2007

Finding Molecular Complexes Through Multiple Layer Clustering of Protein Interaction Networks

Bill Andreopoulos, *York University*
Aijun An, *York University*
Xiangji Huang, *York University*
Xiaogang Wang, *York University*

Finding molecular complexes through multiple layer clustering of protein interaction networks

Bill Andreopoulos* and Aijun An

Department of Computer Science and Engineering,
York University, M3J1P3, Toronto, Ontario, Canada
E-mail: billa@cs.yorku.ca E-mail: aan@cs.yorku.ca
*Corresponding author

Xiangji Huang

School of Information Technology,
York University, M3J1P3, Toronto, Ontario, Canada
E-mail: jhuang@yorku.ca

Xiaogang Wang

Department of Mathematics and Statistics,
York University, M3J1P3, Toronto, Ontario, Canada
E-mail: stevenw@mathstat.yorku.ca

Abstract: Clustering protein-protein interaction networks (PINs) helps to identify complexes that guide the cell machinery. Clustering algorithms often create a flat clustering, without considering the layered structure of PINs. We propose the MULIC clustering algorithm that produces layered clusters. We applied MULIC to five PINs. Clusters correlate with known MIPS protein complexes. For example, a cluster of 79 proteins overlaps with a known complex of 88 proteins. Proteins in top cluster layers tend to be more representative of complexes than proteins in bottom layers. Lab work on finding unknown complexes or determining drug effects can be guided by top layer proteins.

Keywords: clustering; multiple layer; protein; interaction; network; complex; graph; categorical; matrix; neighbourhood similarity; adjacency matrix; bioinformatics.

Reference to this paper should be made as follows: Andreopoulos, B., An, A., Huang, X. and Wang, X. (2007) 'Finding molecular complexes through multiple layer clustering of protein interaction networks', *Int. J. Bioinformatics Research and Applications*, Vol. 3, No. 1, pp.65–85.

Biographical notes: Bill Andreopoulos is a PhD student at the Department of Computer Science and Engineering of York University, Toronto, Canada. He received his MSc at the Department of Computer Science of University of Toronto (2001) and his BSc at the Department of Computing and Software of McMaster University (1999). His research interests include data mining, clustering and computational biology.

Aijun An is an Associate Professor at the Department of Computer Science and Engineering of York University. She received her PhD Degree in Computer Science from the University of Regina in 1997. She worked at the University of Waterloo as a Postdoctoral Fellow from 1997 to 1999 and as a Research Assistant Professor from 1999 to 2001. She joined York University in 2001. Her research interests include data mining, machine learning, and information retrieval.

Xiangji Huang is an Associate Professor of Information Technology at York University. Previously, he was a Post Doctoral Fellow at School of Computer Science, University of Waterloo. He did his PhD in Information Science at City University, London. Before he went into his PhD program, he worked as a Lecturer for four years. He also worked in financial industry in Canada for four years, where he was awarded a CIO Achievement Award. He has published more than 50 refereed papers. His research interests include information retrieval, data mining and bioinformatics.

Xiaogang Wang is an Assistant Professor at the Department of Mathematics and Statistics of York University. He received his PhD Degree in Statistics from the University of British Columbia in 2001. He worked at the Pacific Institute of Mathematical Sciences and Insightful Co. as a Postdoctoral Fellow from 2001 to 2002. He joined York University in 2002. His research interests include data mining and machine learning.

1 Introduction

The amount of Protein Interaction Network (PIN) data in databases has grown exponentially in recent years. Knowledge of the protein complexes in PINs has also increased, but at a slower rate. A complex is a group of proteins that interact with one another to carry out a function. Often, but not always, proteins in a complex have more interactions with one another than they do with proteins from other complexes. This allows clustering tools to find potential protein complexes by identifying the dense areas in a PIN. One of the challenges in analysing PIN data is to develop efficient clustering tools that can fairly accurately identify previously unknown protein complexes (Amau et al., 2005; Barabasi and Oltvai, 2004).

The main contribution of this paper is to propose a novel clustering approach for finding protein complexes in PIN data, using the MULIC algorithm (Andreopoulos et al., 2004). The main strength of this algorithm is that each cluster consists of layers. Moreover, we identify complexes based on neighbourhood similarity, by finding proteins that interact with the same proteins. This differs considerably from previous clustering approaches that have focused on local density and protein degree (Bader and Hogue, 2003; Barabasi and Oltvai, 2004; Bu et al., 2002; Ding et al., 2004). Tightly connected groups of proteins are expected to appear as clusters, but we extend beyond that: we also cluster proteins with ‘similar’ interaction patterns, i.e., proteins that interact with the same proteins. We transform the task into a problem of clustering of categorical data and we leverage our previous work on the MULIC categorical clustering algorithm. However, we modify our algorithm to make it suitable for clustering the PIN topology.

