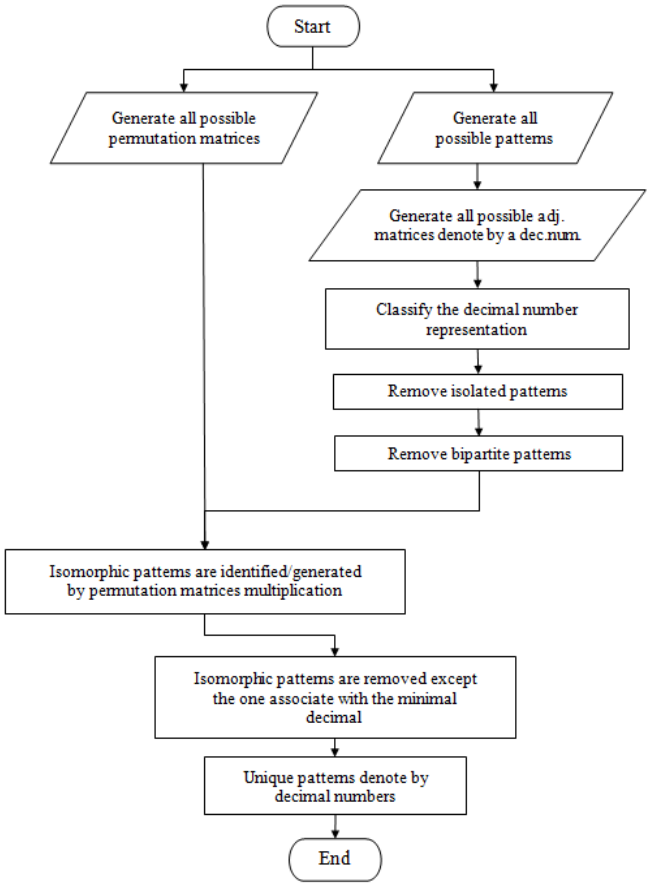


Fig 1. WORKFLOW OF THE PRESENT STUDY

ISBN: 978-988-14048-5-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

TABLE III
THE EIGHT STEPS OF THEWORKFLOW

Step 1 – generate all possible patterns with *N-1* to 2**C(N,2)* edges.
Step 2 –generate all possible permutation matrices.
Step 3 – generate all possible adjacency matrices and denote each matrix by a decimal number.
Step 4 – classify the decimal number representation according to the number of edges.
Step 5 – remove isolated patterns.
Step 6 – remove bipartite patterns.
Step 7 – isomorphic patterns are identified/generated by permutation matrices multiplication.
Step 8 – isomorphic patterns are removed except the one associates with the minimal decimal.

Step 1: Generate all possible patterns with *N-1* to *2*C(N,2)* edges
Input:*N*
Output:[*minimum-edge*, … , *maximum-edge*]

1: *maximum-edge* $=N^2-N$
2: *minimum-edge=N-1*
3: ***While*** *minimum-edge<=maximum-edge:*
4: *edge-index=[minimum-edge++]*

Step 1 created an index parameter to denote a group of motifs that are associated with different decimal numbers. After that, the "while" loop creates the edge-indexed list.

Step2: Generate all possible permutation matrices
Input:*identity_matrix_permutation*
Output:*permutation_matrix*

1: *temp_matrix=identity_matrix_permutation*
2: ***Call****multiset_permutation*
3: ***Function****save_numberof_permutation=multiset_permutation.sum()*
4: ***For****save_numberof_permutation****in****multiset_permutation(list_permutation):*
5: *save_list_permutation(save_numberof_permutation)*

Isomorphic motifs are related by permutation matrix multiplication. Step 2 is to generate the set of all possible permutation matrices, i.e. *N!* for *N* nodes. This is done by calling the library 'multiset_permutation' (available in Python 3.6).

