ECOLE POLYTECHNIQUE FEDERALE DE LAUSANNE
Mathematics section

Master project in Computational Science and Engineering

Motif Discovery for Connectomics: Finding Computational Units in the Brain's Wiring Diagram

Carried out at the Visual Computing Group
At Harvard School of Engineering and Applied Sciences, Boston, USA
Under the supervision of Prof. Hanspeter Pfister, An Wang Professor of Computer Science

HARVARD
John A. Paulson
School of Engineering
and Applied Sciences

Done by

Antoine Alleon

Under the direction of Prof. Kathryn Hess Bellwald
Laboratory for Topology and Neurosciences, EPFL

Lausanne, June 21, 2019

Acknowledgments

I would first like to express my gratitude to Professor Hanspeter Pfister, head of the Visual Computing Group at the Harvard School of Engineering and Applied Sciences, for giving me the incredible opportunity to do my master thesis in his lab.

Then, I would like to particularly thank Bryan Matejek for giving me the opportunity to work with him in this project and for providing me great advice and support to write my thesis.

From the Visual Computing group, I would also like to thank Donglai Wei and Daniel Haehn; and the whole lab for their warm welcome and the great moments shared in and outside of work.

I would like to express my appreciation to Professor Kathryn Hess Bellwald for her supervision at EPFL and her help in coordinating this project.

Finally, I would like to thank my parents for their support and patience throughout my studies.

Abstract

The brain is an immense entanglement of neurons, connected through synapses, and connectomics is the discipline trying to make sense of it all. By imaging at high resolution, microscopes are use to image the brain and deep learning is used to identify the neurons and reconstruct in 3D volumes of the brain. From those volumes wiring diagrams are extracted and can be analyzed. Motif discovery is a field of study which identifies subgraphs over or under represented in graphs. The resulting tools can be applied to brain wiring diagrams to identify building blocks of the brain. As brain networks become larger, the existing algorithms show limitations in their performance. In this thesis, small motifs are identified as statistically significant in the brain network and an algorithm to speed up the discovery of larger motifs is presented. This algorithm takes advantage of the small-world structure of the brain by first clustering the graph before enumerating large subgraphs. The algorithm performs much faster than existing algorithms but fails enumerating subgraphs for one of the used graph. The results obtained from the small motif identification support the claim that the brain displays small-world properties, it is neither entirely random, nor ordered. The results from the algorithms show that using intrinsic features of brain connectomes can help develop faster algorithms for subgraph enumeration. Through faster algorithms applied on larger networks, computational units can be reliably identified and studied. This will lead to a better understanding of the brain's behavior and therefore to some neurological diseases. These computational units also play a major role in bio-inspired artificial intelligence.

Contents

1 Introduction

1.1 Connectomics

Connectomics is the study of the structure of neurons and their connections within the brain. Researchers hope to better understand the computational functions of the brain by studying its underlying wiring diagram. Furthermore, they hope to interpret the effects of genetic, molecular, and pathological changes at the connectivity level, which will lead to a better understanding of mental illnesses, learning disorders, or neuronal pathologies such as Alzheimer's [1], epilepsy [2], and other neuropsychiatric diseases [3]. Additionally, computer scientists hope to create more biologically-inspired artificial intelligence [4].

There are five main steps to go from brain tissue to wiring diagram analysis [5, 6]. The steps are the following:

1 **Acquisition of brain images:** This step begins by slicing a piece of the brain (Fig. 1 & 2) into very thin strips, usually around 30 to 40 nanometers thick. These slices are first stained with heavy metals, collected onto a tape, and sent into a multibeam electron microscope for imaging [4, 7]. The resolution of each image slice is on the order of a few nanometers in the x and y dimensions. Although these images are quite noisy, they provide enough detail for neuroscientists to identify mitochondria, synapses, and each individual neuron. The multibeam electron microscope can image one terabyte of image data every hour. At this throughput, a cubic millimeter of brain tissue can be imaged in less than 6 months. Cubes of this volume contain 30,000 slices each with a resolution of $4nm \times 4nm \times 30nm$ per voxel for a total storage space of two petabytes.

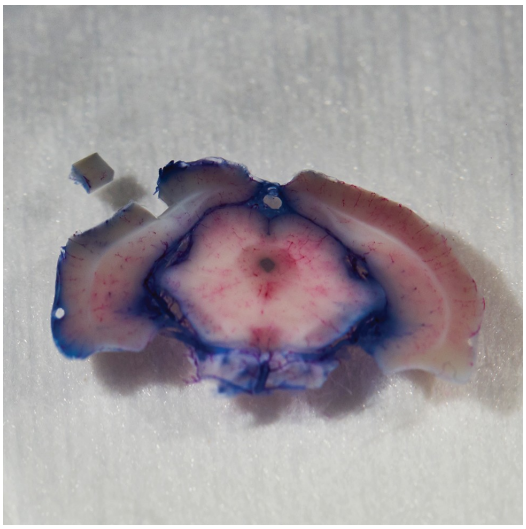


Figure 1: Brain cut and cube to map. [8]

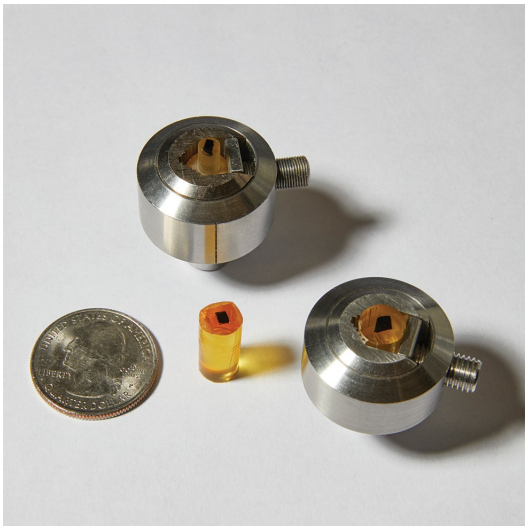


Figure 2: Acrylic encasing for thin slicing. [8]

2 **Registration of image tiles:** The microscope produces numerous sections per image slice that need to be aligned (Fig. 3 & 4). The multibeam microscope produces 61 partially overlapping image sections for a single "multibeam field of view" (MFOVs) (Fig. 5). First, each of these MFOVs is aligned individually. Next, all of the MFOVs for a single slice of brain tissue are aligned (Fig. 6). After each image slice is aligned, they need to be registered to create a coherent 3D volume [5].

Figure 3: Misaligned images. [8]

Figure 4: Aligned images. [8]

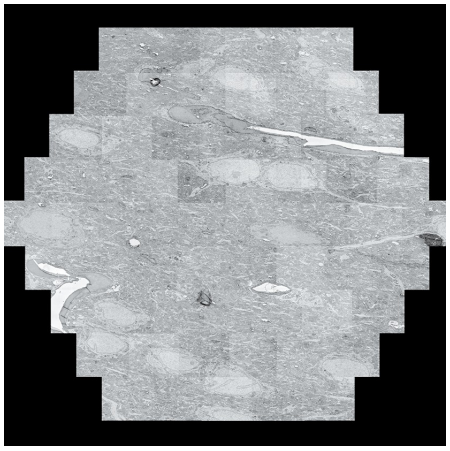


Figure 5: Multibeam field of view. [8]

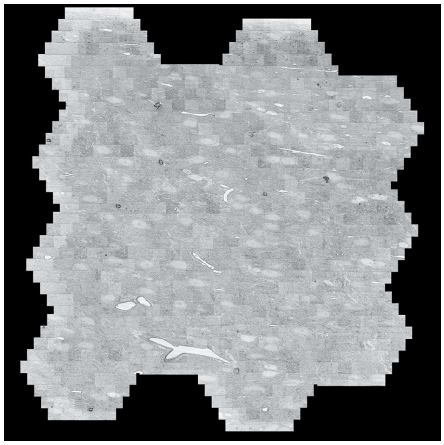


Figure 6: 2D alignment of fields. [8]

3 **Segmentation and Synapse Detection:** The generated image volumes (Fig. 7) are too large for manual segmentation leading to significant research in automatic labelings of neurons. In these label volumes, two voxels receive the same label if and only if they belong to the same neuron. Most current methods predict affinities between voxels using a 3D U-Net [9] and follow with a watershed transform to create supervoxels [10], i.e., small segments of voxels that belong to the same neuron. These supervoxels are agglomerated either using simple hierarchical clustering strategies [11, 12], or machine-learning techniques [13]. Flood-filling networks combine these two steps together and produce accurate reconstructions (Fig. 8) at a high computational cost [14, 15]. Tangentially but equally important for wiring diagram extraction, some research explores how to detect synapses in the volume [16].