Our approach's key characteristics include:

- The PIN topology is decomposed into layered clusters. A cluster consists of layers formed gradually, by relaxing the similarity criterion for inserting objects in clusters. 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.
- Similar clusters can be merged after the initial clustering to further improve the clustering. Merging can capture more complex topological structures and clusters of various shapes and sizes.

We applied this approach to three yeast *Saccharomyces cerevisiae* PINs, one fruitfly *Drosophila melanogaster* PIN and one worm *Caenorhabditis elegans* PIN. We filtered the clusters by cluster size. We compared the filtered clusters with known protein complexes derived from the MIPS yeast database (Mewes et al., 2002).

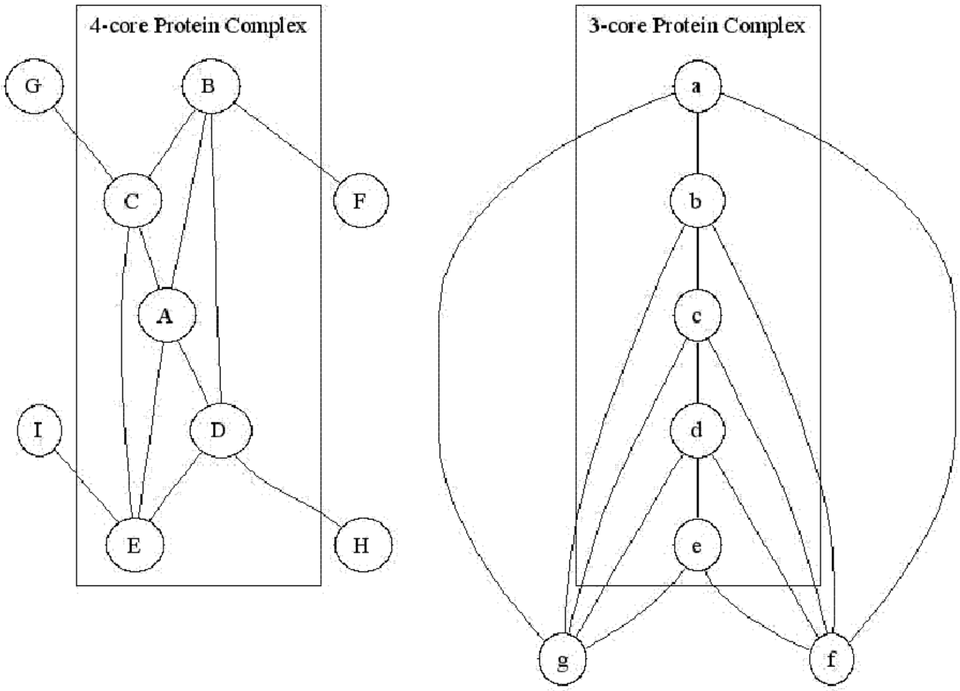
This paper is organised as follows. Section 2 describes previous related work. Section 3 describes the data sets and evaluation measures used. Section 4 describes the MULIC clustering algorithm. Section 5 presents the experimental results. Section 6 discusses the results, compares them to those of other algorithms and discusses the runtimes. 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 (West, 2001). A PIN can be viewed as an undirected graph, where the objects represent proteins and the edges represent the interacting proteins (Hartuv and Shamir, 2000; Alfarano et al., 2003). MULIC clusters objects based on the similarities of their neighbourhoods. The neighbourhood of an object is its set of edges, representing the protein(s) that it interacts with. The main advantage of MULIC is that clusters are layered and objects clustered at top layers have more similar neighbourhoods. Previous 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 and Hogue (2003). *K*-cores in graph theory were introduced by Batagelj and Zavernik (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 neighbours 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 $k = 4$ cannot 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 randomised 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 (Newman and Girvan, 2004). 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 and Lonardi (2005) propose a parallel implementation of Girvan and Newman's clustering algorithm (Newman and Girvan, 2004) 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 *Saccharomyces cerevisiae* PINs originating from Uetz et al. (2000) and von Mering et al. (2001) containing 2455 interactions (988 proteins), 11855 interactions (2617 proteins) and 78390 interactions (5323 proteins). We refer to these data sets as Y^{2K}, Y^{11K} and Y^{78K} respectively. Y^{2K} contains high confidence interactions only. Y^{11K} contains high and medium confidence interactions. Y^{78K} contains high, medium and low confidence interactions. The confidence represents the expected rate of false positives, depending on the experimental methods used to derive the 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 *Drosophila 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 data set as F^{4K}. Finally, we used one worm *Caenorhabditis elegans* PIN containing 5222 interactions (3659 proteins) (Li et al., 2004). We refer to this data set as W^{5K}. We first clustered these PINs 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 data on an organism is categorical. The objects (proteins) have categorical attribute values that are taken from the set of discrete values ('1', '0'). These values have no specified ordering. We represent PIN data as an $N \times N$ symmetric square adjacency matrix $A = (a_{ij})$, where N is the number of proteins in the PIN data set. The rows and columns represent proteins and $a_{ij} = 1$ if there is a known interaction between proteins i and j and $a_{ij} = 0$ otherwise. Figure 2 shows the formulation of the PIN data for our clustering approach.

