Enumeration of Condition-Dependent Dense Modules in Protein Interaction Networks

Supplementary Text

E. Georgii, S. Dietmann, T. Uno, P. Pagel, K. Tsuda

We present a method for systematic module discovery in biological networks. In the first section, we describe our algorithmic approach in detail. In the second section, we give some further information on our experiments and additionally discuss the results of a comparative analysis of module finding methods on the human interaction network.

1 Method

1.1 Problem Definition

We propose an enumerative approach to discover modules in protein-protein interaction networks. Formally, we tackle the problem by extracting all densely connected node sets from a weighted undirected graph. The nodes of this graph correspond to proteins and edge weights indicate the confidence for an interaction between two proteins. In section 2.1 we will describe how the edge weights were computed in this study.

Let us consider an undirected weighted graph with node set V. Let $W = (w_{ij})_{i,j \in V}$ be the corresponding matrix representation, containing the weights of the given edges and zero entries otherwise. In the following we assume $w_{ij} \leq 1$. Then the *density* of a node subset $U \subset V$ with respect to W is defined as

$$\rho_W(U) = \frac{\sum_{i,j \in U, i<j} w_{ij}}{|U|(|U|-1)/2}\,. \tag{1}$$

For non-negative edge weights, the value of $\rho_W(U)$ lies between 0 and 1. In the case of unweighted graphs, W is a binary matrix and the density corresponds to the actual number of edges within U, divided by the possible number of edges. We define $\rho_W(\emptyset) = 1$ and $\rho_W(U) = 1$ for $|U| = 1$. For convenience, we usually write $\rho(U)$ instead of $\rho_W(U)$, as W is given as input and remains unchanged throughout the method.

Now we can define the problem of *dense module enumeration* as follows.

Definition 1 (Dense Module Enumeration)**.** *Given a graph with node set V and weight matrix W, and a density threshold $\theta > 0$, find all modules $U \subset V$ with $\rho_W(U) \geq \theta$.*

Here we use the term *module* to denote the induced subgraph corresponding to a node set.

1.2 Enumeration Algorithm

A canonical strategy to solve set enumeration tasks is to start with the empty set and then iteratively form larger sets by adding one element at a time; if it is evident that no further solutions can be derived from a certain set, the process of extension is stopped, i.e. unnecessary parts of the search space are pruned. One famous example is itemset mining [1]. It solves the problem of finding all itemsets which occur more than k times in a transaction database. In this context, pruning is based on the fact that subsets have at least the same occurrence frequency as their superset. This property is called *anti-monotonicity*:

Definition 2 (Anti-Monotonicity)**.** *A criterion f is anti-monotonic, if $f(U') \geq f(U)$ for all $U' \subset U$.*

Clearly, this property does not hold true for our density criterion. More generally, given the density of some module, it is not possible to make statements about the density of submodules or supermodules. For illustration, consider the modules in Fig. 1. Submodules of a module can have larger density as well as lower density than the module itself. Likewise, there can exist supermodules with higher density as well as supermodules with lower density.

However, it is possible to traverse the search space in a way that allows for straightforward pruning. In fact, we define a tree-based parent-child relationship between modules such that along each path from the root to a leaf, the module size is increasing, whereas the module density is monotonically decreasing. Technically, our algorithm adopts the reverse search paradigm [2]: in each iteration, we generate all direct supersets of the current module and select those which are indeed its children. Due to the monotonicity guarantee in our search tree, only children that fulfill the density criterion have to be further processed. To describe our approach in more detail, we need the definition of *weighted degree*.

Definition 3 (Weighted Degree)**.** *For $u \in U \subset V$, the weighted degree of u with respect to U is defined as*

$$\deg_U(u) = \sum_{j \in U, j \neq u} w_{uj} \tag{2}$$

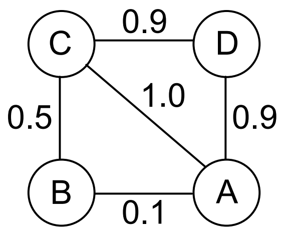
Note that the weighted degree depends on the given weight matrix W. As W remains fixed during the whole algorithm, we omit an explicit reference to W in our notation.

