computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

A PARALLEL TEMPLATE FOR IMPLEMENTING FILTERS FOR BIOLOGICAL CORRELATION NETWORKS

Kathryn Dempsey, Vladimir Ufimtsev, Sanjukta Bhowmick, Hesham Ali

College of Information Science and Technology, University of Nebraska at Omaha
E-mail: hali@mail.unomaha.edu

Abstract: High throughput biological experiments are critical for their role in systems biology – the ability to survey the state of cellular mechanisms on the broad scale opens possibilities for the scientific researcher to understand how multiple components come together, and what goes wrong in disease states. However, the data returned from these experiments is massive and heterogeneous, and requires intuitive and clever computational algorithms for analysis. The correlation network model has been proposed as a tool for modeling and analysis of this high throughput data; structures within the model identified by graph theory have been found to represent key players in major cellular pathways. Previous work has found that network filtering using graph theoretic structural concepts can reduce noise and strengthen biological signals in these networks. However, the process of filtering biological network using such filters is computationally intensive and the filtered networks remain large. In this research, we develop a parallel template for these network filters to improve runtime, and use this high performance environment to show that parallelization does not affect network structure or biological function of that structure.

Keywords: high performance computing, correlation networks, parallel computing, network filters, graph algorithms, noise, biological signal.

1. INTRODUCTION

High-throughput assays are now able to take surveys of the entire cellular landscape at once – be it gene expression, protein function, or any other experimentally quantifiable measure. The technological capacity for examining the minutiae on the grand scale is growing, and with it grows the need for analyses that are both computationally robust and informative. The inherent danger of these experiments and their post-completion analytics lies in the sea of information available. It is possible to find multiple needles in the proverbial haystack, and extremely difficult to discern which needle is the biological candidate for causing any observed phenotypical deviations from the norm, be that disease, aging, or some other biological phenomenon. Simply put, the increase in technological capacity is accompanied, then, by an increase in data heterogeneity, volume, and noise – leading to biological "big data" [10].

To accommodate these specific problem areas, the network model has been employed as an effective tool for data visualization and analysis. Among others, three of the major reasons why network modeling is becoming popular include (1) networks are easy to work with, (2) networks retain the ability to represent relationships between biological entities (not just the entities themselves), and (3) well-established graph theoretic approaches can be used on the network model for analysis. Graph theory has been around at least since the 1700's, ever since Leonhard Euler proposed his solution to the Seven Bridges of Königsberg Problem, and ever since, methods to iterate through and understand the graph model have been identified, solved, and analytically improved.

Consider, then, the problem posed by high-throughput experimentation: large sets of heterogeneous data contain multiple levels of information, (functional ontology, pathway information, gene to protein attributes, etc.), not all relevant to the research query at hand. The network model becomes an ideal tool for the analysis of these datasets, if used cleverly, and if the model is shown to provide useful information. Indeed, as Albert-László Barabási and his team first proposed in their sentinel 1999 work "Emergence of Scaling in Random Networks" and then again in their 2001 follow-up "Lethality and Centrality in Protein Networks," networks can be used to reveal important information about an organism on the

cellular level. In particular, the two publications mentioned established that the degree distribution of many real world networks, including the protein interaction network (PPI), followed a characteristic power-law distribution. In a protein interaction network, nodes are typically representative of proteins, and edges exist between two nodes if there is a measurable, physical interaction between two proteins. This power-law distribution proposed means that the network itself contains few nodes that are highly connected to others, and many nodes that are poorly connected. Since then, the network model in the biological realm has exploded in popularity, and other biological relationships to structural properties of the model have been found, for example, it is now established that within the PPI model, clusters of genes, particularly cliques or completely connected subnetworks, tend to represent protein complexes. The typical protein complex is a conglomeration of proteins that all interact together to perform some function, and will not function without "participation" of all its components. In this way, many new proteins required for cellular function have been identified.

Multiple network models have been proposed to represent biological data: the PPI, the metabolic network, the transcriptome, etc. While they all can be similar (and more importantly, aligned and integrated), there are inherent differences in each model type and benchmarking of the structures native to each model must be performed for that model to become useful to the research of systems biology [4]. In that regard, the correlation network, or a network where nodes represent genes and edges represent a correlation of expression pattern for those genes, is a type of network that is only recently becoming more well understood and finding popularity. Correlation networks have been found to mirror some of the major findings in biological network theory; for example, structures within these networks (hubs, clusters [8], etc.) can point to biological functions, and the relationships between those genes (which may previously may have been unknown). While these networks are increasing in popularity, the issue remains that networks are typically large and noisy [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 has shown that filters imposed on networks that represent gene co-expression (this co-expression can be coincidental or causative) are an effective means for removing "noisy" edges while enhancing biological signal. Duraisamy *et al.* [19] and Dempsey *et al*. [15-17] found that filters that augment networks such that edges that exist as part of a cycle (a path of connected nodes where the original node in the path and the terminal node in the path are the same) typically are found to represent noise. A filter, for example, can remove around 25 % of relationships from the original network, while also maintaining clusters that exist in the original network. The filter can also reveals clusters that were previously "hidden" or undiscoverable by common clustering algorithms due to density or neighborhood distortion. Dempsey *et al*. [17] explored how a maximum weighted 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 [18], or nodes that represent genes that when knocked out *in vivo* results in expiration of the organism at some early stage in development [14].) This study found that by 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, a "hybrid" filter was created 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 three major versions of the network that we are able to test for biological relevance; these variations are shown in Fig. 1.

"Hub" nodes in correlation networks can be disassortative or assortative [27], 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.* [17] show that while the aforementioned maximum spanning tree (MAXST) filter is able 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 MAXST 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, makes identification of these 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 [1] 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.

