Filtering Methods for Subgraph Matching on Multiplex Networks

Jacob D. Moorman*, Qinyi Chen*, Thomas K. Tu*, Zachary M. Boyd*†, Andrea L. Bertozzi*
*Department of Mathematics, University of California, Los Angeles, Los Angeles, California
jdmoorman@math.ucla.edu, qinyichen@ucla.edu, thomastu@math.ucla.edu, bertozzi@math.ucla.edu
†Present affiliation: Department of Mathematics, University of North Carolina at Chapel Hill, Chapel Hill, North Carolina
zachboyd@email.unc.edu

***Abstract*—We present filtering methods for finding all subgraphs of a large multiplex network that are isomorphic to a smaller template network. These methods are shown to be effective on a set of synthetic transaction networks from the DARPA Modeling Adversarial Activity (MAA) program. In some cases, filtering allows us to identify and enumerate all possible isomorphisms. We observe that in some of the MAA networks, the number of subgraphs isomorphic to the template is orders of magnitude larger than the size of the network.**

I. Introduction

Graph theory and network science abstract complicated structures into a collection of actors (called *nodes*) and the links between them (called *edges*). Such a collection is referred to as a graph or network. In the networks we consider, there may be *parallel edges*: multiple edges with the same source and destination nodes. Some networks, known as *multiplex networks* [1], have two or more types of edges. Each type of edge corresponds to a different *channel*. Multiplex networks can be used to model systems found in a wide variety of disciplines, such as social networks [2], ecological networks [3], and neural networks [4].

We wish to solve the subgraph matching problem on multiplex networks. That is, given two multiplex networks, one typically much larger than the other, we look for all isomorphisms between the smaller – referred to hereafter as the *template* – and any subgraph of the larger, which we call the *background*. We refer to any subgraph of the background isomorphic to the template as a *signal*, and any node which participates in at least one signal as a *signal node*. See Figure 1 for an example.

Note that signals are not necessarily induced subgraphs of the background. Every edge in the template will have a corresponding edge in the signal, but signal nodes may have edges between them that do not correspond to edges in the template. Also, a signal need not be unique: as the background is often orders of magnitude larger than the template, there may be many signals that are all isomorphic to the template.

This problem is of particular interest in the context of the DARPA-led Modeling Adversarial Activity (MAA) program [5] in which the goal is to "develop mathematical and computational techniques for modeling adversarial activity for the purpose of producing high-confidence indications and warnings of efforts to acquire, fabricate, proliferate, and/or deploy weapons of mass terror (WMTs)." Since "MAA assumes that an adversary's WMT activities will result in observable transactions," and "transaction data may very naturally be modeled using graphs," it can be seen that the problem of detecting specific patterns of adversarial activity is related to that of finding subgraph isomorphisms in transaction networks.

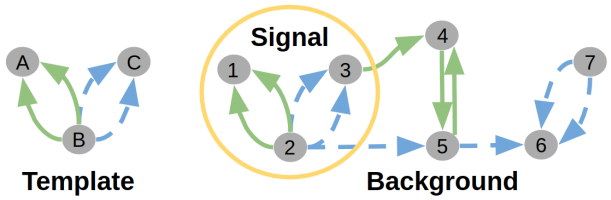

Figure 1: Given the template and background networks above, the only signal is the induced subgraph consisting of nodes **1**, **2**, and **3**.

II. Existing Work

Many subgraph matching algorithms (e.g. [6]–[11]) apply a branch-and-bound approach to search the space of all subgraphs of the background. They grow a partial match between the template and a subgraph of the background, adding one or more nodes at a time to the partial match using various heuristics and backtracking if a complete match becomes impossible. What differentiates subgraph matching algorithms of this type is how they decide what order to add nodes to the partial match, and how they decide which background nodes are considered to be *candidates* for each template node given a partial match [12]. The latter process, that of finding candidate background nodes for each template node, is called *filtering*. It is also sometimes called pruning.

