

2011

## A Parallel Graph Sampling Algorithm for Analyzing Gene Correlation Networks


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

Kanimathi Duraisamy  
*University of Nebraska at Omaha*

Hesham Ali  
*University of Nebraska at Omaha, hali@unomaha.edu*

Sanjukta Bhowmick  
*University of Nebraska at Omaha, sbhowmick@unomaha.edu*

Follow this and additional works at: <https://digitalcommons.unomaha.edu/interdiscipinformaticsfacpub>

 Part of the [Bioinformatics Commons](#), [Genetics and Genomics Commons](#), and the [Numerical Analysis and Scientific Computing Commons](#)

Please take our feedback survey at: [https://unomaha.az1.qualtrics.com/jfe/form/SV\\_8cchtFmpDyGfBLE](https://unomaha.az1.qualtrics.com/jfe/form/SV_8cchtFmpDyGfBLE)

### Recommended Citation

Cooper, Kathryn Dempsey; Duraisamy, Kanimathi; Ali, Hesham; and Bhowmick, Sanjukta, "A Parallel Graph Sampling Algorithm for Analyzing Gene Correlation Networks" (2011). *Interdisciplinary Informatics Faculty Publications*. 25.

<https://digitalcommons.unomaha.edu/interdiscipinformaticsfacpub/25>

This Article 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 Publications by an authorized administrator of DigitalCommons@UNO. For more information, please contact [unodigitalcommons@unomaha.edu](mailto:unodigitalcommons@unomaha.edu).

International Conference on Computational Science, ICCS 2011

# A Parallel Graph Sampling Algorithm for Analyzing Gene Correlation Networks

Kathryn Dempsey, Kanimathi Duraisamy, Hesham Ali, Sanjukta Bhowmick\*

*Department of Computer Science, University of Nebraska at Omaha, Omaha, NE 68182*

---

## Abstract

Efficient analysis of complex networks is often a challenging task due to its large size and the noise inherent in the system. One popular method of overcoming this problem is through graph sampling, that is extracting a representative subgraph from the larger network. The accuracy of the sample is validated by comparing the combinatorial properties of the subgraph and the original network. However, there has been little study in comparing networks based on the applications that they represent. Furthermore, sampling methods are generally applied agnostically, without mapping to the requirements of the underlying analysis.

In this paper, we introduce a parallel graph sampling algorithm focusing on gene correlation networks. Densely connected subgraphs indicate important functional units of gene products. In our sampling algorithm, we emphasize maintaining highly connected regions of the network through parallel sampling based on extracting the maximal chordal subgraph of the network. We validate our methods by comparing both combinatorial properties and functional units of the subgraphs and larger networks. Our results show that even with significant reduction of the network (on average 20% to 40%), we obtain reliable samplings and many of the relevant combinatorial and functional properties are retained in the subgraphs.

*Keywords:* chordal graphs, parallel graph sampling, gene correlation networks.

---

## 1. Introduction

Many disciplines such as social sciences and biology [1, 2], rely on the network analysis for understanding and exploring their experimental data. These data, typically represented as networks, are often very large and can potentially include some noise. In order to obtain accurate information from these large-scale complex systems it is essential to (i) handle the large volume of data efficiently and (ii) filter out a significant portion of the noise. Graph sampling techniques can help achieve these goals.

Graph sampling involves extracting a representative subgraph that exhibits similar characteristics to the original, larger network. The reliability of sampling techniques is generally measured by how closely combinatorial properties, such as degree distribution and clustering coefficient, of the subgraph mimic the original network [3]. Since, the

---

\*Corresponding author

*Email addresses:* [k.dempsey85@gmail.com](mailto:k.dempsey85@gmail.com) (Kathryn Dempsey), [kduraisamy@unomaha.edu](mailto:kduraisamy@unomaha.edu) (Kanimathi Duraisamy), [hali@unomaha.edu](mailto:hali@unomaha.edu) (Hesham Ali), [sbhowmick@unomaha.edu](mailto:sbhowmick@unomaha.edu) (Sanjukta Bhowmick)

underlying purpose of network analysis is to extract meaningful data from an application, it is therefore important that in addition to comparing combinatorial properties, we also verify whether the subgraphs retain the application-based properties of the network. Ideally, graph sampling algorithms should be tuned to the requirements of the application and analysis metrics. However, to date, there has been little research on application-based graph sampling techniques.

In this paper, we introduce a parallel graph sampling algorithm for identifying functional units within gene correlation networks, which are used to identify functional cellular mechanisms *in silico*. Analysis of correlation networks is particularly useful in examining relationships within high-throughput data that were previously impossible to explore using laboratory-based methodology such as microarray analysis.

A correlation network is represented as a graph, where vertices represent genes and edges represent the correlation between the expression levels of two genes. A primary analytical operation of correlation networks is identifying high density clusters of genes, represented by tightly connected vertices in the network. In order to preserve these tightly connected vertices, we designed a graph sampling algorithm based on extracting the maximal chordal subgraph of the network. Structures like maximal chordal subgraphs retain the number of cliques from the original networks, which are representative of high density clusters. Our experiments on correlation networks obtained from brain tissue of different colonies of mice show that, in most cases, our sampling technique preserves both the important functional clusters as well as the key combinatorial properties in the subnetworks, while significantly reducing the size of the network. To the best of our knowledge, this is one of the first tightly coupled parallel implementation of graph sampling.

