

2013


A structure-preserving hybrid-chordal filter for sampling in correlation networks

Kathryn Dempsey Cooper
University of Nebraska at Omaha, kdempsey@unomaha.edu

Tzu-Yi Chen
University of Nebraska at Omaha

Sriram Srinivasan
University of Nebraska at Omaha, sriram882004@gmail.com

Sanjukta Bhowmick
Follow this and additional works at: <https://digitalcommons.unomaha.edu/interdiscipinformaticsfacproc>
University of Nebraska at Omaha, sbhowmick@unomaha.edu

 Part of the [Bioinformatics Commons](#), and the [Health Information Technology Commons](#)

Hesham Ali
Please take our feedback survey at: https://unomaha.az1.qualtrics.com/jfe/form/SV_8cchtFmpDyGfBLE
University of Nebraska at Omaha, hall@unomaha.edu

Recommended Citation

Cooper, Kathryn Dempsey; Chen, Tzu-Yi; Srinivasan, Sriram; Bhowmick, Sanjukta; and Ali, Hesham, "A structure-preserving hybrid-chordal filter for sampling in correlation networks" (2013). *Interdisciplinary Informatics Faculty Proceedings & Presentations*. 12.

<https://digitalcommons.unomaha.edu/interdiscipinformaticsfacproc/12>

This Conference Proceeding is brought to you for free and open access by the School of Interdisciplinary Informatics at DigitalCommons@UNO. It has been accepted for inclusion in Interdisciplinary Informatics Faculty Proceedings & Presentations by an authorized administrator of DigitalCommons@UNO. For more information, please contact unodigitalcommons@unomaha.edu.

A structure-preserving hybrid-chordal filter for sampling in correlation networks

Kathryn Dempsey*^a, Tzu-Yi Chen⁰, Sriram Srinivasan^a, Sanjukta Bhowmick^a, Hesham Ali*^a

^aCollege of Information Science and Technology, University of Nebraska at Omaha

*Department of Pathology & Microbiology, University of Nebraska Medical Center

⁰Department of Computer Science, Pomona College

Contact Email: hali@mail.unomaha.edu

Abstract— Biological networks are fast becoming a popular tool for modeling high-throughput data, especially due to the ability of the network model to readily identify structures with biological function. However, many networks are fraught with noise or coincidental edges, resulting in signal corruption. Previous work has found that the implementation of network filters can reduce network noise and size while revealing significant network structures, even enhancing the ability to identify these structures by exaggerating their inherent qualities. In this study, we implement a hybrid network filter that combines features from a spanning tree and near-chordal subgraph identification to show how a filter that incorporates multiple graph theoretic concepts can improve upon network filtering. We use three different clustering methods to highlight the ability of the filter to maintain network clusters, and find evidence that suggests the clusters maintained are of high importance in the original unfiltered network due to high-degree and biological relevance (essentiality). Our filter highlights the advantages of integration of graph theoretic concepts into biological network analysis.

Keywords—*bioinformatics; clusters; network filters; correlation networks; hub nodes; spanning trees*

I. INTRODUCTION

High-throughput assays that survey the activities of a cell at once are becoming more popular; indeed the growing technological capacity for examining biological processes reflects the current focus on data generation in biomedical research. With this increase in technological capacity comes an exponential increase in heterogeneous data and a massive need for methods to analyze it. Correlation networks are one type of data model employed by bioinformaticians to visualize, analyze, and manipulate these types of datum. Representing genes as nodes and edges as tightly correlated patterns of expression, correlation networks have been found to reflect biological network theory in that structures within these networks (hubs, clusters, etc) [1,10] can point to biological functions, and how genes in those functions are related. While these networks are increasing in popularity, the issue remains that networks are typically large and filled with noise [19], corrupting the biological signal behind observed phenotypes. As such, multiple methods for sorting signal from noise have been proposed. One such general method, network filtering, has found measurable success in reducing network size and noise while enhancing ability to identify relevant biological functions.

Previous work [5, 6, 7, and 9] reveals that filters imposed on networks generated by correlation of gene expression are an effective means for removing coincidental edges while enhancing biological signal. Duraisamy *et al.* [9] and Dempsey *et al.* [5,6] revealed that a filter that removes edges that create large cycles in biological networks (i.e. identifying a chordal subgraph from original graph G) removes about 25% of original edges, maintains clusters that exist in the original network, and also reveals clusters that were previously hidden. Dempsey *et al.* [7] explored the how a spanning tree filter affects biological relevance of high degree or hub nodes in the correlation network. (Biologically relevant nodes in a correlation network can typically be expected to represent lethal nodes [8, 15], or nodes that represent genes that when knocked out *in vivo* results in expiration of the organism at some early stage in development [3].) This study found that using a spanning tree filter, it is possible to more accurately identify biologically relevant hub nodes in the correlation network due to the removal of coincidental edges. Further, this enhanced this type of spanning tree filter using a “hybrid” filter that incorporated a spanning tree and a chordal filter by adding edges back into the network. The focus of the study then became the examination of how the biological relevance of hub nodes is further enhanced (i.e., hub nodes from the original network gain more edges back, making them easier to identify as hub nodes). This filter incorporated edge re-addition in two steps, one where edges were added such that chordality is maintained, and a second where edges were added with a less strict condition--- chordality is preferred, but not some larger cycles are allowed, if they are part of clusters. The best parameters from this study revealed that adding in edges that did not necessarily maintain chordality (but not adding in all edges) was best able to identify biologically relevant hub nodes. In short, we have four major versions of the network that we are able to test for biological relevance; these variations are shown in Figure 1.