III. Our Contribution

We empirically show that a combination of three efficient filtering methods is highly effective at reducing the number of candidates per template node on several datasets from the MAA program. On some of these datasets, our filters find all

This work was supported by DARPA award FA8750-18-2-0066. ZMB was also supported by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate (NDSEG) Fellowship Program.

978-1-5386-5035-6/18/$31.00 ©2018 IEEE

of the signal nodes, and the signals can be counted directly. The number of signals can be quite large. For example, when there exist template nodes with very few (one or two) edges connecting them to the rest of the template, there tend to be many signals which differ only in the corresponding nodes.

IV. Workflow

For each node $\mathbf{u}$ in the template, we look for the set of nodes in the background that correspond to $\mathbf{u}$ in at least one signal. Initially, we consider every node in the background to be a candidate for each node in the template. We then eliminate candidates for nodes in the template by repeatedly applying filters described in Section V until the sets of candidates converge. Next, we remove nodes from the background which are not candidates for any template node. Finally, we repeat these steps until no further nodes are removed from the background.

There are three possible outcomes:

- There are no candidates left for any node in the template. In this case, we can be sure that no signal exists in the background.
- The candidates converge to the nodes participating in signals, allowing us to enumerate the corresponding isomorphisms using the approach discussed in Section VI. See Section VII-B for an example.
- The candidates do not converge to a small enough set, so we cannot determine how many signals exist, if any. See Section VII-C for an example.

V. Filtering

The goal of filtering is, for each template node $\mathbf{u}$, to eliminate candidates in the background which cannot correspond to $\mathbf{u}$ in any signal.

A. Node-Level Statistics Filter

If a node $\mathbf{v}$ in the background belongs to a signal, it must correspond to some node $\mathbf{u}$ in the template. Then, certain statistical properties of the node $\mathbf{v}$ must be larger than the corresponding properties of $\mathbf{u}$. For example, there must be at least as many edges incident to $\mathbf{v}$ as there are on $\mathbf{u}$. Therefore, if there are fewer edges incident on $\mathbf{v}$ than there are on $\mathbf{u}$, $\mathbf{v}$ can be eliminated as a candidate for $\mathbf{u}$. Statistics which can be used to filter in this way include:

- In/out-degree
- Number of direct successors/predecessors
- Number of reciprocated edges
- Number of self-edges

Each of these statistics can be used separately for each channel in the networks.

As an example, consider applying an in/out-degree filter to the problem in Figure 1. Computing the in/out-degree in each channel, shown in Table 1, we see that nodes $\mathbf{2}$, $\mathbf{3}$, $\mathbf{5}$, $\mathbf{6}$, and $\mathbf{7}$ have smaller in-degree in the solid green channel than node $\mathbf{A}$, and can therefore be ruled out as candidates for $\mathbf{A}$. Similarly, we can eliminate nodes $\mathbf{1}$, $\mathbf{3}$, $\mathbf{4}$, $\mathbf{5}$, $\mathbf{6}$, and $\mathbf{7}$ as candidates for node $\mathbf{B}$, and nodes $\mathbf{1}$, $\mathbf{2}$, $\mathbf{4}$, $\mathbf{5}$, and $\mathbf{7}$ as candidates for node $\mathbf{C}$. The remaining candidates for each template node are summarized in Table 2.

Node	**A**	**B**	**C**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Solid green in-degree	2	0	0	2	0	0	2	1	0	0
Solid green out-degree	0	2	0	0	2	1	1	1	0	0
Dashed blue in-degree	0	0	2	0	0	2	0	1	3	0
Dashed blue out-degree	0	2	0	0	3	0	0	1	0	2

Table 1: In/out-degree per channel for nodes in the template and background shown in Figure 1.

Template node	Candidates
A	**1**, **4**
B	**2**
C	**3**, **6**

Table 2: Candidates per template node for the problem shown in Figure 1 after filtering on in/out-degree in each channel.