Figure 2 Cells representing interactions between proteins have attribute values of '1'

$$\begin{array}{l|ccc|}
 & \text{protein 1} & \text{----------} & \text{protein 988} \\
\hline
\text{protein 1} & 0 & \text{----------------} & 0 \\
\text{protein 2} & 0 & \text{----------------} & 1 \\
\text{protein 3} & 1 & \text{----------------} & 1 \\
\hline
\end{array}$$

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. (2004) 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 highlayer 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:

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

and

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

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:

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

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 *matching 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 matching 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 $P_{\text{cluster}} = P_{\text{complex}} = 0.7$ and $P_{\text{contain}} = 0.9$, such that a cluster matches a complex only if there is a significant overlap between them.

In our implementation, one cluster can be matched to more than one MIPS complex. However, this does not bias the number of matching clusters or the matching rate, which only reflect the clusters for which at least one match was found. Multiple clusters can be matched to one MIPS complex; in the next section we describe a cluster merging process that is very effective for identifying similar clusters that are likely to match the same MIPS complex.

4 The MULIC clustering algorithm

MULIC clusters consist of layers, where each layer corresponds to a different value of the similarity criterion used for inserting objects (proteins) in clusters. An optional final step merges similar clusters to find more interesting cluster structures (Andreopoulos et al., 2004).

Each MULIC cluster has a *mode*. Assuming that the objects in the data set are described by *m* categorical attributes, the mode of a cluster *c* is a vector $\mu_c = \{\mu_{c1},\ldots,\mu_{cm}\}$. The *i*th position μ_{ci} is set to '1' if there is at least one object in cluster *c* that has a value of '1' in the *i*th attribute. We do not use the most frequent value for each position of the mode as in the traditional *k*-Modes (Huang, 1998), because with our data set most or all values of the mode would be set to '0'.

MULIC ensures that when each object *o* is clustered it is inserted into the cluster *c* with the most similar mode μ_c, thus maximising the similarity between object and mode. The similarity metric is defined as follows:

$$\text{similarity}\,(o,\mu_c) = \frac{1}{m} \times \sum_{i=1}^{m} \sigma(o_i,\mu_{ci}) \quad \sigma(o_i,\mu_{ci}) = \begin{cases} 1 & (o_i = \mu_{ci} = 1); \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

where *o* is an object in the data set and μ_c is the mode of the cluster *c* in which *o* is to be inserted. The function σ returns 1 if an object *o* and a mode μ_c have identical values of '1' at a position *i* and returns 0 otherwise. When calculating the similarity between a mode μ_c and an object *o*, pairs of '0' attribute values between mode and object are ignored.

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.

The variable ϕ is maintained to indicate how high the dissimilarity is allowed 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 only one value can differ 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 two objects. Objects assigned to clusters of size greater than one are removed from the set of unclassified objects *S*, so those objects will not be re-clustered.

Figure 3 The MULIC clustering algorithm

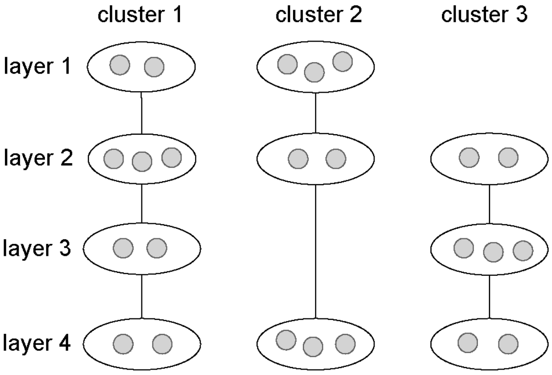
```
Input: (1) a set $S$ of objects;
Parameters: (1) $\delta\varphi$ : the increment for $\varphi$;
            (2) threshold for $\varphi$ : the maximum number of values that
                can differ between an object and the mode of its cluster;
Default parameter values: (1) $\delta\varphi = 1$;
                          (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 $\varphi$>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 iteration no objects were inserted in a cluster with
        size > 1, increment $\varphi$ by $\delta\varphi$.
```

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 $\delta\phi$. 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 very dissimilar, because ϕ can continue increasing until all objects are classified. Even in the extreme case, 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 $\phi = m$ or $\phi = m - 1$, respectively.

