Enumerating graph motifs via dual graphs

Ben Klemens

January 31, 2024

Abstract

This document describes a simple means of enumerating subgraphs within a larger network, via simple iterative generation of dual graphs.

Motifs are consistent patterns in a graph, such as triads, chains, or cycles, that appear more frequently than they appear in reference graphs. An inventory of these motifs goes far in understading the interesting or unique properties of the graph.

Let an M-subgraph be a connected set of M nodes in the parent graph, and an M-motif an M-subgraph which appears relatively frequently, as compared to some baseline. A *dual graph* is a graph derived from a parent graph, where each node in the parent maps to edges in the dual, and each edge in the parent maps to a node in the dual.

The method presented in this article generates a complete list of 2-subgraphs, 3-subgraphs, …, M-subgraphs, by generating a sequence of $M-1$ dual graphs.

An option is given to limit counts for certain subgraphs in certain segments of the graph—segments which are least likely to have motifs in typical cases. Adding this limit makes it possible to produce an otherwise comprehensive inventory, which is computationally feasible for typical social science graphs of millions of nodes even without exceptional resources.

This code base is an offshoot of a study of the network of U.S. business entities, revealing certain close-partnership motifs that are infrequent in simple variants of the standard Barabási-Albert network. Please see Klemens [2024] for a full example of the results of motif-finding using this package on a network of several million nodes.

1 Literature

Most of the systems for detecting motifs in a network use random sampling to draw from the network. Giudice and Ursino [2019] gives an overview the many difficulties of random sampling from a network. Kashtan et al. [2004] randomly sample starting nodes, then trace edges from that node. Alon et al. [2008] offer an unbiased method of making random draws via coloring.[1]

[1]To sample motifs with node count M, assign uniformly at random one of M colors to all nodes in the graph. Then traverse the graph to find only those subgraphs where all elements

A motif is a subgraph in the network that appears rarely relative to a baseline network, meaning that in common cases motifs will be rare. In some cases, motifs that appear only a handful of times in a network of millions of nodes are anomalies to be ignored, but in other applications those infrequent motifs are exactly the motifs of interest. A sampling algorithm over all subgraphs will miss a singleton several-node motif in a million-node network with probability approaching one.

After the census of nodes is done, subgraphs will be grouped by their type. As a final step, each subgraph can be reduced to a canonical type via an algorithm such as NAUTY [McKay and Piperno, 2014]. Because this system is designed for circa 10-subgraphs or smaller, and to simplify logistics, the package described here does its own canonicalization step, which is not innovative and will not be described in detail in this article.[2]

2 Method

The procedure presented in this article is a sequence of transformations from a graph to its dual. The initial graph is a set of unnamed edges and labeled nodes. It is called the 1-graph, because the set of node labels is the set of subgraphs of size one.

To generate the 2-graph: For each edge in the 1-graph, add a node to the 2-graph with the label of the nodes on either end of that edge in the 1-graph. The resulting set of two-element labels is the set of two-node subgraphs. If two edges in the 1-graph share a node, then add an edge between their corresponding nodes in the 2-graph.

For example, consider a 1-graph with hub node H in the center with three adjacent nodes, A, B, and C, and let B also be adjacent to C, as in Figure 1. This generates a 2-graph with four nodes representing the original four edges, with nodes labeled (H, A), (H, B), (H, C), and (B, C). The node H connects three edges in the 1-graph, so that node expands to three edges in the 2-graph, (H, A)—(H, B), (H, A)—(H, C) (H, B)—(H, C). Node B in the 1-graph induces an edge (H, B)—(B, C), and node C in the 1-graph induces an edge (H, C)—(B, C).

To generate the M-graph from the $(M-1)$-graph, repeat this procedure of adding an M-graph node for each $(M-1)$-graph edge, and linking the nodes in the M-graph based on the shared nodes in the $(M-1)$-graph. This completes

are different colors (the subgraph is "colorful"). The authors use this for a traversal over rooted trees. For a given set of M nodes, its likelihood of being colorful is not dependent on any characteristics of the network, such as the edges connecting those nodes or the density of the local graph, so a census of sampled nodes will produce an unbiased sample of motifs, without weighting. The paper can be generalized significantly: *any* method of sampling nodes without regard to edges would have the same property of unbiased subgraph counting; coloring is just one means of doing so.

[2]Any subgraph can be represented as a matrix, and that matrix can be read line-by-line as a single binary number. The algorithm swaps rows and columns in the matrix until the single-number representation of the matrix is maximized, and this is taken as the canonical represenation for the purpose of tallying and recording motifs.

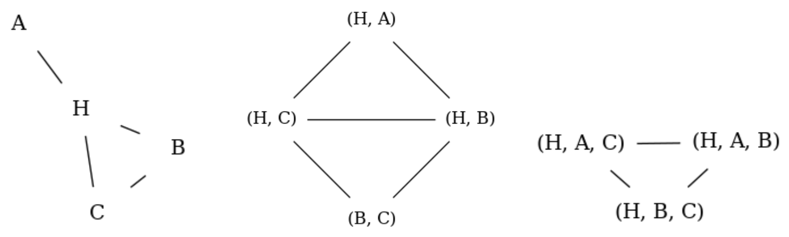

Figure 1: Left: A sample network—a 1-graph. Middle: its dual, a 2-graph. Right: the dual of the dual, a 3-graph.

the description of the core algorithm, save for a detail about deduplication which becomes evident as we continue the example to the 3-graph.

The edge (A, H)—(B, H) generates (A, B, H), and similarly for the others, giving four nodes generated from four edges, some of which are redundant: (H, A, C), (H, A, B), (H, B, C), (H, B, C).[3] The graph needs to be deduplicated, leaving only three nodes, (H, A, C), (H, A, B), and (H, B, C). All edges between nodes are present (a simple triangle).