B. Topology Filter

If nodes $\mathbf{v_1}$ and $\mathbf{v_2}$ in the background belong to a signal in which they correspond to nodes $\mathbf{u_1}$ and $\mathbf{u_2}$ in the template, there must be at least as many edges in each channel between $\mathbf{v_1}$ and $\mathbf{v_2}$ as there are between $\mathbf{u_1}$ and $\mathbf{u_2}$. Therefore, if there are fewer edges in some channel between $\mathbf{v_1}$ and every candidate for $\mathbf{u_2}$, $\mathbf{v_1}$ can be eliminated as a candidate for $\mathbf{u_1}$. Likewise, if there are fewer edges in some channel between $\mathbf{v_2}$ and every candidate for $\mathbf{u_1}$, $\mathbf{v_2}$ can be eliminated as a candidate for $\mathbf{u_2}$.

Consider applying this filter to the problem in Figure 1, supposing that initially the candidates are as shown in Table 2. Since node $\mathbf{4}$ is a candidate for node $\mathbf{A}$, we expect that it should have at least two solid green edges coming from some candidate for node $\mathbf{B}$, which it does not. Therefore, node $\mathbf{4}$ can be eliminated as a candidate of node $\mathbf{A}$. Similarly, node $\mathbf{6}$ can be eliminated as a candidate of node $\mathbf{C}$, since it does not have at least two dashed blue edges coming from any candidate of node $\mathbf{2}$. Thus, once the topology filter has been applied, the only candidates for nodes $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$ are nodes $\mathbf{1}$, $\mathbf{2}$, and $\mathbf{3}$ respectively.

C. Repeated Set Filter

This filter uses the fact that any isomorphism from the template to a signal must be injective. A consequence of this is that if there is a template node which has only one candidate, that candidate can be eliminated from candidacy for every other template node. More generally, if there is a set of k template nodes, the union of whose candidates is a set also of size k, then those candidates cannot be candidates for any template node outside the set. We relax this latter condition and instead impose that if a set of k template nodes each have

exactly the same k candidates, those candidates are removed from the candidates of all other template nodes.

As an example, consider the template and background shown in Figure 2, with candidates as shown in the first two columns of Table 3. Since two nodes in the template, **B** and **D**, each have the same two candidates in the background, **2** and **4**, those candidates are eliminated from candidacy for each of the other template nodes, **A**, **C**, and **E**.

The repeated set filter is most important for templates which contain structures that remain invariant under some permutations of labels e.g. a template containing several leaf nodes, each of which is structurally interchangeable. We observe this invariance across all instances in the dataset discussed in Section VII-B.

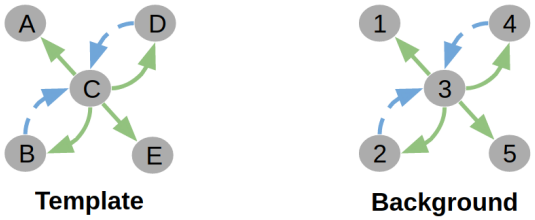

Figure 2: An example of a template that contains nodes which are structurally interchangeable. In particular, there is no way to distinguish between nodes **A** and **E** or nodes **B** and **D**. For illustrative purposes, the background is a duplicate of the template with the labels changed.

Template Node	Candidates without repeated set filter	Candidates with repeated set filter
A	**1**, **2**, **4**, **5**	**1**, **5**
B	**2**, **4**	**2**, **4**
C	**3**	**3**
D	**2**, **4**	**2**, **4**
E	**1**, **2**, **4**, **5**	**1**, **5**

Table 3: Candidates per template node for the problem shown in Figure 2 after filtering on in/out-degree in each channel, followed by filtering on topology. Results are shown with and without additionally applying the repeated set filter.