Step3: Generate all possible adjacency matrices and denote each matrix by a decimal number.
Input:*N*
Output:[2, … , *i*]*(numberof_decimal)*

1: *matrix_adjacency=numpy.ones(change diagonal to zero)*
2: ***For*** *i* ***in*** *reversed(matrix_adjacency(element)):*
3: *max_decimal_number=2^counter x int(i)*
counter+=1
4: ***For*** *I* ***in*** *range(2, max_decimal_number):*
5: *save_numberof_decimal=i*
i+=1
6: ***While*** *i<length(save_numberof_decimal)*
7: *graph_inmatrix =numpy.binary_repr(*
save_numberof_decimal[i],width=NxN)
8: *graph_inmatrix_adj=numpy.array(*
graph_inmatrix)
9: ***While*** *j<length(diagonal_matrix_NxN)*
10: ***If*** *diagonal_matrix_NxN(*
graph_inmatrix_adj) =0:
11: *numberof_decimal[i]=save_numberof_*
decimal[i]
12: *j+=1*
13: *i+=1*

As step 3 shown, one defines a maximum decimal number for a given number of edges. We generate an adjacency matrix consists of '1' equals to the number of edges. Then, we denote each matrix by a decimal number, and save the decimal numbers in a list.

Step 4: Classify the decimal number representation according to the number of edges.
Input:[2, … , *i*]
Output:[[6, … , *i*],[10, … , *i*],[12, … , *i*],…,[Maximum Decimal Number]]

1: ***While*** *i<length(numberof_decimal)*
2: *graph_inmatrix =numpy.binary_repr(*
numberof_decimal[i],width=NxN)
3: *sumof_element[i]=numpy.sum(graph_inmatrix*
[element])
4: *i+=1*
5: ***While*** *j<length(edge-index)*
6: ***While*** *k<length(numberof_decimal)*
7: ***If*** *sumof_element[k]=edge-index[j]:*
8: *classof_decnum.extend([*
numberof_decimal])
9: *k+=1*
10: *j+=1*

In step 4, we are grouped motifs with the similar number of edges (without considering edge direction) together into the same list.

Step5: Remove isolated patterns.
Input:[[6, … , *i*],[10, … , *i*],[12, … , *i*],…,[Maximum Decimal Number]]
Output:[[6, … , *i*],[12, … , *i*],[20, … , *i*],…,[Maximum Decimal Number]]

1: ***While*** *i<length(classof_decnum)*
2: *decnum_classes =classof_decnum[i]*
3: ***While*** *j<length(decnum_classes)*
4: *adj_matrix_trf =numpy.binary_repr(*
decnum_classes[j],width=NxN)
5: *adj_matrix=numpy.array(adj_matrix_trf)*
6: ***If*** *adj_matrix.columnsum[@]=*
adj_matrix.rowsum[@]=0 :
7: *deleteof_decnum =decnum_classes[j]*
8: *j+=1*
9: *classof_decnum[i]=*
classof_decnum[i]-deleteof_decnum
10: *sorted(classof_decnum[i])*
11: *i+=1*

ISBN: 978-988-14048-5-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

Consider a list of decimal numbers correspond to the same number of edges, we convert them to the adjacency matrices. Step 5 compute the in- and out- degree matrix(Eq. (1)). Isolated motifs are determined by the condition where both the i^{th} column and i^{th} row are zero.

Step 6: Remove bipartite patterns.
Input:[[6, … , *i*],[12, … , *i*],[20, … , *i*],…,[Maximum Decimal Number]]
Output:[[6, … , *i*],[12, … , *i*],…,[Maximum Decimal Number]]