Figure 4 illustrates what the results of MULIC look like. Each cluster consists of one or more different 'layers'. The layer of an object represents how high the object's dissimilarity 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. Bottom layers, such as 1000, correspond to higher values of ϕ and have a lower coherence – meaning a higher average dissimilarity between all pairs of objects in the layer. MULIC starts by inserting as many objects as possible in top layers, such as layer 1, and then moves to bottom layers, creating them as ϕ increases.

Figure 4 A MULIC cluster consists of one or more layers representing dissimilarities between the objects and mode. Ovals are layers and circles are objects



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 Ordering the objects before clustering

When running MULIC with different random orderings of the data set objects (proteins), the result is often different. The modes and clusters are influenced most by the attribute values of the proteins that are clustered first in top cluster layers. It makes more sense to cluster first the proteins of low degree (i.e., proteins that interact with few proteins) and last the proteins of high degree. Two proteins of high degree are unlikely to interact with the exact same proteins, thus there are unlikely to be many proteins of high degree in top cluster layers. By ordering the proteins and presenting them to the clustering process from low to high degree, and by relaxing φ gradually, the clusters get an onion-layered structure where proteins in top layers interact with similar sets of proteins and proteins in bottom layers interact with less similar sets of proteins.

4.2 Merging of clusters

Sometimes the dissimilarity of the top layers of two clusters is less than the dissimilarity of the top and bottom layers of one of the two clusters. 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 ($\mu_c = \{\mu_{c1}, \ldots, \mu_{cm}\}$ and $\mu_d = \{\mu_{d1}, \ldots, \mu_{dm}\}$) is defined as:

$$\text{dissimilarity}\,(\mu_c\ ,\mu_d) = \sum_{i=1}^{m} \delta(\mu_{ci}, \mu_{di}) \quad \delta(\mu_{ci}, \mu_{di}) = \begin{cases} 0 & (\mu_{ci} = \mu_{di}); \\ 1 & (\mu_{ci} \neq \mu_{di}). \end{cases}$$

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).

4.4 Characteristics for PIN data clustering

We implemented several characteristics specific for PIN data clustering, such as the mode's updating and the similarity metric as described above. Proteins are ordered from low to high degree.

While the MULIC clustering algorithm follows the basic framework of *k*-Modes (Huang, 1998), it has substantially different characteristics:

- Clusters are layered.
- The number of clusters is not specified by the user - clusters are created, removed or merged, as the need arises. *K*-Modes requires the user to specify the number of clusters and the algorithm builds and refines the specified number of clusters.
- All MULIC clusters are of size two or greater.

5 Experimental results

Our tests involve various values of $\delta\phi$, *threshold*, as well as both merging and not merging the clusters. The default values of $\delta\phi$ is 3. 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. We have placed the detailed results of our experiments including clustering outputs and matches with known MIPS complexes online.[1]

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 (without merging) 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

PIN	$\delta\phi$	*Total clusters*	*Passing clusters*
Y^{2K}	3	306	85
Y^{11K}	3	480	178
Y^{78K}	5	936	130

5.2 MULIC clusters matching MIPS complexes by overlap and by containment

In most of our Y^{2K} 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 30 or more 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, three of the proteins in the cluster do not overlap with the complex. All three 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 306. 110 small clusters were merged into 14 large merged clusters. As shown, most of these merged clusters match by overlap known MIPS complexes. What is most interesting is the size of some merged clusters. One merged cluster is of size 79 and it matches by overlap the MIPS complex '550.1.149' of size 88 that is involved in RNA metabolism (Gavin et al., 2003). Another 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 finding large complexes.

Table 2 Pairs of MIPS protein complexes and Y^{2K} clusters that match by overlap and their overlapping proteins. 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

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. 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 ten proteins) and large complexes are relatively infrequent. Large clusters are more interesting than small clusters for biologists who want to confirm them in a lab, since they are likely to match large protein complexes.

Table 3 Numbers of merged and unmerged Y^{2K} clusters passing the size filter and matching a MIPS complex (by overlap or containment) after reducing the total number of clusters through merging. The number of clusters before merging was 306 of which 85 were passing and 45 were matching clusters

Total clusters after merging	*Merged clusters*	*Unmerged clusters*	*Passing merged clusters*	*Matching merged clusters*	*Passing unmerged clusters*	*Matching unmerged clusters*	*Matching rate for merged clusters* (%)
220	13	207	13	8	41	24	61.5
210	14	196	14	8	37	21	57.1
200	11	189	11	7	36	20	63.6
190	12	178	12	9	33	18	75
180	18	162	18	13	29	16	72.2
170	16	154	16	11	24	15	68.75
160	15	145	15	11	21	14	73.3
150	15	135	15	11	17	10	73.3
136	17	119	17	13	12	6	76.5