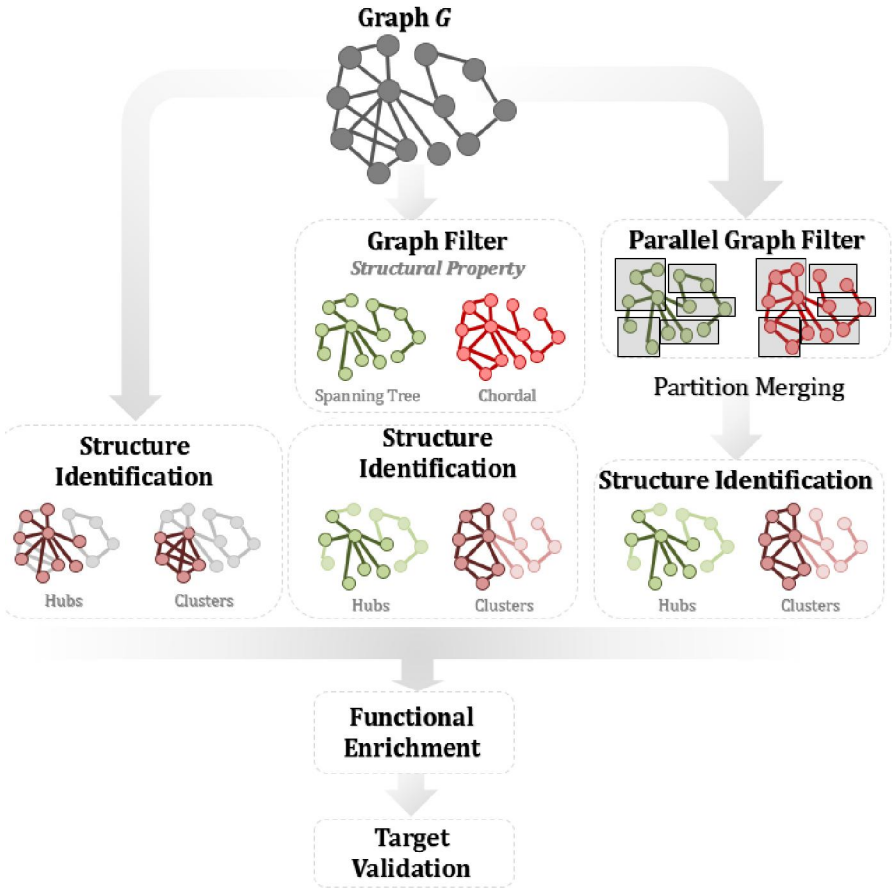

Fig. 1 – An overview flowchart of this approach.

In this study, we further examine the applicability of this hybrid filter by examining its effectiveness in enhancing clusters in correlation networks. Previous studies on chordal filters by [15-17] 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 parallel filtering approach to networks generated from an aging mouse gene expression study to show its effectiveness in identifying clusters and the speedup that results. An overview of our method is shown in Fig. 1. Previous work in this area of biological correlation network filters used relatively small networks, and as such it is crucial to show that these filters are able to scale and maintain the same result.

2. HYPOTHESIS

In this study, we further examine the applicability of the previously studied filters by examining their effectiveness in enhancing clusters and hubs in correlation networks. Gene expression correlation networks tend to get large when analyzed on the whole-genome scale, so it is important to be able to parallelize the process and also reduce runtime of the clustering method (typically the longest step in the analytic pipeline), while still being able to identify relevant biological clusters. From our previous results using network filters, we have observed the following phenomena:

- Chordal filters maintain network clusters [15]
- Spanning tree filters maintain lethal hub nodes [16]
- The hybrid filter maintains lethal hub nodes *best* when edges are added back into the

network without necessarily maintaining chordality [17]
- Almost all the chordal-like edges are added to the spanning tree in 1 to 2 iterations [8]

Based on these observations, we can propose our hypothesis for how well the hybrid filter is able to identify clusters, in addition to the rather straightforward hypothesis that the process of filtering networks should reduce their density and in turn reduce the search space and the computational time of extracting information from the networks.

- *H0a: There are significant independent tasks associated with graph-theoretic network filtering which implies that Parallelization of these filters results in decreased runtimes, indirectly leading to faster analysis.*
- *H0b: Filtered networks retain their relevant biological structures, including hubs and clusters.*
- *H0c: The changes due to parallelization do not significantly effect the utility of the filters.*

3. METHODS

3.1. NETWORK CREATION

Networks are created by using data from NCBI's Gene Expression Omnibus, which houses data from microarray, RNA-seq, and other high-throughput assays [2]. The data was taken from GSE8150 [5], which uses brain tissue from mice at ages 5 months and 30 months. Three datasets total resulted from the GSE8150 experiment:

1. Untreated mice at 5 months (YMBC)
2. Treated with α-tocopherol, 30 months (MOBATOP)
3. Treated with γ-tocopherol, 30 months (MOBAGTOP)

Briefly, mouse-aging networks were created using the pairwise correlation coefficient calculated for each pair of genes by measuring correlation of pattern expression. Genes or gene products are represented in the network as nodes. Correlation can fall between -1.00 and 1.00, and correlations passing hypothesis testing using the Student's T-test (p-val < 0.0005) are used to draw an edge between the representative nodes, with the weight of the edge set to the actual correlation score.

3.2. TEMPLATE FOR FILTERING ALGORITHMS

The primary goal of filtering algorithms used in our experiments is to preserve the structural properties of the networks that highlight the corresponding system properties. Specifically, we use maximum spanning tree (MAXST) filters to identify hubs, which represent lethal nodes and chordal graph based (CHD) filters to extract important communities in the networks. Both these filters, as well other structural sampling methods such as random walk, forest fire, breadth first search (BFS), etc. are based on network traversal. As part of our implementation we propose a template that can be easily modified for all such graph traversal-based sampling algorithms.

To create such a template, we observe that all graph traversal algorithms follow this pattern;

(i) Select a start vertex

(ii) Identify its neighboring nodes that have not been visited

(iii) Put a priority value to the neighboring nodes.

For example, the priority for BFS is the distance from the root; for MAXST using Prim's method [27] it is the neighbor with the highest weight; for CHD using Dearing's Algorithm [28] it is the neighbor with the most connections to the filter graph

(iv) Add the neighboring nodes to a maximum priority queue. Mark them as visited.

(v) Remove the top node from the priority queue

(vi) Add this node to the filtered network, if it maintains certain structural properties

For BFS and MAXST the network should remain acyclic. For CHD the size of a cycle cannot be more than three.

(vii) This new node becomes the start vertex.

(viii) The process is continued until all the vertices have been visited.