1: ***While*** *i<length(classof_decnum)*
2: *decnum_classes =classof_decnum[i]*
3: ***While*** *j<length(decnum_classes)*
4: *adj_matrix_trf =numpy.binary_repr(decnum_classes[j],width=NxN)*
5: *adj_matrix=numpy.array(adj_matrix_trf)*
6: ***While*** *k<N:*
7: ***While*** *l<N:*
8: ***If*** *k!=l:*
9: ***If*** *adj_matrix[k][l]!=adj_matrix[l][k]:*
10: *adj_matrix[k][l]=1*
11: *k+=1*
12: ***If*** *adj_matrix.sum(all)/2 <minimum-edge: deleteof_decnum =decnum_classes[j]*
13: *j+=1*
14: *classof_decnum[i]= classof_decnum[i]-deleteof_decnum*
15: *sorted(classof_decnum[i])*
16 *i+=1*

One can use step6 to remove bipartite motifs. Step 6 is similar to step 5 except from lines 6 to 13. We treat the pattern as an undirected graph as we mention in Section II. If the sum divided by 2 is less than the minimum of edges, we removed this bipartite motif and the associated adjacency matrix.

Step 7: Isomorphic patterns are identified/generated by permutation matrices multiplication
Input:[[6, … , *i*],[12, … , *i*],…,[Maximum Decimal Number]]
Output:[[6, 40, 192],[12, 34, 66, 96, 132, 136],[14, 42, 70, 168, 196, 224],[36, 72, 130],[38, 44, 104, 134, 194, 200],[46, 198, 232],[74, 76, 100, 138, 162, 164],[78, 170, 228],[98, 140],[102, 106, 142, 172, 204, 226],[108, 166, 202],[110, 174, 206, 230, 234, 236],[238]]

1: ***While*** *i<length(classof_decnum)*
2: *decnum_classes =classof_decnum[i]*
3: ***While*** *j<length(decnum_classes)*
4: *adj_matrix_trf =numpy.binary_repr(decnum_classes[j],width=NxN)*
5: *adj_matrix=numpy.array(adj_matrix_trf)*
6: ***While*** *k<length(save_list_permutation*):*
7: *permutation=save_list_permutation[k]*
8: *permutation_transpose= permutation.transpose()*
9: *matrix_isomorphm=((permutation × adj_matrix)×permutation_transpose)*
10: *decnum_isomorphm=convert(matrix_isomorphm)*
11: *k+=1*
12: *collectof_decnum_isomorphm.extend(decnum_isomorphm)*
13: *j+=1*
14: *classof_decnum[i]= collectof_decnum_isomorphm*
15: *sorted(classof_decnum[i])*
16: *i+=1*
**save_list_permutation is call permutation in step 3*

Step 7 is to group together all the isomorphic motifs that are related by permutation matrix multiplication. For instance, the group [6, 40, 192] means that motif_6, motif_40 and motif_192 are isomorphic, these three motifs area related by permutation matrix multiplication. Group [12, 34, 66, 96, 132, 136], represents another group of isomorphic motifs, it has the same number of edges as group [6, 40, 192]. Output from step 7 serves as the input for step 8.

Finally, for each group of isomorphic motifs, step 8 selected the minimal decimal number to represent that group, which is shown as underline and bold-face font below.

Step8: Isomorphic patterns are removed except the one associate with the minimal decimal.
Input:[[**6**, 40, 192],[**12**, 34, 66, 96, 132, 136],[**14**, 42, 70, 168, 196, 224],[**36**, 72, 130],[**38**, 44, 104, 134, 194, 200],[**46**, 198, 232],[**74**, 76, 100, 138, 162, 164],[**78**, 170, 228],[**98**, 140],[**102**, 106, 142, 172, 204, 226],[**108**, 166, 202],[**110**, 174, 206, 230, 234, 236],[**238**]]
Output:[6, 12, 14, 36, 38, 46, 74, 78, 98, 102, 108, 110, 238]

1: ***While*** *i<length(classof_decnum)*
2: *decnum_classes =classof_decnum[i]*
3: ***While*** *j<length(decnum_classes)*
4: *decnum_minimal=decnum_classes[0]*
5: *collectof_decnum_minimal=decnum_minimal*
6: *j+=1*
7: *i+=1*
8: *classof_decnum=collectof_decnum_minimal*
9: *sorted(classof_decnum)*

III. Experiment

A. Computer environment

This study aims to determine complete sets of connected 3-node motifs and 4-node-motifs. We are implemented our algorithm in Python 3.6. Experiments are performed on two computers, which one contains2-Intel Xeon E5-2609 @2.40 GHz CPU with 4-cores and 8 GB memory and the other is a computer composes of an Intel Core™ i7-3770 CPU @3.40 GHz @3.90 GHz and 12 GB memory. The result of comparison of execution time in two computers is given in Table 4.