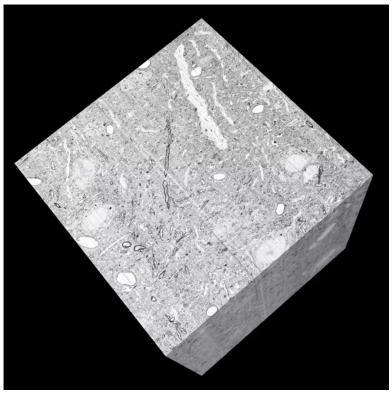


Figure 7: 3D reconstruction. [8]

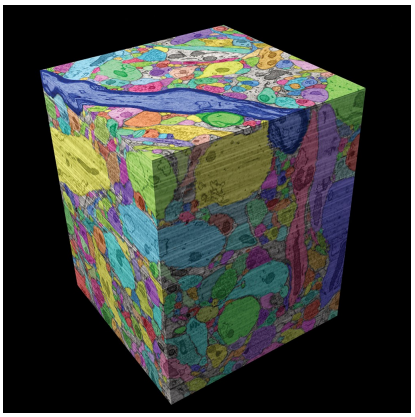


Figure 8: Final segmentation. [8]

4 **Proofreading:** Despite a decade of significant progress in automatic segmentation and synapse detection, the method still produce errors, particularly at large scales. Therefore, proofreading and error correction is an important component of the connectomics pipelines. Existing research either focuses on streamlining human involvement [17, 18], or creating automatic error correction methods [19, 20, 21].

5 **Analysis:** At this point, the connectome contains information such as label of the cell, properties (size, position or type) of the cell and synapse properties (pre and postsynaptic neuron label or distance between somas). All this information can be regarded as a graph with nodes corresponding to neurons and edges to the synapses between two neurons, the graph is ideally directed (information about pre and postsynaptic neuron available) and contains extra information in nodes and edges. From there, there exists many graph analysis tools that exist and that can be used to analyze the connectome, from visualization to motif discovery depending on the research focus [22, 23].

Figure 9 visually summarizes those five steps, from acquisition to analysis.

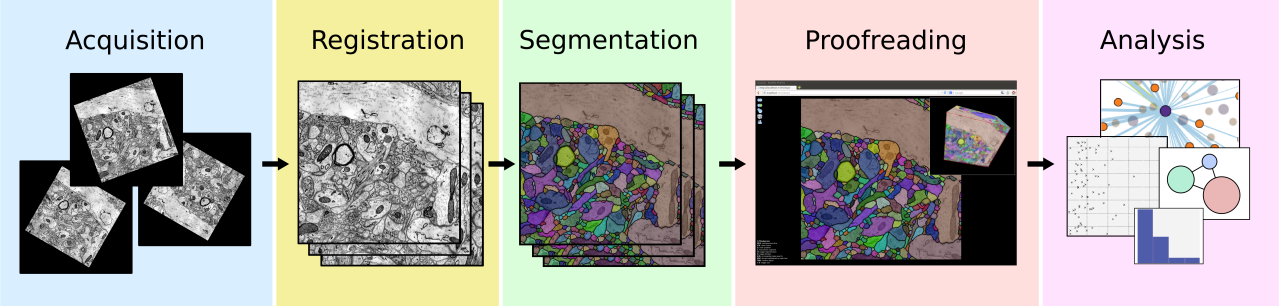


Figure 9: The typical connectomics workflow includes several steps: image tiles of brain tissue are *acquired* using an electron microscope, *registered* in 2D and 3D, and automatically *segmented* into neurons. Since the output of the automatic segmentation is not perfect, it is mandatory to *proofread* the result prior to any *analysis*.

1.2 Motif discovery

Neuroscientists have explored the brain from many levels. Some have focused at the microscopic level, trying to model the behavior of a neuron through several models such as the Hodgkin-Huxley model or the leaky integrate-and-fire model by carefully looking at chemical interactions happening in the the cell [24]. Others have focused on the mesoscopic level, understanding populations of neurons and their activity around other population using probabilistic models [25]. Finally, at the macroscopic level, researchers have worked hard to determine the role of different parts of the brain such as the visual thalamus, the hippocampus or the medulla. However, very few studies have been done on the structural interaction between neurons at the microscopic level. Motif discovery on brain networks could take us a step further in comprehending how the brain works.

During my thesis, I have focused on the connectome analysis and more specifically on the motif discovery in the wiring diagram of different connectomes. Over millions of years, the brain has evolved from a simple network to the very complex and large network that it is today. Systematic analysis of the connectome of C. Elegans or mammalian brains have shown that the topology in the brain network is not entirely random but exhibits populations of short path connections between components, revealing small-world properties. These properties can

be observed in many other networks such as the metabolic system, power-grids or voter networks as suggested by B. Uzzi & Al [26].

Recent studies as well as personal motif discovery experiments for simple structures (3, 4 nodes) on brain networks shows that there are a small number of basic motifs very frequently identified in the brain. This supports the hypothesis that the small structures encountered in the brain are the building blocks of the global structure of the brain, each having a unique role to play in the functionality of the brain. In fact, many of these small subgraphs can be interpreted as functions with a very simple role, for example the feed-forward loop can be interpreted as a delayed 'AND' gate (short impulses will not trigger impulses in C, but persistent signal will) or oppositely as a pulse generator (if B inhibits C, C will not trigger but if impulses are emitted from A, B will inhibit C with delay) depending on the type of neuron B, excitatory or inhibitory respectively, as seen in Fig. 10. By identifying separately those motifs and then linking them could help gain insight as to how the brain works.

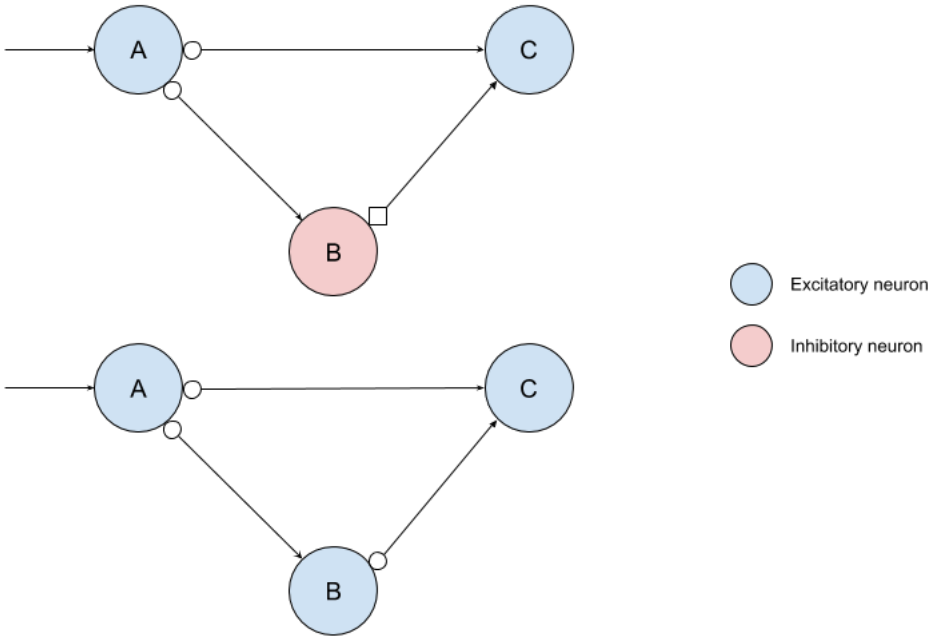


Figure 10: positive and negative feed forward loop.

1.3 Our contribution

During my master thesis, I have worked on developing an algorithm that approximates motif enumeration of large subgraphs by taking advantage of the small world property of certain networks. I believe this algorithm can faithfully retrieve the most frequent motifs in such graphs, as well as most of its occurrences, greatly reducing the computing time other enumeration techniques would need.

2 Related works

2.1 Motif Discovery

In the 1980s, the first milestone, implicitly in motif discovery, was achieved by McKay [27]. From previous work on canonical graph labelling, he built the *Nauty* algorithm to canonically label graphs faster than any existing methods at the time. Today, *Nauty* still is the state of the art algorithm for this problem [28] and is used in a large number of motif discovery algorithms.

Between the 80s and 90s, many algorithms were developed to enumerate motifs. These algorithms were meant to find specific motifs, especially cycles or cliques. Furthermore, sampling methods were developed for non-exact enumeration of other graph types such as spanning trees. Other algorithms were aiming at finding only certain frequent motifs in the original graphs, but the first algorithm to approximate motifs frequency in an undirected labeled graph came in 1995 from Duke & Al [29]. This method had strong constraint on the subgraph size for large networks and the runtime grows polynomially with the graph size.

