Finding Molecular Complexes through Multiple Layer Clustering of Protein Interaction Networks

Bill Andreopoulos

Aijun An

Xiangji Huang

Xiaogang Wang

Technical Report CS-2005-13

October 2005

Department of Computer Science and Engineering
4700 Keele Street North York, Ontario M3J 1P3 Canada

Finding Molecular Complexes through Multiple Layer Clustering of Protein Interaction Networks

Bill Andreopoulos

Department of Computer Science and Engineering, York University

Aijun An

Department of Computer Science and Engineering, York University

Xiangji Huang

Department of Information Technology, York University

Xiaogang Wang

Department of Mathematics and Statistics, York University

Motivation: One of the purposes of studying and analyzing protein-protein interaction networks (PINs) is to identify new protein complexes that guide the workings of a cell. Clustering algorithms for PIN data presented in the literature often do not consider the layered structure of protein complexes, creating instead a flat clustering.

Results: We propose the MULIC clustering algorithm that produces layered clusters of PIN data. We applied MULIC clustering to five PINs, including three yeast PINs. MULIC clusters correlate with known protein complexes in the MIPS database. For example, a large cluster of 79 proteins significantly overlaps with a known complex of 88 proteins.

Conclusions: MULIC clustering can assist in predicting protein complexes. Given the layered structure of the MULIC clusters, the proteins in top layers tend to be more representative of protein complexes than proteins in bottom layers. Lab experiments on finding an unknown complex or determining the potential effects of a drug can initially be guided by proteins in top layers and later move to bottom layers of clusters.

Supplementary Information: http://www.cs.yorku.ca/~billa/MULICppi05/

Keywords: Clustering, multiple layer, protein interaction network, complex.

1 Introduction

The amount of PIN data in databases has increased exponentially in recent years. Knowledge of the protein complexes in PINs has also increased, but at a slower rate. Often, but not always, proteins in a specific complex have more interactions with one another than they do with proteins from other complexes. This often allows clustering tools to predict protein complexes by identifying the dense areas in a PIN. One of the challenges in analyzing PIN data is to develop efficient clustering tools that can fairly accurately identify previously unknown protein complexes.

The main contribution of our work is to propose a novel method for finding protein complexes in PIN data using the MULIC clustering algorithm. The main strength of this algorithm is that each cluster consists of layers. Proteins in the top layer of a cluster have very similar sets of interactions to other proteins, while proteins in lower layers have less similar sets of interactions. A new cluster is created only when a set of proteins with very similar interactions is found. We applied this algorithm to three yeast *S. cerevisiae* PINs, one fruitfly *D. melanogaster* PINs and one worm *C. elegans* PIN. We filtered the clusters by cluster size. We compared the filtered clusters with known protein complexes in the MIPS database.

This paper is organized as follows. Section 2 describes previous related work. Section 3 describes the data sets and evaluation measures used. Section 4 describes the MULIC clustering algorithm. Section 5 presents and discusses the experimental results. Section 6 discusses the results in detail comparing them to those of other algorithms and discusses the advantages of this approach. Finally, Section 7 concludes the paper.

2 Related Work

Several clustering algorithms applied to PINs have been proposed so far, often based on graph theoretic techniques. These algorithms often do not consider the layered structure of protein complexes, creating instead a flat clustering. Moreover, the focus of these algorithms is often on finding the most densely connected or largest hubs of a PIN and not on the similarities between the proteins' sets of interactions with all other proteins.

An application of the identification of k-cores algorithm was proposed by (*Bader & Hogue, 2003*). K-cores in graph theory were introduced by (*Batagelj et al., 2001*). Given a graph $G = \{V,E\}$ with vertices set V (proteins) and edges set E (interactions), the k-core is computed by pruning all the vertices and their respective edges with degree (number of edges) less than k. That means that if a vertex u has degree d_u and it has n neighbors with degree less than k, then u's degree becomes $d_u - n$ and it will be also pruned if $k > d_u - n$. Figure 1 shows simple examples of protein complexes: a 4-core that can be found by both MULIC and k-cores with k=4; and two 3-cores that can be found by MULIC but not k-cores with k=4. K-cores with k=4 can not find the 3-core complexes, since some proteins have 3 edges only. MULIC can find all of these complexes, since most proteins have similar edge sets.

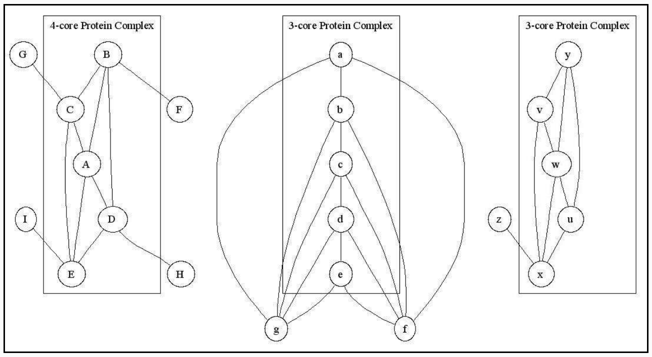


Figure 1: A 4-core protein complex and two 3-core protein complexes.

The Restricted Neighbourhood Search Clustering algorithm (RNSC) (*King et al.,2004*). is a cost-based local search algorithm based loosely on the tabu search metaheuristic (*Glover, 1989*). A clustering of a network *G = {V,E}* is equivalent to a partitioning of the node set V. The RNSC efficiently searches the space of partitions of V, each of which is assigned a cost, for a clustering with low cost. RNSC searches for a low-cost clustering by first composing an initial random clustering, then iteratively moving one node from one cluster to another in a randomized fashion to improve the clustering's cost. The algorithm searches using a simple integer-valued cost function as a preprocessor before it searches using a more expressive (but less efficient) real-valued cost function.

(*Ding et al., 2004*) present a representation of PINs based on an underlying bipartite graph model that allows generating the protein complex - protein complex association network. This representation allows viewing the PIN as consisting of protein complexes that share components.

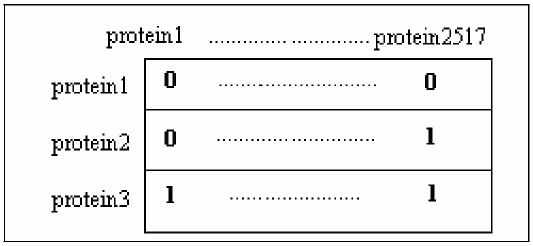
(*Dunn et al., 2005*) describe separating PIN graphs into subgraphs (protein clusters) of interconnected proteins, using the JUNG implementation of Girvan and Newman's Edge-Betweenness algorithm. Functions are sought for the subgraphs by detecting significant correlations with the distribution of Gene Ontology functional annotations which had been used to annotate the proteins within each cluster. The method was implemented using freely available software (JUNG and the R statistical package). (*Yang & Lonardi, 2005*) propose a parallel implementation of Girvan and Newman's clustering algorithm that runs on clusters of computers. This parallel implementation achieves almost linear speed-up and allows running this computationally intensive algorithm on large PINs.