“Hub” nodes in correlation networks can be disassortative or assortative [14, 18] (), the former indicating that its neighbors are poorly connected and the latter indicating that the hub is very well connected; in such cases the assortative hub can be found to exist within clusters as a member of a dense community. Results from Dempsey *et al.* [7] show that while the aforementioned spanning tree (ST) only filter is able

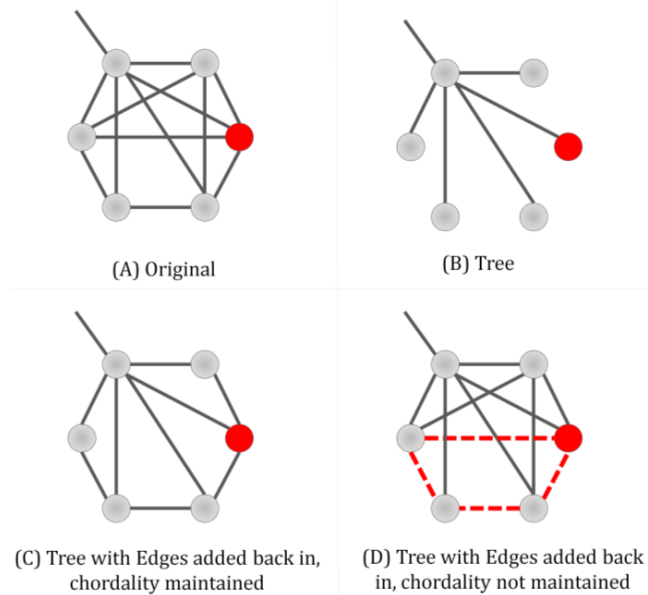


Figure 1. (A) The original network with lethal hub nodes identified in red. (B) The network filtered to a spanning tree. (C) A version of the “chordal” implementation of the hybrid filter, where edges are added back and chordality is maintained. (D) A version of the “all” implementation where edges are added back but chordality is not maintained. A 4-cycle is highlighted in red in Figure 1D. Note that while the lethal hub that is not contained in the cluster is maintained as a hub throughout each version, the lethal hub in the original network cluster only becomes a hub again after edges are added back in at stage (C), and becomes even “hubbi-er” as edges are added back in (D). The cluster density change from (C) to (D) for the 6 nodes involved goes from 46.7% (7 edges) to 73.3% (11 edges) and the lethal hub node goes from degree = 2 to degree = 4.

to identify lethal hub nodes better than the original network (according to degree), the edge-addition methods are both better than the spanning tree only approach. We speculate that this is because the ST only approach only identifies disassortative nodes within the network; adding edges back in allows for the assortative hubs, which by definition require more edges between neighbors, which by definition makes more hubs possible. Theoretically speaking, a biological network is self-organizing and as contains multiple built-in redundancies to ensure survival in structural breakdown; this characteristic of self-organizing systems [17] is consistent with the need for clusters in a correlation network—it reflects the inherent need for a set of genes to be co-expressed and working in concert toward some discrete function.

In this study, we further examine the applicability of this hybrid filter by examining its effectiveness in enhancing clusters in correlation networks. The study on chordal filters by [5], [6] and [9] revealed that a chordal filter is able to maintain current clusters from the original network and identify new clusters that were previously hidden. Previous studies on the hybrid chordal filter have only examined its effectiveness in identifying biologically relevant hub nodes, not clusters. Therefore, in this study we implement and apply a hybrid chordal filter to networks generated from an aging mouse gene expression study to show its effectiveness in identifying clusters. We use three different methods

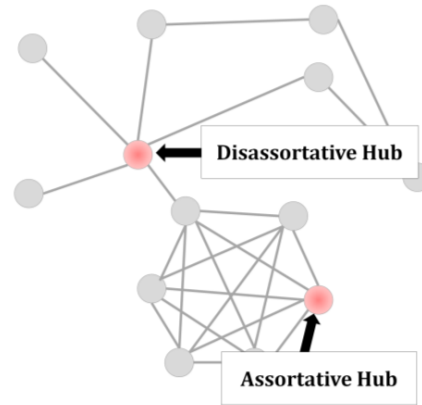


Figure 2. The assortativity of hub nodes. The disassortative hub in this definition has a low clustering coefficient, or its neighbors are not well connected. The assortative hub is very well connected to its neighbors. Both types of hubs have been found to be relevant in various kinds of biological networks.

(AllegroMCODE, MCL, and our own CliqueCode) to indicate how well the filter is able to identify clusters in the network, and for each clustering type we compare clusters from the original network to clusters from the filtered network. This comparison reveals that the hybrid filter is able to identify biologically relevant clusters stemming from cores in the original network and remove coincidental edges. The networks contained here are relatively small for gene expression correlation networks, so it is important to be able to parallelize the clustering method (typically the longest step in the analytic pipeline) and still be able to identify relevant biological clusters. We show in our results that the parallel implementation of CliqueCode approach is very scalable and yields same results as the sequential version.