The second big milestone was set by Milo & Al in 2002 [30], they were first to compute the exact enumeration of subgraphs in a graph. From there they were able to find the relative frequencies of particular subgraphs. They did so by comparing motif frequency in the original subgraph with motif frequency in similar randomly generated networks (Fig. 11). In their experiment, they were able to successfully extract size 3 and 4 network motifs (more than the mean random number of motifs plus twice the standard deviation in the random graphs) from different directed networks setting a milestone in the area of motif discovery. However, this approach was not computationally feasible for larger motifs and more work had to be put to reduce these limitations.

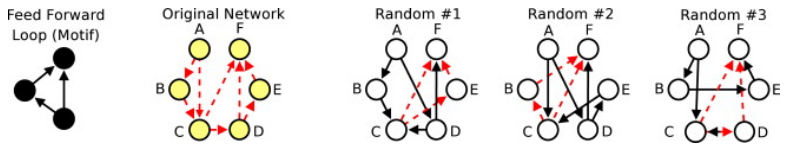


Figure 11: Randomly generated networks highlighting occurrences of the feed forward motif.

Two years later, Kashtan & Al [31] proposed an algorithm, based on Milo's previous work, called MFinder. The algorithm uses a sampling method to approximate the relative frequency of network motifs in the original graph. The motif is sampled by randomly choosing edges from a size k subgraph (k=2,..,n) until a motif of size n is found. The motif edges are then expanded to include those not previously sampled at random, and the probability that the motif is sampled is computed, this is done so that all motifs have the same likeliness of being sampled. The advantage of this new algorithm is that the sampling method is much faster than the exact enumeration of network motifs and is not dependant on the network size but rather on the subgraph size and on the number of sampled motifs. However, it has drawbacks, namely, the sampling can be greatly biased by the presence of 'hub-nodes', which have much more probability of being in the sampled motif than nodes with low degrees, also, there is no guarantee the motif will not be sampled several times, again biasing the approximation. Finally, the algorithm can only find motifs of size up to 6 due to its implementation.

One of the answer to mfinder came from Wernicke with the algorithms ESU and RAND-ESU as part of the tool FANMOD [32], deployed in 2006. His algorithm is based on an ESU-tree built

by recursively adding nodes from an "extension" set into a "subgraph" set, following certain conditions:

- If w_1 is a node distinct from the root, w_1 has a child for all vertices in its "extension" set, this insures all subgraphs are covered.
- For each node w in the tree, all nodes in the "extension" set of w are labelled larger than the smaller label in the "subgraph" set of the vertices, this ensures that no subgraph is found twice.
- Let w_1 and w_2 be two nodes in the ESU-tree with a common parent node, and w_1 comes before w_2, then "subgraph" set of w_1 contains exactly one vertex u_1 not in the "subgraph" set of w_2 and similarly for w_2. Furthermore, For every descendant w node of w_2, u_1 will not be in the "subgraph" set of w, this ensures that no subgraph is repeated neither.

Once the tree is built, as seen in Fig. 12, the different subgraphs present in the original graph are the leaves of the ESU-tree. The *Nauty* algorithm is then used to group all isomorphic subgraphs together before enumerating them. The sampling method RAND-ESU follows the same principle but instead of expanding each child at level d, they are expanded with probability p_d. This allows to partially explore the ESU-tree, greatly reducing the computation time using an unbiased sampling of subgraphs as all subgraphs have the same probability of being sampled ($\Sigma_d p_d$). Even though ESU is still widely used today and one of the fastest algorithm to find the exact relevance of motifs in a graph, the sampling method has drawbacks. Indeed, if a node is pruned in the first layers of the ESU-tree, there is a big chance that an entire area of the graph will be briefly explored while another would be extensively explored, furthermore, defining p_d takes experience and can impact greatly on the number of leaves in the ESU-tree (pruning subtrees whose root is close to the root of ESU-tree has a higher influence than pruning subtrees whose root are farther from it) as well as on the running time (pruning more in lower levels will induce more nodes in the tree than pruning in the higher levels and therefore increase the running time). Moreover, FANMOD has a motif size limit of 9 nodes due its implementation.

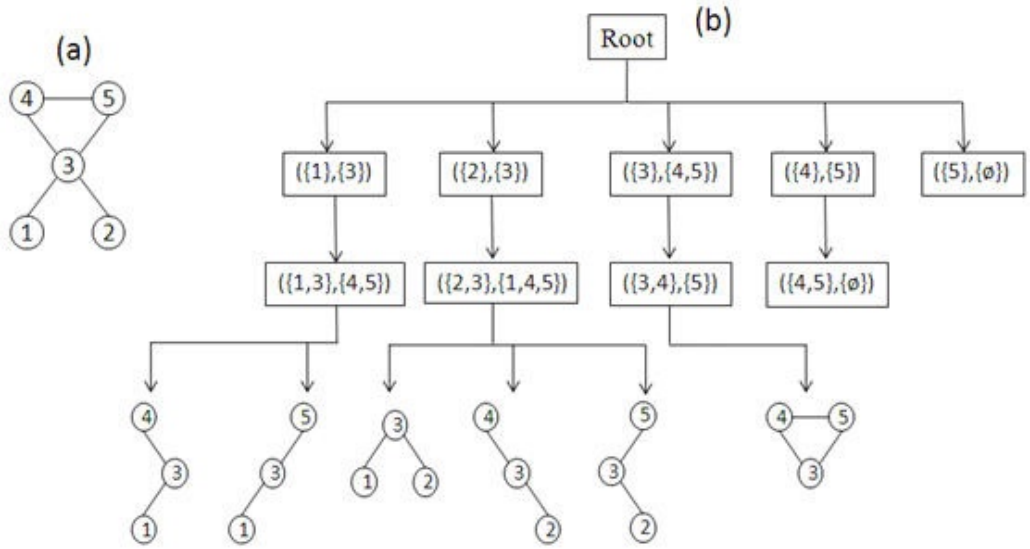


Figure 12: ESU tree with the example graph it is built from.

In 2009, Kashani & Al introduced a new algorithm called *Kavosh* to find exact enumeration of network motifs in undirected and directed networks [33]. The algorithm first find all size k

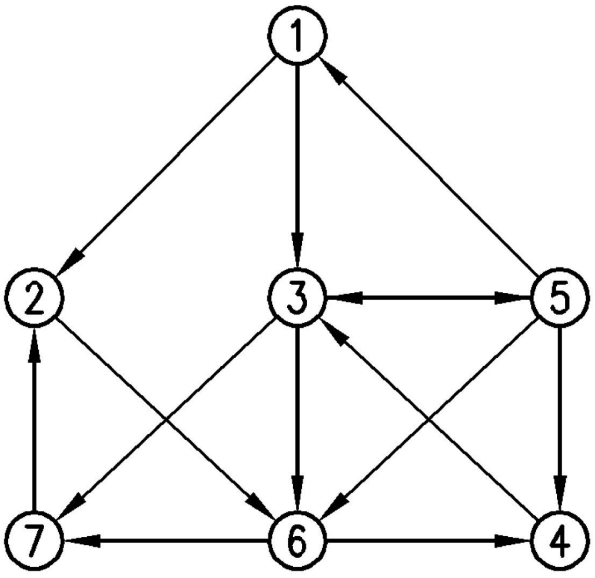


Figure 13: Example graph used to build the trees in Fig. 14

subgraphs that a vertex u participates in and then moves on to a different node, removing u from the graph. For each node in the graph, the algorithm builds trees with maximum depth of k following certain rules when descending the tree, ensuring that subgraphs are not duplicated in the enumeration:

- A child can be included in a particular level of the tree only if it has not been included in the previous levels.
- All children in a tree must have numerical labels greater than the root of the tree.
- When a tree has been descended to the lowest level, it is ascended again and the process is repeated and vertices visited in earlier paths are now considered unvisited.

In order to find all occurrences of size k subgraphs, all possible compositions of the integer k-1 must be considered. To do this, the algorithm uses the "revolving door algorithm" ensuring that all combinations of $k_2, k_3, ..., k_m$ such that $k_i = k - 1$ are considered. To enumerate subgraphs based on the composition, k_i nodes are selected from the ith level of the tree and added to the root to be part of the subgraphs ($i = 2, 3, .., m$). An example graph and the corresponding trees are shown in Fig. 13 and 14 respectively. The subgraph is then classified using the *Nauty* algorithm. Motif relevance is then computed by repeating the subgraph enumeration on random networks much like Milo & Al did in their tool MFinder. *Kavosh* is today the most efficient algorithm, with low memory usage and relatively low running time. Moreover, *Kavosh* does not have a limit on the size of the network motif size.