This paper is organized as follows; In Section 2 we provide brief overviews of correlation networks and graph sampling techniques. In Section 3 we describe our parallel graph sampling algorithm. In Section 4 we present our experimental results and analysis which include comparing both combinatorial and functional properties of the original and subnetworks. We conclude in Section 5 with a discussion of our future research plans.

## 2. Background and Previous Work

In this section we give a brief description of how gene correlation networks are generated and provide an overview of research on graph sampling.

### 2.1. Generating Gene correlation Networks

Gene correlation networks are created based on the correlation between expression levels of different genes as obtained from microarray data analysis. Different measurements of correlation have been used to build these networks, such as the partial correlation coefficient [4], the Spearman correlation coefficient [5] and more commonly, the Pearson correlation coefficient [6]. Analysis of correlation networks where all vertices (genes) are connected to each other is computationally very expensive. In such cases, thresholding is used for network reduction by removing edges with low correlations [7].

The degree distribution of correlation network (filtered to remove low level expression edges) follow a power-law distribution that indicates a scale-free network structure [2]. This indicates that there are many vertices in the network that are poorly connected and a few vertices that are very well connected (hubs). Hubs have been found in the yeast interactome to correspond to essential genes [8] and have been found to be critical for maintenance of structure in other biological networks. Biologically relevant network characteristics such as hubs or clustering coefficient can be found using combinatorial algorithms and more importantly, without the help of extra biological data. However, this is only plausible for smaller networks. Networks built from microarray data are too large to analyze in reasonable time creating the need for efficient sampling mechanisms. Furthermore, correlation networks can have noise or unnecessary edges, [9] which can adversely impact the accuracy of the results.

### 2.2. Combinatorial Terminology and Graph Sampling Techniques

We now define some terms used in graph theory. Unless mentioned otherwise, the terms used here are as they are defined in [10]. A graph  $G = (V, E)$  is defined as a set of vertices  $V$  and a set of edges  $E$ . An edge  $e \in E$  is associated with two vertices  $u, v$  which are called its *endpoints*. A vertex  $u$  is a *neighbor* of  $v$  if they are joined by an edge. The *degree* of a vertex  $u$  is the number of its neighbors. Vertices with high degrees are termed as *hubs*. A *path*, of length  $l$ , in a graph  $G$  is an alternating sequence of  $v_0, e_1, v_1, e_2, \dots, e_l, v_l$  vertices and edges, such that for  $j = 1, \dots, l$ ;  $v_{j-1}$  and

$v_j$  are the endpoints of edge  $e_j$ , with no edges or internal vertices repeated. A *cycle* is a path whose initial and final vertices are identical. A *clique* is a set of vertices which all are connected (neighbors to) each other.

We define high density subgraphs or modules as set of vertices in a graph which have a high percentage of edges between them. Two important combinatorial measures of graph modularity are clustering coefficient and core number. The *clustering coefficient* of a vertex is computed as the ratio of the edges between the neighbors of a vertex to the total possible connections between the neighbors. A large clustering coefficient indicates presence of dense modules. The *core number* of a vertex is defined as the largest integer  $c$  such that the vertex exists in a graph where all degrees are greater than equal to  $c$ . This metric relates to how closely high degree vertices are connected.

Graph sampling encompasses many different algorithms and applications ranging from generating spanning tree based support graph preconditioners for faster linear system solution [11] to network reduction for improved visualization [12]. Most sampling methods for large scale-free networks are based on random sampling, such random node selection [13] or random walks on the network [14]. A comprehensive comparison of different randomized sampling algorithms can be found in [3]. Most sampling algorithms assume that the network is connected, which in the case of correlation networks, may not always be true. Parallel random walk implementations generally involve starting simultaneous multiple walks [15]. To the best of our knowledge, our algorithm is one of the few tightly coupled parallel implementations of graph sampling.

Our sampling technique is based on a polynomial time algorithm for extracting the maximal chordal subgraphs introduced in [16]. A chordal graph is a graph where there are no cycles greater than length three. The algorithm builds the maximal chordal subgraph using a modified version of the maximum cardinality search method. Initially the chordal subgraph consists of the starting vertex and its associated edges. In the subsequent steps, the vertex with the maximum number of visited neighbors (which are in the chordal subgraph) is selected. An edge from the current selected vertex  $a$ ,  $(a, b)$  is added to the chordal subgraph if the number of visited neighbors of  $b$  is a subset of the number of visited neighbors of  $a$ . The complexity of this algorithmic is  $O(Ed)$ , where  $E$  is the number of edges in the graph and  $d$  is the maximum degree. A detailed description of this method can be found in [16].

### 3. Parallel Algorithm for Sampling by Extracting Quasi-Chordal Subgraphs of Networks

#### Pseudocode For Extracting Quasi-Chordal Subgraphs

**Input:** The original network  $G$

**Output:** Reduced quasi-chordal network  $\tilde{G}$

1. Partition the network across  $P$  processors
2. For subgraphs in each processor  $P_i$ 
  - (a) Find maximal chordal subgraph in processor  $P_i$
  - (b) Mark the border edges
  - (c) Send border edges to receiver processor,  $P_j$
3. After receiving border vertices from sender processor,  $P_k$ 
  - (a) For any two border edges  $(n_a, n_b)$  and  $(n_a, n_c)$ , if  $(n_b, n_c)$  is a chordal edge then, include  $(n_a, n_b)$  and  $(n_a, n_c)$  in the subgraph
4. End