A. Hypothesis

Our approach uses an original network G and applies our hybrid chordal filter to that network. Our filter creates an augmented spanning tree by first computing a spanning tree, and then adding back selected non-tree edges that create cycles of length three in the filter. This augmentation can be performed over several iterations—at each iterations $T+1$, the distance-2 nodes of the graph created at iteration T are considered and new triangles are added. As the number of iterations increase, we will finally recreate most of the original network. Therefore an important parameter for an effective filter is to judiciously select the number of levels of iterations. The different parameters that affect the performance of the hybrid chordal filter include:

1. *Tree selection*: The node selection process for the initial tree can use a breadth-first-search (BFS) or maximum weighted spanning tree (MST).
2. *Augmentation*: This determines how edges are added back to the tree. The tree itself is named as the 'None' filter. We add back a subset of the edges from the original networks between nodes at distance-2 in the tree." The subset can be chosen to ensure chordality, or made looser

to allow for some larger cycles. In this paper we consider the second case and add back all distance-2 edges. A final option is to add even more edges back to the network via iterations (described below). This filter was called the 'All' filter.

3. *Iterations*: This parameter determines how many times the augmentation should be performed, and applies only to the “all” augmentations.

From our previous results using network filters, we have observed the following phenomena:

- Chordal filters maintain network clusters [4,8]
- Spanning tree filters maintain lethal hub nodes [5,6]
- The hybrid filter maintains lethal hub nodes *best* when edges are added back into the network without necessarily maintaining chordality [6]

We are able to define biological relevance as a node that is essential or lethal, meaning the removal of that particular gene results in a lethal organism phenotype. Based on these observations, we propose our hypothesis for how well the hybrid filter is able to identify clusters:

H₀: A hybrid filter based on tree and chordal structure will identify clusters from the original network that contain high degree nodes of biological relevance.

II. METHODS

For the majority of our studies, networks from aging mouse studies were used. Young (Yng) and Middle-Aged (MID) mouse aging networks were created using gene expression data from GEO Series [12] GSE5078 [20] using Pearson Correlation Coefficient [as described in 11] ($p\text{-val} < 0.005$) as described in Dempsey [6]. Briefly, the pairwise correlation coefficient was calculated for each gene pair. Genes or gene products are represented in the network as nodes. If the correlation was within the threshold range (0.70-1.00), an edge with the weight of the correlation was drawn between the two nodes for those genes/gene products. For parallel studies, we used larger networks GSE5140 [2] which study the effect of creatine supplementation on older mice (treated vs. untreated) and GSE17072 [16] which compares breast cancer in humans in normal tissue, familial breast cancer, and non-familial breast cancer (Control vs. Familial vs. Non). Duplicate edges and self loops were removed from networks before filtering and clustering. Our hybrid chordal filter (coded in MATLAB) was applied to each network under a variety of conditions; for each network there were a total of 9 implementations including the original (Table 1):

TABLE I. DESCRIPTION OF FILTERS

Network	Name	Node Selection	Iterations
Original	Orig	-	-
Spanning Tree Only	NONE-BFS	Breadth First Search	-
Spanning Tree Only	NONE-MST	Maximum Spanning Tree	-
Chordal	Chordal-BFS-1	Breadth First Search	1
Chordal	Chordal-MST-1	Maximum Spanning Tree	1
Non-chordal	All-BFS-1	Breadth First Search	1
Non-chordal	All-MST-1	Maximum Spanning Tree	1
Non-chordal	All-BFS-2	Breadth First Search	2
Non-chordal	All-MST-2	Maximum Spanning Tree	2
Non-chordal	All-BFS-3	Breadth First Search	3
Non-chordal	All-MST-3	Maximum Spanning Tree	3

The number of edges removed for each network and the resulting edge density is contained in Table 2. We used three clustering methods for our study, including AllegroMCODE v1.0 (implemented in Cytoscape v2.8.3) [23], MCL 09-308, v1.088 [13], and CliqueCode v1.0 in sequential and parallel versions. A description of the CliqueCode implementation is contained under Methods – CliqueCode. AllegroMCODE was run using degree cutoff=4, node score cutoff = 0.2, K-core=4, and max depth = 100. These parameters were chosen to find all small, dense clusters of minimum size 4 with a minimum core density of a K₄. MCL was run under default parameters. The fill-in parameters of the CliqueCode were selected to find very dense subgraphs. When fill-in is set to zero, the set of vertices form a complete clique. We also relax the fill-in to 1, which indicates that the subgraph is a complete clique minus one edge. Larger values of fill-in lead to less dense cliques. For the networks considered in this paper, we found fill-in of 0 and 1 to give the best value. The size of the clique also plays an important role in determining the significance. In this paper we considered all cliques of size 4 or larger. Clusters from each method were then compared in from the original networks to the filtered networks. For example, AllegroMCODE original clusters were only compared to AllegroMCODE filtered clusters; clusters from different methods were not compared for accuracy. Filtered clusters were measured against original clusters using sensitivity measures for both nodes and edges. To measure this, we used the following where x = node or edge:

- xTP = an element in the original cluster set was also found in the filtered set
- xFP = an element in the filtered set was not found in the original set
- xFN = an element in the original set was not found in the filtered set

Using these metrics, we are able to identify sensitivity for each filter where $x\text{Sensitivity} (xS_n) = xTP / (xTP + xFN)$. In addition to sensitivity, we also measure cluster size, count, and filter speed. Based on our hypothesis, the ideal result for our

networks is to identify small, dense clusters with biologically relevant nodes [10].