3 Data Sets and Evaluation Measures

We used three yeast S. cerevisiae PINs originating from (*von Mering et al., 2002*) containing 2455 interactions (988 proteins), 11855 interactions (2617 proteins) and 78390 interactions (5323 proteins). We refer to these networks as Y2K, Y11K and Y78K respectively. Y2K contains high confidence interactions only, Y11K contains high and medium confidence interactions and Y78K contains high, medium and low confidence interactions. We used two more PINs of organisms for which little knowledge of protein complexes exists, making the evaluation of the results difficult. We used one fruitfly D. melanogaster PIN containing the set of 4637 interactions (4603 proteins) that have confidence greater than 0.5, as given in (Giot et al., 2003). We refer to this network as F4K. Finally, we used one worm C. elegans PIN containing 5222 interactions (3659 proteins) (Li et al., 2004, King et al., 2004). We refer to this network as W5K. We first clustered these networks using the MULIC algorithm. Then we filtered the results based on cluster size, to preserve only the clusters that are large enough and more likely to represent true biological complexes.

3.1 Representation of PIN Data Sets

PIN information on an organism is categorical, meaning that the objects (proteins) have attribute values that are taken from a set of discrete values and the values have no specified ordering. We represent PIN information as a categorical data set by creating a symmetric square *N*×*N* matrix, where *N* is the number of proteins of an organism. Figure 2 shows the representation of a PIN data set. The categorical attribute value (CA) in cell *(i,j)* of the matrix is 'zero' or 'one', where 'one' represents that protein *i* interacts with protein *j* and 'zero' represents that protein *i* does not interact with protein *j*.



	protein1	....	protein2517
protein1	0	....	0
protein2	0	....	1
protein3	1	....	1

Figure 2: Cells representing interactions between proteins have attribute values of 'zero' or 'one'.

3.2 Filtering Clusters by Size

We filter the clusters by size so that clusters of size less than a lower bound are ignored. The lower bound is determined experimentally for each PIN. One reason for ignoring small clusters is that an overlap of *x%* between a large cluster and a known complex is less likely to be by chance than an overlap of *x%* for a small cluster. Furthermore, small known complexes have low protein interaction rates and thus it is difficult to detect these complexes through clustering of PINs. Thus, small clusters are less likely to represent true protein complexes.

In the previous work by King et al. the results were also filtered by cluster density (i.e. the average number of interactions between proteins in a cluster) and functional homogeneity (i.e. whether a known functional annotation occurs in a cluster more frequently than would be expected by random). We do not filter the results by cluster density or functional homogeneity, because the clusters resulting from our algorithm have a more complex structure and we want all clusters to be investigated for structural properties. We do not filter the results by functional homogeneity because we want to evaluate the results independently of whether a function occurs frequently in the cluster – for example, a function might occur frequently at a high layer but a totally different function might occur at a lower layer and this may show something interesting about the complex's structure.

3.3 Matching Clusters to Complexes

We used matching criteria proposed in (*King et al., 2004*) to match the filtered clusters of proteins to the known protein complexes in the MIPS complex database (Mewes et al., 2002). According to the matching criteria, a cluster matches a known MIPS complex by *overlap* if there are sufficient overlapping proteins between them and preference is given to larger overlapping clusters and complexes. A cluster matches a known MIPS complex by *containment* if the cluster is nearly entirely contained in the complex. A large cluster containing a small complex is not useful for researchers, so we ignore this case.

The notation *O(C)* represents the set of all objects (proteins) in a cluster or complex *C*. We consider a cluster *Cl* to match a complex *Co* by overlap if both criteria are satisfied:

$$\frac{|O(Cl) \cap O(Co)|}{|O(Cl)|} \geq \frac{P_{cluster}}{\log_{10}(7+|O(Cl)|)}$$

and

$$\frac{|O(Cl) \cap O(Co)|}{|O(Co)|} \geq \frac{P_{complex}}{\log_{10}(7+|O(Co)|)}$$

This means that for *Cl* to match *Co* by overlap: *a*. the proportion of *Cl's* proteins that are contained in *Co* should be larger than a percentage which decreases as the size of *Cl* increases, and *b*. the proportion of *Co's* proteins that are contained in *Cl* should be larger than a percentage which decreases as the size of *Co* increases. Thus, matches by overlap occur easier for larger overlapping clusters and complexes rather than smaller ones.

We consider a cluster to match a complex by containment if:

$$\frac{|O(Cl) \cap O(Co)|}{|O(Cl)|} \geq P_{contain}$$

This means that for *Cl* to match *Co* by containment, the proportion of *Cl's* proteins that are contained in *Co* should be at least $P_{contain}$. The constants $P_{cluster}$, $P_{complex}$ and $P_{contain}$ are user-defined, experimentally derived proportions between 0 and 1. More details on these matching criteria and their experimental derivation are given in (*King et al., 2004*).

3.4 Evaluation of Results

To evaluate the effectiveness of our clustering algorithm for finding protein complexes, we filter the clusters by size (Sections 3.2) and then match them to the MIPS complexes according to the matching criteria (Section 3.3). Our goal is to achieve a high number of *passing clusters*, *matching clusters* and high *prediction rate*. *Passing* clusters are those that pass the size filter. *Matching* clusters are passing clusters that match at least one known MIPS complex according to the matching criteria. The *prediction rate* is the proportion of passing clusters that are also matching clusters. Another goal of our work is for the matched complexes to be of a large size and to have a large overlap with the matching clusters.

We use strict values for the matching criteria of $P_{cluster}=P_{complex}=0.7$ and $P_{contain}=0.9$, such that a cluster matches a complex only if there is a significant overlap between them.

4 The MULIC Clustering Algorithm

MULIC is an extension of the k-Modes clustering algorithm for categorical data sets (*Huang, 1998*). The k-Modes clustering algorithm requires the user to specify the number of clusters to be produced and the algorithm builds and refines the specified number of clusters. Each cluster has a mode associated with it. Assuming that the objects (proteins) in the data set are described by *m* categorical attributes, the mode of a cluster is a vector $Q=\{q_1, q_2, \ldots, q_m\}$ where q_i is the most frequent value for the *i*th attribute in the given cluster.

The MULIC clustering algorithm makes substantial changes to k-Modes. MULIC ensures that when each object is clustered it is inserted into the cluster with the most similar mode, thus maximizing the similarity between the object and the mode:

$$similarity(o_i, \text{mode}_i) \qquad (1)$$

where o_i is the *i*th object in the data set and $mode_i$ is the mode of the *i*th object's cluster. The similarity metric is defined in Section 4.1.