B. Comparison Algorithm between machine 1 and machine 2

TABLE IV
COMPARISON OF EXECUTION TIME IN TWO COMPUTERS

Node	3 (mm:ss,######)	4 (mm:ss,######)	5 (hh:mm:ss)
M1time	00:00.028420	00:05.743595	1:52:28
M2time	00:00.024324	00:03.446612	1:07:09

ISBN: 978-988-14048-5-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

To evaluate this algorithm, we compare the execution time using two computers. As shown in Table IV, this algorithm works in microseconds using serial programming. As well known, increasing number of node is made the time more complexity. It indicate the process obtain consuming time [21, 22].

IV. RESULTS

As shown in Table V we have identified sets of 3-node motifs and 4-node-motifs. For 5-node-motifs, we obtain 9378 motifs, which are different from 9364 [11, 21]. There are 14 motifs are not removed by the above method. We are extending our algorithm so that it can be applied to construct 5-node-motifs without bipartite, isolate and isomorphic patterns.

TABLE V
MOTIFS ID IDENTIFIED USING THE PRESENTED ALGORITHM

Node	3	4	5
Numbers of Motifs found	13	199	9378

A. Results for3-node

The output for 3-node is 13 unique motifs, it is shown below:

Number of Edges	Patterns ID[Decimal Number]
2	[6, 12, 36] Total Number is 3
3	[14, 38, 74, 98] Total Number is 4
4	[46, 78, 102, 108] Total Number is 4
5	[110] Total Number is 1
6	[238] Total Number is 1
Result:	
[6, 12, 14, 36, 38, 46, 74, 78, 98, 102, 108, 110, 238]	
13 patterns	

B. Results for 4-node

The output for 4-node is 199 unique motifs, it is shown below:

Number of Edges	Patterns ID [Decimal Number]
3	[14, 28, 74, 76, 280, 328, 392, 2184] Total Number is 8
4	[30, 78, 90, 92, 204, 282, 330, 332, 344, 390, 394, 396, 404, 408, 456, 904, 2186, 4370, 4418, 4420, 4424, 4740] Total Number is 22
5	[94, 206, 286, 334, 346, 348, 398, 406, 410, 412, 454, 458, 460, 468, 472, 856, 906, 908, 2190, 2202, 2204, 2252, 4374, 4422, 4426, 4428, 4434, 4436, 4440, 4546, 4548, 4678, 4682, 4692, 4742, 4748, 4994] Total Number is 37
6	[222, 350, 414, 462, 470, 474, 476, 858, 910, 922, 924, 972, 2206, 2254, 2458, 2506, 4382, 4430, 4438, 4442, 4444, 4550, 4556, 4562, 4564, 4686, 4694, 4698, 4700, 4750, 4758, 4764, 4812, 4946, 4952, 4998, 5002, 5004, 5010, 5012, 5016, 5058, 5064, 6342, 6348, 6356, 6552] Total Number is 47
7	[478, 862, 926, 974, 2270, 2462, 2510, 2524, 4446, 4558, 4566, 4572, 4702, 4766, 4814, 4950, 4954, 5006, 5014, 5018, 5020, 5062, 5066, 5068, 5074, 5076, 5080, 6350, 6358, 6364, 6550, 6554, 6598, 6602, 6604, 6616, 6854, 6858] Total Number is 38
8	[990, 2526, 4574, 4830, 4958, 5022, 5070, 5078, 5082, 5084, 6366, 6558, 6606, 6614, 6618, 6620, 6862, 6870, 6874, 6876, 7128, 13142, 13146, 13148, 13260, 14678, 14790] Total Number is 27
9	[3038, 5086, 6622, 6878, 7126, 7130, 13150, 13262, 14686, 14798, 14810, 14812, 15258] Total Number is 13
10	[7134, 13278, 14814, 15262, 15310] Total Number is 5
11	[15326] Total Number is 1
12	[31710] Total Number is 1
Result:	
[14, 28, 30, 74, 76, 78, 90, 92, 94, 204, 206, 222, 280, 282, 286, 328, 330, 332, 334, 344, 346, 348, 350, 390, 392, 394, 396, 398, 404, 406, 408, 410, 412, 414, 454, 456, 458, 460, 462, 468, 470, 472, 474, 476, 478, 856, 858, 862, 904, 906, 908, 910, 922, 924, 926, 972, 974, 990, 2184, 2186, 2190, 2202, 2204, 2206, 2252, 2254, 2270, 2458, 2462, 2506, 2510, 2524, 2526, 3038, 4370, 4374, 4382, 4418, 4420, 4422, 4424, 4426, 4428, 4430, …, 13150, 13260, 13262, 13278, 14678, 14686, 14790, 14798, 14810, 14812, 14814, 15258, 15262, 15310, 15326, 31710]	
199 patterns	