Few months later, Ribeiro & Silva released a tool based on a new data structure used to store subgraphs in an efficient manner called GTrie [34]. GTries are conceptually similar to prefix trees, taking advantage of the common substructures of network motif. Each node in the GTrie stores information about a vertex in the subgraph and its connections to ancestors in the tree. The tree is built by inserting one subgraph at a time, starting with an empty tree until all size k subgraphs are inserted, as shown in Fig. 15. In order to cope with the different isomorphic classes, each subgraph adjacency matrix is first transformed into its canonical representation so that each leaf of the tree corresponds to a non repeated isomorphic class within the other leaves. This structure is especially efficient when looking for small subgraphs and

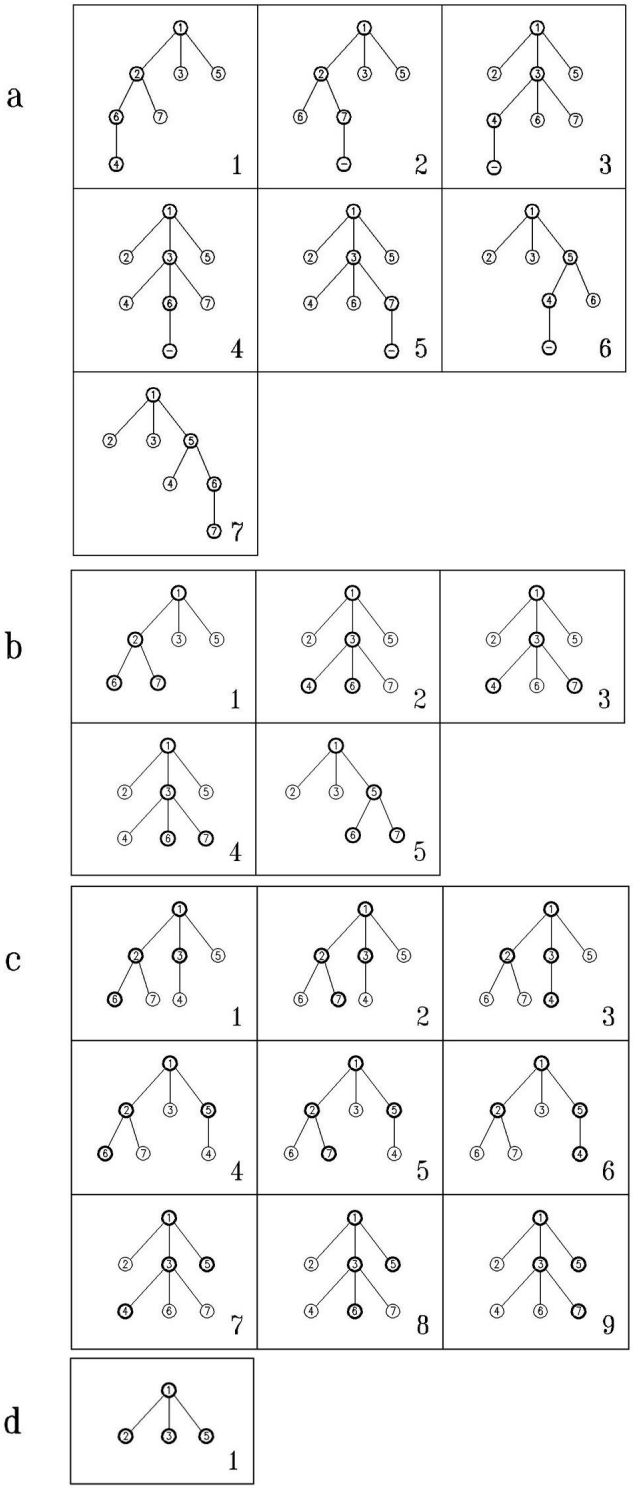


Figure 14: All Kavosh trees built by the algorithm.

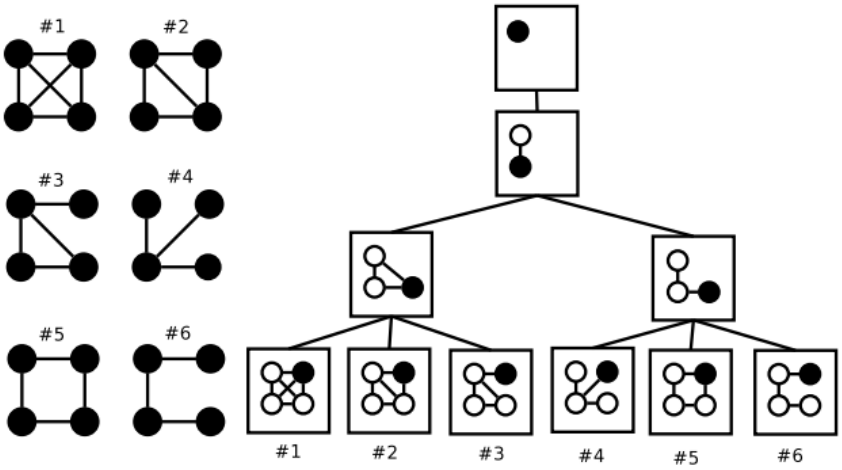


Figure 15: GTrie built from the 6 possible undirected 4 nodes graphs

for a high number of random networks when computing motif relevance. In fact, once the tree is built (for any subgraph size k), the enumeration is done by inserting subgraphs one by one in the tree. This method can be fast, especially if the set of subgraphs found in the original network is small compared to the set of possible subgraphs of the same size (the subgraphs not present in the original network will be discarded if present in the random networks). Moreover, the implemented tool, called *gtrieScanner*, allows to output the occurrences of each subgraph with the included nodes, feature that the other algorithms do not offer.

2.2 Multicut clustering

Graph decomposition or clustering (Fig. 16) can be formulated as a Minimum Cost Multicut Problem (MP), well known in image segmentation [35, 36]. This formulation has many advantages. First, the feasible solutions of the MP problem match the decomposition of a graph, with the number of compositions given by the solution. Second, the problem is very easily stated, taking as argument just an adjacency matrix [37]. However, this problem is NP hard and therefore any exact solution is computationally very demanding, few algorithms have been proposed to approximate the solution to this problem.

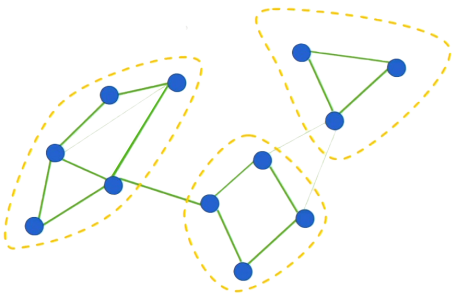


Figure 16: Graph clustering problem.

In 2015, Keuper & Al proposed a couple of algorithms to solve a generalized MP problem called Lifted Multicut Problem, where the objective function is redefined to overcome the fact that there are no long-range term present in it in the original formulation [37]. The first algorithm, Greedy Additive Edge Contraction, it starts from as many compositions as there are

nodes, then iteratively, nodes are grouped together to maximally reduce the objective function until it is at minimum. The second is an extension of the original Kernighan-Lin algorithm [38] that takes as input an initial composition and at each iteration tries to improve the partition, maximizing the decrease in the objective function, by greedily trying three different composition transformations iteratively: 1. move nodes from two neighbouring components; 2. move nodes from a component to a new one; 3. join two components.

3 Methods

3.1 Notation

A network considered as a graph is defined by a set of vertices and edges, $G = V, E$. V is a finite set of vertices, and E is a finite set of edges, $E \subseteq (V, V)$. Each edge $e_{ij} \in E$ is defined as (v_i, v_j) where $v_i, v_j \in E, i \neq j$ (self loops are not considered in our case). v_i and v_j are *adjacent* and in a directed graph, v_i is called the source and v_j the target of the edge. A subgraph of G is defined as $G' = V', E'$ with $V' \subseteq V$ and $E' \subseteq (V', V') \cap E$. The subgraph size is defined as the number of vertices in V'. A graph is weighted when each edge is assigned with a positive weight w_{ij} from v_i to v_j. For a weighted graph, a in-degree and an out-degree can be defined for each vertex. The in-degree d_{in} of a vertex u is the total number of edges from which u is the target, similarly, the out-degree is the number of edges for which u is the source. The total degree of a vertex can finally be defined as the sum of the in-degree and out-degree.

Two subgraphs $G_1 = V_1, E_1$ and $G_2 = V_2, E_2$ are isomorphic if there is a one to one correspondence F between V_1 and V_2 (e.g $V_{1_1} : V_{2_3}; V_{1_2} : V_{2_2}; V_{1_3} : V_{2_1}$) with $F(V_1) = V_2$ that also satisfies $E_2 = F'(E_1) = (F(V_{1_i}), F(V_{1_j}))$ for all edges in E_1. Two isomorphic graphs G_1 and G_2 are said to belong to the same isomorphic class I_i. The frequency of an isomorphic class I_i is defined as the number of subgraphs found in the graph that belong to I_i. An isomorphic class is said to be a motif if its frequency is higher than in similar random graphs. A metric, Z-Score, is defined for this purpose $Z_m = (f_m - \tilde{f}_m)/\sigma_m$, for each motif m, and where $\tilde{f}_m$ is the mean frequency of motif m in n random graphs, and σ_m the standard deviation of the frequency of motif m in those same random graphs.