5.4 Results after treating objects as outliers

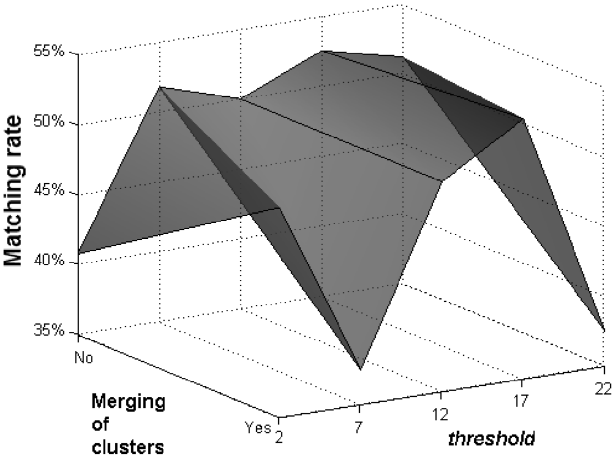
Objects are treated as outliers by setting the *threshold* for ϕ to a value less than the number of attributes m, as discussed in Section 4.3. When ϕ exceeds the maximum allowed value specified by *threshold*, any remaining objects are treated as outliers by placing them independently in clusters of 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 matching 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 Y^{2K} clusters passing the size filter and matching a MIPS complex (by overlap or containment) using various values of *threshold*

Threshold	*Total clusters*	*Passing clusters*	*Matching clusters*	*Matching rate* (%)
20	298	77	44	57.14
25	304	79	44	56
30	306	79	43	54.43
35	306	80	43	53.75
40	306	83	45	54.22

Figure 5 illustrates the matching rates for the Y^{78K} data set, for various values of *threshold* and both merging and not merging clusters. The value of $\delta\phi$ is 5. As shown, the highest matching rates are derived using a low value of *threshold* of 17. The matching rates for Y^{78K} are not very dissimilar from Y^{2K}, 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 Y^{78K} matching rates (by overlap or containment) using various values of *threshold* and both merging and not merging the clusters



5.5 Results for various values of $\delta\phi$

Table 5 shows the MULIC results for Y^{2K} using various values of $\delta\phi$ and without merging clusters. We notice that a value of $\delta\phi$ set to 3 results in more clusters matching complexes than other values. The reason why a $\delta\phi$ 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 $\delta\phi$ that is too large, on the other hand, decreases the matching rate and the quality of the results because many proteins are assigned to clusters to which they are not so similar.

Table 5 Numbers of Y^{2K} clusters passing the size filter and matching a MIPS complex (by overlap or containment) using various values of $\delta\varphi$

$\delta\phi$	*Total clusters*	*Passing clusters*	*Matching clusters*	*Matching rate* (%)
1	315	79	44	55.7
3	306	85	45	53
5	292	83	43	52
10	273	84	40	47.61
25	251	74	36	49
50	246	69	31	45
75	241	71	33	46.5
100	239	69	32	46.4

6 Discussion

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 4-40 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 less similar proteins. This is in accordance with a recent study (Dezso et al., 2003; Bu et al., 2003) in which protein complexes were discovered to feature centres 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 (2003) 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. King et al. (2004) generate a set of 28 clusters filtered by size, density and functional annotation, of which 23 match the MIPS protein complex database in the Y^{2K} yeast PIN of 2,000 interactions. Our matching 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 ($P_{\text{cluster}} = P_{\text{complex}} = 0.7$ and $P_{\text{contain}} = 0.9$) such that a cluster matches a complex only if there is a significant overlap. Table 6 shows that relaxing the matching criteria increases the number of clusters that match a known MIPS complex and the matching rate. Table 7 shows a comparison of the MULIC results with the results of the RNSC clustering algorithm. The RNSC results were evaluated using the same matching criteria as in our MULIC evaluation (King et al., 2004). Even with our strict matching 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 Y^{2K}). 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 down 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 Y^{2K} clusters matching a MIPS complex (by overlap or containment) increases. Since there are 85 passing clusters for Y^{2K}, the matching rate for Y^{2K} also increases. Clusters are *not* merged

Matching criteria	*Matching clusters*	*Matching rate* (%)
$P_{\text{cluster}} = P_{\text{complex}} = 0.7$, $P_{\text{contain}} = 0.9$	45	53
$P_{\text{cluster}} = P_{\text{complex}} = 0.5$, $P_{\text{contain}} = 0.7$	74	87
$P_{\text{cluster}} = P_{\text{complex}} = 0.3$, $P_{\text{contain}} = 0.5$	81	95.3

Table 7 The number of Y^{2K} clusters matching a MIPS complex (by overlap or containment) 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 match by overlap criteria of $P_{cluster} = P_{complex} = 0.7$ and match by containment criteria of $P_{contain} = 0.9$