Biological networks can often have disconnected components (generally, one giant component and many small ones). This template for graph traversal can also be modified for disconnected components by adding a check to ensure that when a traversal ends, i.e. there are no more new vertices to add to the priority queue, there are also no unvisited vertices remaining. If an unvisited vertex is found, then it belongs to a new component, and it is selected as the next start node. Note that the only change required to implement a new traversal method is at steps (iii) and (vi). Thus this template facilitates easy modification and experimentation of new traversal techniques.

3.3. PARALLEL FILTERING ISSUES AND SOLUTIONS

We now discuss a parallel implementation for the traversal template. The template by itself does not lend easily to parallelization since the start nodes are selected sequentially from the priority queue one by one. Although BFS has a wave front like expansion of neighbors from which traversal can be done in parallel, this is a specialized case and does not hold

for CHD or MAXST. Another area of parallelization is when searching for the neighbors. However most biological networks are scale-free and therefore most of the nodes have low degree. Thus parallelizing the neighbors search will not significantly affect the running time.

We therefore decided to partition the network across different processing units (in this case, threads), execute the traversal for each partition separately and then combine the filters obtained from each partition. One issue in this method is that although the individual filters maintain the specified traversal properties, it is more difficult to ensure that the combined filter will also do so. In fact, attempting to maintain the BFS or MAXST tree or a chordal graph across the combined filters, often results in the combination process becoming sequential and extremely time consuming. As a compromise we decided to opt for quasi-filters instead of exact filters. This helps reduce the time and makes the combination process parallel as well. Moreover, the quasi-filters maintain most if not all of the properties of the exact filters and therefore do not affect the analysis results significantly.

The combination process for BFS and MAXST is as follows; add exactly one edge between partitions Pi and Pi+1, and no edge between the first and the last partition. It is easy to see that the resulting quasi-filter is still a tree. However, for BFS some of the nodes may not be in the shortest path from the root, as would be the case for an exact BFS tree and for MAXST the edges connecting two partitions may not be the ones with the maximum weight. In the combination process for CHD one node from partition Pi is connected to all its neighbors in Pi+1. If the neighbors are connected then the chordal graph property is maintained, otherwise we will end up with a few cycles of length greater than three. So long as the percentage of such larger cycle is significantly smaller than the number of triangles the benefits of the chordal graph are maintained. Note that since Pi connects to Pi+1 and Pi+1 connects to Pi+2, each of the combination steps can be done in parallel by each partition (except for the last partition which does not combine). This increases the scalability of the filter.

3.4. CLUSTERING

Clustering was performed using MGClus [5] under default parameters. MGClus aims to identifying clusters that exist in *large biological networks* and has been shown to perform well in PPI's, whose clusters tend to be dense and range from very small (5 nodes) to large. In particular, MGClus runs very quickly at the command line which is why it was chosen for these very large biological networks.

3.5. DESCRIPTION OF EXPERIMENTS

Experiments were performed in quadruplicate for each of the three networks (YMBC, MOBATOP, MOBAGTOP) to highlight the consistency of our approach. There are several parameters to be measured within this set of research. These major parameters are:

- *Filter*
  - *Node Selection*
  - *Filter Iteration*
- *Speedup*

The first measure that we will address is the filter itself – what type of filter is applied, and how many iterations of edges are added back in. Two sub-parameters of the filter itself are node selection – how nodes used to filter the network is selected. The node selection process for the initial tree can use a breadth-first-search (BFS) or maximum weighted spanning tree (MAXST), or a chordal filter (CHD). The chordal filter itself is not a node selection method per se, but it filters the network such that the final network model is a chordal subgraph of the original. The second sub-parameter of the filter is the augmentation of the network, which determines how edges are added back to the tree. Edges are added back only if they are present in the original network, by adding them between nodes at distance-2 in the tree. This operation creates triangles, which is required for chordal graphs. The constraints can be tight such that only chordal graphs are created or loose (quasi) where some larger cycles are allowed. This parameter also can be iterated over many times, or none. In this paper, we use iterations of 0-3, meaning that at iteration 0, no augmentation is performed, at 1, only one round of augmentation is performed, and at 2, two rounds are performed, and so on. In a recent paper by West *et* al., it was found that the filtered network rarely changes its inherent base structure after the first few iterations, so extension beyond 3 iterations should not be required. Finally, we compare each of these parameterizations sequentially and in parallel.

4. EXPERIMENTAL RESULTS

The goal of this study is to establish benchmarks for the time required to build and analyze networks, and establish how changes in parameters affects this runtime in sequential and parallel environments. The structure of our experiments and research in this manuscript follows as thus: the effect of filter on network size (3.1), the effect of filters on cluster count and overlap with original clusters (3.2), how scalable are the parallelizations (3.3.). Biological impact of these results is described using hubs (3.4.) and clusters (3.5).

4.1. FILTERED NETWORK SIZE

One of the first measures that can indicate the power of a filter is edge density. For each network, we measure the number of nodes present after filtering (n) and the number of edges present after filtering (e). If the network was completely connected, we know the total number of edges present would be equal to $E = n*(n-1)/2$, assuming no self-loops or multiple edges. Next, to determine edge density, we take the number of edges in the filtered network (e) divided by the total number of edges possible (E) and present it as a percentage to give the edge density. The edge densities of all networks during sequential runs are shown in Fig. 2.

Fig. 2 – Edge Density Results for Sequential Runs of each network for each filter. Edge density is represented on the y-axis, as Total Edges/ total Possible Edges *100. The x-axis represents the network, filter, and iteration used.

Typically, correlation networks tend to be sparse, and have low levels of edge density, but this does not necessarily mean the network is small or easily manageable. For example, the ORIG YMBC network (shown at bottom in green, Fig. 2), has an edge density of 0.11 %, meaning that the number of edges in the represents less than 1 % of the possible edges given the number of nodes. However, this low number is deceiving. The network has 43,021 nodes and 1,050,293 edges, which is too large for graphic visualization even with the most current network GUIs; a network of this size must be handled at the command line. This task might be challenging for a person not primarily trained as a bioinformatician or computer scientist.

It is clear from each of the three networks in Fig. 2 (MOBAGTOP at top in blue, MOBATOP at middle in red, and YMBC at bottom in green) that the MAXST filter, at every increasing iteration, reduces edge density drastically from the network.