The MULIC algorithm has the following characteristics. First, the number of clusters is not specified by the user. Clusters are created, removed or merged during the clustering process, as the need arises. Second, it is possible for all objects to be assigned to clusters of size two or greater by the end of the process. However, outliers are assigned to separate clusters of size one. Third, clusters are layered.

Figure 3 shows the main part of the MULIC clustering algorithm. The algorithm starts by reading all objects from the input file and storing them in *S*. The first object is inserted in a new cluster, the object becomes the mode of the cluster and the object is removed from *S*. Then, it continues iterating over all objects that have not been assigned to clusters yet, to find the closest cluster. In all iterations, the closest cluster for each unclassified object is the cluster with the highest similarity between the cluster's mode and the object, as computed by the similarity metric.

```
Input:  (1)  a set $S$ of objects;
Parameters:  (1)  $\delta\varphi$ : the increment for $\varphi$;
             (2)  threshold for $\varphi$ : the maximum number
                  of values that can differ between an
                  object and the mode of its cluster;
Default parameter values: (1)  $\delta\varphi = 1$;
                          (2)  threshold = the number of
                               categorical attributes $m$;
Output: a set of clusters;
Method:
  1.  Insert the first object into a new cluster, use the
      object as the mode of the cluster, and remove the
      object from $S$;
  2.  Initialize $\varphi$ to 1;
  3.  Loop through the following until S is empty or $\varphi$
      is greater than the specified threshold
      a.  For each object $o$ in $S$
              i.    Find $o$'s closest cluster $c$ by using the
                    similarity metric to compare $o$ with the
                    modes of all existing cluster(s);
              ii.   If the number of different values
                    between $o$ and $c$'s mode is larger than
                    $\varphi$, insert $o$ into a new cluster
              iii.  Otherwise, insert $o$ into $c$ and update
                    $c$'s mode;
              iv.   Remove object $o$ from $S$;
      b.  For each cluster $c$, if there is only one object in
          $c$, remove $c$ and put the object back in $S$;
      c.  If in this loop no objects were placed in a
          cluster with size > 1, increment $\varphi$ by $\delta\varphi$.
```

Figure 3: The MULIC clustering algorithm.

The variable φ is maintained to indicate how strong the similarity has to be between an object and the closest cluster's mode for the object to be inserted in the cluster – initially φ equals 1, meaning that the similarity has to be very strong between an object and the closest cluster's mode. If the number of different values between the object and the closest cluster's mode is greater than φ then the object is inserted in a new cluster on its own, else, the object is inserted in the closest cluster and the mode is updated.

At the end of each iteration, all objects assigned to clusters of size one have their clusters removed so that the objects will be re-clustered at the next iteration. This ensures that the clusters that persist through the process are only those containing at least 2 objects for which the required similarity can be found. Objects assigned to clusters with size greater than one are removed from the set of unclassified objects *S*, so those objects will not be re-clustered.

At the end of each iteration, if no objects have been inserted in clusters of size greater than one, then the variable φ is incremented by $\delta\varphi$. Thus, at the next iteration the criterion for inserting objects in clusters will be more flexible. The iterative process stops when all objects are classified in clusters of size greater than one, or φ exceeds a user-specified *threshold*. If the *threshold* equals its default value of the number of attributes m, the process stops when all objects are assigned to clusters of size greater than one.

The MULIC algorithm can eventually classify all objects in clusters, even if the closest cluster to an object is not that similar, because φ can continue increasing until all objects are classified. Even in the extreme cases, where an object o with m attributes has only zero or one value similar to the mode of the closest cluster, it can still be classified when $\varphi = m$ *or* $\varphi = m\text{-}1$, respectively.

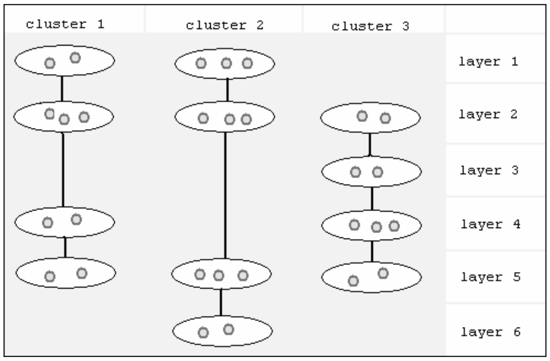


Figure 4: MULIC results. Each cluster consists of one or more different layers representing different similarities of the objects attached to the cluster.

Figure 4 illustrates what the results of MULIC look like. Each cluster consists of many different "layers" of objects. The layer of an object represents how strong the object's similarity was to the mode of the cluster when the object was assigned to the cluster. The cluster's layer in which an object is inserted depends on the value of φ. Lower layers have a lower coherence - meaning a lower average similarity between all pairs of objects in the layer - and correspond to higher values of φ. MULIC starts by inserting as many objects as possible in top layers – such as layer 1 - and then moves to lower layers, creating them as φ increases.

If an unclassified object has equal similarity to the modes of the two or more closest clusters, then the algorithm tries to resolve this 'tie' by comparing the object to the mode of the top layer of each of these clusters – the top layer of a cluster may be layer 1 or 2 and so on. Each cluster's top layer's mode was stored by MULIC when the cluster was created, so it does not need to be recomputed. If the object has equal similarity to the modes of the top layer of all of its closest clusters, the object is assigned to the cluster with the highest bottom layer. If all clusters have the same bottom layer then the object is assigned to the first cluster, since there is insufficient data for selecting the best cluster.

4.1 MULIC Characteristics for PIN Data Clustering

MULIC includes characteristics specific for PIN data clustering. A position of the mode of a cluster is set to 'one' if there is at least one object in the cluster that has an attribute value of 'one' in the corresponding position. We do not use the most frequent value for each position of the mode as in the traditional k-Modes, because with the PIN data sets most or all values of the mode would be set to 'zero'.

When calculating the similarity between a mode and an object, pairs of 'zero' attribute values between mode μ and object o are ignored. The similarity metric is defined as follows:

$$similarity(o,\mu) = \sum_{i=1}^{m} \sigma(o_i,\mu_i) \qquad \sigma(o_i,\mu_i) = \begin{cases} 1 & (o_i = \mu_i = 1); \\ 0 & otherwise\ . \end{cases}$$

The function σ returns 1 if an object o and a mode μ have identical CAs of 'one' at a position, and returns 0 otherwise.