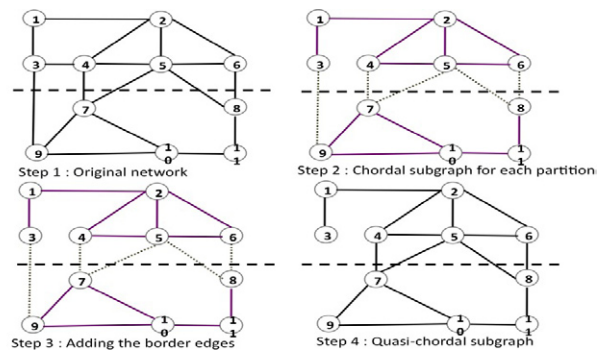


Figure 1: Left Figure: Pseudocode for extracting quasi-chordal subgraphs. Right Figure: Steps in applying the parallel graph sampling algorithm to a small graph. Step 1 shows the original network partitioned into two processors. Step 2 shows how the maximal chordal subgraph is obtained in each individual partition. The solid lines show the chordal edges and the dotted lines the border edges. In Step 3, the border edges that form a triangle with one chordal edge are selected. In this example, they are the pair (4,7) and (5, 7) and the pair (5,8) and (6,8). Step 4 shows the final quasi-chordal subgraph. There is now a cycle of length 5, (5,7,10,11,8), in the subgraph.

The objective of our sampling algorithm is to maintain the highly connected subgraphs from the original network, while removing some of the associated noise. We assume that the effect of noise is more likely to be prevalent in structures formed by loosely connected vertices, since even the addition of a small number of edges can significantly affect their clustering coefficient. Based on this assumption, we have developed a parallel algorithm for extracting the maximal chordal subgraphs from the network. Since, chordal subgraphs retain properties like the number of cliques

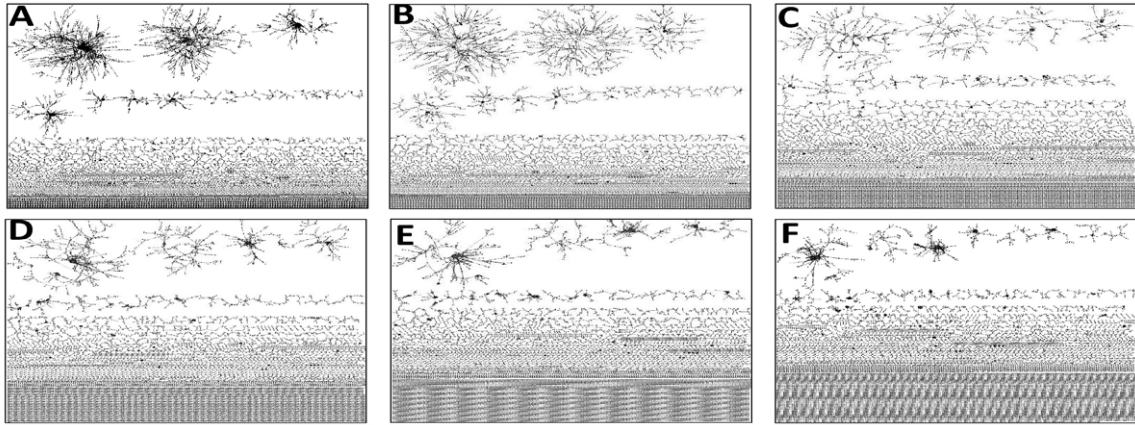


Figure 2: Visualization of Networks from the Creatine Treated Mice. A: Original Network. B: QCS with 1 Partition. C: QCS with 2 Partitions. D: Left Figure: QCS with 4 Partitions. E: QCS with 8 Partitions. F: QCS with 16 Partitions.

of the original network while increasing the distance between vertices, we anticipate that this method of sampling would be effective in preserving the high density clusters and eliminating the low density ones.

Our initial implementation was based on the sequential algorithm described in [16]. We adapted the original algorithm to be applied to networks with disconnected components, like the correlation networks. We, however, discovered that a completely chordal subgraph is a very strict restriction, and such sampling may disintegrate some of the clusters, which are almost, but not exact, cliques. To counteract this effect we modified the algorithm to extract quasi-chordal subgraphs, which include some cycles of length greater than three as follows;

We divide the network into  $P$  partitions. Within each partition, we obtain a maximal chordal subgraph formed only of the edges, called *chordal edges*, whose endpoints lie completely within the partition. The edges that lie across the partitions, called *border edges*, are included in the subgraph only if two border edges, with a common vertex, combine with a perviously marked chordal edge to form a triangle. Since the other neighbors of the border edges are not further checked, part of the network falling across the partitions may contain cycles larger than three. Figure 1 provides the pseudocode and demonstrates how our parallel algorithm obtains quasi-chordal subgraphs (QCS).

This algorithm can be effectively implemented in parallel, by substituting the number of partitions by the number of processors. Also note that the condition for inclusion of border edges requires comparison only across two processors. This pair-wise communication lowers the complexity and improves the scalability of the algorithm. For each possible pair of processors we designated one processor as *sender* and the other as *receiver*. The sender processor  $P_i$  sends the border edges to the receiver processor  $P_j$ . The receiver processor then checks whether the vertices form a triangle with a chordal edge. The pairs of senders and receivers are assigned such that every processor sends and receives border edges from an almost equal number of processors. Assuming nearly equal distribution of edges across processors, this ensures that the computation load is balanced across processors.