TABLE II. THE SIZES AND EDGE DENSITIES OF THE ORIGINAL AND FILTERED NETWORKS

Age	Node Selection	Filter	Iterations	Nodes	Edges	Edge Density	
YNG	Original			5,348	7,274	0.0509%	
	BFS	None	-	5,348	3,885	0.0272%	
		Chordal	-	5,348	4,206	0.0294%	
		All	1	5,348	5,379	0.0376%	
		All	2	5,348	6,153	0.0430%	
	MST	All	3	5,348	6,596	0.0461%	
		None	-	5,348	3,885	0.0272%	
		Chordal	-	5,348	4,280	0.0299%	
		All	1	5,348	4,449	0.0311%	
	MID	Original			5,549	7,178	0.0466%
		BFS	None	-	5,549	4,154	0.0270%
			Chordal	-	5,549	4,542	0.0295%
All			1	5,549	5,267	0.0342%	
All			2	5,549	5,726	0.0372%	
MST		All	3	5,549	6,005	0.0390%	
		None	-	5,549	4,154	0.0270%	
		Chordal	-	5,549	4,808	0.0312%	
		All	1	5,549	5,117	0.0332%	
		All	2	5,549	5,924	0.0385%	
		All	3	5,549	6,490	0.0422%	

A. Description of Clustering Methods

AllegroMCODE is an implementation of MCODE as a Cytoscape Plug-in that weights nodes according to high k-core values. The more dense the local community around a node, with high core value, the heavier the weight. The code was originally designed to find clusters in protein-protein interaction networks [2], which tend to be small, dense clusters representative of protein complexes. MCL was originally designed for clustering in protein-protein interaction networks as well; it uses Markov clustering and network topology to rapidly identify groups of nodes in weighted or unweighted networks [13].

B. CliqueCode

We developed CliqueCode as a more efficient and scalable alternative to AllegroMCODE. As the name suggests, CliqueCode focuses on finding near-cliques in the network. For each vertex, we check the connections between its neighbors, and compute the *fill-in*, i.e. the number of edges required to create a complete clique comprising of the vertex and its neighbors. The value of fill-in can be adjusted according to the tightness of cliques required. In these experiments, fill-in was set to zero (perfect clique) and one (clique with one edge missing).

In contrast to the other clustering methods, finding cliques provides a very simple yet effective method for finding clusters of biological importance. In our algorithm a clique is detected by identifying a seed vertex with low fill-in(0 or 1) and adding its neighbors to form the clique. In general, if the seed has low fill-in, the neighbors do not, since they can be connected to vertices outside of the clique as well. However, if

a vertex is part of two cliques we use a tie breaking scheme to assign one vertex to one clique only. Thus we will never include the same clique more than once in our analysis. The results, in most cases are comparable to those obtained by AllegroMCODE (see section on Experiments). However because the algorithm only considers distance-2 neighbors of each vertex, it is faster and more amenable to parallelization as compared to the k-core discovery method of AllegroMCODE.

The parallelization of CliqueCode is very simple. Ideally, each vertex in parallel can determine whether it is part of a clique. In practice the number of available processing units determines the degree of parallelism. The parallelization of CliqueCode is implemented as follows: the fill-in for each vertex can be computed in parallel, and based on the threshold of fill-in, the cliques are formed, also in parallel, by including the neighbors of each vertex. Each vertex is associated with the id of the seed vertex of the clique. If the vertex is found to belong to two cliques, the clique with the smaller seed id is selected. The results of this implementation are also stable under parallelization—that is they are not affected by the number of processing units used, or the ordering of the network. However, this simple code also produces redundant results. For example, the same clique is returned for each of its constituent vertices. After the initial phase, we cull out the duplicate cliques. Another issue is when a group of vertices belong to multiple cliques. In this case we merge the cliques containing the common vertices.

III. EXPERIMENTAL RESULTS

For each experimental run, we have recorded clustering accuracy in identifying original clusters, network size, cluster counts, and a number of other variables that indicate the hybrid chordal filter indeed identifies clusters with biologically relevant high degree nodes.

A. Filtered Network Size

Table 2 contains network node and edge counts as well as edge density (where $\text{edge Density} = \frac{\text{edge count}}{[(\text{node count}^2 - \text{node count})/2]}$ as a percentage). If the filter performs correctly, we expect that the edge count and edge density should increase in the following order: None < Chordal < All $i=1$ < All $i=2$, All $i=3$ < Original. We find that this is indeed the case, with the None augmentation containing around ~50% of the original edges and the All $i=3$ augmentation containing around 75% of original edges.

B. Sensitivity and Cluster Count

In comparing original network clusters to filtered network clusters, we again note that the ideal cluster for this type of network is a small number of dense clusters with relatively few nodes. This guarantees that the search space for further biological testing is narrowed (small number of clusters) and that the clusters are tightly correlated and thus more likely to retain biological function. Further, we want to ensure that if clusters are found, they are also found in the original network. To measure performance of the filter in finding original clusters, we use the aforementioned definitions. The node and