4.2 Merging of Clusters

We should generally avoid the situation where the similarity of the top layers of two different clusters is stronger than the similarity of the top and bottom layer of the same cluster. To avoid this, after the clustering process MULIC can merge pairs of clusters whose top layers' modes' dissimilarity is less than the maximum layer depth of the two clusters. For this purpose, MULIC preserves the modes of the top layers of all clusters. This process reduces the total number of clusters and may improve the quality of the results. This process is described as follows:

```
for (c = first cluster to last cluster)
    for (d = c+1 to last cluster)
        if the dissimilarity between c's mode and d's mode is less than the maximum
          layer depth of c and d, merge c into d and break the inner loop;
```

where the dissimilarity between two modes ($Q_c = \{q_{c1}, ..., q_{cm}\}$ and $Q_d = \{q_{d1}, ..., q_{dm}\}$) is defined as:

$$\text{dissimilarity}(Q_c\ , Q_d) = \sum_{i=1}^{m} \delta(q_{ci}, q_{di}) \qquad \delta(q_{ci}, q_{di}) = \begin{cases} 0 & (q_{ci} = q_{di}); \\ 1 & (q_{ci} \neq q_{di}). \end{cases}$$

4.3 Detection of Outliers

MULIC will eventually put all the objects in clusters if the *threshold* for φ equals its default value of the number of attributes m. When φ equals m, any object that remains unclassified will be inserted in the lowest layer of a cluster. This is undesirable if the object is an outlier and has little similarity with any cluster. The user can disallow this situation from happening by specifying a value for *threshold* that is less than m. In this case when φ exceeds the maximum allowed value specified by *threshold*, any remaining objects are treated as outliers by classifying each object in a separate cluster of size one. We showed that top layers are more reliable than lower layers in (*Andreopoulos et al., 2004*).

5 Experimental Results

Our tests involve various values of $\delta\varphi$, *threshold*, as well as both merging and not merging the clusters. For most of our experiments we set *threshold* to its default value of the total number of objects (proteins) because we do not want any proteins to be treated as outliers and we want all proteins to be assigned to clusters with at least one other protein, since a protein does not function independently but in protein complexes. The detailed results of our experiments including clustering outputs and matches with known MIPS complexes are available on the supplementary information website (see Abstract).

5.1 Filtering the Clusters by Cluster Size

Increasing the lower bound for the cluster size decreases the number of passing clusters. The lower bound for the cluster size filter was set to a value of 4, to allow plenty of clusters to pass the filter while ensuring they had a good chance of matching known MIPS complexes. Table 1 shows the number of clusters that pass the size filter for the chosen lower bound for different yeast PINs.

Table 1: Numbers of total and passing clusters for the yeast PINs. The lower bound for the cluster size filter is 4. The value of *threshold* is set to its default value. The clusters are not merged after the clustering process.

PIN	$\delta\varphi$	Total clusters	Passing clusters
Y2K	3	232	73
Y11K	3	480	178
Y78K	5	936	130

5.2 MULIC Clusters Matching MIPS Complexes by Overlap and by Containment

In most of our Y2K tests without merging clusters, there were at least 10 MULIC clusters that matched known MIPS complexes by overlap (cluster and complex are large enough and have significant proportions of overlapping proteins). Furthermore, there were approximately 20 MULIC clusters that matched known complexes by containment (a significant proportion of the cluster is contained in the complex). Table 2 shows that all of the MULIC clusters that match known MIPS complexes by overlap have a large number of overlapping proteins. A MULIC cluster of size 12 matches by overlap the MIPS protein complex “550.3.60” of size 13. A MULIC cluster of size 10 matches the MIPS protein complex “550.2.163” of size 10. In this case, 3 of the proteins in the cluster do not overlap with the complex. All 3 of the non-overlapping proteins were in the bottom layer of the MULIC cluster. For the matched complex “500.10.40” there is also one protein in the bottom layer of the cluster that does not overlap with the complex. Relations of a cluster’s bottom layer proteins with the matched MIPS protein complexes can be further investigated in the lab.

5.3 Results after Merging of Clusters

Similar MULIC clusters can be merged after the clustering process, as described in Section 4.2. Table 3 shows that merging the clusters has the effect of reducing the total number of clusters. Many of the original clusters get merged into few large clusters and all or most of these large merged clusters match a known MIPS complex. For example, the second row in Table 3 shows reducing the number of clusters to 210 after merging. The original number of clusters was 232, so 22 small clusters were merged into 2 large merged clusters. As shown, both of these merged clusters match by overlap known MIPS complexes. What is most interesting is the size of these merged clusters. One merged cluster is of size 104 and it matches by overlap the MIPS complex “550.1.149” of size 88 that is involved in RNA metabolism (Gavin AC, et al., 2003). The second merged cluster is of size 14 and it matches by overlap the MIPS complex "360.10.20" of size 18, that is involved in 19/22S regulation. Clearly, these matches point to the effectiveness of MULIC combined with merging for predicting large complexes.

One would expect that some small clusters that match different complexes would be merged and some of the resulting merged clusters would match more than one complex. However, this never happens in our detailed results (see supplementary info page). In fact, all of the matching merged clusters match by overlap single complexes, despite their large size. This is another testament to the effectiveness of this method, given that the majority of known protein complexes are of a small size (typically of a size less than 10 proteins) and large complexes are relatively infrequent. Large clusters that are likely to match large protein complexes are more interesting in a lab setting than small clusters.

Table 2: Pairs of MIPS complexes and Y2K clusters that match by overlap and their overlapping proteins. The value of $\delta\varphi$ is 3. The value of *threshold* is set to its default value. Clusters are not merged after the clustering process.

Matches by overlap	Overlapping proteins between matching cluster and complex	Proteins contained in the cluster but not in the complex
Complex 550.3.60 (20S Proteosome) of size 13 matches cluster 179 of size 12	YJL001W, YGR253C, YPR103W, YOL038W, YMR314W, YML092C, YGR135W, YGL011C, YER012W, YBL041W, YOR362C	YER094C
Complex 550.2.163 of size 10 matches cluster 133 of size 10	YNL147W, YMR268C, YJL124C, YER112W, YDL160C, YCR077C, YBL026W	YNL118C, YER146W, YPR182W
Complex 550.2.241 of size 4 matches cluster 80 of size 4	YPR101W, YMR213W, YLR117C, YLL036C	
Complex 260.90 (Arp2p/Arp3p complex) of size 6 matches cluster 92 of size 8	YNR035C, YLR370C, YKL013C, YJR065C, YIL062C, YDL029W	YGR196C, YBR234C
Complex 260.30.10 (Coat complexes) of size 8 matches cluster 125 of size 8	YNL287W, YIL076W, YFR051C, YDR238C, YDL145C, YGL137W, YPL010W	YKR067W
Complex 550.1.4 (probably cell cycle) of size 5 matches cluster 135 of size 5	YLR314C, YJR076C, YHR107C, YDL225W, YCR002C	
Complex 500.10.40 (elF3) of size 7 matches cluster 199 of size 6	YNL244C, YDR429C, YMR309C, YMR146C, YBR079C	YBL076C
Complex 160 (exocyst complex) of size 7 matches cluster 204 of size 6	YIL068C, YGL233W, YER008C, YDR166C, YPR055W, YLR166C	
Complex 550.1.166 (probably signalling) of size 10 matches cluster 209 of size 9	YDR422C, YDR028C, YGL208W, YER027C, YDR477W, YGL115W	YEL022W, YDR099W, YDR001C

Table 3: Numbers of merged and unmerged Y2K clusters passing the size filter and matching a MIPS complex after reducing the total number of clusters through merging. The number of clusters before merging was 232 of which 73 were passing and 29 were matching clusters. The value of $\delta\varphi$ is 3. The value of *threshold* is set to its default value.