The scalability of our parallel algorithm can be computed as follows; Let the number of edges in the network be  $E$ , the maximum degree of the network be  $d$  and the number of processors involved be  $p$ . The complexity of the sequential algorithm is given by  $T_{seq}(E) = O(Ed)$ . The parallel overhead  $T_{over}(E, p)$  consists of communicating the border nodes from each processor, denoted by  $b$ , and subsequently checking, for each pair of border nodes, if they form a triangle with a chordal edge. Assuming equal distribution of border nodes, the total communication volume is  $O(bp)$  and the computation volume over all processors is  $O(b^2p)$ . Therefore,  $T_{over}(E, p) = O(b^2p)$ . In order maintain isoeficiency,  $T_{seq}(E) \geq T_{over}(E, p)$ , which implies  $E \geq Cb^2p/d$ , where  $C$  is a constant.

The memory required by to store the network is approximately  $O(E)$ . Therefore, the scalability function for this algorithm can be computed as  $\frac{Cb^2p}{dp} = O(b^2/d)$ . Thus, the parallel overhead increases with the number of border edges which in turn increases propotional to the number of processors. We see in Section 4 that more partitions lead to better sampling, so there exists a trade-off between algorithmic efficiency and quality of solution. Since the quasi-chordal subgraph (QCS) includes only border edges that are part of at least one triangle, we believe that more tightly



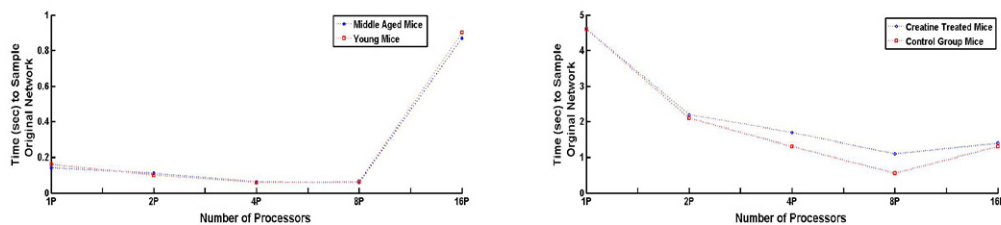


Figure 3: Time for sampling subgraphs in parallel. Left: Time for GSE5078. Right: Time for GSE5140.

coupled structures (not necessarily cliques) are included while loosely coupled structures (which can include noise) are potentially eliminated. Figure 2 shows the QCS generated from one of our sample networks.