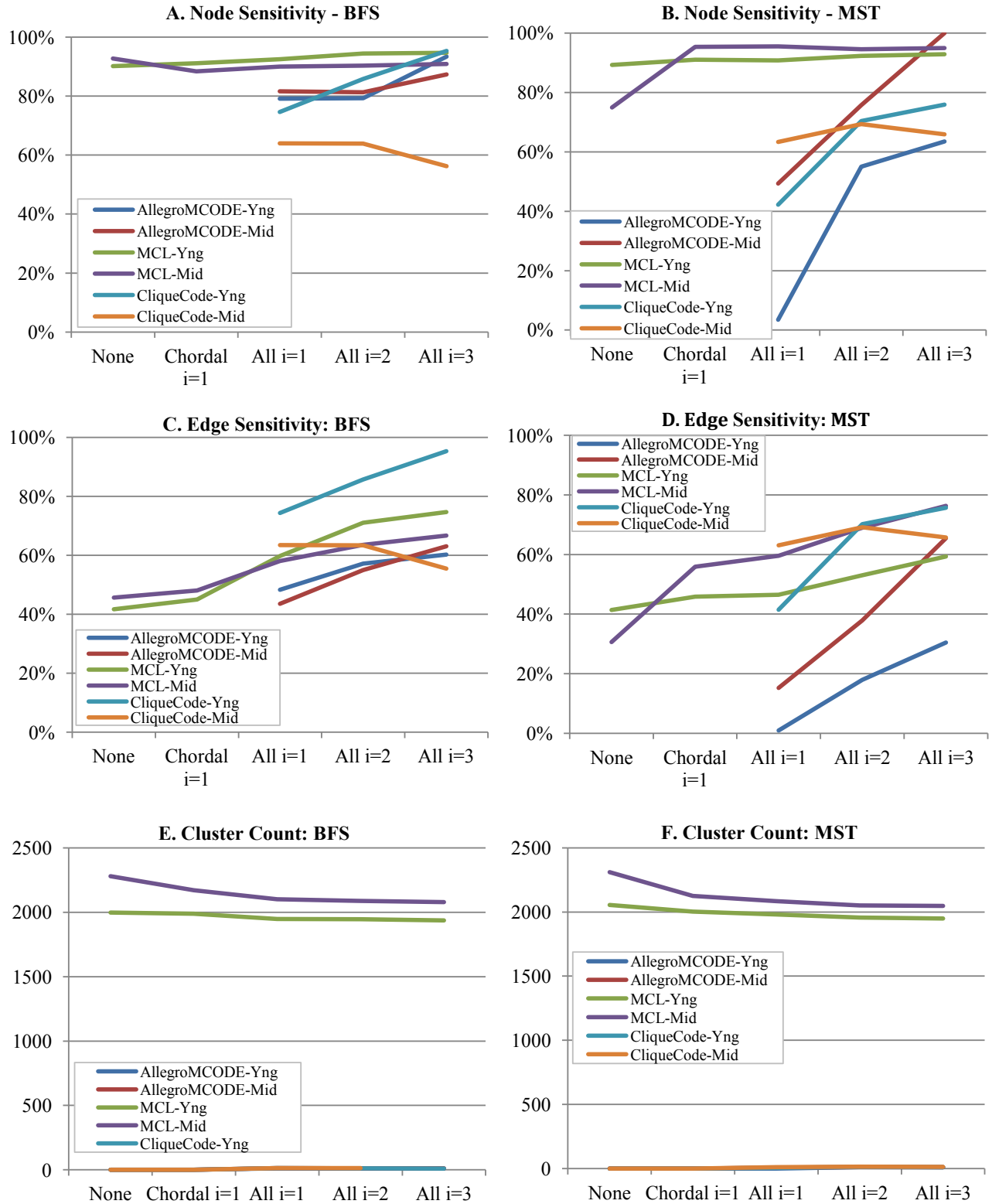


Figure 3. (A) Node Sensitivity for BFS runs. (B) Node Sensitivity for MST runs. (C) Edge Sensitivity for BFS runs. (D) Edge Sensitivity for MST runs. (E) Cluster count for BFS. (F) Cluster count for MST. X-axis: Filter type. Y-axis for A-D: Sensitivity, Y-axis for E-F: Cluster count. If lines are not shown for a particular run, that is because no clusters were found for that particular version of the filter.