Total clusters after merging	Merged clusters	Unmerged clusters	Passing merged clusters	Matching merged clusters	Passing unmerged clusters	Matching unmerged clusters	Prediction rate for merged clusters
220	1	219	1	1	65	28	100%
210	2	208	2	2	61	27	100%
200	3	197	3	2	57	27	66%
190	5	185	5	2	52	26	40%
180	6	174	6	3	48	23	50%
170	9	161	9	6	42	20	66%
160	11	149	11	6	38	18	55%
150	8	142	8	5	37	17	63%
100	10	90	10	3	19	9	30%
67	8	59	8	4	11	5	50%

5.4 Results after Treating Objects as Outliers

Objects are treated as outliers by setting the *threshold* for φ to a value less than the number of attributes *m*, as discussed in Section 4.3. When φ exceeds the maximum allowed value specified by *threshold*, any remaining objects are treated as outliers by placing them independently in clusters of size one. Table 4 shows the results for various values of *threshold* without merging clusters. A lower value of *threshold* leads to treating more proteins as outliers which is beneficial for the prediction rate. When setting *threshold* to its default value of the number of attributes *m*, many proteins that have little interaction similarity to any other protein will likely be clustered incorrectly with proteins of different complexes; then fewer clusters will match known complexes. On the other hand, by setting threshold to a lower value these proteins are treated as outliers; they are placed in independent clusters of size one and then filtered out though the cluster size filter.

Table 4: Numbers of Y2K clusters passing the size filter and matching a MIPS complex using various values of *threshold*. The value of $\delta\varphi$ is 3. Clusters are not merged after the clustering process.

threshold	Total clusters	Passing clusters	Matching clusters	Prediction rate
20	219	65	32	50%
25	227	67	32	48%
30	228	67	30	45%
35	230	69	29	43%
40	232	72	31	43%

Figure 5 illustrates the prediction rates for the Y78K data set, for various values of *threshold* and both merging and not merging clusters. As shown, the highest prediction rates are derived using a low value of *threshold* of 17. The prediction rates for Y78K are not very dissimilar from Y2K, even though many interactions of low confidence are used in the clustering process. This supports that the clustering process is not significantly affected by the high rate of false positives in data from high-throughput interaction experiments.

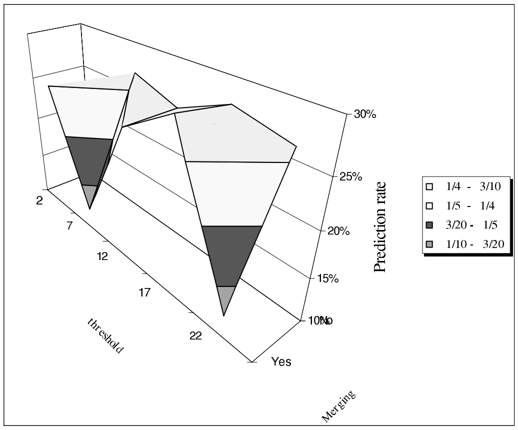


Figure 5: This graph illustrates the Y78K prediction rates, using various values of *threshold* and both merging and not merging the clusters. The value of $\delta\varphi$ is 5.

Table 5: Numbers of Y2K clusters passing the size filter and matching a MIPS complex using various values of $\delta\varphi$. The value of *threshold* is set to its default value. Clusters are not merged after the clustering process.

δφ	Total clusters	Passing clusters	Matching clusters	Prediction rate
1	251	64	25	39%
3	232	73	29	40%
5	218	75	27	36%
10	189	72	23	32%
25	160	68	22	32%
50	156	61	23	38%
75	150	56	20	36%
100	151	56	19	34%

5.5 Results for Various Values of $\delta\varphi$

Table 5 shows the MULIC results for Y2K using various values of $\delta\varphi$ and without merging clusters. We notice that a value of $\delta\varphi$ set to 3 results in more clusters matching complexes than other values. The reason why a $\delta\varphi$ value greater than 1 is used is that it allows sufficient proteins to be clustered at each iteration so that the modes of the clusters are given the opportunity to change, as opposed to remaining static. Then, at the next iteration more unclassified proteins will be attracted to the cluster. A value of $\delta\varphi$ that is too large, on the other hand, decreases the prediction rate and the quality of the results because many proteins are assigned to clusters to which they are not so similar.

6 Discussions

MULIC has characteristics specific to PINs that allow it to find unknown protein complexes. In PINs, there are many complexes of small sizes that have high internal connectivity, where the connectivity is the number of interactions divided by the number of proteins. For example, in the yeast proteome of 6,000 proteins most complexes have sizes of 3-30 proteins. MULIC does not require for the number of clusters to be specified - a new cluster is created when a set of proteins is discovered that have similar (highly overlapping) interaction sets. As the process continues MULIC relaxes its criterion for assigning proteins to clusters, forming cluster layers of lower connectivity. This is in accordance with a recent study (*Dezso et al., 2004*) in which protein complexes were discovered to feature centers of highly co-expressed proteins which mostly display the same deletion phenotype.

6.1 Comparisons

MULIC is able to achieve high matching rates between PIN clusters and known protein complexes. In comparison, Bader and Hogue generate a set of 209 protein complexes, of which 54 match the MIPS database in at least 20% of their proteins in a yeast PIN of 15,000 interactions (*Bader & Hogue, 2003*). King et al. generate a set of 28 clusters filtered by size, density and functional annotation, of which 18 match the MIPS protein complex database in the Y2K yeast PIN of 2,000 interactions (*King et al., 2004*). Our results complement these efforts to better understand protein complexes. Our prediction rate is lower than that of (*King et al., 2004*) and one reason for this is that we get more passing clusters since we do not filter the results by density and functional homogeneity as in their work. Furthermore, we use strict values for the matching criteria ($P_{cluster}$ = $P_{complex}$ = 0.7 and $P_{contain}$ = 0.9) such that a cluster matches a complex only if there is a significant overlap. Table 6 shows relaxing the matching criteria increases the number of matching clusters – with relaxed matching criteria, 92% of the passing clusters match a known MIPS complex. Table 7 shows a comparison of the MULIC results with the results of the RNSC clustering algorithm (*King et al., 2004*). Even with strict criteria, our number of clusters that match a known MIPS complex is higher and our cluster size is often larger (both works used a lower bound of 4 for the cluster size filter for Y2K). With MULIC, before merging clusters there was a cluster of size 55 proteins matching the MIPS complex "550.1.149" of size 88 proteins. After merging to 220 clusters, there was a cluster of 79 proteins matching the same complex.