This would be expected, as the filter is more stringent – originally it creates a tree which by nature has no cycles. The CHD filter, in which edges are added to increase the neighborhood connectivity where clusters exist, far better maintains original edge density with every increasing iteration, particularly at $i = 3$. The only iteration of CHD that is similar in edge density to the MAXST filter is for $i=0$; after this, we see a large jump for every filter and every network from $i=1$ and on. Whether or not this is beneficial will be revealed in the biological and clustering analysis.

4.2. CLUSTER COUNT AND OVERLAP

The clusters identified by MGClus are based on shared neighbors, and it should be noted that any clustering algorithm will perform differently based on its core techniques. MGClus was designed for large biological networks and was shown to perform well in random networks (does not identify noise as signal) and well in PPIs (identifies dense clusters based on shared neighborhoods and neighborhood topology). The results of the clusters in terms of cluster size are shown in Fig. 3. It is clear that in terms of clustering consistency, the CHD filter far outperforms MAXST; in fact, the MAXST filters behave in an unexpected fashion in that the number of clusters per network actually decreases with addition of edges. One might speculate that this occurs due to the fact that adding edges back in will create clusters where there were *dense* neighborhoods in the original network, creating actual neighborhoods instead of small groups of poorly connected nodes (resulting in the initial high number of clusters). Therefore, the smaller clusters in the earlier trees get merged.

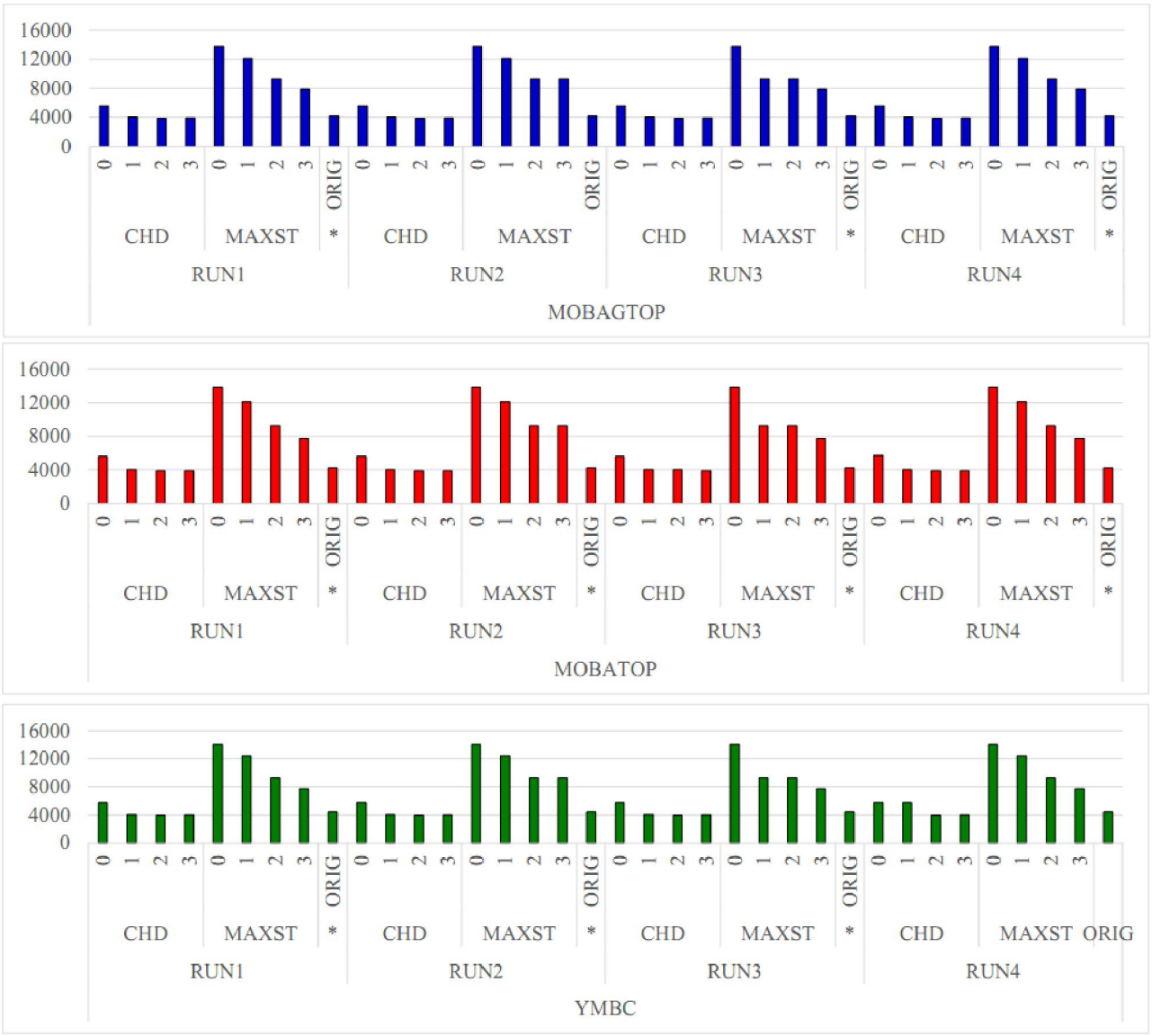

Fig. 3 – Cluster Count Results for Sequential Runs of each network for each filter. Clusters found per network is represented on the y-axis. The x-axis represents the network, filter, and iteration used.

One of the methods used to compare how well clusters in original networks are also identified in filtered networks is through cluster overlap. To determine cluster overlap, a list of each cluster in the original network and a list of each cluster in the filtered network was compared. The comparison or score (*Cluster Overlap Score)* was determined by checking the node overlap of clusters. For each cluster in original network G_O, each node was placed in a hash H_O. Then for each cluster in the filtered network G_F, each node was placed in a hash H_F. The two hashes are then compared, and if a node occurs in both clusters, it is scored as a match. The final match score for each cluster comparison is defined as the number of matches divided by the *total number of nodes in the original cluster*. Each original cluster is compared to each filtered cluster, and the *Maxscore* is defined as the highest final match score for that original cluster, or the overlap score of the original cluster compared and the highest overlapping cluster in the filtered network according to their node comparison. Note that the higher the *Maxscore*the better. *Maxscores* with a value greater than 1 indicate that multiple clusters within the filtered network overlapped with the original cluster, and overlaps are counted if the *Maxscore* of each of those clusters is equal. A *Maxscore* with a value greater than 1 is considered a sign of robustness of the cluster and the filter.