	Number of Y^{2K} matching clusters	*Largest size of a cluster that matches a MIPS complex by overlap*
MULIC	45	MIPS complex '550.1.149' of size 88 matches MULIC merged cluster of size 79 by overlap. Their overlap is 44
RNSC	23	MIPS complex of size 29 matches RNSC cluster of size 17 by overlap. Their overlap is 10
k-Modes	18	MIPS complex of size 20 matches k-Modes cluster of size 15 by overlap. Their overlap is 10
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 and Cheeseman, 1995) to the same PIN data sets. We evaluated their results using the same matching criteria as in our MULIC evaluation. Table 7 summarises the results. *K*-Modes does not have the MULIC characteristics specific to PIN clustering (described in Section 4.4) 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 Meaning of Layered Cluster Structure

The multiple layer structure of the MULIC clusters reveals several things about the structures of the protein complexes that could not be identified with other algorithms. For clusters that match known MIPS complexes, the top-layer proteins (Layer 1) often have the highest connectivity to other proteins in the complex. In other words, top-layer proteins are often locally central points of connectivity for the matched complex. For example, the well-studied FKS1p (YLR342W) and FKS2p (YGR032W) proteins have a 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 synthesise beta-1,3-glucan, a major structural polymer of the cell wall in yeast. The drug caspofungin binds to FKS1p and FKS2p to disturb the interactions of the glucan synthase complex (Markovich, 2004; Reinoso-Martin et al., 2003). 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 Proteins in top layers of Y^{2K} clusters matching a known MIPS complex are more likely to be contained in the matched complex

Layers	*Percentage of proteins that are contained in the matched complex*
1–4	75
7–10	66
13–19	40

The multiple layer structure of the derived MULIC clusters can be useful in cases where few protein complexes are known for the PIN of an organism, such as fruitfly and worm. Lab experiments can initially focus on the proteins clustered in top layers. Later, proteins in lower cluster layers can guide the lab experiments on finding protein complexes. Table 8 shows that for the Y^{2K} clusters that match a MIPS complex (by overlap or containment) the proteins in top layers (1–4) are likely to be contained in the matched complexes. While the proteins in bottom layers (7–19) are less likely to be contained in the matched complexes. This table was derived by averaging the percentage of the proteins in various layers of matching clusters that are contained in the matched complex.

Figure 6 shows an example of a layered MULIC cluster. As shown, all of the proteins in the cluster interact with YPL082C. Proteins in the top layer have the same neighbourhoods, while proteins in the bottom layer have less similar neighbourhoods.

MULIC provides the capability to merge similar clusters, which can help to identify and capture more complex topological structures. When two clusters are merged, it usually means that some protein interacts with both clusters. Merging of clusters allows finding proteins the influence of which traverses across neighbourhoods (clusters). This may lead to a richer classification of proteins. Figure 7 shows how YOL115W can be classified as being related to two neighbourhoods, since YOL115W interacts with both YDR175C and YDR036C.

Figure 6 A MULIC cluster with 3 layers. Circles represent proteins and edges represent interacting partners

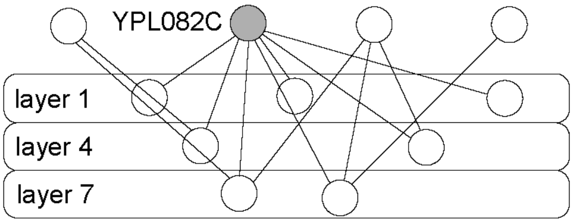
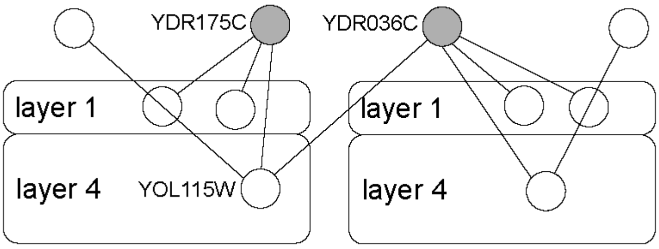