Table 6: As the matching criteria are relaxed, the number of Y2K matching clusters increases. Since there are 73 passing clusters for Y2K, the prediction rate for Y2K also increases. The value of $\delta\varphi$ is 3. The value of *threshold* is set to its default value. Clusters are not merged after the clustering process.

Matching criteria	Matching clusters	Prediction rate
$P_{cluster} = P_{complex} = 0.7$, $P_{contain} = 0.9$	29	40%
$P_{cluster} = P_{complex} = 0.5$, $P_{contain} = 0.7$	52	71%
$P_{cluster} = P_{complex} = 0.3$, $P_{contain} = 0.5$	67	92%

Table 7: The number of Y2K matching clusters and the largest size of a cluster that matches a MIPS complex by overlap, for the MULIC, RNSC, k-Modes and AutoClass algorithms. All works used a lower bound of 4 for the cluster size filter. MULIC used strict values for the matching criteria of $P_{cluster}$=$P_{complex}$=0.7 and $P_{contain}$=0.9. The value of $\delta\varphi$ is 3. The value of *threshold* is set to its default value.

	Number of Y2K matching clusters	Largest size of a cluster that matches a MIPS complex by overlap
MULIC	32	MIPS complex "550.1.149" of size 88 matches MULIC merged cluster of size 79 by overlap. Their overlap is 44.
RNSC	18	MIPS complex of size 29 matches RNSC cluster of size 17 by overlap. Their overlap is 7.
k-Modes	16	MIPS complex of size 20 matches k-Modes cluster of size 15 by overlap. Their overlap is 7.
AutoClass	10	MIPS complex of size 15 matches AutoClass cluster of size 14 by overlap. Their overlap is 6.

We also applied k-Modes (*Huang, 1998*) and AutoClass (*Stutz & Cheeseman, 1995*) to the same PIN data sets and compared their results with the MULIC results. Table 7 summarizes the results. K-Modes does not have the MULIC characteristics specific to PIN clustering (described in section 4.1) and we modified the source code to implement them. Without these characteristics, the clusters' modes would have all values set to zero. To evaluate the k-Modes and AutoClass results on our PIN data sets we compared the clusters to known MIPS complexes. For the k-Modes experiments, we did trials by setting the number of clusters *k* to values between 2 and 1500. For the k-Modes experiments we set the convergence threshold to 0 and we set the modes of the initial clusters equal to the values of the first objects inserted. For the AutoClass experiments we did not specify the number of clusters beforehand as the software considers results for numbers of clusters varying from a minimum of 2; we set the prior distribution for the categorical attributes to the *single multinomial* distribution, with no attributes ignored, which was also the distribution chosen by the developers of the software for their tests on the soybean data sets.

6.2 Cluster Structure

The multiple layer structure of the MULIC clusters reveals several things about the structures of the predicted protein complexes that could not be identified with other algorithms. In all of the derived MULIC clusters the top-layer proteins (layer 0 and 1) have the highest connectivity to the other protein members of the cluster. For clusters that match known MIPS complexes, the proteins in the top layer are often 'central' points of connectivity for the matched complex and perhaps even the entire cell – in other words, interactions occur with top-layer proteins more frequently than other proteins in the complex. For example, the well-studied FKS1p (YLR342W) and FKS2p (YGR032W) proteins have a very high connectivity to the other proteins in their complex and were clustered in the top layers of MULIC clusters. FKS1p and FKS2p are catalytic subunits of the beta-1,3-glucan synthase complex, which synthesize beta-1,3-glucan, a major structural polymer of the cell wall in Saccharomyces cerevisiae. The drug caspofungin binds to FKS1p and FKS2p to disturb the interactions of the glucan synthase complex

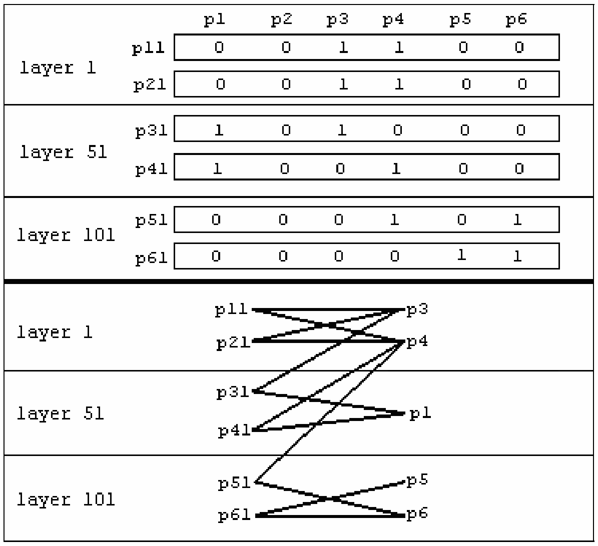
(*Markovich et al., 2004*). Thus, a biologist could start by testing a new drug on the proteins in top layers, instead of all proteins in the cluster.

Table 8: For the Y2K matching clusters, the percentages of the proteins in different layers that are contained in the matched complex.

Layers	% of the proteins in these layers contained in matched complex
1-4	72%
7-10	66%
13-19	49%

Furthermore, the multiple layer structure of the derived MULIC clusters can be very useful in cases where few protein complexes are known for the PIN of an organism, such as fruitfly and worm. A researcher's experiments can initially focus on the proteins clustered in the top layers. Later, the proteins in lower cluster layers can guide the experiments for the growing set of predicted protein complexes. Table 8 shows that proteins in top layers of Y2K clusters matching a known MIPS complex are more likely to be contained in the matched complex, than proteins in bottom layers.

A MULIC cluster can be viewed as a graph, where the nodes correspond to proteins and the edges correspond to interactions between proteins. A MULIC cluster represented as a graph is referred to as an *outward decreasing density* (ODD) graph. An ODD graph has a set of 'central' nodes, with a dense set of undirected edges between them, and a set of 'peripheral' nodes. The nodes are organized in 'layers' such that the central nodes are considered to belong to layer 1 and the peripheral nodes to layers 2 to *n*. The layers 2 to *n* with the peripheral nodes have sparsely occurring edges to the central nodes in layer 1.



		p1	p2	p3	p4	p5	p6
layer 1	p11	0	0	1	1	0	0
	p21	0	0	1	1	0	0
layer 51	p31	1	0	1	0	0	0
	p41	1	0	0	1	0	0
layer 101	p51	0	0	0	1	0	1
	p61	0	0	0	0	1	1

Figure 6: A typical MULIC cluster with 3 layers and its representation as an ODD graph. *density_layer(1)*=4/4, *density_layer(51)*=4/7, *density_layer(101)*=4/11.