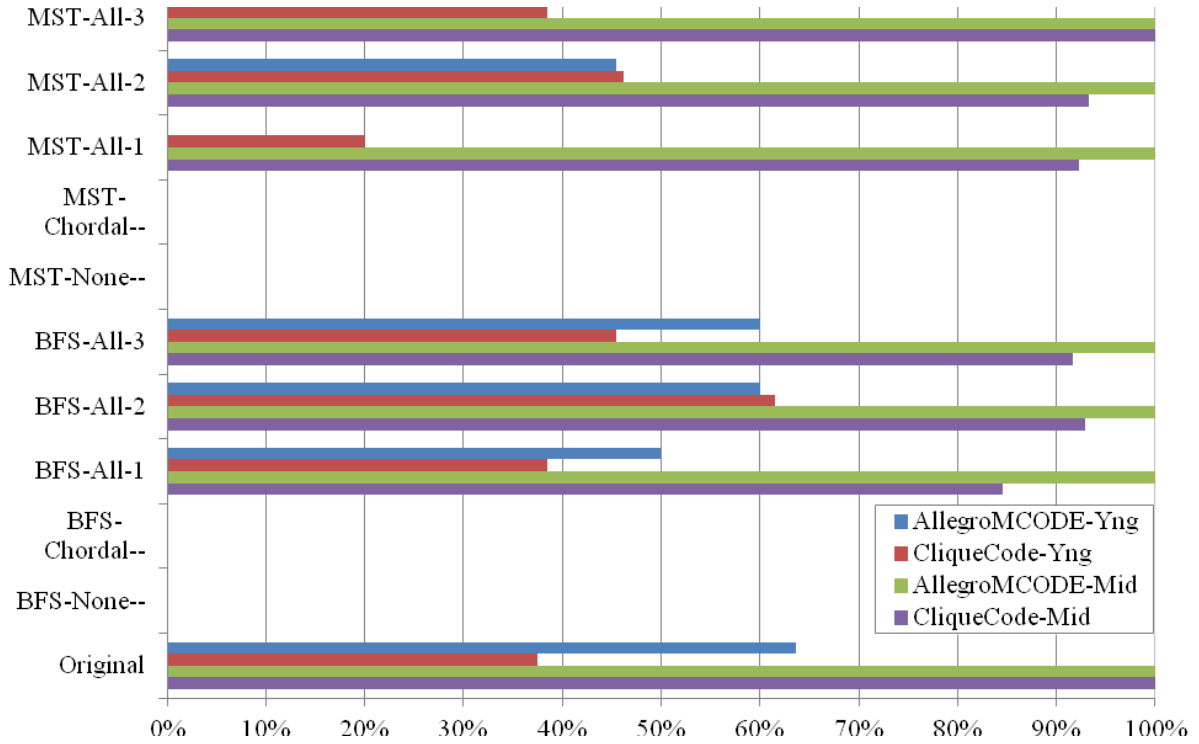


Figure 4. Percent of clusters from results containing lethal nodes for Original, MST All and BFS All networks. (None and chordal networks did not find clusters using our methods).

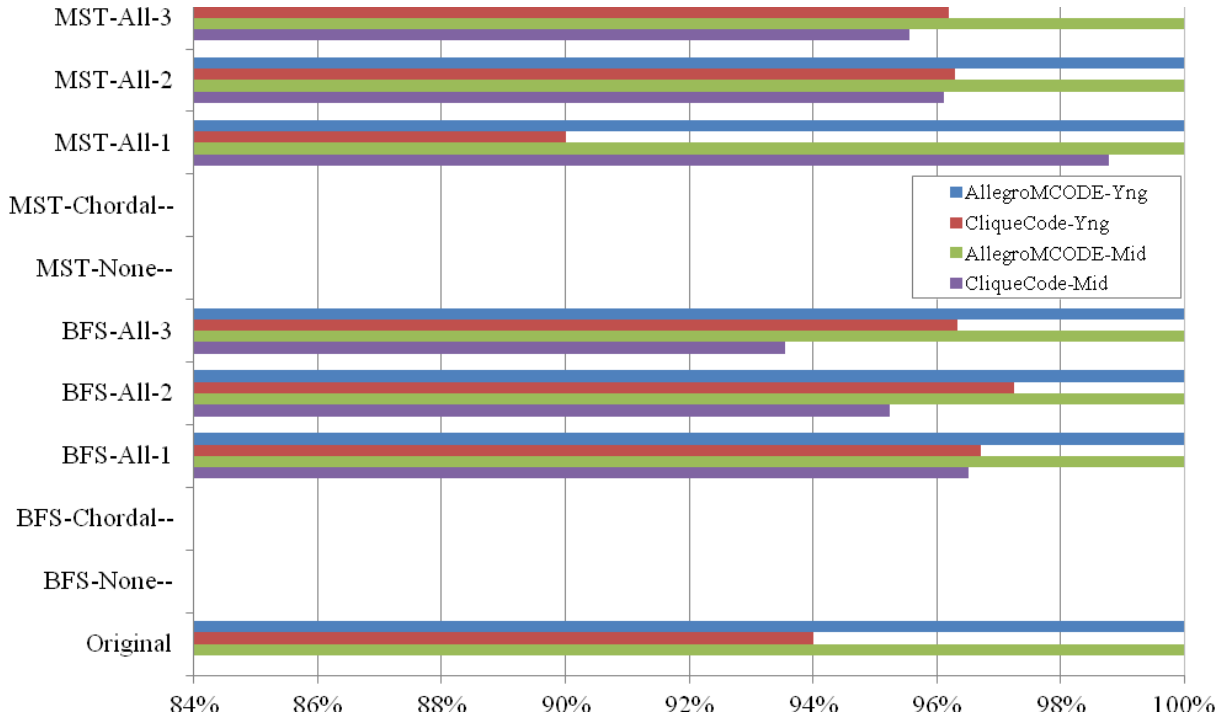


Figure 5. Percent of clusters from results containing hub nodes from the original network for Original, MST All and BFS All networks. (None and chordal networks did not find clusters using our methods).