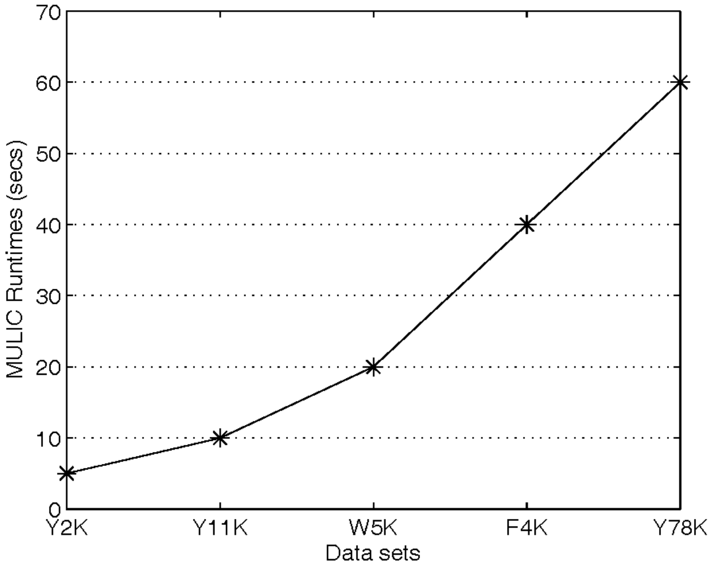
Figure 7 Two merged MULIC clusters



6.3 Complexity and Runtime

The worst case complexity of MULIC is $O(N^2)$, where N is the number of objects. Figure 8 shows MULIC runtimes on PINs of various sizes. The experiments were performed on a Sun Ultra 60 with 256 MB of memory and a 300 MHz processor.

Figure 8 MULIC runtimes on PIN data sets



The most costly test run was on Y^{78K}. The runtimes of MULIC are comparable to those of other algorithms, but MULIC can find more complexes and the cluster structure is more interesting for analysis. One reason for the rather low runtimes is that most objects are clustered during the initial iterations when the top cluster layers (e.g., 1–30) are created. Thus, relatively 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. 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 when $\varphi = m$.

Several optimisations can be implemented in the MULIC source code to significantly reduce the runtimes. A bitwise *OR* can be used to update a mode when a new protein is inserted in the cluster. A bitwise *AND* can be used to evaluate the similarity between a protein and a mode. Alternatively, MULIC can be given its input in market-basket format instead of square matrix format. In this case, a mode is a vector storing the set of all interacting partners of proteins that are cluster members. Updating a mode involves inserting in it any new interacting partners. Comparing a protein to a mode involves finding the overlap between the protein's interacting partners and the mode.

7 Conclusion and future work

We have proposed a clustering method for finding protein complexes in PINs. This approach finds proteins with 'similar' interaction patterns, i.e., proteins that interact with the same proteins. The main advantage of this method is that clusters consist of layers, where top layers are created first to contain proteins with very similar interaction sets – the similarity criterion is gradually relaxed at lower layers. This method does not require the number of clusters to be specified by the user – it returns as many coherent clusters as it can find. This method is effective for detecting proteins that are outliers. Moreover, it can find complexes of 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 significant pieces of the interactome and validate the potential complexes in the lab.

The cluster merging process can be used to merge similar clusters, potentially leading to finding 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 the uncertainty on the correctness of interactions. In many PIN data sets the interactions have annotations of high, medium or low confidence. The confidence annotations represent the expected rate of false positives, which depends on the experimental method used to derive the interactions. If the high confidence interactions are given a heavier weight in the clustering process, this may lead to improved complex finding. 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

The authors thank the reviewers for their valuable comments. The authors gratefully acknowledge the financial assistance of the National Science and Engineering Research Council (NSERC) and Ontario Graduate Scholarship (OGS).

References

Alfarano, T.B., Dewar, C.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. and Tyers, M. (2003) 'Systematic identification of protein complexes in saccharomyces cerevisiae by mass spectrometry', *Nature*, Vol. 415, No. 6868, pp.180–183.

Andreopoulos, B., An, A. and 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.

Amau, V., Mars, S. and Marin, I. (2005) 'Iterative cluster analysis of protein interaction data', *Bioinformatics*, Vol. 21, No. 3, pp.364–378.

Barabasi, A.L. and Oltvai, Z.N. (2004) 'Network biology: understanding the cell's functional organization', *Nature Reviews Genetics*, Vol. 5, pp.101–113.

Bader, G. and Hogue, C. (2003) 'An autormated method for finding molecular complexes in large protein interaction networks', *BMC Bioinformatics*, Vol. 4, No. 2.

Batagelj, V. and Zavernik, M. (2001) 'Cores decomposition of networks', *Recent Trends in Graph Theory, Algebraic Combinatorics, and Graph Algorithms*, September 24–27, Bled, Slovenia.

Bu, D., Zhao, Y., Cai, L., Xue, H., Zhu, X., Lu, H., Zhang, J., Sun, S., Ling, L., Zhang, N., Li, G. and Chen, R. (2003) 'Topological structure analysis of the protein-protein interaction network in budding yeast', *Nucleic Acids Research*, Vol. 31, No. 9, pp.2443–2450.

Dezso, Z., Oltvai, Z.N. and Barabási, A-L. (2003) 'Bioinformatics analysis of experimentally determined protein complexes in the yeast Saccharomyces cerevisiae', *Genome Res.*, Vol. 13, pp.2450–2454.

Ding, C., He, X., Meraz, R. and Holbrook, S. (2004) 'Multi-protein complex data clustering for detecting protein interactions and functional organizations', *Interface 2004: Computational Biology and Bioinformatics*, Baltimore, MD, May 26–29.