The following lemma states a fundamental property of ρ.

Lemma 1. *Let $v \in U$ be a node with minimum weighted degree in U, i.e. $\forall u \in U : \quad \deg_U(u) \geq \deg_U(v)$. Then, $\rho(U \setminus \{v\}) \geq \rho(U)$.*

Proof.

$$\begin{aligned}
&\rho(U \setminus \{v\}) - \rho(U) \\
&= \frac{\sum_{i,j \in U \setminus \{v\}, i<j} w_{ij}}{(|U|-1)(|U|-2)/2} - \frac{\sum_{i,j \in U, i<j} w_{ij}}{|U|(|U|-1)/2} \\
&= \frac{\frac{1}{2}\sum_{u \in U \setminus \{v\}} \deg_{U \setminus \{v\}}(u)}{(|U|-1)(|U|-2)/2} - \frac{\frac{1}{2}\sum_{u \in U} \deg_U(u)}{|U|(|U|-1)/2} \\
&= \frac{\frac{1}{2}\left(\sum_{u \in U} \deg_U(u) - 2 \cdot \deg_U(v)\right)}{(|U|-1)(|U|-2)/2} - \frac{\frac{1}{2}\sum_{u \in U} \deg_U(u)}{|U|(|U|-1)/2} \\
&= \frac{\frac{1}{2}\sum_{u \in U} \deg_U(u) - \deg_U(v) - \frac{\frac{1}{2}\sum_{u \in U} \deg_U(u)}{|U|(|U|-1)/2}\left(((|U|-1)(|U|-2)/2\right)}{(|U|-1)(|U|-2)/2} \\
&= \frac{\frac{1}{2}\sum_{u \in U} \deg_U(u) - \deg_U(v) - \frac{1}{2}\sum_{u \in U} \deg_U(u) \cdot \frac{|U|-2}{|U|}}{(|U|-1)(|U|-2)/2} \\
&= \frac{\frac{1}{2}\sum_{u \in U} \deg_U(u)\left(1 - \frac{|U|-2}{|U|}\right) - \deg_U(v)}{(|U|-1)(|U|-2)/2} \\
&= \frac{\frac{1}{2}\sum_{u \in U} \deg_U(u)\frac{2}{|U|} - \deg_U(v)}{(|U|-1)(|U|-2)/2} \\
&= \frac{\frac{1}{|U|}\sum_{u \in U} \deg_U(u) - \deg_U(v)}{(|U|-1)(|U|-2)/2} \\
&\geq \frac{\frac{1}{|U|}\left(|U| \cdot \deg_U(v)\right) - \deg_U(v)}{(|U|-1)(|U|-2)/2} \\
&= 0
\end{aligned}$$



Module	Density
{A,B,C,D}	0.57
{A,B,D}	0.33
{A,C,D}	0.93
{A,D}	0.90

Figure 1: Densities of some modules in an example graph, showing that the density criterion is neither anti-monotonic nor monotonic.

The inequality holds because of the assumption that v is a node with minimum weighted degree in U, i.e. $\deg_U(v) \leq \deg_U(u)$ for $u \in U$. Altogether it follows that $\rho(U \setminus \{v\}) \geq \rho(U)$. $\square$

This property is the key for defining the search tree. In order to fix a unique parent for each module, we introduce a function ord, which defines a strict total ordering on the nodes,i.e. for each node pair u, v with $u \neq v$ either $\mathrm{ord}(u) > \mathrm{ord}(v)$ or $\mathrm{ord}(u) < \mathrm{ord}(v)$ holds. With this, we define the *parent-child relationship* for modules as follows.