#### 4. Experimental Results

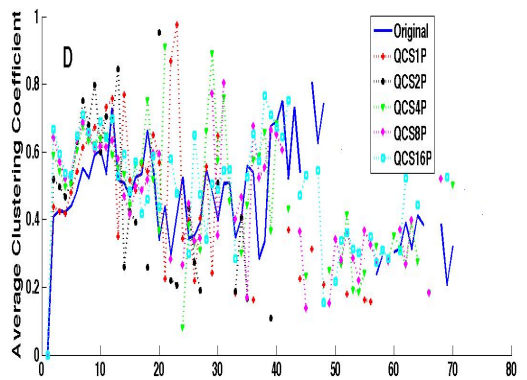
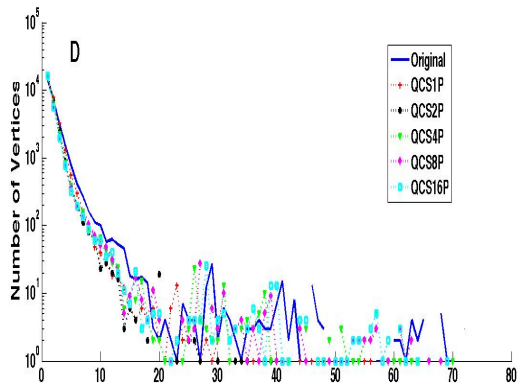
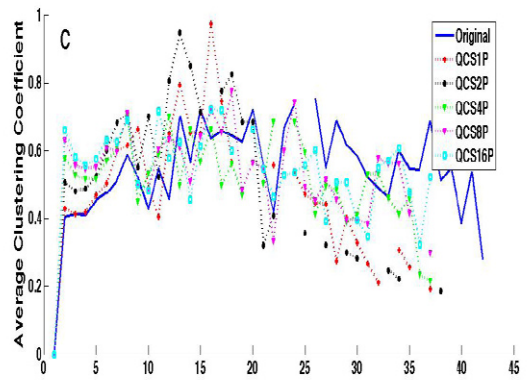
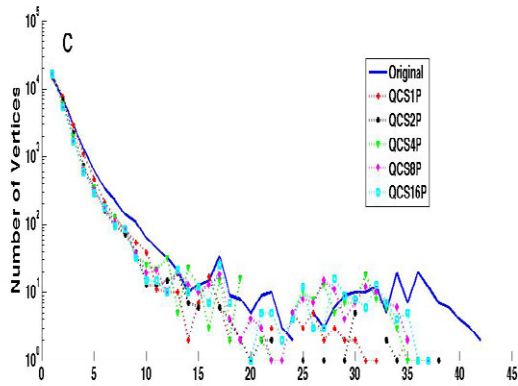
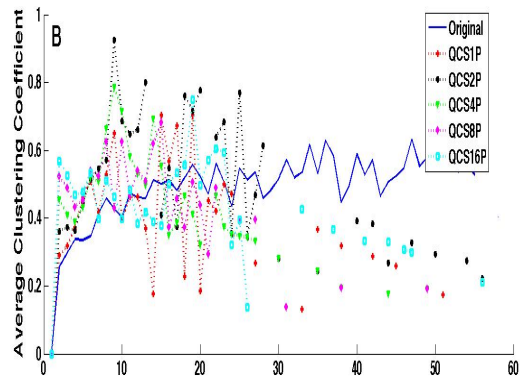
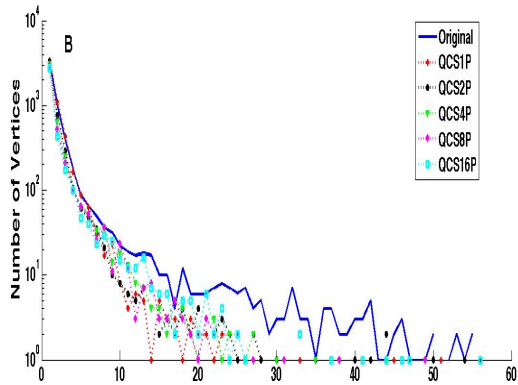
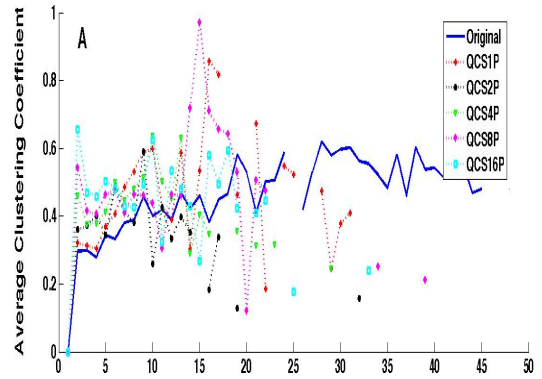
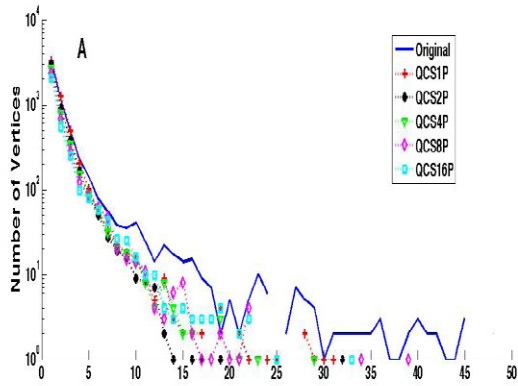
We evaluated the effectiveness of our sampling algorithm by comparing the combinatorial and functional properties of the original and subnetworks. The data sets for our experiments were obtained from NCBI's Gene Expression Omnibus (GEO) website (<http://www.ncbi.nlm.nih.gov/geo/>) [17]. The first dataset is the GSE5078 series generated by Verbitsky et al. [18], and is derived from hippocampal samples of young and middle-aged mice (2 months and 15 months, respectively). The second dataset is the GSE5140 series generated by Bender et al. [19], and is derived from the brain tissue of middle aged mice, treated with oral supplementation of creatine and an untreated control group. Networks were created by pairwise computation of the Pearson correlation coefficient for each possible pairing of the genes and thresholding was applied to eliminate low correlation edges.

We obtained quasi-chordal subgraphs for these four networks, by the process described in Section 3 on 1,2,4,8 and 16 processors on a distributed memory system using MPI. The codes were executed on the University of Nebraska at Omaha's Blackforest linux computing cluster, consisting of subclusters of Intel Pentium D, Dual-core Opteron and 2 Quad-core AMD Opteron processors. As shown in Figure 3, our algorithm is highly scalable up to 8 processors. The execution time on 16 processors increases due to the presence of more border edges.

##### 4.1. Analysis of Combinatorial Properties

Our first set of analysis involved comparing the combinatorial properties of the networks and the subgraphs. Since the primary purpose of gene correlation networks is to identify clusters of similar genes, we emphasize the retention of properties related to cluster membership such as highly connected vertices or hubs, average clustering coefficients and core numbers. We compared the following properties between the networks using the MatlabBGL library ([http://www.stanford.edu/dgleich/programs/matlab\\_bgl/](http://www.stanford.edu/dgleich/programs/matlab_bgl/)).

1. *Number of Edges*: The subgraphs will have a lower number of edges. The percentage reduction is computed by  $1 - (\text{Edges in Subgraph} / \text{Edges in Original Network})$ . The higher the number the more the reduction.
2. *Degree Distribution*: Degree distribution is computed as number of vertices with degree  $d$ . Degree distributions in the correlation networks follow the power law. The subgraphs should maintain a similar pattern.
3. *Clustering Coefficient*: We compare the average clustering coefficients per degree between the networks. The average clustering coefficient of the subgraphs should be close to the original network.
4. *High Degree Vertices*: We compare how many of the same vertices appear as hubs (top 1% of the highest connected vertices) in both the original and the sub-networks. The percentage of common hubs is computed as  $\text{Common Vertices} / \text{Total Vertices in the Original Network}$ . The higher the number, the better the sampling.
5. *Core Number Distribution*: We selected the vertices in the top 5 core number groups and identified the vertices that are grouped together both in the original network and the subgraphs. The percentage for core numbers is computed as the ratio of the number of vertices grouped together both in the subgraph and the original network by the total number of vertices in the top 5 core number group of the original network. The higher the value, the better the clustering property should be maintained.