VI. Isomorphism Counting

After applying the filters described in Section V, we frequently find that most template nodes have exactly one candidate. However, a few template nodes still have multiple candidates; we refer to these as *unspecified* nodes. To count the number of isomorphisms, we must enumerate the valid ways that candidates can be matched to these unspecified nodes, determining a mapping from the template nodes to the nodes of a signal. We call such a mapping a *node-match*.

When an edge exists between two unspecified nodes, we have to enforce that a corresponding edge exists between the two candidates we choose for them. As this makes enumerating node-matches computationally complex, we start by finding a set of unspecified nodes which, if specified, would cause the remaining unspecified nodes to have no edges between them. This set is called a *node cover*, and the smallest such set is called the *minimal node cover*.

For example, in Figure 2, if we suppose all five template nodes have multiple candidates, the minimal node cover would be $\{\mathbf{C}\}$. Since the minimal node cover is expensive to compute in general, we settle for a small node cover [13].

Next, we iterate through all possible choices for candidates of nodes in the node cover. For each choice, we reapply the topology and repeated set filters so we can be sure that any remaining candidates belong to signals. Since the remaining unspecified nodes have no edges between them, it is much simpler to enumerate the ways to choose their candidates. The only constraint is that the same candidate cannot be chosen for more than one node. The problem of choosing candidates in this way is known as the alldifferent constraint satisfaction problem [14].

Because networks may have parallel edges, a node-match may correspond to more than one isomorphism. Each node-match leads to a number of isomorphisms equal to the number of ways edges can be chosen between the signal nodes. Since the distinction between isomorphisms arising from the same node-match is not very interesting, we omit counting them and instead focus on enumerating node-matches.

Similarly, when a template contains structures invariant under some permutations, one set of signal nodes can give rise to multiple node-matches, each corresponding to a permutation. Thus, we also count the number of distinct sets of signal nodes.

VII. Experiments

To test the efficacy of our filtering methods, we apply the workflow described in Section IV to several datasets created by Pacific Northwest National Laboratory (PNNL), the Graphing Observables from Realistic Distributions In Activity Networks (GORDIAN) team, and IvySys Technologies for the MAA program. The PNNL and GORDIAN datasets consist of multiple instances, each of which has its own template and background, while the IvySys dataset only has a single instance. The size of each instance can be found in Table 4. Each instance is known to have one *hidden signal* embedded in the background by its creator; however, there may also be many naturally occurring signals. These natural signals may overlap with the hidden signal or be completely disjoint.

In some cases, filtering solves the whole problem by finding a single candidate for each template node (see Section VII-A). Even in cases where there are multiple candidates, filtering can still sometimes reduce the problem to the point that node-matches can be directly enumerated (see Section VII-B).

However, there are cases where filtering fails to reduce the problem to this point (see Section VII-C). And, when the template is disconnected, we can have varying results for each connected component in the template (see Section VII-D).

A. GORDIAN Version 4 Probabilistic

Many of the template nodes in each instance of this dataset have much higher degrees in at least one channel than the

Dataset	Instance	Template			Background			Channels	Comments
		Nodes	Edges	Distinct edges	Nodes	Edges	Distinct edges		
PNNL Version 4 10k	B0	75	857	780	22,154	480,242	466,667	6	Ignored self-edges
	B4	75	896	818	22,059	488,549	475,220		
	B5	88	1,099	900	22,260	545,773	530,199		
	B8	88	1,135	929	21,974	445,130	433,499		
GORDIAN Version 4 Probabilistic	5k	25	21,261	151	5,003	4,755,911	1,113,816	5	Template only uses 3 of the 5 channels
	10k	39	61,803	453	10,003	25,380,079	6,012,215		
	35k	39	47,644	424	35,003	39,674,679	10,029,209		
GORDIAN Version 4 Agent-Based	5M	43	652,544	38	19,338	11,237,548	39,987	2	Template is disconnected
	10M	48	521,312	41	31,215	20,201,977	72,339		
	35M	43	664,047	34	76,625	75,289,654	230,808		
IvySys Version 4		91	194	127	4875	9,584,969	45,366	3	