Definition 4 (Parent-Child Relationship)**.** *Let U be a module and $v \in V \setminus U$. $U^* := U \cup \{v\}$ is a child of U if and only if*

$$\forall u \in U: \quad (\deg_{U^*}(v) < \deg_{U^*}(u)) \vee (\deg_{U^*}(v) = \deg_{U^*}(u) \wedge \mathrm{ord}(v) < \mathrm{ord}(u)) \ .$$

In other words, we obtain the parent of a certain module by removing the smallest of the nodes with minimum weighted degree.

Obviously, we cannot directly derive all children of a given module U with size k. Instead, we have to check for all possible extended modules of size $k+1$ whether U is their parent or not (reverse search principle). From the lemma we know that each module has a smaller or equal density than its parent. That means, if the density check fails for a certain module, none of its descendants can satisfy it. Based on this, the DME algorithm starts with the empty set and recursively generates children as long as the density threshold is not violated (Alg. 1). In other words, it performs a depth-first traversal of the search tree, pruning whenever possible and thereby avoiding the generation of non-dense descendants. Since we investigated all dense modules which have the empty set as ancestor, the result set is complete. Fig. 2 shows the module tree traversed in an example run of DME.

In a straightforward implementation, we use an array of length $|V|$ where we store the weighted degree of each node with respect to the current module U; more precisely, it contains the value $\deg_U(u)$ for nodes $u \in U$, and the value $\deg_{U \cup \{v\}}(v)$ for nodes $v \in V \setminus U$. Update of this array after having added one node requires $O(|V|)$ operations using the matrix W. In each iteration of the algorithm, we first check for each node v not included in the current module U whether the density criterion holds for $U \cup \{v\}$ (if we compute the density incrementally using $\deg_{U \cup \{v\}}(v)$, a single density check takes constant time). If that is the case, we check the conditions of $U \cup \{v\}$ being a child of U according to definition 4 (this requires a temporary update of $|U|$ array entries; if the entries of W can be accessed directly, this corresponds to $O(|U|)$ operations).[1] Therefore, each iteration of the algorithm takes at most $O(|V| + |V \setminus U| \cdot |U|)$ time. This is also an upper bound for the longest computation time between two outputs (also called *delay*). To see this, we introduce a small modification of the algorithm according to the odd-even output method [6]. If the recursion depth is odd, we output the solution before the recursive calls; if the recursion depth is even, we do it afterwards. Then any three consecutive iterations of the algorithm yield at least one solution. Hence, DME is a polynomial-delay algorithm. The delay is the common time complexity measure for enumeration algorithms because by the definition of enumeration tasks, there might exist an exponential number of solutions.

By changing the density threshold, the user can regulate the size of the output. Also note that the computation can easily be parallelized. Finally, dense modules that are subsets of other solutions are not so informative; we call them non-maximal. While these redundant results could be eliminated by checking for each new module all previous solutions, it is possible to identify *locally maximal* modules without requiring additional computation or storage, as shown in Alg. 1. A module U is locally maximal if and only if for all $v \in V \setminus U$, $U \cup \{v\}$ does not satisfy the minimum density threshold. Although a module with this property could still be non-maximal, it happens rarely in practice.

[1]For weight matrices containing only non-negative values, an additional speed-up is possible by keeping track of the minimum weighted degree value in U, d^*_U, and applying some special rules: If $\deg_{U \cup \{v\}}(v) < d^*_U$, we can infer directly that $U \cup \{v\}$ is a child of U. If $\deg_{U \cup \{v\}}(v) > d^*_U + 1$, then $U \cup \{v\}$ cannot be a child of U (assuming a maximum weight of 1).