Combinatorial Properties	Original Network	Quasi-Chordal Subgraphs with				
		1 Partition	2 Partitions	4 Partitions	8 Partitions	16 Partitions
Middle-Aged Mice (GSE5078) (Vertices: 5,549)						
Number of Edges	7,178	5,206 (27.4)	4,127 (42.5)	3,878 (45.9)	3,637 (49.3)	<b>3,362 (53.1)</b>
Mean Clust. Coeff.	.46	.45	.31	.38	.44	.41
High Degree Vertices	144	83 (57)	60 (41)	77 (53)	80 (55)	<b>92 (63)</b>
Core Numbers	50	30 (60)	26 (52)	26 (52)	28 (56)	<b>44 (88)</b>
Young Mice (GSE5078) (Vertices: 5,349)						
Number of Edges	7,277	4,949 (31.9)	4,269 (41.3)	4,029 (44.6)	<b>3,657 (49.7)</b>	3,753 (48.4)
Mean Clust. Coeff.	.48	.39	.47	.40	.41	.41
High Degree Vertices	146	73 (50)	<b>106 (72)</b>	96 (65)	86 (58)	95 (65)
Core Numbers	46	26 (56)	25 (54)	<b>39 (84)</b>	36 (78)	26 (56)
Control Group (GSE5140) (Vertices: 27,320)						
Number of Edges	29,719	25,281 (14.9)	22,284 (25)	22,986 (22.6)	22,272 (25)	<b>21,898 (26.3)</b>
Mean Clust. Coeff.	.54	.47	.50	.49	.51	.52
High Degree Vertices	595	368 (61)	335 (56)	<b>451 (75)</b>	430 (72)	431 (72)
Core Numbers	200	34(17)	37 (18)	106 (53)	112 (56)	<b>106 (53)</b>
Creatine Treated Mice (GSE5140) (Vertices: 28,161)						
Number of Edges	33,099	27,278 (17.5)	23,867 (27.8)	25,268 (23.6)	24,719 (25.3)	<b>24,641 (25.5)</b>
Mean Clust. Coeff.	.46	.45	.43	.45	.44	.48
High Degree Vertices	662	387 (58)	360 (54)	478 (72)	494 (74)	<b>502 (76)</b>
Core Numbers	187	58 (31)	45 (24)	101 (54)	98 (52)	<b>117 (62)</b>

Table 1: Comparison of combinatorial properties between the original networks and subgraphs. The numbers in the parenthesis denote the reduction percentages. The best values of edge reduction, and hub and core number retention, for each network, have been marked in bold.

Figure 4 plots the degree distribution and the distribution of the average clustering coefficient per degree of the four networks and their subgraphs. As can be seen from the figures, for both metrics, barring slight fluctuations, the subgraphs follow the same pattern as the original network.

Table 1 compares the other combinatorial properties including reduction in edges, mean clustering coefficient over the entire network, hubs and core numbers. The reduction patterns are similar within the same group, i.e. the GSE5078 or the GSE5140 networks, but changes across groups. For the smaller networks in GSE5078, the sampling technique achieves high reductions from 27% to as much as 53%, while still maintaining nearly 50% or more of the hubs and the core number grouping. For the larger networks in GSE5140, the reduction is around 14% to 26%. The percentage of hubs retained is 50% to as much as 75%. The core number grouping is higher (above 50%) for subgraphs having larger partitions (4-16). The mean clustering coefficients of all the subgraphs are very close to the corresponding original network. In general, the results are better when there are more partitions. We conjecture that this is because by increasing the number of partitions we include more almost-clique structures (by keeping the triangles at border edges), as well as filter out noise (by removing border edges that do not form any triangle).

#### 4.2. Analysis of Biological Properties

We now compare the functional units in the correlation networks. We used the Cytoscape plug-in MCODE (<http://baderlab.org/Software>) [20] on each network to identify clusters, which can potentially overlap. We extracted the top five clusters of the original network and subgraphs and compared the clusters based on maximum common genes in each set. The names of genes in each cluster were given to the PANTHER Classification System (<http://www.pantherdb.org/>) [21] to identify common molecular functions. The results contain only the most represented Gene Ontology (GO) molecular function terms per cluster. A large number of the common vertices within each cluster may not yet have a known molecular function. Therefore, to ascertain biological relevance, it is critical to



Cluster ID	Original Network	QCS on 1 Partition	QCS on 2 Partitions	QCS on 4 Partitions	QCS on 8 Partitions	QCS on 16 Partitions
1	protein binding	protein binding		protein binding	protein binding	protein binding
	nucleic acid binding					nucleic acid binding
	binding	binding		binding	binding	binding
		receptor activity		receptor activity		
			transcription regulator activity transcription factor activity			
2	catalytic activity	catalytic activity	catalytic activity	catalytic activity		
	protein binding	protein binding		protein binding	protein binding	protein binding
	binding	binding	binding	binding		
					transmembrane transport. activity	transmembrane transport. activity
			receptor activity	receptor binding		
3	transferase activity		transferase activity	transferase activity		
	transporter activity		transporter activity	transporter activity		
	catalytic activity		catalytic activity	catalytic activity		
		nucleic acid binding transmembrane transport. activity binding			nucleic acid binding transmembrane transport. activity binding	
4	structural molecule activity	structural molecule activity	structural molecule activity	structural molecule activity	structural molecule activity	structural molecule activity
		hydrolase activity	hydrolase activity	hydrolase activity	hydrolase activity	hydrolase activity
		catalytic activity	catalytic activity	catalytic activity	catalytic activity	catalytic activity
	receptor activity carbohydrate transmem transport					