Graph clustering is the task of grouping together vertices within a graph. Usually the clustering is done so that vertices that are highly connected are separated from other highly connected groups of vertices, however, the cutting criteria can be different and take advantage of edge weights, minimizing the total weights between compositions. The cut is defined as $C(V)$ where C maps every node V_i to a cluster c, $c = 1, ..., n_c$.

3.2 Algorithm

3.2.1 Small subgraph enumeration

The first step of the algorithm is to build a metric from which to cluster the different populations. In our algorithm, we define new edges that are built up from the sum of different adjacency matrices. First, we use a *GTrie* motif enumeration for subgraphs of size 3, 4 and 5 on the entire graphs, outputting the occurrences of each. From those, we create an adjacency matrix by adding 1 to the adjacency matrix each time two nodes are found in the same motif. For each motif, we give a weight computed from the Z-Score of the motif. If the motif has a Z-Score smaller than 1, the weight is set to zero. We finally add to this adjacency matrix the original adjacency matrix, giving a positive weight to edges that exists in the original graph.

3.2.2 Clustering

After we obtained the modified adjacency matrix, we apply the second algorithm proposed by Kernighan & Al [38] (modified Kernighan-Lin) using the modified adjacency matrix. The algorithm classifies nodes together in order to minimize the cost (sum of removed edges). From the colors obtained through the clustering algorithm, we create k new graphs for each different

cluster. By doing the clustering in such a way, the most frequent small subgraphs should all be included in populations, cutting the graph as much as possible on non-relevant subgraphs.

3.2.3 Large graph enumeration

Once the original graph is divided into several smaller subgraphs, the enumeration of large subgraphs is conducted on each of them. The frequency of each isomorphic class is finally summed for all clusters to obtain the final frequencies for each subgraph.

3.2.4 Large graph motif relevance

Finally, to support our algorithm, the frequency of large subgraphs obtained in the original network will be compared to random networks.A routine has been developed (Algo. 1) to generate random graphs with small-world properties. The probabilities of having an edge between any two populations (edge inside a same cluster as well) in the base network is computed and, starting from an empty edge list, this probability is applied between every pair of nodes to create random edges.
Once this routine has been applied to create enough random networks, we can repeat the enumeration process for large graphs using the random ones. From the subgraph frequency, we can then compute the Z-Score for each of them, indicating if the subgraph is specific to our graph or not.

Algorithm 1 Create a random network with small-world properties:

Require: an edge list E of the original graph and a labelling $C(V)$ of the vertices.
  for all permutations of 2 clusters, ij; $i, j = 1, ..., n_c$ **do**
    $n_e = \sum_{v_k, v_l \| v_k \in c_i, v_l \in c_j} \mathbb{1}\{e_{kl} \in E\}$ (number of existing edges between both clusters)
    if $i = j$ **then**
      $n_{tot} = \sum_{v_k, v_l \| v_k, v_l \in c_i; k \neq l} 1$ (number of possible edges within a cluster)
    else
      $n_{tot} = \sum_{v_k, v_l \| v_k \in c_i, v_l \in c_j} 1$ (number of possible edges between two clusters)
    end if
    $p_{ij} \leftarrow \frac{n_e}{n_{tot}}$
  end for
  $E_r \leftarrow \{\}$
  for node u in N **do**
    for node v in $N \setminus u$ **do**
      Add (u, v) to E_r with probability p_{ij}, where $i = C(u)$ and $j = C(v)$.
    end for
  end for

4 Experiments

4.1 Datasets

To test the algorithm, three graphs have been used. For the purpose of the algorithm, the graphs should possess small-world properties, as it is the case in brain wiring diagrams or community interaction graphs.

4.1.1 *LGN* dataset

The first dataset that was used to test our algorithm is referred to as the *LGN* dataset. This wiring diagram was taken from a mouse dorsal lateral geniculate nucleus (LGN) [39], extracted by J. Morgan in the Lichtman lab at Harvard. This part of the brain is taken from the visual thalamus, which is a major relay center, receiving major input from the retina. In the extraction used, the wiring diagram is composed of 412 nodes and 820 directed edges. The graph has a tree-like structure (Fig. 17), with few nodes going back and forth. This segmentation has been thoroughly proofread by experts in the neuroscience field.

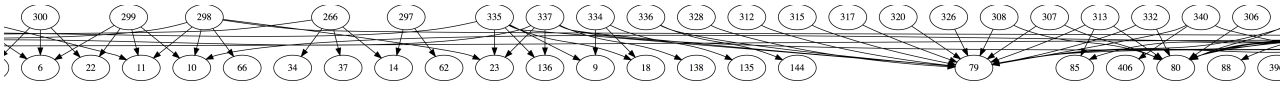


Figure 17: Visualization of sampled nodes from the LGN wiring diagram.

4.1.2 *email* dataset

The second dataset that was used to test our algorithm is referred to as the *email* dataset. It was taken from the Stanford large network dataset collection [40]. The graph represent emails sent between members from a large European institution. There is an edge (u_1, u_2) if employee u_1 has sent at least one email to employee u_2. Emails sent to and from the outside of the company are not included. Each individual in the graph belongs to one of the 42 departments - label of the department is available for each individual. This graph clearly exhibits small world properties with over 34% of emails sent between two employees of the same department. Relations between department 2 and 3 can be seen in Fig. 18. The full graph contains 986 nodes and 24929 edges after removing self loops. In order to reduce computing time on the algorithm, only 4 departments are considered, leaving 72 nodes and 595 edges.

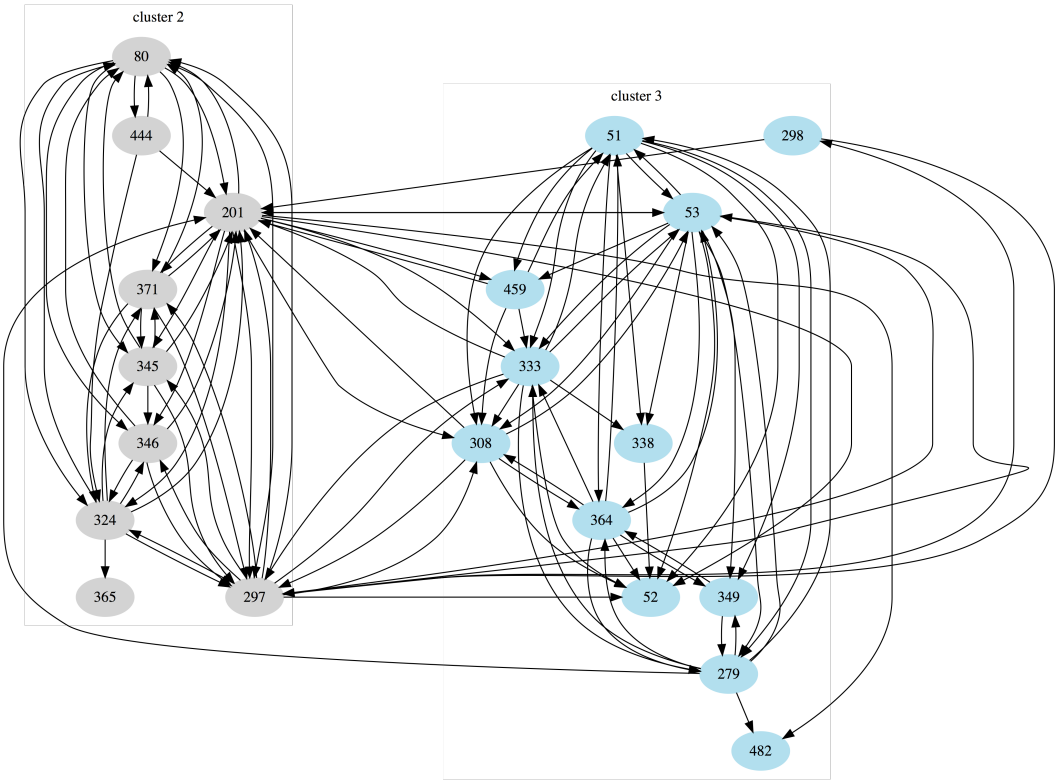


Figure 18: Emails sent within and between two departments (2 and 3).

4.1.3 *Fib-25* dataset

The third dataset that was used to test our algorithm is referred to as the *Fib-25* dataset. This dataset is extracted from EM data from the FlyEM project at HHMI Janelia [41]. The dataset contains 8000 images taken from the medulla oblongata of a fruit fly. From these images, cells have been segmented and synapses identified and the segmentation has been densely proofread. The original segmentation features 798 neurons segments and over 31000 synapses. We applied a filter to keep only pairs of segments which have at least 10 synapses between them, thus reducing the graph to 279 cells or nodes and 441 unique directed connections or edges.