Algorithm 1 Dense module enumeration for node set V, weight matrix W, and minimum density θ. U represents the current module. DME is called with $U = \emptyset$.

```
1:  DME $(V, W, \theta, U)$ :
2:     locallyMaximal = true
3:     for each $v \in V \setminus U$ do
4:        if $\rho_W(U \cup \{v\}) \geq \theta$ then
5:           locallyMaximal = false
6:           if $U \cup \{v\}$ is child of $U$ then
7:              DME $(V, W, \theta, U \cup \{v\})$
8:           end if
9:        end if
10:    end for
11:    if locallyMaximal then
12:       output $U$
13:    end if
```

1.3 Constraint Integration

Often the user is interested in modules that have certain coherency properties with respect to side information from additional data sources. We here consider the case that discrete profiles are attached to each node. For proteins, the profiles could for instance indicate presence or absence under certain cellular conditions. In this context, a useful requirement would be that all proteins in a module share the same profile across a subset of conditions. This can be formalized using minimum frequency constraints.

Definition 5. *Let V be again the node set of our graph and $U \subset V$ a module. Let $M = (m_{ij})_{i \in V, j \in C}$ be a side information matrix, where C denotes a set of conditions. The entries m_{ij} take values from a set of discrete states S, e.g. $S = \{0, 1\}$.*

Given a state $s \in S$, a condition $j \in C$ is said to be s-consistent with respect to U, if $m_{ij} = s$ for all $i \in U$. The number of s-consistent conditions is denoted by $f_s(U)$.

To control the consistency of module profiles, we can fix minimum thresholds for the frequencies $f_s(U)$ of the different states s. Without loss of generality, let us consider the case of binary side information, i.e. $S = \{0, 1\}$. Then the constrained dense module enumeration task is described as follows.

Definition 6. *Given a graph with node set V and weight matrix W, a binary profile matrix M, a density threshold $\theta > 0$, and frequency thresholds*

n_1 and n_0, find all modules $U \subset V$ such that $\rho(U) \geq \theta$, $f_1(U) \geq n_1$, and $f_0(U) \geq n_0$.

To integrate the additional frequency constraints into the search, the anti-monotonicity property of $f_s(U)$ is crucial.

Lemma 2. *$U' \subset U \Rightarrow f_s(U') \geq f_s(U)$.*

Proof. Each column that is s-consistent with respect to U is also s-consistent with respect to U'. $\square$

In other words, module extension either reduces the frequency of consistent columns or leaves it unchanged. This is illustrated in Fig. 3. When extending the module {A,B,C,D,E} by node F, the number of consistently green columns (f_1) shrinks, whereas the number of consistently black columns (f_0) stays equal. To determine the frequencies f_s for {A,B,C,D,E,F}, only the columns that are s-consistent with respect to {A,B,C,D,E} need to be considered.

The lemma implies that if a module does not satisfy the frequency constraints, no extension of it can satisfy them. Thus, we can use the frequency constraints as additional pruning criteria. That means we simply add them as conditions to line 4 of Alg. 1. By this, the search is guided directly to the modules of interest.

2 Experiments

2.1 Interaction Weights

One main challenge in the analysis of protein interaction networks are false positive edges, i.e. interactions that arise from experimental artifacts or measurement errors. To deal with this, we computed edge weights based on the experimental evidence. A large variety of scoring schemes are conceivable; here, we followed the method in [4]. For that purpose, we first determined for each individual interaction the set of sources supporting it. The MPact database reports for each measured interaction the set of experimental techniques. Also in the IntAct, MINT, DIP, and BIND databases, the corresponding experimental methods are listed, following the PSI-MI standard. In these cases, we considered each experimental technique as separate source. Further, the HPRD data set and the core data sets from Gavin [3] and Krogan [5] were labeled as individual sources.

Given a set of sources as evidence for a certain interaction, we estimate a reliability score using gold standard sets of correct and incorrect protein pairs. For the gold standard set of positive examples, we collected protein pairs that share the same MIPS functional category[2]. To avoid introducing a bias in our precision-recall analysis, we excluded the pairs which belong to MIPS reference complexes used for the evaluation. So the positive set is not specifically tailored to currently known protein complexes, but rather represents functional relatedness in a more general sense. Nevertheless, our module analysis reproduces reference complexes with high accuracy (see main paper). For the negative set, we used proteins with different subcellular localization as given in the Gene Ontology database[3]. To score a specific set of sources, we selected all interactions being associated with exactly this source set. We then computed the true positive rate and the false positive rate for this interaction set with respect to the gold standard sets. More precisely, the true positive rate is the fraction of positive pairs covered by the interaction set, and the false positive rate is the fraction of negative pairs covered by it. The ratio between the true positive rate and the false positive rate is assigned as score to the corrresponding interactions.