Cluster Id	Original Network	QCS on 1 Partition	QCS on 2 Partitions	QCS on 4 Partitions	QCS on 8 Partitions	QCS on 16 Partitions
1	protein binding	protein binding	protein binding	protein binding	protein binding	protein binding
	catalytic activity	catalytic activity	catalytic activity			catalytic activity
	binding	binding	binding	binding	binding	binding
				enzyme regulator activity	enzyme regulator activity	
2	receptor binding	receptor binding	receptor binding	receptor binding		
	protein binding	protein binding	protein binding	protein binding	protein binding	protein binding
	binding	binding	binding	binding	binding	binding
					transmembrane transport. activity	transmembrane transport. activity

Figure 5: Comparison of functional units of the networks of GSE5078. The leftmost column of functionalities is from the original network, proceeding to the functionalities obtained for QCS with 1,2,4,8 and 16 partitions respectively. Similar color within each cluster represents similar functionality.

determine if these clusters have common function(s). If this is the case, then unknown vertices in the cluster become targets for future experimentation. The top three functions per cluster are reported in Figure 5 and 6.

We define clusters to be overlapped if the same vertex (gene) is classified as being in more than one cluster. There were no overlapping clusters in the GSE5140 series. There was one small overlap of two clusters in the middle-aged mice network. However, the young mice network exhibited significant overlap of more than two clusters, which affected the comparison of the functional units. We conjecture that was because the gene pathways of the young mice are in a more fluid and volatile state than the more fixed gene pathways of older mice. One of our future research plans is to investigate further into the biological relevance of this phenomena.

Except for the young mice network (due to high overlap), most clusters from the original network are present in the subgraphs. The results show that sampling from the larger networks from the GSE5140 match to more functional units than the smaller networks, and the matching improves with larger number of partitions. We also note that several subgraphs show common functional units (such as in cluster 3 of the middle aged mice and in cluster 1 of the creatine treated mice) which are not present in the original cluster. We conjecture that the removal of noisy edges has exposed these functional units previously hidden in the original network.

Functional Units For Control Group Mice Network						
Cluster ID	Original Network	QCS with 1 Partition	QCS with 2 Partitions	QCS with 4 Partitions	QCS with 8 Partitions	QCS with 16 Partitions
1	nucleic acid binding	nucleic acid binding		nucleic acid binding	nucleic acid binding	nucleic acid binding
		catalytic activity	catalytic activity	catalytic activity	catalytic activity	catalytic activity
	binding	binding	binding	binding	binding	binding
	protein binding		hydrolase activity			
2	catalytic activity	catalytic activity	catalytic activity	catalytic activity	catalytic activity	catalytic activity
		protein binding	protein binding			
	binding	binding	binding	binding	binding	binding
3	transferase activity			transferase activity	transferase activity	transferase activity
	catalytic activity	catalytic activity		catalytic activity		
	protein binding	protein binding	protein binding	binding	protein binding	protein binding
4	binding	binding	binding		binding	binding
			receptor activity		receptor activity	receptor activity
				nucleic acid binding		
5	protein binding	protein binding	protein binding		protein binding	protein binding
	structural molecule activity	structural molecule activity	structural molecule activity		structural molecule activity	structural molecule activity
	binding	binding	binding		binding	binding
5	transcription regulator activity		transcription regulator activity			
	binding			binding		
	nucleic acid binding		transcription factor activity	oxidoreductase activity		
			DNA binding	catalytic activity		

Functional Units For Creatine Treated Mice Network						
Cluster ID	Original Network	QCS with 1 Partition	QCS with 2 Partitions	QCS with 4 Partitions	QCS with 8 Partitions	QCS with 16 Partitions
1	receptor activity	receptor activity		receptor activity		
	protein binding	protein binding	protein binding	protein binding	protein binding	protein binding
	binding	binding	binding	binding	binding	binding
2			enzyme regulator activity		ligand-gated ion chan act	ligand-gated ion chan act
	nucleic acid binding			nucleic acid binding	nucleic acid binding	nucleic acid binding
	binding	binding	binding	binding	binding	binding
3	catalytic activity	catalytic activity	catalytic activity	catalytic activity	catalytic activity	catalytic activity
		peptidase activity	protein binding			
	protein binding					protein binding
4	acyltransferase activity					acyltransferase activity
	binding					binding
	calmodulin binding	calmodulin binding	calmodulin binding	calmodulin binding	calmodulin binding	calmodulin binding
5	protein binding	protein binding	protein binding	protein binding	protein binding	protein binding
	binding	binding	binding	binding	binding	binding
	transferase activity			transferase activity	transferase activity	transferase activity
5	binding	binding	binding	binding	binding	binding
	catalytic activity	catalytic activity		catalytic activity	catalytic activity	catalytic activity
		lyase activity	protein binding			
5			enzyme regulator activity			

Figure 6: Comparison of functional units of the networks of GSE5140. The leftmost column of functionalities is from the original network, proceeding to the functionalities obtained for QCS with 1,2,4,8 and 16 partitions respectively. Similar color within each cluster represents similar functionality.