C. Resulst for 5-node

The output for 5-node is 9378 motifs, because of space limitation, only a partial list of the pattern ID is shown below:

Result:
[30, 60, 62, 154, 156, 158, 184, 186, 188, 190, 404, 406, 412, 414, 438, 444, 446, 924, 926, 958, 1080, 1082, 1086, 1176,...,7598014, 7974318, 7974326, 7974332, 7974334, 7974814, 7974846, 7988654, 7988670, 7989134, 7989150, 7989174, 7989178, 7989180, 7989182, 7990966,7990970, 7990974, 7991098, 7991100, 7991102, 7991198, 7991230, 8113846, 8113854, 8114078, 8114110, 8120254, 8122302, 16510910]
9378 patterns

V. DISCUSSION AND CONCLUSION

We utilized the permutation matrices, a systematic method, to generate all the allowed unique network motifs with isolated, bipartite and isomorphic motifs removed. We illustrated our approach by applying it to generate the complete sets of both 3-node-motifs and 4-node-motifs. It is expected that our algorithm can be applied to construct 5-node-motifs by parallelizing [15, 23, 24] the algorithm and remove bipartite patterns with two disconnected parts, i.e. a triangle pattern and a pair of connected nodes.

ISBN: 978-988-14048-5-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

REFERENCES

[1] U. Alon, "Network motifs: theory and experimental approaches," *Nature Reviews Genetics*, vol. 8, no. 6, p. 450, 2007.

[2] S. S. Shen-Orr, R. Milo, S. Mangan, and U. Alon, "Network motifs in the transcriptional regulation network of escherichia coli," *Nature genetics*, vol. 31, no. 1, p. 64, 2002.

[3] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon, "Network motifs: simple building blocks of complex networks," *Science*, vol. 298, no. 5594, pp. 824–827, 2002.

[4] J. Fodor, M. Brand, R. J. Stones, and A. M. Buckle, "Intrinsic limitations in mainstream methods of identifying network motifs in biology," *bioRxiv*, p. 272401, 2018.

[5] M. B. Z. Joveini and J. Sadri, "Application of fractal theory on motifs counting in biological networks," *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 15, no. 2, pp. 613–623, 2018.

[6] K. Mukherjee, M. M. Hasan, C. Boucher, and T. Kahveci, "Counting motifs in dynamic networks," *BMC systems biology*, vol. 12, no. 1, p. 6, 2018.