If there did not exist any experimental support for a protein pair, we gave a score of 0. Before scaling the scores to the range from 0 to 1, we truncated the score distribution as follows. To avoid that a few large positive outliers dominate the analysis, we fixed $t = 2$ as an upper threshold for reliability scores, i.e. all ratios above or equal to that value were considered as highly reliable and received the top score of 2 (11% of the yeast interactions). Equivalently, interactions with ratio scores below $1/t = 0.5$ were considered as spurious and discarded (set to zero). Now we normalized all scores with the maximum score value t (in our case 2), obtaining the final interaction weights between 0 and 1. For example, a source set producing equal fractions of false and correct pairs would yield a ratio of 1 and final interaction weights of 0.5. The resulting network for yeast consisted of 3559 nodes with 14212 non-zero interactions having an average weight of 0.67. The human network contained 9371 nodes and 32048 non-zero interactions having an average weight of 0.47.

2.2 Comparative Analysis for Yeast

This section contains some add-ons to the comparative analysis of the yeast network described in section 3.2 of the main paper.

[2]`http://mips.gsf.de/projects/funcat`

[3]`http://www.geneontology.org/`

2.2.1 Parameter Setting

For each method, we tested a wide range of parameters and selected the configuration with the largest area under the precision-recall curve for the ranked results. For DME, we varied the density threshold from 1.0 to 0.955 using decrements of 0.005. As there was a large difference in the number of solutions between 0.96 and 0.955, we further analyzed this range using decrements of 0.001. The best result was achieved at 0.957. Clique and CPM take as parameter a minimum edge weight threshold to select the edges which are used for computing the cliques. We varied this threshold from 1 to 0.35, using decrements of 0.05 (optimal setting for Clique: 0.4). In addition, CPM has an integer parameter k to control the clique size and the required node overlap for joining cliques, which was tested in the default range between 3 and the maximum clique size (optimal setting for CPM: edge selection threshold 0.9, k=9). This parameter k also exists for CPMw, but instead of preselecting edges, CPMw expects a threshold for the geometric mean of the edge weights in a clique. Only cliques satisfying this threshold are further processed. We tried the same thresholds as for DME; k was tested from 3 to 7, as the filtering step gets very expensive for higher values. We obtained the best results for k=6 and clique selection threshold 0.97. For MCL, there exist two main parameters affecting the cluster granularity: the inflation parameter, which we varied from 1.5 to 8 using increments of 0.5, and the centering, varied between 1 and 5. Furthermore, we set the parameter to retain potentially generated cluster overlaps. Here, the best result was obtained for inflation 3.0 and centering 2.0.

2.2.2 Overlap Precision-Recall Analysis

DME and Clique produced heavily overlapping modules, that means a large number of interactions were shared by several modules (see Table 1 in the main paper). To evaluate the accuracy of interactions in module overlaps, we sorted them according to the number of modules in which they occur (in descending order). Then we computed precision and recall values with respect to the MIPS protein complexes. Figure 4 shows the resulting precision-recall curves. Remarkably, the accuracy of DME interactions clearly increases with the number of modules in which they occur. However, this is not the case for Clique: while middle-ranked interactions are more accurate than low ranked interactions, the top interactions are less reliable. The reason is that Clique modules are solely based on the topology, whereas DME respects edge weights and therefore mainly reuses dense core modules, i.e. node sets that are densely connected by high-weight edges.

2.3 Comparative Analysis for Human

Similarly to the experiments on the yeast interaction network we described in section 3.2 in the main paper, we compared different module detection methods on the human interaction network. For validation, we used the set of human complexes published in [7].