Let *density_layer(i)* represent the density of the edges occurring between the nodes in layer *i* and the nodes in layers 1 to *i*. It is defined as:

density_layer(i) = number of edges between nodes in layer i and nodes in layers 1 to i / number of nodes in layers 1 to i.

Then, an ODD graph has the following property:

density_layer(i) > density_layer(i+1)

Figure 6 illustrates a typical MULIC cluster and its representation as an ODD graph structure. Modeling the PIN topology as an ODD graph can help researchers to predict complexes, as well as the proteins that are likely to be central to complexes.

6.3 Complexity and Runtime

The worst case complexity of MULIC is $O(N^2)$, where N is the number of objects. A high runtime might occur in the rare situation where all objects (proteins) were extremely dissimilar to one another, such that the algorithm had to go through all m (number of attributes) iterations and all N objects were clustered in the last iteration. Table 9 shows the runtimes of our trials on the PINs. The experiments were performed on a Sun Ultra 60 with 256 MB of memory and a 300 MHz processor.

Table 9: Runtimes of MULIC on the PIN data sets.

PIN	Runtime
Y2K	10 seconds
Y11K	30 seconds
Y78K	7 minutes
F4K	2 minutes
W5K	1 minute

We did not encounter a very time-consuming data set in our clustering experiments. The most intensive test run was on Y78K which took seven minutes. The runtime of MULIC is better than or comparable to other algorithms, such as k-Modes and AutoClass, but MULIC can find more complexes and the cluster structure is more complex and more interesting for analysis. The reason for the low runtime is that most objects are clustered during the initial iterations when the top cluster layers (1, 2, 3) are created. Thus, few comparisons between objects and modes need to be done during the clustering process. Moreover, decreasing the value of *threshold* or increasing the value of $\delta\varphi$ improves the runtime significantly. Changing these parameters does not necessarily imply weakening the quality of the results. Decreasing the value of *threshold* is useful for detecting outliers. Increasing the value of $\delta\varphi$ often improves the quality of the resulting clusters (*Andreopoulos et al., 2004*).

7 Conclusion and Future Work

We have proposed a method for finding protein complexes based on clustering PINs represented as categorical data sets. The main advantage of this method is that clusters have multiple layers, where top layers are created first to contain proteins with very similar interaction sets - the similarity criterion is progressively relaxed at lower layers. Furthermore, this method does not require the number of complexes to be specified by the user – it returns as many coherent complexes as it can find. Furthermore, this method is effective for detecting proteins that are outliers. Moreover, this method can find complexes of greatly varying sizes. Comparison with MIPS complexes shows that the clusters are representative of known protein complexes, including many complexes of relatively large size. Researchers can label the proteins in top cluster layers as potentially significant pieces of the interactome and validate the predicted complexes in the lab.

The cluster merging process can be used to merge similar clusters, thus leading to predicting complexes of large sizes. We have shown that merged clusters significantly overlap with complexes of relatively large sizes, pointing to the method's effectiveness. The merging process may eventually place an object in more than one cluster, which is in accordance with the reality of proteins being involved in more than one complex. However, we have focused on single membership in this paper, assuming that a researcher will initially seek specific hints for guiding the experiments.

One direction worth pursuing is to extend our method so that it incorporates uncertainty on the interactions. In many PIN data sets the interactions have annotations of high, medium or low confidence. If the high confidence interactions are given a heavier weight in the clustering process, this may lead to improved complex prediction. This may also help to identify small protein complexes that have sparsely occurring interactions and connectivity, which is a drawback of current clustering algorithms applied to PINs. Another direction is to develop an improved method for merging clusters that will hopefully improve the results.

Another direction worth pursuing is to implement a parallel implementation of the MULIC clustering algorithm that will be capable of running on clusters of computers. This parallel implementation will ideally achieve linear speed-up on very large PINs.

Acknowledgements

This research was supported in part by a research grant from the Natural Science and Engineering Research Council (NSERC) of Canada and by an Ontario Graduate Scholarship.

References

Albert, R. & Barabasi, A.-L. (2002) Statistical mechanics of complex networks. Reviews of Modem Physics, 74,47-97.

Andreopoulos, B., An, A., Wang, X. (2004) MULIC: Multi-Layer Increasing Coherence Clustering of Categorical Data Sets. Department of Computer Science and Engineering, York University, Technical Report CS-2004-07.

Bader, G. & Hogue, C. (2003) An autormated method for finding molecular complexes in large protein interaction networks. BMC Bioinformatics, 4 (2).

Barabasi, A.-L. & Oltvai, Z. N. (2004) Network biology: understanding the cell's functional organization. Nature Reviews Genetics, 5, 101-113.

Batagelj, V., Zaveršnik M. Cores Decomposition of Networks. Recent Trends in Graph Theory, Algebraic Combinatorics, and Graph Algorithms, September 24-27, 2001, Bled, Slovenia.

Bu, D., Zhao, Y., Cai, L, Xue, H., Zhu, X., Lu, H., Zhang, J., Sun, S., Ling, L., Zhang, N., Li, G. & Chen, R. (2003) Topological structure analysis of the proteinprotein interaction network in budding yeast. Nucleic Acids Research, 31 (9), 2443-2450.

Dezso, Z., Oltvai, Z., Barabasi, A.-L., Genome Res. 2004, 13, 2450–2454.

Ding, C., He, X., Meraz, R. & Holbrook, S. Multi-protein Complex Data Clustering for Detecting Protein Interactions and Functional Organizations. Interface 2004: Computational Biology and Bioinformatics. Baltimore, MD. May 26-29, 2004.

Dunn R, Dudbridge F, Sanderson CM. (2005) The use of edge-betweenness clustering to investigate biological function in protein interaction networks. BMC Bioinformatics. 2005 Mar 1;6(1):39.

Gavin, A. C., Bosche, M., Krause, R., Grandi, P., Marzioch, M. & Bauer, A. (2003) A functional organization of the yeast proteome by systematic analysis of protein complexes. Nature, 415 (6868),141-147.

Giot, L., Bader, J. S., Brouwer, C., Chaudhuri, A., Kuang, B., Li, Y., Hao, Y. L., Ooi, C. E., Godwin, B., Vi- tols, E., Vijayadamodar, G., Pochart, P., Machineni, H., Welsh, M., Kong, Y., Zerhusen, B., Malcolm, R., VaITone, Z., Collis, A., Minto, M., Burgess, S., McDaniel, L., Stimpson, E., Spriggs, E, Williams, J., Neurath, K., Joime, N., Agee, M., Voss, E., Fur- J' tak, K., Renzulli, R., Aanensen, N., Can-olla, S., Bickelhaupt, E., Lazovatsky, Y, DaSilva, A., Zhong, J., Stanyon, C. A., Finley, R. L., Jr., White, K. P., Braven-nan, M., Jarvie, T., Gold, S., Leach, M., Knight, J., Shimkets, R. A., McKenna, M. P., Chant, K J. & Rothberg, J. M. (2003) A protein interaction map of Drosophila melanogaster. Science, 302 (5651), 1727-1736.