[7] J. Wang, Y. Huang, F.-X. Wu, and Y. Pan, "Symmetry compression method for discovering network motifs," *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, vol. 9, no. 6, pp. 1776–1789, 2012.

[8] K. A. Zweig, C. Brugger, V. Grigorovici, C. De Schryver, and N. Wehn, "Automated determination of network motifs," Jun. 7 2018, uS Patent App. 15/579,518.

[9] F. Harary and E. Palmer, "Graphical enumeration, acad," *Press, NY*, 1973.

[10] N. Sloane, "A and plouffe, s," *The Encyclopedia of Integer Sequences. San Diego*, 1995.

[11] J. Y. Kim, Y. Y. Kim, S. J. Kim, J. C. Park, Y. H. Kwon, M. K. Jung, O. K. Kwon, H. Y. Chung, W. Yu, J. Y. Park *et al.*, "Predictive factors for lymph node metastasis in signet ring cell gastric cancer and the feasibility of endoscopic submucosal dissection," *Journal of gastric cancer*, vol. 13, no. 2, pp. 93–97, 2013.

[12] P. Bloem and S. de Rooij, "Finding network motifs in large graphs using compression as a measure of relevance," *arXiv preprint arXiv:1701.02026*, 2017.

[13] Y. Zhang, P. Smolen, D. A. Baxter, and J. H. Byrne, "Computational analyses of synergism in small molecular network motifs," *PLoS computational biology*, vol. 10, no. 3, p. e1003524, 2014.

[14] X. Zhang, J. Han, and W. Zhang, "An efficient algorithm for finding all possible input nodes for controlling complex networks," *Scientific Reports*, vol. 7, no. 1, p. 10677, 2017.

[15] Y. Chen and Y. Chen, "An efficient sampling algorithm for network motif detection," *Journal of Computational and Graphical Statistics*, vol. 27, no. 3, pp. 503–515, 2018.

[16] C. Ma, B.-B. Xiang, H.-F. Zhang, H.-S. Chen, and M. Small, "Detection of core-periphery structure in networks by 3-tuple motifs," *arXiv preprint arXiv:1705.04062*, 2017.

[17] B. I. Simmons, M. J. Sweering, M. Schillinger, L. V. Dicks, W. J. Sutherland, and R. Di Clemente, "bmotif: a package for motif analyses of bipartite networks," *bioRxiv*, p. 302356, 2018.

[18] M. Kivelä and M. A. Porter, "Isomorphisms in multilayer networks," *IEEE Transactions on Network Science and Engineering*, vol. 5, no. 3, pp. 198–211, 2018.

[19] S. Patra and A. Mohapatra, "Discovery of large disjoint motif in biological network using dynamic expansion tree," *bioRxiv*, p. 308254, 2018.

[20] M. Houbraken, S. Demeyer, T. Michoel, P. Audenaert, D. Colle, and M. Pickavet, "The index-based subgraph matching algorithm with general symmetries (ismags): exploiting symmetry for faster subgraph enumeration," *PloS one*, vol. 9, no. 5, p. e97896, 2014.

[21] W. Kim, M. Diko, and K. Rawson, "Network motif detection: Algorithms, parallel and cloud computing, and related tools," *Tsinghua Science and Technology*, vol. 18, no. 5, pp. 469–489, 2013.

[22] D. Quang, Y. Guan, and S. C. Parker, "Yamda: thousandfold speedup of em-based motif discovery using deep learning libraries and gpu," *Bioinformatics*, vol. 1, p. 3, 2018.

[23] W. Lin, X. Xiao, X. Xie, and X.-L. Li, "Network motif discovery: A gpu approach," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 3, pp. 513–528, 2017.

[24] H.-N. Tran and E. Cambria, "A survey of graph processing on graphics processing units," *The Journal of Supercomputing*, vol. 74, no. 5, pp. 2086–2115, 2018.

ISBN: 978-988-14048-5-5
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)