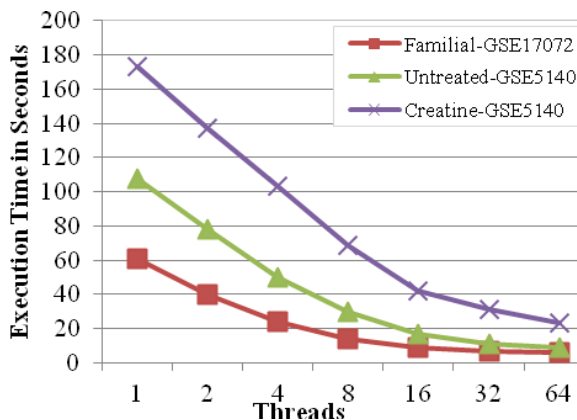
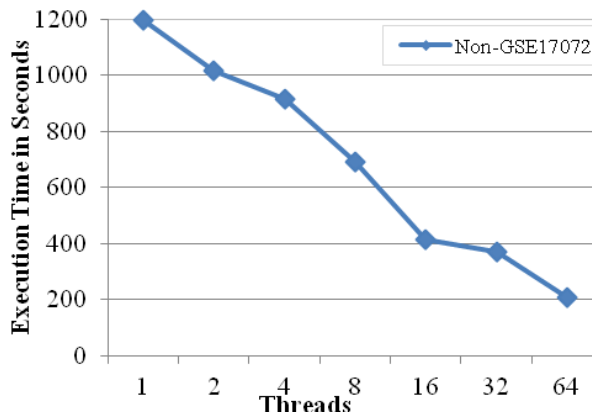


Figure 6. Strong scalability for the parallel implementation of CliqueCode.

edge sensitivity results are contained in Figure 3a-d. We find that in terms of node sensitivity, MCL is the best performer. However, this result is misleading as MCL clusters the entire network together; the majority of MCL clustering results contain around 2,000 clusters on average and finds every node in the network. Inherently, even though sensitivity is high, guessing everything does not yield the type of results we desire (few small dense clusters). AllegroMCODE and CliqueCode both have moderately sensitive results; AllegroMCODE finds more clusters and thus has more sensitivity whereas CliqueCode is more conservative and finds fewer smaller clusters. As was the case in Dempsey *et al.* [7], the BFS is a better performer overall than MST. Comparing cluster counts (Figure 3e,3f) we see that AllegroMCODE and CliqueCode are better at identifying fewer clusters; examining the individual density of those clusters we find that they are indeed small and dense, with CliqueCode clusters being smaller (results not shown).

C. Lethal and Hub Node Identification

To further probe our hypothesis that clusters that contain high-degree nodes tend to be biologically relevant, we look at the original and filtered network clusters and examine how many of them contain lethal nodes (Figure 5) and how many of them contain hub nodes as shown in Figure 6 (hub nodes as defined in the original network). This analysis was performed only for AllegroMCODE and sequential CliqueCode. We find that overall, the Yng original networks have 40-60% of clusters that contain lethal nodes; in the Mid network 100% of clusters found contain lethal nodes. The BFS augmentation is a slightly better performer at maintaining clusters with lethal nodes. If our hypothesis is correct, we should find that the chordal network contains fewer on average clusters with lethal nodes. In the current case, the chordal networks had no clusters identified. Therefore, in AllegroMCODE, we loosened the parameters for the chordal network (degree cutoff=2, node score cutoff = 0.2, K-core=2, and max depth = 100), which identified between 100-200 small chordal clusters for each network. For these networks we examined the ratio of clusters with lethal nodes to total clusters for the top 25 clusters of each result; we found that the following: Yng-BFS-



Chordal = 28%, Yng-MST-Chordal = 32%, Mid-BFS-Chordal = 44%, and Mid-MST-Chordal = 46%. Indeed, the chordal approach finds less lethal nodes per cluster than any “All” augmentation result, lending evidence that re-addition of more edges makes assortative hubs more clearly evident.

To further probe our hypothesis that clusters contain high degree nodes from the original network, we perform the same analysis with hub nodes. We took the top 15% (determined to be an optimal hub threshold cutoff by [8]) of nodes according to degree from the original networks (top 802 nodes for Yng, top 832 for Mid) and examined how many clusters contained hub nodes. We find that for all clusters found by AllegroMCODE and CliqueCode, 90-100% of the clusters are composed of hub nodes from the original network. This confirms our hypothesis that the clusters we identify contain high-degree nodes, and that these nodes point towards those biologically relevant assortative hubs.

D. Parallel Results

We implemented a parallel algorithm for our Clique Code, where each vertex identifies whether its neighbors and itself together form a clique conforming to the bounds on fill-in. Since the networks obtained from the young and middle aged mouse were too small, we tested our scalability on larger networks obtained from creatine and untreated mice and breast cancer networks. The node and edge counts for the networks are as following: Untreated: 45020 nodes, 655698 edges; Creatine: 45023 nodes, 714628 edges; Familial: 48803 nodes, 687783; Non: 48803 nodes, 1109553 edges. The experiments were conducted on an Opteron multicore processor with 64 cores per node and 256 GB Ram per node. We used a shared memory OpenMP and tested the scalability of the code by execution over 1 to 64 threads. As shown in Figure 6, our code shows good scalability.

IV. DISCUSSION

In this study, we have examined how well our hybrid filter identifies dense clusters with high-degree nodes and biologically relevant nodes in correlation networks. It has been shown previously that network filters can remove noise from biological networks. We have identified that our filter,

which begins with a spanning tree and fills edges back in a quasi-chordal way, allows for edge removal with maintenance of high-degree and biologically relevant nodes in clusters. We speculate that the biologically relevant nodes that are maintained are assortative hubs, or those hub nodes whose neighbors are very well connected. Our results lend credence to this concept because as more edges are added in, cluster identification increases (node sensitivity and edge sensitivity), and the numbers of clusters containing lethal nodes and hub nodes from the original network matches or usurps the same levels in the original network. We have also implemented our own clustering method, CliqueCode, that identifies near-cliques within the network, and have also implemented a scalable version of this method in parallel. As network sizes continue to increase, it is important to know that methods for assessment of networks will be able to be parallelized and maintain integrity of results. Future work includes trying other values for the fill-in step and running a shared memory implementation of MCODE to compare its performance with parallel CliqueCode as well as parallelization of the filter.