Table 4: Overview of the size of each dataset considered. The *edges* columns count each parallel edge separately, whereas the *distinct edges* columns count all edges with the same source and destination as one. For example, the template in Figure 1 has four *edges*, but only two *distinct edges*.

non-signal nodes in the background. In the 35k instance, there are 25 such nodes out of the 39 nodes in the template. After running the node-level statistics filter to convergence, those 25 template nodes have fewer than 20 candidates each. The remaining template nodes have thousands of candidates as seen in Figure 3.

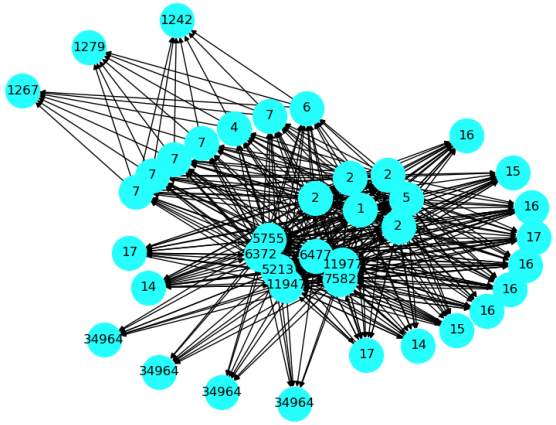

Figure 3: Number of candidates per template node in GORDIAN Version 4 Probabilistic 35k after repeatedly applying the **node-level statistics** filter until convergence.

After applying the topology filter in addition to the node-level statistics filter, one template node has three candidates, while the rest have only a single candidate each as seen in Figure 4. Upon inspection of the three candidates, two are already the sole candidates of other template nodes. After additionally running the repeated set filter, every template node has a single candidate corresponding exactly to the nodes of the hidden signal. The same result holds for the 5k and 10k instances.

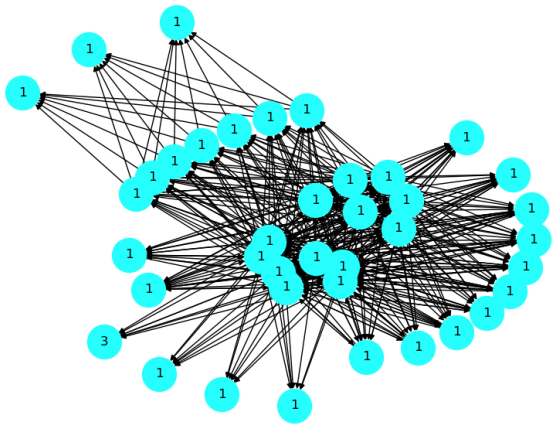

Figure 4: Number of candidates per template node in GORDIAN Version 4 Probabilistic 35k after repeatedly applying the **node-level statistics** and **topology** filters until convergence.

B. PNNL Version 4 10k

For the PNNL Version 4 10k dataset, filtering on node-level statistics and topology proves useful in eliminating background nodes and narrowing down candidates for the template nodes. The resulting candidate counts for the B0 instance are shown in Figure 5. Upon inspection, six of the template nodes share the same six candidates, and several of the other template nodes with six or eight candidates also share these six candidates. By additionally applying the repeated set filter, which specifically targets such situations, we see the improvements shown in Figure 6.