4.2 Setup

All performance experiments ran on an Intel Core i7-6800K CPU 3.40 GHz with a Titan X Pascal GPU.

5 Results

In this section, small motif discovery on brain networks that inspired the development of the algorithm will be presented first. Then, computing performance of the *Kavosh* and *gtrieScanner* algorithms will be shown for all three graphs. Finally, large motif enumeration for both presented networks as well as motif relevance on the *LGN* network will be shown.

5.1 Motif discovery on brain networks

This section features subgraphs or motifs that have a particular role to play in the two brain networks that we consider. Motifs are defined as subgraphs that have a frequency much larger in the original network than in similar random networks, or in other term, a large Z-Score. From both the *LGN* and *Fib-25* we can apply the previously cited algorithm *gtrieScanner* and look at subgraph frequency and Z-Scores. The number of subgraphs found in each graph of size 4 and 5 as well as the number of isomorphic groups identified are registered in Table 1. The small number of isomorphic groups in the *LGN* graph is due to the tree like structure, preventing many isomorphic groups from occurring.

	LGN		Fib-25	
Motif size	Subgraphs	Isomorphic classes	Subgraphs	Isomorphic classes
4	89641	18	17239	91
5	1289357	85	157160	666

Table 1: Number of subgraphs and isomorphic groups enumerated from the *LGN* and *Fib-25* graphs.

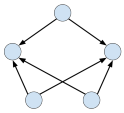
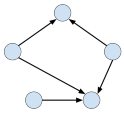
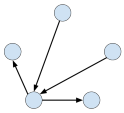
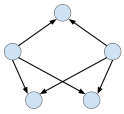
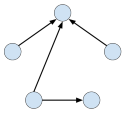
		LGN		Fib-25	
Motif ID	Motif	normalized frequency	Z-Score	normalized frequency	Z-Score
Size 4 motifs					
4.1		1.8%	7.7	0.68%	14.1
4.2		0.15%	4.0	1.6%	5.2
4.3		42%	-10.9	18%	-3.9
Size 5 motifs					
5.1		0.15%	1.4	0.06%	52.1
5.2		3.0%	1.1	0.79%	10.7
5.3		0.04%	-2.3	6.3%	-0.7
5.4		0.08%	14.57	0.01%	9.4
5.5		37%	-6.9	6.3%	-2.7

Table 2: Different motifs of size 4 and 5 found in both brain networks: *LGN* and *Fib-25*.

The results in Table 2 show that the two brain networks are very similar. Both networks exhibit a high Z-Score for the first two size 4 motifs (4.1 and 4.2), as well as a non negligible frequency. Large Z-Scores in motif of size 5 are credited to motifs that include the top Z-Score size 4 motif (5.1, 5.2 and 5.4), these motifs still have high frequencies compared to other subgraphs of similar Z-Score. Furthermore, the most frequent size 4 motif (4.3) is the same in both graphs with a large negative value for both, this is also the case for size motifs, where the most frequent ones (5.3 and 5.5) have both low Z-Score.

5.2 Computing performance

Comparing the computation performance of both algorithms can help understand how to address the problem of finding larger motifs in such networks. In a first step, the three datasets are enumerated for subgraphs of size 4 and 5 using *Kavosh* and *gtrieScanner*, and the running time for each is recorded using 1000 random networks to compute the Z-Score. Both the run times for the original graph and one random graph are also written down.

The results for the *LGN* graph are shown in Table 3 below. For reminder, the graph possesses 412 nodes and 820 edges.

	Total Running time		Original graph		Single random graph	
Motif size	Kavosh	gtrieScanner	Kavosh	gtrieScanner	Kavosh	gtrieScanner
4	61.7	2.42	0.067	0.074	0.062	0.0023
5	1910	36.0	1.5	1.3	1.9	0.035

Table 3: Runtime (in seconds) of both algorithms with 1000 random networks on *LGN* dataset for motifs of size 4 and 5.

The same results are shown for the *Fib-25* dataset in table 4. This dataset has much less nodes or edges and the enumeration is therefore much faster.

	Total Running time		Original graph		Single random graph	
Motif size	Kavosh	gtrieScanner	Kavosh	gtrieScanner	Kavosh	gtrieScanner
4	5.66	1.06	0.013	0.016	0.0056	0.0010
5	56.0	8.61	0.16	0.16	0.056	0.0085

Table 4: Runtime (in seconds) of both algorithms with 1000 random networks on *Fib-25* dataset for motifs of size 4 and 5.

Table 5 shows the running time on the last dataset *email*. Even though the number of nodes is smaller than in the *Fib-25*, the number of edges is larger and the number of subgraphs in the graph as well. Indeed, there are almost twice the number of subgraphs of size 4 and 5 in the *email* dataset than in the *Fib-25*.

	Total Running time		Original graph		Single random graph	
Motif size	Kavosh	gtrieScanner	Kavosh	gtrieScanner	Kavosh	gtrieScanner
4	18.7	11.7	0.022	0.021	0.019	0.012
5	228	137	0.15	0.15	0.23	0.14

Table 5: Runtime (in seconds) of both algorithms with 1000 random networks on *email* dataset for motifs of size 4 and 5.

Even though both algorithms are equivalent when enumerating subgraphs on the original graph, *gtrieScanner* is much faster to compute each random graph, making it the fastest tool to compute Z-Scores on graphs of size 4 and 5. Furthermore, there is an increase in computing time as the size of the graph increases, especially the number of edges. Indeed, the *email* dataset has a smaller number of nodes but a larger number of edges than the *Fib-25* graph, and the time spent on both algorithms is much larger for the former.

When enumerating larger subgraphs, the two algorithms behave differently. *Kavosh* is well optimized for this purpose and the increase of time for the graph enumeration is less than for *gtrieScanner*, as can be seen in Fig. 19. However, *gtrieScanner* makes up for this weakness enumerating random graphs. The time required to enumerate a random graphs increases very little for *gtrieScanner* compared to *Kavosh* as evidenced in Fig. 20.

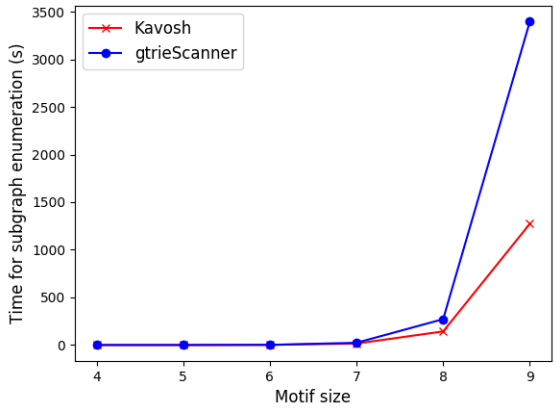


Figure 19: Evolution of subgraph enumeration with motif size using *Kavosh* and *gtrieScanner* on *Fib-25* graph.

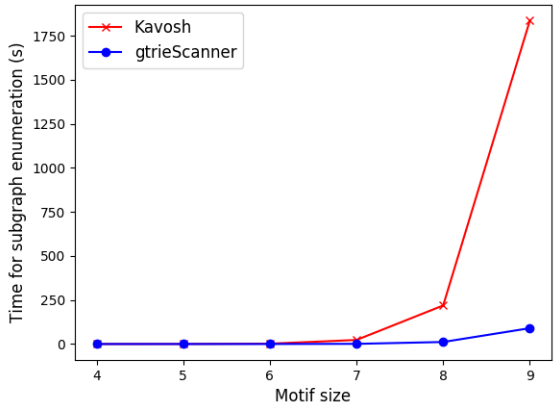


Figure 20: Evolution of subgraph enumeration with motif size using *Kavosh* and *gtrieScanner* on a randomly generated graph from *Fib-25*.

5.3 Large motif enumeration

In this section, the results of the large subgraph enumeration after clustering will be revealed. For the purpose of the algorithm, only the *LGN* and *email* datasets have been used.

5.3.1 The *LGN* dataset

The first step of the algorithm is to cluster, using the previously computed frequencies and Z-Scores of size 4 and 5 motifs, the original graph. From the clustering algorithm, we obtain 17 clusters, from which only 9 contain internal edges and 5 of them have more than 10 nodes. The clusters and their properties are listed in Table 6.

Cluster ID	Number of nodes	Number of edges
0	200	345
1	38	45
2	20	19
3	5	4
5	10	7
8	10	9
10	36	43
15	4	3
16	78	70

Table 6: Cluster properties

From now, only clusters with more than 10 nodes (clusters 0,1,2,10 & 16) will be considered. From these 5 clusters, subgraphs of size 8 are enumerated and compared with the actual subgraph frequency in the original graph. A total of $3'622'227'539$ subgraphs occur in the original graph from a total $13'068$ isomorphic groups, whereas the clusters account for $1'633'037'684$ subgraphs in $11'230$ groups.
From there, isomorphic groups from the original enumeration are identified with high frequency as well as high Z-Score and counted in the different clusters. Eight of the most relevant motifs

have been studied and the results are shown in Table. 7.