ACKNOWLEDGMENT

This publication was made possible by the College of Information Science and Technology, University of Nebraska at Omaha and Grant P20 RR16469 from the NCRR, a component of the National Institutes of Health.

REFERENCES

- [1] Barabasi, A. L., & Oltvai, Z. N. (2004). Network biology: Understanding the cell's functional organization. *Nature Reviews Genetics*, 5(2), 101-113.
- [2] Bender, A., Beckers, J., Schneider, I., Holter, S.M. *et al.* (2008). Creatine improves health and survival of mice. *Neurobiol Aging* 29(9):1404-11.
- [3] Bult, Cj, Eppig JT, Kadin JA, Richardson JE, Blake JA; and the members of the Mouse Genome Database Group. 2008. The Mouse Genome Database (MGD): mouse biology and model systems. *Nucleic Acids Res* 36(Database Issue):D724-8.
- [4] Christakis, NA., *et al.* (2007). The spread of obesity in a large social network over 32 years. *New England Journal of Medicine*.357 no 4:370-379.
- [5] Dempsey, K., Duraisamy, K., Ali, H., & Bhowmick, S. (2011). A Parallel Graph Sampling Algorithm for Analyzing Gene Correlation Networks. Proceedings of the 2011 International Conference on Computational Science.
- [6] Dempsey, K., Duraisamy, K., Bhowmick, S., and H. Ali (2012). The Development of Parallel Adaptive Sampling Algorithms for Analyzing Biological Networks. 11th IEEE International Workshop on High Performance Computational Biology (HiCOMB 2012). May 21, 2012: Shanghai, China.
- [7] Dempsey, K., Bhowmick, S., and H. Ali. (2012). Function-preserving filters for Sampling in Biological Networks. *Proc Comp Sci* (9):587-595. Proceedings of the 2012 International Conference on Computational Science.
- [8] Dempsey, K. and Ali, H. On the discovery of Cellular subsystems in correlation networks using centrality measures (2012). *Current Bioinformatics*, 7(4).. Publication pending.
- [9] Duraisamy, K., Dempsey, K., Ali, H., and S. Bhowmick (2011). A noise reducing sampling approach for uncovering critical properties in large scale biological networks. *High Performance Computing and Simulation 2011 International Conference (HPCS)*: July 4-8. Istanbul, Turkey.
- [10] Dong, J., & Horvath, S. (2007). Understanding network concepts in modules. *BMC Systems Biology*, 1, 24.
- [11] Ewens, W. J., & Grant, G. R. (2005). *Statistical methods in bioinformatics* (Second Edition ed.). New York, NY: Springer.
- [12] Edgar, R., Domrachev, M., and AE. Lash (2002). Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nuc Acid Res* 30(1):207-10.
- [13] Enright A.J., Van Dongen S., Ouzounis C.A. *An efficient algorithm for large-scale detection of protein families*. *Nucleic Acids Research* 30(7):1575-1584 (2002).
- [14] Hao D, Li C (2011) The dichotomy in degree correlation of biological networks. *PLoS one* 6: e28322. doi: [10.1371/journal.pone.0028322](https://doi.org/10.1371/journal.pone.0028322).
- [15] Jeong, H., Mason, S. P., Barabasi, A. L., & Oltvai, Z. N. (2001). Lethality and centrality in protein networks. *Nature*, 411(6833), 41-42.
- [16] Lim E, Vaillant F, Wu D, Forrest NC *et al.* Aberrant luminal progenitors as the candidate target population for basal tumor development in BRCA1 mutation carriers. *Nat Med* 2009 Aug;15(8):907-13. PMID: [19648928](https://pubmed.ncbi.nlm.nih.gov/19648928/)
- [17] Linker, R. (1988). Self-organization in a perceptual network. *Computer* 21(3):105-117.
- [18] Newman, MEJ. Assortative mixing in networks. *Phys. Rev. Lett.*, 89:208701-1 – 208701-4, Oct 2002.
- [19] Oppen-Rhein, R., & Strimmer, K. (2007). From correlation to causation networks: A simple approximate learning algorithm and its application to high-dimensional plant gene expression data. *BMC Systems Biology*, 1, 37.
- [20] Verbitsky, M., Yonan, A. L., Malleret, G., Kandel, E. R., Gilliam, T. C., & Pavlidis, P. (2004). Altered hippocampal transcript profile accompanies an age-related spatial memory deficit in mice. *Learning & Memory* (Cold Spring Harbor, N.Y.), 11(3), 253-260.
- [21] Subramanian, A., Tamayo, P., Mootha, VK., Mukherjee, S., Ebert, BL., Gillette, MA., Paulovich, A., Pomeroy, SL., Golub, TR., Lander, ES., and JP Mesirov (2005). Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proc Natl Acad Sci* 102(43):15545-15550.
- [22] Gleich, D. 16 May 2009. Gaimc: Graph Algorithms in Matlab Code. *Mathworks.com*. Obtained on 01.17.2012, from <http://www.mathworks.com/matlabcentral/fileexchange/24134>
- [23] Yoon, JS and WH Jung (2011). A GPU-accelerated bioinformatics application for large-scale protein interaction networks. APBC poster presentation.