The *Maxscore*s of each network at RUN1 are shown in ranked order (lowest to highest) in Fig. 3. (Runs 2-4 are not shown as the filtered networks are extremely similar to RUN1). In Fig. 3, it is evident that there is a wide range of *Maxscores* for each original cluster, but it is quite evident that the CHD filters (at each iteration 0,1,2, and 3) are better performers than the MAXST filters in terms of finding clusters that are robust and having good overlap.

4.3. PARALLEL RESULTS AND SCALABILITY

For each of the two filters (CHDand MAXST) the scalability results are shown in Fig. 4. The machine used was a 64-bit with two physical Quad-Core AMD Opteron Processors (2394.112 Mhz CPU) with 32 GB of RAM per processor. The number of threads used ranges from 1 to 16 (specifically, the data points are for 1,2,4,8, and 16 threads) and the run time is just for the parallel portion of the algorithm not for the I/O of reading in the file and writing out the results.

After 16 threads, in each network, the partition becomes too small for effective parallelization and the overhead from the communication and combining all of the parts back together dominates the runtime.

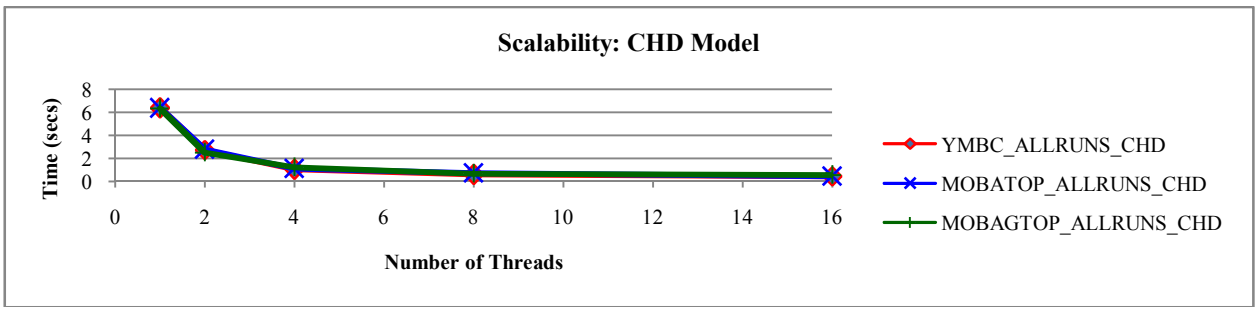

Fig. 4 – Scalability Results for the three networks with no iterations per CHD filter.

For each model there are 3 charts (MOBATOP, MOBAGTOP, and YMBC) and each chart shows the runtimes for different numbers of threads for each version of the network. For example, for the CHD model the chart for MOBATOP contains runtimes for MOBATOP_RUN1, MOBATOP_RUN2, MOBATOP_RUN3, and MOBATOP_RUN4. Table 1gives the number of vertices and edges in each network used.

Table 1. Sizes of networks tested.

Network	Vertices	Edges
MOBATOP	44577	2026962
MOBAGTOP	44564	1987326
YMBC	44875	2100586

Chordal Model. As is seen in Fig. 4, the CHD model performs well on all variants of each network. Notice that the shape of each of the curves is close to perfect scaling. When graphed using a log-log plot,

the curve actually becomes a straight line (with slope close to 1) meaning that when the number of threads is doubled, the runtime is cut in half. In fact, when scaling from 1 thread to 2 threads, the runtime is reduced by more than one half in every case. The same is true when comparing the runtimes for 2 and 4 threads. However, for 8 and 16 threads the runtimes do not get reduced as much and after 16 threads they begin to increase indicating that there is no benefit to increasing the number of threads after 16. *MAXSTModel*. The MAXST model (as shown in Fig. 5) performs the best out of the two models on all variants of each network. Notice that the shape of each of the curves is closer to perfect scaling than for the CHD model. When graphed using a log-log plot, the curve becomes a straight line (with slope greater than 1) meaning that when the number of threads is doubled, the runtime is cut by more than half. When doubling the number of threads starting from 1 all the way up to 16 threads the factor by which the runtime is reduced is more or less the same (around 60 %) i.e. runtime is cut down by the same proportion each time the number of threads are doubled (up to 16).

However, as was seen in the CHD model, after 16 threads the runtimes increase indicating that for networks of this size, it makes no sense to increase the number of threads past 16.

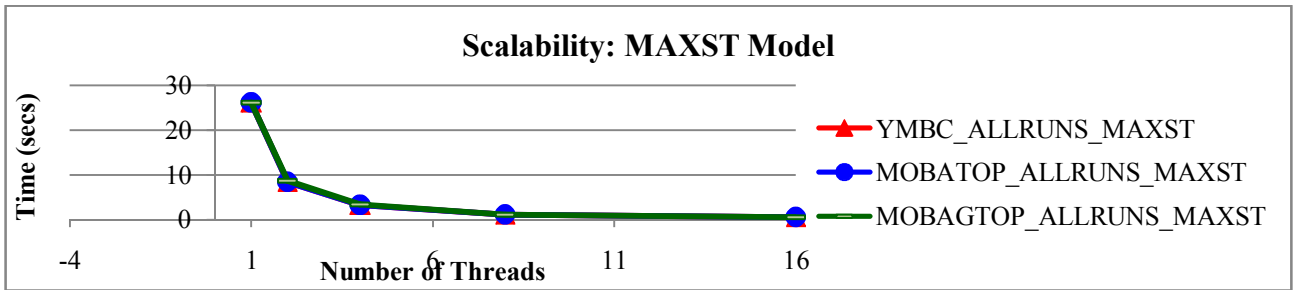

Fig. 5 – Scalability Results for the three networks with no iterations per MAXST filter.