At this point, since there are so few template nodes with more than one candidate, we can identify all possible node-matches. We count 57,139,200 node-matches for the B0 instance using the approach discussed in Section VI. Note that several template nodes are structurally equivalent, which causes the number of node-matches to skyrocket. However,

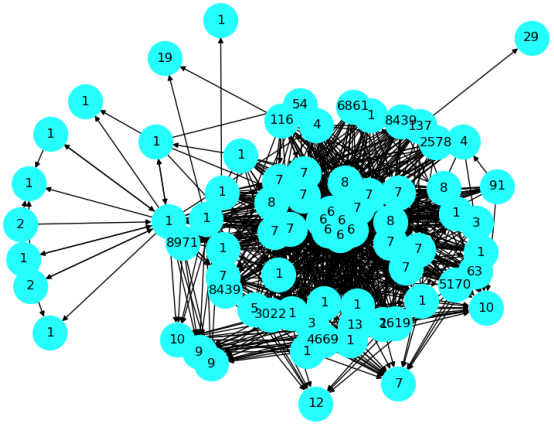

Figure 5: Number of candidates per template node in PNNL Version 4 10k B0 after repeatedly applying the **node-level statistics** and **topology** filters until convergence.

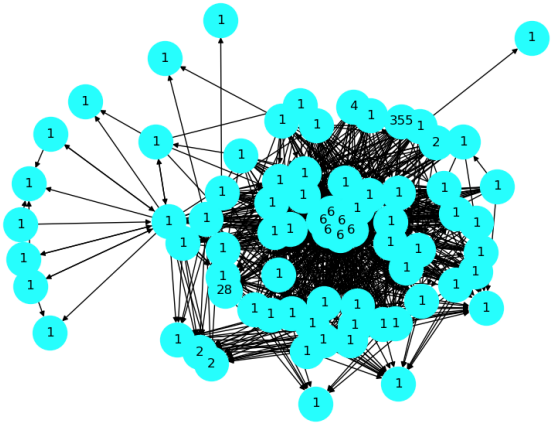

Figure 6: Number of candidates per template node in PNNL Version 4 10k B0 after repeatedly applying the **node-level statistics**, **topology**, and **repeated set** filters until convergence.

these node-matches correspond to only 39,368 distinct sets of signal nodes, which share 71 of their 75 nodes. These results are summarized for B0 and other instances in Table 5.

C. IvySys Version 4

Due to the sparsity of the template in this dataset, and the lack of any easily identifiable nodes, the filters are unable to narrow down the candidates to find the hidden signal. As shown in Figure 7, the nodes with the fewest candidates have 45, 62, and 115 candidates after applying all of the filters. It's possible that there are many naturally occurring signals, which could be causing the observed abundance of candidates for each node.

D. GORDIAN Version 4 Agent-Based

This dataset consists of three instances. Each instance has a template with 10 or 11 connected components, and a different

Instance	Node-matches	Signal node sets	Signal nodes	Essential signal nodes
B0	57,139,200	39,368	446	71 of 75
B4	9,617,097,600	6,579,768	221	67 of 75
B5	19,801,877,760	2,289,808	661	79 of 88
B8	22,029,096,960	4,751,162	639	85 of 88

Table 5: Summary of results for the B0, B4, B5, and B8 instances from PNNL Version 4 10k. The discrepancy between the number of *Node-matches* and *Signal node sets* comes from signal node sets that can be matched to the template in multiple ways. *Essential signal nodes* are nodes that participate in every signal.

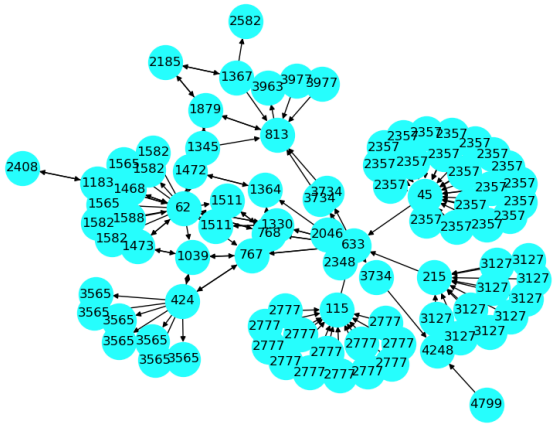

Figure 7: Number of candidates per template node in IvySys Version 4 after repeatedly applying the **node-level statistics**, **topology**, and **repeated set** filters until convergence.