Glover, E (1989) Tabu search, part I. ORSA Journal on K Computing, 1 (3), 190-206. "ORSA" is called Informs today.

Hartuv, E. & Shamir, R. (2000) A clustering algorithm based on graph connectivity. Information Processing Letters, 76 (4-6),175-181.

LHo, Y., Gruhler, A., Heilbut, A., Bader, G. D., Moore, L., Adams, S. L., Millar, A., Taylor, P., Bennett, K., Boutilier, K., Yang, L., WoIting, c., Donaldson, I., Schandorff, S., Shewnarane, J., Vo, M., Taggart, J., Goudreault, M., Muskat, B., Alfarano, c., Dewar, D., Lin, Z., Michalickova, K., Willems, A. R., Sassi, H., Nielsen, P. A., Rasmussen, K. J., Andersen, 1. R., Johansen, L. E., Hansen, L. H., Jespersen, H., Podtelejnikov, A., Nielsen, E., Crawford, J., Poulsen, V., Sorensen, B. D., Matthiesen, J., Hendrickson, R. c., Gleeson, E, Pawson, T., Moran, M. E, Durocher, D., Mann, M., Hogue, C. W., Figeys, D., & Tyers, M. (2003) Systematic identification of protein complexes in saccharomyces cerevisiae by mass spectrometry. Nature, 415 (6868), 180-183.

Huang, Z. (1998) Extensions to the k-Means Algorithm for Clustering Large Data Sets with Categorical Values. Data Mining and Knowledge Discovery 2(3): 283-304.

Ito, T., Chiba, T., Ozawa, R., Yoshida, M., Hattori, M. & N Sakaki, Y. (2001) A comprehensive two-hybrid analysis to explore the yeast protein interactome. Proc. P Nat'!. Acad. Sci. USA, 98 (8), 4569-4574.

Ito, T., Tashiro, K., Muta, S., Ozawa, R., Chiba, T., Mishizawa, M., Yamamoto, K. & S. Kuhara, a. Y S. (2000) Toward a protein-protein interaction map of P the budding yeast: a comprehensive system to examine two-hybrid interactions in all possible combinations between the yeast proteins. *Proc. Nat'!. Acad. Sci. USA,* 97 (3), 1143-1147.

Jansen, R., Lan, N., Qian, J. & Gerstein, M. (2002) Integration of genomic datasets to predict protein com p1exes in yeast. *J. Struct. Funct. Genomics,* 2, 71-81.

King, A. D., Prulj, N., & Jurisica, I. (2004). Protein Complex Prediction via Cost-based Clustering. *Bioinformatics,* 20 (3), 340-348.

Li, S., Armstrong, C. M., Bertin, N., Ge, H., Milstein, S., Boxem, M., Vidalain, P.-O., Han, J.-D. J., Chesneau, A., Hao, *T.,* Goldberg, D. S., Li, N., Martinez, M., Rual, J.-F., Lamesch, P., Xu, L., Tewari, M., Wong, S. L., Zhang, L. v., Berriz, G. F., Jacotot, L., Vag1io, P., Reboul, J., Hirozane-Kishikawa, *T.,* Li, Q., Gabel, H. W., Elewa, A., Baumgartner, B., Rose, D. J., Yu, H., Bosak, S., Sequerra, R., Fraser, A., Mango, S. E., Saxton, W. M., Strome, S., van den Heuvel, S., Piano, F., Vandenhaute, J., Sardet, c., Gerstein, M., Doucette-Stamm, L., Gunsalus, K. C., Harper, J. w., Cusick, M. E., Roth, F. P., Hill, D. E. & Vidal, M. (2004) A map of the interactome network of the metazoan C. elegans. *Science,* 303 (5657), 540-543.

Mewes, H. W., Frishman, D., Guldener, U., Mannhaupt, G., Mayer, K., Mokrejs, M., Morgenstern, B., Munsterkotter, M., Rudd, S. & Weil, B. (2002) Mips: a database for genornes and protein sequences. *Nucleic Acids Research,* 30 (1),31-34.

Markovich S, *et al.* (2004) Genomic approach to identification of mutations affecting caspofungin susceptibility in Saccharomyces cerevisiae. *Antimicrob Agents Chemother* 48(10):3871-6

Newman, M. E. J. (2003) The structure and function of complex networks. *SIAM Review,* 45 (2), 167-256.

Reinoso-Martin C, Schuller C, Schuetzer-Muehlbauer M, Kuchler K (2003) The yeast protein kinase C cell integrity pathway mediates tolerance to the antifungal drug caspofungin through activation of Slt2p mitogen-activated protein kinase signaling. *Eukaryot Cell* 2(6):1200-10. http://www.pdg.cnb.uam.es/UniPub/iHOP/gs/31559.html

Shamir, R., Sharan, R. (2000) CLICK: A Clustering Algorithm for Gene Expression Analysis. RECOMB 2000, Tokyo, Japan.

Strogatz, S. H. (2001) Exploring complex networks. *Nature,* 410,268-276.

Stutz J. and Cheeseman P. (1995) Bayesian Classification (AutoClass): Theory and results. Advances in Knowledge Discovery and Data Mining, 153-180, Menlo Park, CA, AAAI Press.

Uetz, P., Giot, L., Cagney, G., Mansfield, T. A., Judson, R. S., Knight, J. R., Lockshon, D., Narayan, V., Srinivasan, M., Pochart, P., Qureshi-Emili, A., Li, Y., Godwin, B., Conover, D., Kalbfleish, T., Vijayadamodar, G., Yang, M., Johnston, M., Fields, S. & Rothberg, J. M. (2000) A comprehensive analysis of protein-protein interactions in saccharomyces cerevisiae. *Nature,* 403 (6770), 623-627.

von Mering, C., Krause, R., Snel, B., Cornell, M., Oliver, S. G., Fields, S. & Bork, P. (2002) Comparative assessment of large-scale data sets of protein-protein interactions. *Nature,* 417 (6887), 399-403.

West, D. B. (2001) *Introduction to Graph Theory, Second Edition.* Prentice Hall, Upper Saddle River, NJ.

Yang, Q., Lonardi, S. (2005) A Parallel Algorithm for Clustering Protein-Protein Interaction Networks. 2005 IEEE Computational Systems Bioinformatics Conference (CSB2005). Stanford University, CA. August 8-11, 2005.

Yu, H., Zhu, X., Greenbaum, D., Karro, J. & Gerstein, M. (2004) Topnet: a tool for comparing biological subnetworks, correlating protein properties with topological statistics. *Nucleic Acids Research,* 32 (l), 328-337.