4.4. BIOLOGICAL RESULTS – CLUSTERS

Due to the large amount of networks and subsequently, clusters generated, (over 777,000), biological significance of each will not be performed. Previous studies have shown that the chordal filters tend to maintain and/or enhance the biological function of found clusters if such a function exists. To highlight this, one example of cluster overlap is taken from the parallel code: cluster 2 from RUN4 of the YMBC original network was found to have a 61 % overlap with cluster 4037 from RUN4 of the YMBC CHD-1 filtered network. Between the two clusters, 20 nodes were found to overlap and 39 were unique to either the original or filtered network. A list of nodes found in both clusters is shown in Table 2.

The Gene Ontology profiles for both sampled clusters are shown in Fig. 6. .

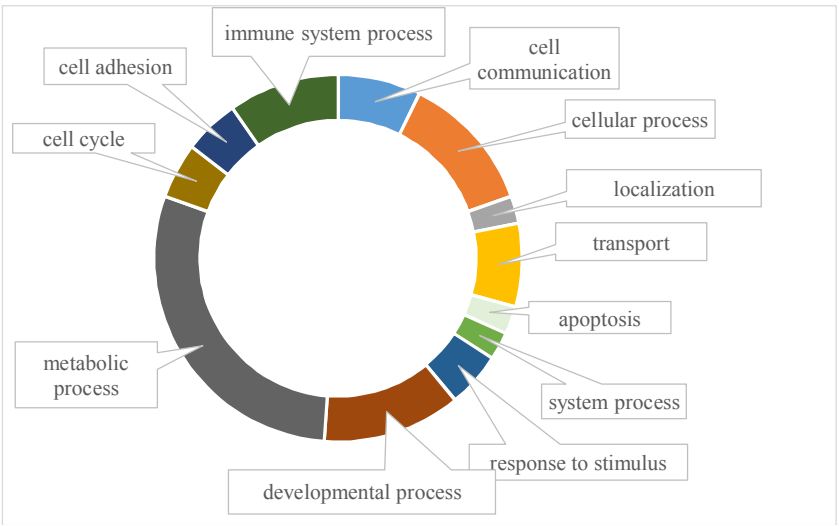

Fig. 6 (a). Gene Ontology profiles (Biological Process tree) for the sampled clusters from the original network.

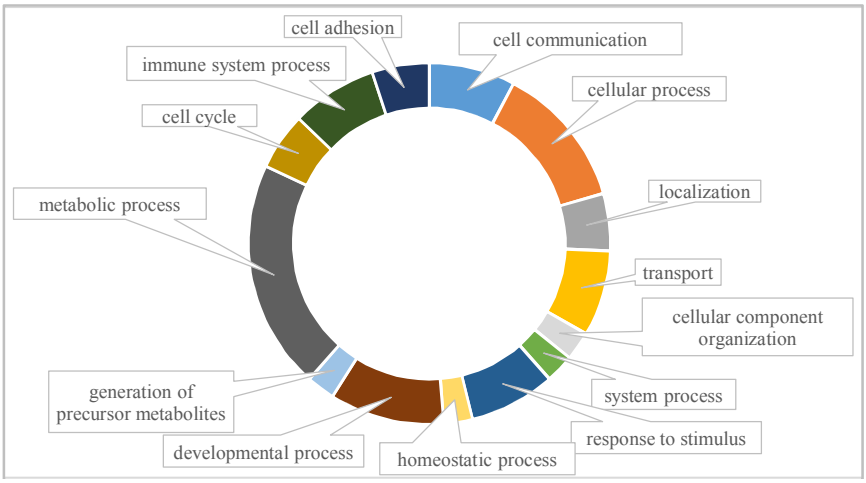

Fig. 6 (b). Gene Ontology profiles (Biological Process tree) for the sampled clusters from the filtered network.

Table 2. Node IDs, MGI IDs, and Gene Symbols for Original and Filter sample clusters from the YMBC RUN4 networks.

ORIGINAL CLUSTER			**FILTERED CLUSTER**		
Input	*ID*	*Symbol*	*Input*	*ID*	*Symbol*
1443866_at	MGI:2442106	Lrtm1	1429534_a_at	MGI:1923864	Immt
1451208_at	MGI:2385071	Etfl	1443866_at	MGI:2442106	Lrtm1
1460331_at	MGI:1915309	Tm9sf2	1453367_a_at	MGI:1923442	Abhd12
1422621_at	MGI:894323	Ranbp2	1422621_at	MGI:894323	Ranbp2
1453367_a_at	MGI:1923442	Abhd12	1457846_at	MGI:1917052	Cox11
1438511_a_at	MGI:1913464	Rgcc	1416514_a_at	MGI:1352745	Fscn1
1456035_at	MGI:2685230	Nxf3	1451655_at	MGI:2672859	Slfn8
1435569_at	MGI:2143561	D630029K05Rik	1424360_at	MGI:2384576	Tti2
1427672_a_at	MGI:1095419	Kdm6a	1435569_at	MGI:2143561	D630029K05Rik
1429534_a_at	MGI:1923864	Immt	1421233_at	MGI:1201409	Pknox1
1417086_at	MGI:109520	Pafah1b1	1430307_a_at	MGI:97043	Me1
1434508_at	MGI:1917343	Ube2q1	1460331_at	MGI:1915309	Tm9sf2
1437086_at	MGI:96919	Ascl1	1451208_at	MGI:2385071	Etfl
1421815_at	MGI:2145369	Epdr1	1417086_at	MGI:109520	Pafah1b1
1453214_at	MGI:1921738	Lrrc15	1450080_at	MGI:1920115	Cxx1c
1421874_a_at	MGI:1928138	Mrps23	1460726_at	MGI:87948	Adss
1451655_at	MGI:2672859	Slfn8	1419703_at	MGI:1858212	Col5a3
1457846_at	MGI:1917052	Cox11	1422409_at	MGI:104877	Hes3
1416514_a_at	MGI:1352745	Fscn1	1421878_at	MGI:1346862	Mapk9
1424360_at	MGI:2384576	Tti2			
1420598_x_at	MGI:99592	Defa-rs2			
1418751_at	MGI:1889342	Sit1			
1421233_at	MGI:1201409	Pknox1			
1422409_at	MGI:104877	Hes3			
1422699_at	MGI:87998	Alox12			
1421878_at	MGI:1346862	Mapk9			

While there are a few differences in the cluster profiles (apoptosis present in the Original cluster only, and three terms – generation of precursor metabolites, homeostatic process, and cellular component organization – are present in the Filtered cluster only) – the two clusters largely have a similar ontological profile, even considering that the filtered cluster has 7 less genes than the original and out of both clusters, only 20 genes overlap. (The Original cluster has 26 genes and the Filtered has 19). This highlights how we are able to maintain the biological integrity of the original network structure while reducing network size, cluster size, and noise. A level of discovery is even added with the finding of three new possible functions performed by the noted gene cluster.