size background network. After running all of the filters until convergence on the 35M instance, all but two of the connected components consist of nodes which have a single candidate each. In other words, their corresponding hidden signal nodes have been found exactly. The remaining two connected components have many more candidates, as seen in Figure 8. The smaller connected component's node-matches can easily be enumerated, since there are only two template nodes participating in that component. However, we were unable to enumerate the node-matches of the largest connected component, as each of its template nodes has over 2,000 candidates. Similar results were found for the 10M and 5M instances.

Counterintuitively, it appears at first from Figure 8 that a structure as simple as a pair of nodes has fewer node-matches than the largest, most complicated structure. However, closer inspection of the template reveals that each of these simple pairs in the template is actually connected by many edges, not just one. This illustrates the importance of parallel edges; without them, these structures would be undetectable.

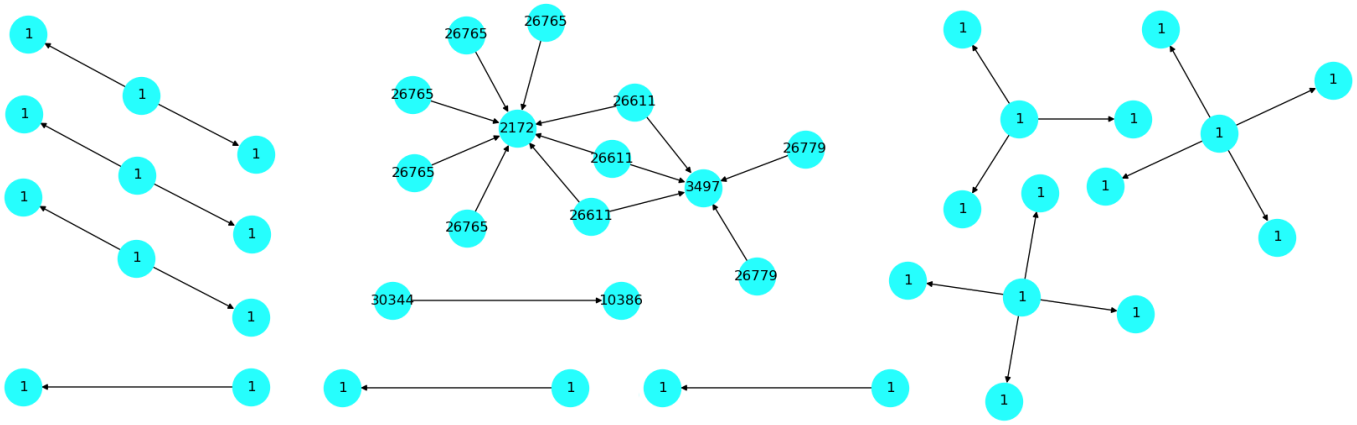

Figure 8: Number of candidates per template node in GORDIAN Version 4 agent-based 35M after repeatedly applying the **node-level statistics**, **topology**, and **repeated set** filters until convergence.

VIII. Conclusion

We propose effective filtering methods for finding signals isomorphic to a template inside of a large multiplex network by reducing the search space based on local statistics and topology to the point where less scalable counting methods can be applied. We test our methods on datasets created by PNNL, GORDIAN, and IvySys for the MAA program. Our methods find a unique signal in each of the GORDIAN Version 4 Probabilistic instances, and find many signals in each of the PNNL Version 4 10k instances. They do not find the signal(s) in the IvySys Version 4 dataset, indicating the need for a more restrictive approach.

In the future, we plan to extend the filter to the noisy case, where the background is not fully observed. In this case, the observed background may not contain an exact match of the template, since some edges of a hidden signal may not be observed. This will require us to relax the requirements of our filter, and search for subgraphs of the background *most likely* to be a signal.