One could repeat the procedure of dual graph generation and deduplication to transition to the 4-graph, but this is unnecessary, as the only 4-subgraph in this connected graph of four nodes is the full graph.

To summarize the procedure of transitioning from the $(M-1)$-graph to the M-graph:

- For each edge between two nodes in the $(M-1)$-graph with labels L_1 and L_2, add a node to the M-graph with label (L_1, L_2).
- Each node label will have one $(M-1)$-graph label on both sides of the edge. Deduplicate that node label.
- If two edges in the $(M-1)$-graph share a node, add an edge to the M-graph between the nodes corresponding to those two edges.[4]
- Check for any adjacent nodes in the M-graph with identical labels and merge them.
- Record the names of the nodes as the set of M-subgraphs.

[3]This pattern matches the basic combinatorics, that a node H with k neighbors will generate $\binom{k}{M}$ subgraphs of size M, meaning one subgraph of size k.

[4]The algorithm will be run only up to some desired motif size, M_{max}. For the M_{max}-graph, writing down the edges is unnecessary if the list of nodes can be deduplicated after output, such as writing to a a database which stores only unique keys or a text file that can be run through the POSIX-standard `sort` and `uniq` commands.

2.1 Efficiency

Deduplication of a label in the M-graph, such as merging (A, B, C) with (B, C, D) to form (A, B, C, D) is an $\mathcal{O}(M \log M)$ operation. Deduplication of adjacent nodes is an $\mathcal{O}(k)$ operation, where k is the average link count for a node.

Apart from these two deduplication steps, the process of writing down nodes in the M-graph given the $(M-1)$-graph is the process of writing down the M-subgraphs, so no lower-order operation is possible given the task.

Social and a great many other networks have link densities that follow a power law, meaning some hub nodes with a large number of links, which therefore induce a large number of motifs. There are a few ways of ameliorating the problem.

Remove identically-behaving nodes The M-subgraph has especially high concentration around hub nodes with either a large number of parents or a large number of children. This can be ameliorated by combining all nodes adjacent to a hub node that themselves have no other edges of the same type to a single node with high weight. For the business entity network of Klemens [2024], 437,137 nodes are parents whose sole entry in the network is a single child.

Limit counts around hubs-of-hubs Because the method's goal is a complete catalog of subgraphs, if the count of subgraphs is infeasible to count, the method is infeasible. This happens when n nodes with a large number of links are all in the same $(M-n)$-graph. In a typical graph with a link count approximating an exponential distribution, there will be relatively few nodes with large edge counts (by definition), and the subgraph count explodes iff those nodes are linked amongs themselves.

The solution taken by the algorith as written is to check a potential new subgraph for nodes whose edge count will be above a specified limit. If the subgraph meets the criterion, it is thrown out, and no subgraphs of larger size are generated from it. Of course, this means motifs using those most-linked nodes will be missed. Whether this is qualitatively important depends on the application. If they are, a sampling method on the subgraph of those nodes plus those $M-1$ steps away will enumerate subgraphs of size M or smaller.

Probabilistic draws If sampling is desired, it is easily added as an initial step.

If we keep any node in the original graph with only probability p, then an M-subgraph is retained with probability $(1-p)^M$, regardless of the shape of that subgraph or which other subgraphs it is adjacent to or overlaps. That is, removing nodes with probability $(1-p)$ and then finding all subgraphs will draw subgraphs from the full set of subgraphs with uniform probability.

This process can be repeated R times. For a given subgraph that occurs in the full graph η times, we now have R subsets which in expectation have $\eta(1-$

$p)^M$ instances of the subgraph, and the expectation approximates a Binomial distribution with variance $\eta p(1-p)$.[5]

3 Conclusion

The most immediate benefit of the subgraph enumeration algorithm presented in this paper is its simplicity. Calculating a sequence of dual graphs means there is no traversal algorithm needed, meaning no special handling of cycles, no different treatment of rooted trees versus other directed acyclic graphs (DAGs) versus non-DAGs, no probabilistic weighting according to node degrees. Apart from deduplication, the process of writing down the nodes in the $M+1$-graph from the M-graph is almost exactly the process of writing down the M-subgraphs, bringing the computational complexity of the algorithm close to the computational complexity of the theoretical problem itself.

References

N. Alon, P. Dao, I. Hajirasouliha, F. Hormozdiari, and S. C. Sahinalp. Biomolecular network motif counting and discovery by color coding. June 2008. doi: 10.1093/bioinformatics/btn163.

Paolo Lo Giudice and Domenico Ursino. Algorithms for graph and network analysis: Traversing/searching/sampling graphs. In *Encyclopedia of Bioinformatics and Computational Biology*, pages 89–94. 2019. doi: 10.1016/b978-0-12-809633-8.20323-3.

N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon. Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. March 2004. doi: 10.1093/bioinformatics/bth163.

Ben Klemens. Measures of the capital network of the U.S. economy. 2024. doi: 10.48550/ARXIV.2401.12118.

Brendan D. McKay and Adolfo Piperno. Practical graph isomorphism, II. January 2014. doi: 10.1016/j.jsc.2013.09.003.

[5]The coloring method of Alon et al. [2008] is itself a method of randomly drawing from the population of subgraphs, but it adds numerous complications: the draw probability is determined by the number of colors, and it is not possible to reduce it to an arbitrarily small level; many paths must be checked before confirming that a subgraph is not colorful and should be thrown out; coloring for a 4-subgraph search is entirely different from coloring for a 5-subgraph search.