Dunn, R., Dudbridge, F. and Sanderson, C.M. (2005) 'The use of edge-betweenness clustering to investigate biological function in protein interaction networks', *BMC Bioinformatics*, March 1, Vol. 6, No. 1, p.39.

Gavin, A.C., Bosche, M., Krause, R., Grandi, P., Marzioch, M. and Bauer, A. (2003) 'A functional organization of the yeast proteome by systematic analysis of protein complexes', *Nature*, Vol. 415, No. 6868, pp.141–147.

Giot, L., Bader, J.S., Brouwer, C., Chaudhuri, A., Kuang, B., Li, Y., Hao, Y.L., Ooi, C.E., Godwin, B., Vitols, E., Vijayadamodar, G., Pochart, P., Machineni, H., Welsh, M., Kong, Y., Zerhusen, B., Malcolm, R., Varrone, Z., Collis, A., Minto, M., Burgess, S., McDaniel, L., Stimpson, E., Spriggs, F., Williams, J., Neurath, K., Ioime, N., Agee, M., Voss, E., Furtak, K., Renzulli, R., Aanensen, N., Carrolla, S., Bickelhaupt, E., Lazovatsky, Y., DaSilva, A., Zhong, J., Stanyon, C.A., Finley Jr., R.L., White, K.P., Braverman, M., Jarvie, T., Gold, S., Leach, M., Knight, J., Shimkets, R.A., McKenna, M.P., Chant, J. and Rothberg, J.M. (2003) 'A protein interaction map of Drosophila melanogaster', *Science*, Vol. 302, No. 5651, pp.1727–1736.

Glover, E. (1989) 'Tabu search, part I', *ORSA Journal on Computing*, Vol. 1, No. 3, pp.190–206.

Hartuv, E. and Shamir, R. (2000) 'A clustering algorithm based on graph connectivity', *Information Processing Letters*, Vol. 76, Nos. 4–6, pp.175–181.

Huang, Z. (1998) 'Extensions to the *k*-means algorithm for clustering large data sets with categorical values', *Data Mining and Knowledge Discovery*, Vol. 2, No. 3, pp.283–304.

King, A.D., Prulj, N. and Jurisica, I. (2004) 'Protein complex prediction via cost-based clustering', *Bioinformatics*, Vol. 20, No. 3, pp.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., Berriz, G.F., Jacotot, L., Vaglio, 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., Cusick, M.E., Roth, F.P., Hill, D.E. and Vidal, M. (2004) 'A map of the interactome network of the metazoan C. elegans', *Science*, Vol. 303, No. 5657, pp.540–543.

Markovich, S. (2004) 'Genomic approach to identification of mutations affecting caspofungin susceptibility in Saccharomyces cerevisiae', *Antimicrob Agents Chemother*, Vol. 48, No. 10, pp.3871–3876.

Mewes, H.W., Frishman, D., Guldener, U., Mannhaupt, G., Mayer, K., Mokrejs, M., Morgenstern, B., Munsterkotter, M., Rudd, S. and Weil, B. (2002) 'Mips: a database for genornes and protein sequences', *Nucleic Acids Research*, Vol. 30, No. 1, pp.31–34.

Newman, M.E.J. and Girvan, M. (2004) 'Finding and evaluating community structure in networks', *Physical Review E*, Vol. 69, No. 2, p.026113.

Reinoso-Martin, C., Schuller, C., Schuetzer-Muehlbauer, M. and Kuchler, K. (2003) 'The yeast protein kinase C cell integrity pathway mediates tolerance to the antifungal drug caspofungin through activation of Slt2p mitogen-activated proteinkinase signaling', *Eukaryot Cell*, Vol. 2, No. 6, pp.1200–1210, http://www.pdg.cnb.uam.es/UniPub/iHOP/gs/31559.html.

Stutz, J. and Cheeseman, P. (1995) 'Bayesian classification (AutoClass): theory and results', *Advances in Knowledge Discovery and Data Mining*, AAAI Press, Menlo Park, CA, pp.153–180.

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. and Rothberg, J.M. (2000) 'A comprehensive analysis of protein-protein interactions in saccharomyces cerevisiae', *Nature*, Vol. 403, No. 6770, pp.623–627.

von Mering, C., Krause, R., Snel, B., Cornell, M., Oliver, S.G., Fields, S. and Bork, P. (2002) 'Comparative assessment of large-scale data sets of protein-protein interactions', *Nature*, Vol. 417, No. 6887, pp.399–403.

West, D.B. (2001) *Introduction to Graph Theory*, 2nd ed., Prentice-Hall, Upper Saddle River, NJ.

Yang, Q. and 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.

Note

[1] Supplementary data: http://www.cs.yorku.ca/~billa/MULICppi05/.