Acknowledgments

This material is based on research sponsored by the Air Force Research Laboratory and DARPA under agreement number FA8750-18-2-0066. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory and DARPA or the U.S. Government.

We thank Jamie Atlas, Omri Azencot, Jeremy Budd, Yoni Dukler, Xie He, Matthew Jacobs, Blaine Keetch, Hao Li, Kevin Miller, Mason A. Porter, Bao Wang, Yotam Yaniv, Baichuan Yuan, and others for helpful discussions.

References

[1] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter, "Multilayer networks," *J. of Complex Networks*, vol. 2, no. 3, pp. 203–271, 2014.

[2] L. M. Verbrugge, "Multiplexity in adult friendships," *Social Forces*, vol. 57, pp. 1286–1309, 1979.

[3] S. Pilosof, M. A. Porter, M. Pascual, and S. Kéfi, "The multilayer nature of ecological networks," *Nature Ecology & Evolution*, vol. 1, no. 4, p. 0101, 2017.

[4] B. Bentley, R. Branicky, C. L. Barnes, Y. L. Chew, E. Yemini, E. T. Bullmore, P. E. Vértes, and W. R. Schafer, "The multilayer connectome of caenorhabditis elegans," *PLoS Computational Biology*, vol. 12, no. 12, p. e1005283, 2016.

[5] C. Schwartz. Modeling Adversarial Activity (MAA). Defense Advanced Res. Projects Agency. "(2018, Oct 2)". [Online]. Available: https://www.darpa.mil/program/modeling-adversarial-activity

[6] J. R. Ullmann, "An algorithm for subgraph isomorphism," *J. ACM*, vol. 23, no. 1, pp. 31–42, Jan. 1976.

[7] L. P. Cordella, P. Foggia, C. Sansone, and M. Vento, "A (sub)graph isomorphism algorithm for matching large graphs," *IEEE Trans. on Pattern Analysis and Machine Intell.*, vol. 26, no. 10, pp. 1367–1372, Oct 2004.

[8] C. Solnon, "AllDifferent-based Filtering for Subgraph Isomorphism," *Artificial Intell.*, vol. 174, pp. 850–864, Aug. 2010.

[9] G. Audemard, C. Lecoutre, M. Samy-Modeliar, G. Goncalves, and D. Porumbel, "Scoring-based neighborhood dominance for the subgraph isomorphism problem," in *Int. Conf. on Principles and Practice of Constraint Programming*, B. O'Sullivan, Ed. Springer Int. Publishing, 2014, pp. 125–141.

[10] C. McCreesh and P. Prosser, "A parallel, backjumping subgraph isomorphism algorithm using supplemental graphs," in *Int. Conf. on Principles and Practice of Constraint Programming*, G. Pesant, Ed. Springer Int. Publishing, 2015, pp. 295–312.

[11] V. Carletti, P. Foggia, A. Saggese, and M. Vento, "Introducing VF3: A new algorithm for subgraph isomorphism," in *Graph-Based Representations in Pattern Recognition*, P. Foggia, C.-L. Liu, and M. Vento, Eds. Cham: Springer Int. Publishing, 2017, pp. 128–139.

[12] J. Lee, W.-S. Han, R. Kasperovics, and J.-H. Lee, "An in-depth comparison of subgraph isomorphism algorithms in graph databases," *Proc. VLDB Endow.*, vol. 6, no. 2, pp. 133–144, Dec. 2012.

[13] R. Bar-Yehuda and S. Even, "A local-ratio theorem for approximating the weighted vertex cover problem," in *Analysis and Des. of Algorithms for Combinatorial Problems*. Elsevier, 1985, pp. 27–45.

[14] J. Régin, "A filtering algorithm for constraints of difference in CSPs," in *Proc. of the 12th Nat. Conf. on Artificial Intell, Seattle, WA, USA, July 31 - August 4, 1994, Volume 1.*, 1994, pp. 362–367.