Motif ID	Adjacency matrix	Z-Score	Original frequency	Clusters frequency
1	00110000 10110010 00000000 00000000 10110000 10010010 00100000 00100000	5100	510	510
2	00110000 10110000 00000000 00000000 10110000 10110000 10110000 00100000	307	2385	2385
3	00110000 10110000 00000001 00000000 10110000 10110000 00100000 00000000	115	1945	1945
4	00110000 10110000 00000000 00000000 10110000 10110000 00100000 00100000	83.3	32235	32235
5	00000000 10001000 11001100 11001000 00000000 10000000 10000000 10000000	72.8	8415	8415
6	00110001 10110000 00000000 00000000 10110000 10110000 00100000 00000000	56.7	10055	10055
7	01001000 00000000 11000000 11000000 00000000 00001010 00000000 00000010	39.3	49182	465
8	00000000 10001000 11001000 11001000 00000000 10000000 10000000 10000000	22.9	186620	186620

Table 7: Comparison between original frequency and clusters frequency for high Z-Score motifs.

The results show that for 7 out of the 8 considered motifs, all the occurrences have been completely conserved by the clustering algorithm. One motif has not been conserved and very few occurrences have been collected on the clusters. Furthermore, it can be noted that motif 5.1 from Table 2 appears in most of the considered isomorphic groups, namely motifs 1,2,3,4 and 6. While it does not appear in motifs 5,7 and 8, the 4.1 is contained by those motifs.

In terms of time performance, the original size 8 subgraph enumeration takes $15'860s$, the equivalent enumeration on the clusters takes $4'821s$ without taking into account the Z-Score

calculations for size 4 and 5 motifs (original enumeration and 1000 random networks), presented in Table 3. With the clustering algorithm, which takes $4s$, the entire enumeration takes a total of $4860s$.

5.3.2 The *email* dataset

From the clustering approach explained in section 3.2.4, we obtain 21 different node clusters, most of which contain no edge. After filtering, we are left with 7 clusters containing edges, and 3 containing at least 8 nodes. Properties of the clusters are indexed in Table 8.

cluster ID	Number of nodes	Number of edges
0	19	108
1	11	72
3	2	1
5	6	17
6	20	265
9	2	1
11	2	1

Table 8: Cluster properties

After enumeration on the original *email* graph, $186'650'317$ size 8 subgraphs occurrences have been listed for a total of $7'848'039$ isomorphic classes. From clusters 0, 1 and 6, only $126'499$ subgraphs are identified. Because the number of subgraphs enumerated are so different between the original and the clusters, no further analysis has been carried out.

6 Discussion

The first results of this thesis showed that brain networks display small-world like properties, through small motif enumeration. Brain wiring diagrams are not entirely random nor are they entirely established. Many subgraphs are found to be very populated in these networks while others are very unlikely to be found compared to completely random similar graphs. The results presented here are particularly important as they are some of the first to be extracted from actual brain wiring diagrams. They offer a view of what are the main computational units that build up our brain. However, having a broader vision by considering the most relevant motifs of larger sizes could be of even more use to understand the brain. By looking at larger motifs, more complex functional units can be identified and more mysteries about the brain could be solved.

Comparing both *Kavosh* and *gtrieScanner* algorithms have shown that although *Kavosh* deals much better with large motifs, the tree structure of *gtrieScanner* makes it much faster to enumerate randomly generated networks from the pre-built tree. However, they both have exponentially increasing run time with increasing motif size as well as graph size, especially the number of edges. This is especially important as the surface has barely been scraped in terms of wiring diagram sizes that are produced. With new microscopes and imaging technologies as well as improvements in 3D segmentation and proofreading softwares, access to very large (millions of edges) and reliable wiring diagrams will become easier.

To tackle this issue, an algorithm consisting of graph clustering before enumeration has been proposed during this project. The clustering algorithm takes as input an adjacency matrix of the graph, which can be modified to encompass more information about the graph, such as motifs. This algorithm proved to be worthy, reducing time for subgraph enumeration by 2/3 from the original size 8 subgraph enumeration, while finding the vast majority of the relevant subgraphs. Nevertheless, the algorithm has shown weaknesses on the *email* dataset. The graph has an average total degree of 16.5, so removing one node from the graph greatly reduces the number of possible subgraph occurrences. Therefore, clustering on graphs which have high total degree has a great impact on the population of motifs in each clusters.

While results on the *LGN* dataset are satisfactory, the clustering algorithm yields very uneven clusters. In fact, cluster 0 from Table 6 has almost twice the number of edges and more nodes than all other clusters put together. This is in part the reason there are so little subgraphs missing from the clusters: the clustering algorithm trimmed the graph to keep the relevant large part. Also, the clustering algorithm [38] requires some experience to adapt the input adjacency matrix to improve the quality of the clusters.

The main concern with the algorithm is finding a proper method to cluster the graph. When using this clustering algorithm, there are many ways the adjacency matrix can be manipulated to include other graph characteristics. For example, a weighted motif based adjacency matrix, based on the Z-Score and frequency of each motif present in the graph, can be built. The approach in this project was to use the previously cited algorithm, but other algorithms to solve this problem also exist. There are several graph clustering algorithms, especially using motifs as a variable [42], which can be tried to obtain a more robust clustering step.

Other solutions to reduce the computing time of motif enumeration is to use sampling methods [43, 44, 45]. These methods approximate the subgraph count by considering only a subset of the nodes in the graph, and inferring statistics from this subset. These methods are very

fast and received a lot of attention lately. However, they have their own drawbacks and usually never take into account the inherent structure of the graph, such as the one exhibited by brain wiring diagrams.

Concerning future perspectives, the next steps to adopt will be to try motif discovery tools on larger and more diverse connectomes. By continuing further down in this direction, more motifs will be identified with more certainty. Over time, those functional units will explain more of the brain's behavior. Another perspective that should be taken into account is the computing time of the tools used. Indeed acquisition of brain wiring diagrams will be more widespread, and their number and actual sizes will increase drastically. Faster algorithms are thus going to be needed. By continuing developing new algorithms to enumerate motifs, the field will be able to keep up pace with the amount of ongoing generated data.

7 Conclusion

To conclude, brain wiring diagrams are becoming increasingly easy to generate and accessible. Motif discovery tools need to be adapted for these new networks and taking advantage of the intrinsic features will help improve these tools.

Being able to reliably identify small blocks that build up the brain at the microscopic level could lead to the discovery of underlying mechanisms that govern certain neurological diseases, such as Alzheimer's. Besides, these computing units are used to develop bio-inspired artificial intelligence, always more human.

References

[1] Jade Marsh and Pavlos Alifragis. Synaptic dysfunction in alzheimer's disease: the effects of amyloid beta on synaptic vesicle dynamics as a novel target for therapeutic intervention. *Neural regeneration research*, 13(4):616, 2018.

[2] Jerome Engel Jr, Paul M Thompson, John M Stern, Richard J Staba, Anatol Bragin, and Istvan Mody. Connectomics and epilepsy. *Current opinion in neurology*, 26(2):186, 2013.

[3] Menachem Fromer, Andrew J Pocklington, David H Kavanagh, Hywel J Williams, Sarah Dwyer, Padhraig Gormley, Lyudmila Georgieva, Elliott Rees, Priit Palta, Douglas M Ruderfer, et al. De novo mutations in schizophrenia implicate synaptic networks. *Nature*, 506(7487):179, 2014.

[4] Narayanan Kasthuri, Kenneth Jeffrey Hayworth, Daniel Raimund Berger, Richard Lee Schalek, José Angel Conchello, Seymour Knowles-Barley, Dongil Lee, Amelio Vázquez-Reina, Verena Kaynig, Thouis Raymond Jones, et al. Saturated reconstruction of a volume of neocortex. *Cell*, 162(3):648–661, 2015.

[5] Adi Suissa-Peleg, Daniel Haehn, Seymour Knowles-Barley, Verena Kaynig, Thouis R Jones, Alyssa Wilson, Richard Schalek, Jeffery W Lichtman, and Hanspeter Pfister. Automatic neural reconstruction from petavoxel of electron microscopy data. *Microscopy and Microanalysis*, 22(S3):536–537, 2016.

[6] Daniel Haehn, John Hoffer, Brian Matejek, Adi Suissa-Peleg, Ali Al-Awami, Lee Kamentsky, Felix Gonda, Eagon Meng, William Zhang, Richard Schalek, et al. Scalable interactive visualization for connectomics. In *Informatics*, volume 4, page 29. Multidisciplinary Digital Publishing Institute, 2017.