An example of a cluster with low overlap to an original cluster can be found in the YMBC network using the CHD-1 filter. The first cluster found in this network contains 42 genes but has just 30 % overlap with any original clusters, meaning that this cluster is one that has been "found" or revealed with the removal of noise. This cluster was analyzed for functional enrichment using the GeneTrail [26] tool using default parameters. The cluster was found enriched in three main biological processes (P-val<0.05): cell development, cellular component morphogenesis, and cell morphogenesis. The Gene Ontology subtree for these functions is presented in Fig. 7 below. This is just one example of how a cluster that is not found in the original network can be discovered in the filtered network, or how noise removal can strengthen biological signal.

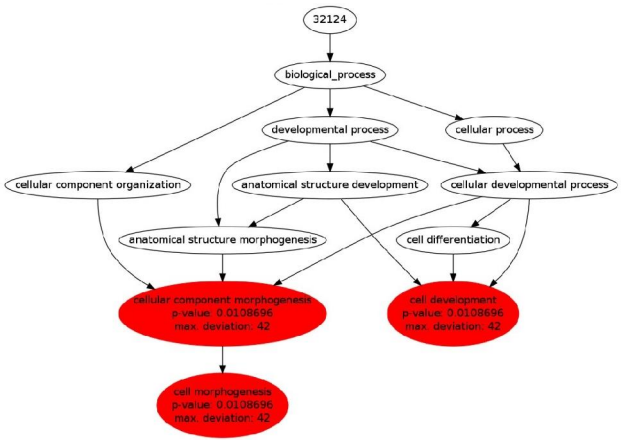

Fig. 7 – TheGene Ontology tree generated by GeneTrail Express [28] after enrichment of the second cluster of the filtered CHD-1 YMBC network.

5. 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. The filter presented here can identify chordal or maximum spanning tree subgraphs, and also has the ability to add edges from the original network back in to bias that network toward dense communities. The speedup curves of the parallelized filter highlights the necessity of this parallelization, proving our original hypotheses, *H0a,* that *there are significant independent tasks associated with graph-theoretic network filtering which implies that Parallelization of these filters results in decreased runtimes, indirectly leading to faster analysis.* Secondly, the networks were shown to be consistent in terms of filter edge removal and cluster identification across multiple networks and multiple runs, proving our third hypothesis, *H0c*: That t*he changes due to parallelization do not significantly effect the utility of the filters.* Finally, we use examples of biological relevance to highlight our second hypothesis, showing that biological structures and their meanings are not affected by the parallelization of the filters. As the amount of information that needs to be incorporated into the network model grows, this parallel template can be trusted to improve computational runtimes for a faster and robust analysis.

6. 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.

7. REFERENCES

[1] A. L. Barabasi, & Z. N. Oltvai. Network biology: Understanding the cell's functional organization, *Nature Reviews Genetics*, (5) 2 (2004), pp. 101-113.

[2] T. Barrett, S. E. Wilhite, P. Ledoux, et al. NCBI GEO: archive for functional genomics data sets – update 2013, *Nucleic Acids Research*, (41(D1), (2013), pp. D991-D995.

[3] O. Frings, A. Alexeyenko, and E. L. Sonnhammer, MGclus: network clustering employing shared neighbors, Molecular Biosystyems, (9) 7 (July 2013), pp. 1670-1675.

[4] J. Reichardt, Structure in Complex Networks, *Lecture Notes in Physics*, Vol. 766, Springer, Berlin, 2009.

[5] E. Reiter, Q. Jiang, and S. Christen, Anti-inflammatory properties of alpha- and gamma-tocopherol, *Molecular Aspects in Medicine*, (28) 5-6 (October-December 2007), pp. 668-691.

[6] What is Big Data | Big Data Explained, Villanova University, n.p. n.d., January 5, 2014 www.villanovau.com/university-online-programs/what-is-big-data/

[7] S. West, K. Dempsey, S. Bhowmick, and H. Ali. Analysis of incrementally generated clusters in biological networks using graph-theoretic filters and ontology enrichment, *IEEE International Conference on Data Mining (ICDM 2014)*, Dallas, Texas, USA, December 7-10, 2013.

[8] G. D. Bader, and C. H. W. Hogue. An automated method for finding molecular complexes in large protein interaction

networks, *BMC Bioinformatics*, (4) 2 (January 2003), pp. 2.

[9] C. J. Bult, J. T. Eppig, J. A. Kadin, J. E. Richardson, J. A. Blake et al. The Mouse Genome Database (MGD): mouse biology and model systems, *Nucleic Acids Ressearch*, (36) Database Issue (January 2003), pp. D724-D728.

[10] K. Dempsey, K. Duraisamy, H. Ali, & S. Bhowmick. A parallel graph sampling algorithm for analyzing gene correlation networks, *Proceedings of the 2011 International Conference on Computational Science*, Vol. 4, 2011, pp. 136-145.

[11] K. Dempsey, K. Duraisamy, S. Bhowmick, and H. Ali. The development of parallel adaptive sampling algorithms for analyzing biological networks, *Proceedings of the 11th IEEE International Workshop on High Performance Computational Biology (HiCOMB 2012)*, May 21, 2012, www.hicomb.org/proceedings.html

[12] K. Dempsey, S. Bhowmick, and H. Ali. Function-preserving filters for sampling in biological networks, *Proceedings of the 2012 International Conference on Computational Science*, Vol. (9), 2012, pp. 587-595.

[13] K. Dempsey, and H. Ali. On the discovery of cellular subsystems in correlation networks using centrality measures, *Current Bioinformatics*, (7) 4 (2012), publication pending.