2.3.1 Parameter Setting

We considered the same five module detection methods as in the yeast experiments: DME, Clique, CPM, CPMw, and MCL. For each of them, we tested a range of parameters. In each run, we sorted the results and calculated the precision-recall curve exactly as in the case of yeast. Finally, we selected for each method the parameter configuration which yielded the highest value for the area under the precision-recall curve.

For DME, we checked density thresholds between 1.0 and 0.8 in decrements of 0.01. The best results were obtained with 0.94. Like for the yeast data, the edge selection threshold of Clique and CPM was varied from 1.0 to 0.35 and the clique size parameter k of CPM was checked in the whole range between 3 and the maximum detected clique size. Clique was optimal for the edge selection threshold 0.35, and CPM was optimal for the edge selection threshold 0.35 and k=6. In the case of CPMw, the clique selection threshold was set to the same values as the DME density threshold (from 1.0 to 0.8 in decrements of 0.01) and k was varied from 3 to 7. Here, the optimal configuration was achieved for a clique selection threshold of 0.81 and k=6. Finally, the MCL parameters were tested in the same range as for the yeast experiments, the inflation parameter from 1.5 to 8 using increments of 0.5 (best value: 2.0) and the centering parameter from 1.0 to 5.0 (best value: 3.0).

2.3.2 Results

Figure 5 shows the precision-recall curves we obtained for the different module detection methods. In general, the performance is much worse than on the yeast data, irrespective of the chosen method. This can be explained by the fact that the human data are less comprehensive and, consequently, known complexes are less exposed with respect to the density criterion. DME has a high peak in the beginning, but otherwise its precision up to a recall value around 0.04 is comparable to CPM and CPMw. Interestingly, CPM and CPMw achieve this by producing very few modules, one of which is very large, whereas DME detects many small modules and therefore captures more diverse known complexes (see also Table 1). Clique is always below DME, but for higher recall values, DME is significantly outperformed by MCL, which also yields the largest area under the curve (see Table 1). Again, this is an effect of the data sparseness: one cannot gain much accuracy by explicitly considering the module density, therefore partitioning methods, which take the global network structure into account, are advantageous.

In Table 1, we collected various statistics for the results of the different methods, like in the yeast analysis. MCL required the longest computation time. To determine the number of distinct modules for each method, strongly overlapping modules were grouped together. More precisely, a module was added to a certain group if it matched another module of the group, according to the matching criterion defined in the main paper. DME modules produced the highest number of matches to distinct known complexes, closely followed by Clique. With respect to partially recovered complexes (i.e. complexes from which at least one protein pair was contained in a module), Clique is leading, followed by MCL and DME. Moreover, we performed for each distinct module a GO enrichment analysis using the TANGO tool [8] (for each group of overlapping modules, we took the top-ranking module). The number of enriched modules was generally very low: CPM and CPMw produced only very few modules, and the other methods predicted mainly modules which are too small to satisfy the enrichment threshold, irrespective of their purity. Among the top-50 distinct modules, approximately one half is enriched. Regarding the number of protein pairs in the overlaps between different modules, the results are negligible for all methods except DME and Clique. Although the number is much higher for Clique, the area under the precision-recall curve is not that different, which means that overlapping interactions of DME are more accurate. However, the accuracy is generally not higher than for the total set of predicted interactions (at comparable recall levels). A reason for this is that the overlaps tend to be very small. Finally, the overlapping modules of Clique and DME also recovered a large number of overlaps between known complexes (see section 3.2 of the main paper for details on our measure).

Table 1: Module statistics of the comparative analysis on the human network.