[7] AL Eberle, S Mikula, R Schalek, J Lichtman, ML Knothe Tate, and D Zeidler. High-resolution, high-throughput imaging with a multibeam scanning electron microscope. *Journal of microscopy*, 259(2):114–120, 2015.

[8] MS Windows NT kernel description. `http://web.archive.org/web/20080207010024/http://www.808multimedia.com/winnt/kernel.htm`. Accessed: 2010-09-30.

[9] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *International conference on medical image computing and computer-assisted intervention*, pages 424–432. Springer, 2016.

[10] Toufiq Parag, Fabian Tschopp, William Grisaitis, Srinivas C Turaga, Xuewen Zhang, Brian Matejek, Lee Kamentsky, Jeff W Lichtman, and Hanspeter Pfister. Anisotropic em segmentation by 3d affinity learning and agglomeration. *arXiv preprint arXiv:1707.08935*, 2017.

[11] Kisuk Lee, Jonathan Zung, Peter Li, Viren Jain, and H Sebastian Seung. Superhuman accuracy on the snemi3d connectomics challenge. *arXiv preprint arXiv:1706.00120*, 2017.

[12] Jan Funke, Fabian David Tschopp, William Grisaitis, Arlo Sheridan, Chandan Singh, Stephan Saalfeld, and Srinivas C Turaga. A deep structured learning approach towards automating connectome reconstruction from 3d electron micrographs. *arXiv preprint arXiv:1709.02974*, 2017.

[13] Juan Nunez-Iglesias, Ryan Kennedy, Stephen M Plaza, Anirban Chakraborty, and William T Katz. Graph-based active learning of agglomeration (gala): a python library to segment 2d and 3d neuroimages. *Frontiers in neuroinformatics*, 8:34, 2014.

[14] Yaron Meirovitch, Alexander Matveev, Hayk Saribekyan, David Budden, David Rolnick, Gergely Odor, Seymour Knowles-Barley, Thouis Raymond Jones, Hanspeter Pfister, Jeff William Lichtman, et al. A multi-pass approach to large-scale connectomics. *arXiv preprint arXiv:1612.02120*, 2016.

[15] Michał Januszewski, Jörgen Kornfeld, Peter H Li, Art Pope, Tim Blakely, Larry Lindsey, Jeremy Maitin-Shepard, Mike Tyka, Winfried Denk, and Viren Jain. High-precision automated reconstruction of neurons with flood-filling networks. *Nature methods*, 15(8):605, 2018.

[16] Benedikt Staffler, Manuel Berning, Kevin M Boergens, Anjali Gour, Patrick van der Smagt, and Moritz Helmstaedter. Synem, automated synapse detection for connectomics. *Elife*, 6:e26414, 2017.

[17] Daniel Haehn, Verena Kaynig, James Tompkin, Jeff W Lichtman, and Hanspeter Pfister. Guided proofreading of automatic segmentations for connectomics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9319–9328, 2018.

[18] Daniel Haehn, Seymour Knowles-Barley, Mike Roberts, Johanna Beyer, Narayanan Kasthuri, Jeff W Lichtman, and Hanspeter Pfister. Design and evaluation of interactive proofreading tools for connectomics. *IEEE transactions on visualization and computer graphics*, 20(12):2466–2475, 2014.

[19] Brian Matejek, Daniel Haehn, Haidong Zhu, Donglai Wei, Toufiq Parag, and Hanspeter Pfister. Biologically-constrained graphs for global connectomics reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2089–2098, 2019.

[20] Jonathan Zung, Ignacio Tartavull, Kisuk Lee, and H Sebastian Seung. An error detection and correction framework for connectomics. In *Advances in Neural Information Processing Systems*, pages 6818–6829, 2017.

[21] Konstantin Dmitriev12, Toufiq Parag, Brian Matejek, Arie E Kaufman12, and Hanspeter Pfister. Efficient correction for em connectomics with skeletal representation. *British Machine Vision Conferemce (BMVC)*, 2018.

[22] A. Al-Awami, J. Beyer, H. Strobelt, N. Kasthuri, J.W. Lichtman, H. Pfister, and M. Hadwiger. Neurolines: A subway map metaphor for visualizing nanoscale neuronal connectivity. *IEEE Transactions on Visualization and Computer Graphics (Proceedings IEEE InfoVis 2014)*, 20(12):2369–2378, 2014.

[23] Ting Zhao, Donald J Olbris, Yang Yu, and Stephen M Plaza. Neutu: software for collaborative, large-scale, segmentation-based connectome reconstruction. *Frontiers in Neural Circuits*, 12, 2018.

[24] LF Abbott and Thomas B Kepler. Model neurons: from hodgkin-huxley to hopfield. In *Statistical mechanics of neural networks*, pages 5–18. Springer, 1990.

[25] Chi-Tin Shih, Olaf Sporns, and Ann-Shyn Chiang. Toward the drosophila connectome: structural analysis of the brain network. *BMC neuroscience*, 14(1):P63, 2013.

[26] Brian Uzzi, Luis AN Amaral, and Felix Reed-Tsochas. Small-world networks and management science research: A review. *European Management Review*, 4(2):77–91, 2007.

[27] Brendan D McKay et al. *Practical graph isomorphism*. Department of Computer Science, Vanderbilt University Tennessee, USA, 1981.

[28] Brendan D McKay and Adolfo Piperno. Practical graph isomorphism, ii. *Journal of Symbolic Computation*, 60:94–112, 2014.

[29] Richard A Duke, Hanno Lefmann, and Vojt e˘ ch Rödl. A fast approximation algorithm for computing the frequencies of subgraphs in a given graph. *SIAM Journal on Computing*, 24(3):598–620, 1995.

[30] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[31] Nadav Kashtan, Shalev Itzkovitz, Ron Milo, and Uri Alon. Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics*, 20(11):1746–1758, 2004.

[32] Sebastian Wernicke. Efficient detection of network motifs. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 3(4):347–359, 2006.

[33] Zahra Razaghi Moghadam Kashani, Hayedeh Ahrabian, Elahe Elahi, Abbas Nowzari-Dalini, Elnaz Saberi Ansari, Sahar Asadi, Shahin Mohammadi, Falk Schreiber, and Ali Masoudi-Nejad. Kavosh: a new algorithm for finding network motifs. *BMC bioinformatics*, 10(1):318, 2009.

[34] Pedro Ribeiro and Fernando Silva. G-tries: an efficient data structure for discovering network motifs. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 1559–1566. ACM, 2010.

[35] Jörg Hendrik Kappes, Markus Speth, Björn Andres, Gerhard Reinelt, and Christoph Schn. Globally optimal image partitioning by multicuts. In *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 31–44. Springer, 2011.

[36] Margret Keuper, Bjoern Andres, and Thomas Brox. Motion trajectory segmentation via minimum cost multicuts. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3271–3279, 2015.

[37] Margret Keuper, Evgeny Levinkov, Nicolas Bonneel, Guillaume Lavoué, Thomas Brox, and Bjorn Andres. Efficient decomposition of image and mesh graphs by lifted multicuts. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1751–1759, 2015.

[38] Brian W Kernighan and Shen Lin. An efficient heuristic procedure for partitioning graphs. *Bell system technical journal*, 49(2):291–307, 1970.

[39] Josh Lyskowski Morgan, Daniel Raimund Berger, Arthur Willis Wetzel, and Jeff William Lichtman. The fuzzy logic of network connectivity in mouse visual thalamus. *Cell*, 165(1):192–206, 2016.

[40] Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):2, 2007.

[41] Shin-ya Takemura, C Shan Xu, Zhiyuan Lu, Patricia K Rivlin, Toufiq Parag, Donald J Olbris, Stephen Plaza, Ting Zhao, William T Katz, Lowell Umayam, et al. Synaptic circuits and their variations within different columns in the visual system of drosophila. *Proceedings of the National Academy of Sciences*, 112(44):13711–13716, 2015.

[42] Hao Yin, Austin R. Benson, Jure Leskovec, and David F. Gleich. Local higher-order graph clustering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 555–564, New York, NY, USA, 2017. ACM.

[43] Jason M Klusowski and Yihong Wu. Counting motifs with graph sampling. *arXiv preprint arXiv:1802.07773*, 2018.

[44] Chen Yang, Min Lyu, Yongkun Li, Qianqian Zhao, and Yinlong Xu. Ssrw: A scalable algorithm for estimating graphlet statistics based on random walk. In *International Conference on Database Systems for Advanced Applications*, pages 272–288. Springer, 2018.

[45] Guyue Han and Harish Sethu. Waddling random walk: Fast and accurate mining of motif statistics in large graphs. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 181–190. IEEE, 2016.