[14] K. Duraisamy, K. Dempsey, H. Ali, and S. Bhowmick. A noise reducing sampling approach for uncovering critical properties in large scale biological networks, *Proceedings of the High Performance Computing and Simulation International Conference (HPCS)*, July 4-8, 2011, pp. 721-728.

[15] J. Dong, & S. Horvath, Understanding network concepts in modules, *BMC Systems Biology*, (1) 24 (June 1, 2007).

[16] W. J. Ewens, & G. R. Grant, *Statistical Methods in Bioinformatics*, 2nd edition, New York, NY: Springer, 2005.

[17] R. Edgar, M. Domrachev, and A. E. Lash. Gene expression omnibus: NCBI gene expression and hybridization array data repository, *Nucleic Acids Research*, (30) 1 (2002), pp. 207-210.

[18] A. J. Enright, S. Van Dongen, C. A. Ouzounis. An efficient algorithm for large-scale detection of protein families, *Nucleic Acids Research*, (30) 7 (2002), pp. 1575-1584.

[19] H. Jeong, S. P. Mason, A. L. Barabasi, & Z. N. Oltvai. Lethality and centrality in protein networks, *Nature*, (411) 6833 (2001), pp. 41-42.

[20] R. Linkser, Self-organization in a perceptual network, *Computer*, (21) 3 (1998), pp. 105-117.

[21] M. E. J. Newman. Assortative mixing in networks, *Physical Review Letters*, (89) 20 (October 2002), pp. 208701.

[22] R. Opgen-Rhein, & K. Strimmer. 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 (August 2007).

[23] M. Verbitsky, A. L. Yonan, G. Malleret, E. R. Kandel, T. C. Gilliam, & P. Pavlidis. Altered hippocampal transcript profile accompanies an age-related spatial memory deficit in mice, *Learning & Memory*, (11) 3 (2004), pp. 253-260.

[24] A. Subramanian, P. Tamayo, V. K. Mootha, S. Mukherjee, B. L. Ebert, M. A. Gilette, A. Paulovich, S. L. Pomeroy, T. R. Golub, E. S. Lander, and J. P. Mesirov. Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wise expression profiles, Proceeding of the National Academy of Sciences, (102) 43 (2005), pp. 15545-15550.

[25] D. Gleich. Gaimc: Graph Algorithms in Matlab Code. 16 May 2009. Mathworks.com. Obtained on 01.17.2012, from http://www.mathworks.com/matlabcentral/fileexchange/24134.

[26] A. Keller, C. Backes, M. Al-Adwahi, A. Gerasch, J. Keuntzerm, O. Kohlbacher, M. Kaufmann, and H. P. Lenhof. GeneTrailExpress: a web-based pipeline for the statistical evaluation of microarray experiments, *BMC Bioinformatics*, (9) 552 (December 2008).

[27] R. C. Prim. Shortest connection networks and some generalizations, *Bell System Technical Journal*, (36) (1957), pp. 1389–1401.

[28] P. M. Dearing, D. R. Shier, and D. D. Warner, Maximal chordal subgraphs, *Discrete Applied Mathematics*, (20) 3 (1988), pp. 181-190.

***Dr. Kate Dempsey**: Research Associate in the School of Interdisciplinary Informatics within the College of IS&T at UNO. She has been a member of the UNO Bioinformatics Research Group since September 2005, and has been involved with the UNO Bioinformatics Core Facility since its inception. She has publications in peer-reviewed conferences and journals in the areas of motif detection, parallel*

computing, and biological networks in systems biology. Her doctoral research involved using network theory and high-throughput biomedical data to identity structure-function relationships in a variety of graph theoretic data models. As a research associate, she has sought to expand her research from systems biology to complex systems, applying her knowledge of the biological world to the social world. In this time, she has collaborated with top companies in the Omaha metro-area to manage, analyze, and predict anomalies within masses of data, regardless of system type.

***Vladimir Ufimtsev** is a Ph.D. in IT student in the College of Information Science & Technology at the University of Nebraska at Omaha. He received his M.S. degree in Mathematics from Northeastern University (2009) and B.Sc. degree in Mathematics and Physics from University of Nebraska at Omaha (2006). His current research focus is on complex network analysis, high performance computing, efficiently identifying important vertices/edges in a network, group testing, the effects of network noise, and combinatorial algorithms. He has worked with the UNO Bioinformatics Research Group since 2011.*

***Sanjukta Bhowmick** is an Assistant Professor in the College of Information Science and Technology at the University of Nebraska at Omaha. She received her Ph.D. from the Pennsylvania State University. Her core research area is in high performance computing with a focus on the synergy of combinatorial and numerical methods. Her current projects focus on designing parallel, efficient and robust algorithms for analyzing large-scale dynamic networks. In particular, she is interested in evaluating how factors such as the discrepancy between the network model and the application and permutations in the input affect network analysis results and developing algorithms to minimize the influence of this noise.*

***Hesham H. Ali** is a Professor of Computer Science and the Lee and Wilma Seaman Distinguished Dean of the College of Information Science and Technology (IS&T), at the University of Nebraska at Omaha (UNO). He currently serves as the director of the UNO Bioinformatics Core Facility that supports a large number of biomedical research projects in Nebraska. He has published numerous articles in various IT areas including scheduling, distributed systems, data analytics, wireless networks, and Bioinformatics. He has also published two books in scheduling and graph algorithms, and several book chapters in Bioinformatics. He is currently serving as the PI or Co-PI of several projects funded by NSF, NIH and Nebraska Research Initiative (NRI) in the areas of data analytics, wireless networks and Bioinformatics. He has been leading a Research Group at UNO that focuses on developing innovative computational approaches to classify biological organisms and analyze big bioinformatics data. The research group is currently developing several next generation data analysis tools for mining various types of large-scale biological data. This includes the development of new graph theoretic models for assembling short reads obtained from high throughput instruments, as well as employing a novel correlation networks approach for analyzing large heterogeneous biological data associated with various biomedical research areas, particularly projects associated with aging and infectious diseases. He has also been leading two funded projects for developing secure and energy-aware wireless infrastructure to address tracking and monitoring problems in medical environments, particularly to study mobility profiling for healthcare research.*