## 5. Discussion and Future Research

In this paper we introduced a scalable parallel graph sampling algorithm based on extracting the maximal chordal subgraphs from large networks. We showed that our sampled subgraphs retain most of the combinatorial and functional properties of the original network. In the case of networks with significant overlap of clusters, fewer functional units were retained in the subgraphs, despite good mapping of the combinatorial characteristics. This observation highlights that focusing only on combinatorial metrics might in some cases be misleading. Our future research plans, therefore, include investigating sampling techniques that can accurately represent graphs with multiple overlapping clusters. Also we currently can not identify any strong mapping between the combinatorial and functional properties, which would allow us to predict preservice of function units based on graph properties and therefore plan research on other combinatorial metrics that provide greater insight to the gene cluster functionality. We also observe that sampling sometimes reveals new clusters in the reduced networks. We believe that this occurs due to the elimination of noise in the graphs. In our next projects we plan to look into noise reducing sampling techniques for large networks.

## Acknowledgment

The authors would like to thank the Nebraska EPSCoR First Award, the Nebraska INBRE Program and the College of IS&T, University of Nebraska at Omaha for supporting this research.

## References

- [1] K. Voevodski, S. H. Teng, Y. Xia, Finding local communities in protein networks, *BMC Bioinformatics* 10 (10) (2009) 297.
- [2] A. L. Barabasi, Z. N. Oltvai, Network biology: Understanding the cell's functional organization, *Nature Reviews.Genetics* 5 (2) (2004) 101–113.
- [3] J. Leskovec, C. Faloutsos, Sampling from large graphs, in: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '06, 2006*, pp. 631–636.
- [4] N. S. Watson-Haigh, H. N. Kadarmideen, A. Reverter, Pcit: An r package for weighted gene co-expression networks based on partial correlation and information theory approaches, *Bioinformatics(Oxford, England)* 26 (3) (2010) 411–413.
- [5] W. J. Ewens, G. R. Grant, *Statistical methods in bioinformatics (Second Edition ed.)*, New York, NY: Springer, 2005.
- [6] M. Mutwil, U. B., S. M., A. Loraine, O. Ebenhoh, S. Persson, Assembly of an interactive correlation network for the arabidopsis genome using a novel heuristic clustering algorithm, *Plant Physiology* 152 (1) (2010) 29–43.
- [7] S. L. Carter, C. M. Brechbuhler, M. Griffin, A. T. Bond, Gene co-expression network topology provides a framework for molecular characterization of cellular state, *Bioinformatics (Oxford, England)* 20 (14) (2004) 2242–2250.
- [8] H. Jeong, S. P. Mason, A. L. Barabasi, Z. N. Oltvai, Lethality and centrality in protein networks, *Nature* 411 (6833) (2001) 41–42.
- [9] 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.
- [10] J. L. Gross, J. Yellen, *Handbook of Graph Theory and Applications*, CRC Press, 2004.
- [11] B. Marshall, J. R. Gilbert, B. Hendrickson, N. Nguyen, S. Toledo, Support-graph preconditioners, *SIAM Journal on Matrix Analysis and Applications* 27 (4) (2006) 930–951.
- [12] D. Rafiei, S. Curial, Effectively visualizing large networks through sampling, *Visualization Conference, IEEE O* (2005) 48.
- [13] V. Krishnamurthy, M. Faloutsos, J.-H. Chrobak, M. and Cui, L. Lao, A. G. Percus, Sampling large internet topologies for simulation purposes, *Computer Networks* 51 (2007) 4284–4302.
- [14] J. Leskovec, J. Kleinberg, C. Faloutsos, Graphs over time: densification laws, shrinking diameters and possible explanations, in: *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, 2005*, pp. 177–187.
- [15] A. Rasti, M. Torkjazi, R. Rejaie, N. Duffield, W. Willinger, D. Stutzbach, Respondent-driven sampling for characterizing unstructured overlays, 2009, pp. 2701–2705.
- [16] P. M. Dearing, D. R. Shier, D. D. Warner, Maximal chordal subgraphs, *Discrete Applied Mathematics* 20 (3) (1988) 181 – 190.
- [17] T. Barrett, D. B. Troup, S. E. Wilhite, P. Ledoux, D. Rudnev, C. Evangelista, I. F. Kim, A. Soboleva, M. Tomashevsky, K. A. Marshall, K. H. Phillippy, P. M. Sherman, R. N. Muerter, R. Edgar, Ncbi geo: archive for high-throughput functional genomic data, *Nucleic Acids Res.(Database issue)* Jan;37.
- [18] 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 and Memory (Cold Spring Harbor, N.Y.)* 11 (3) (2004) 253–260.
- [19] A. Bender, J. Beckers, I. Schneider, S. Hlter, T. Haack, T. Ruthsatz, D. Vogt-Weisenhorn, L. Beckerand, J. Genius, D. Rujescu, M. Irmeler, T. Mijalski, M. Mader, L. Quintanilla-Martinez, H. Fuchs, V. Gailus-Durner, M. H. de Angelis, W. Wurst, J. Schmidt, T. Klopstock, Creatine improves health and survival of mice., *Neurobiol Aging* Sep;29 (9) (2008) 1404–1411.
- [20] G. D. Bader, C. W. Hogue, An automated method for finding molecular complexes in large protein interaction networks, *BMC Bioinformatics* 4 (2).
- [21] P. Thomas, M. J. Campbell, A. Kejariwal, M. Huaiyu, B. Karlak, R. Daverman, K. Diemer, A. Muruganujan, N. A., Panther: a library of protein families and subfamilies indexed by function, *Genome Res.* 13 (2003) 2129–2141.