	DME	Clique	CPM	CPMw	MCL
# distinct modules	2321	2982	9	3	2092
average size of distinct modules	2	3	12	19	3
# raw modules	3005	3420	9	3	2093
average size of raw modules	3	3	12	19	3
# matched complexes	136	133	18	1	113
average complex size	3	4	11	32	4
# partially recovered complexes	378	403	77	32	387
# predicted interactions	3925	7055	1131	1026	6616
area under prec.-rec. curve (AUC)	0.025	0.021	0.019	0.015	0.030
# enriched distinct modules	24	58	8	3	64
# enriched among top-50	23	25	-	-	21
# overlapping proteins	970	1225	3	0	103
# overlapping interactions	428	2405	3	0	7
AUC for overlapping interactions	0.0046	0.0055	0.0002	-	0.0000
# recovered complex overlaps	942	1618	6	0	0
running time (in seconds)	24	2	2	12	107

References

[1] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *VLDB '94: Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[2] David Avis and Komei Fukuda. Reverse search for enumeration. *Discrete Appl. Math.*, 65(1-3):21–46, 1996.

[3] A. C. Gavin, P. Aloy, P. Grandi, R. Krause, M. Boesche, M. Marzioch, C. Rau, L. J. Jensen, S. Bastuck, B. Dumpelfeld, A. Edelmann, M. A. Heurtier, V. Hoffman, C. Hoefert, K. Klein, M. Hudak, A. M. Michon, M. Schelder, M. Schirle, M. Remor, T. Rudi, S. Hooper, A. Bauer, T. Bouwmeester, G. Casari, G. Drewes, G. Neubauer, J. M. Rick, B. Kuster, P. Bork, R. B. Russell, and G. Superti-Furga. Proteome survey reveals modularity of the yeast cell machinery. *Nature*, 440(7084):631–6, 2006.

[4] Ronald Jansen, Haiyuan Yu, Dov Greenbaum, Yuval Kluger, Nevan J Krogan, Sambath Chung, Andrew Emili, Michael Snyder, Jack F Greenblatt, and Mark Gerstein. A Bayesian networks approach for predicting

protein-protein interactions from genomic data. *Science*, 302(5644):449–53, 2003.

[5] N. J. Krogan, G. Cagney, H. Yu, G. Zhong, X. Guo, A. Ignatchenko, J. Li, S. Pu, N. Datta, A. P. Tikuisis, T. Punna, J. M. Peregrin-Alvarez, M. Shales, X. Zhang, M. Davey, M. D. Robinson, A. Paccanaro, J. E. Bray, A. Sheung, B. Beattie, D. P. Richards, V. Canadien, A. Lalev, F. Mena, P. Wong, A. Starostine, M. M. Canete, J. Vlasblom, S. Wu, C. Orsi, S. R. Collins, S. Chandran, R. Haw, J. J. Rilstone, K. Gandi, N. J. Thompson, G. Musso, P. St Onge, S. Ghanny, M. H. Lam, G. Butland, A. M. Altaf-Ul, S. Kanaya, A. Shilatifard, E. O'Shea, J. S. Weissman, C. J. Ingles, T. R. Hughes, J. Parkinson, M. Gerstein, S. J. Wodak, A. Emili, and J. F. Greenblatt. Global landscape of protein complexes in the yeast saccharomyces cerevisiae. *Nature*, 440(7084):637–43, 2006.

[6] Shin-Ichi Nakano and Takeaki Uno. Constant time generation of trees with specified diameter. In *LNCS 3353*, pages 33–45, 2004.

[7] Andreas Ruepp, Barbara Brauner, Irmtraud Dunger-Kaltenbach, Goar Frishman, Corinna Montrone, Michael Stransky, Brigitte Waegele, Thorsten Schmidt, Octave Noubibou Doudieu, Volker Stumpflen, and H. Werner Mewes. CORUM: the comprehensive resource of mammalian protein complexes. *Nucl. Acids Res.*, 36(suppl_1):D646–650, 2008.

[8] Ron Shamir, Adi Maron-Katz, Amos Tanay, Chaim Linhart, Israel Steinfeld, Roded Sharan, Yosef Shiloh, and Ran Elkon. Expander - an integrative program suite for microarray data analysis. *BMC Bioinformatics*, 6(1):232, 2005.

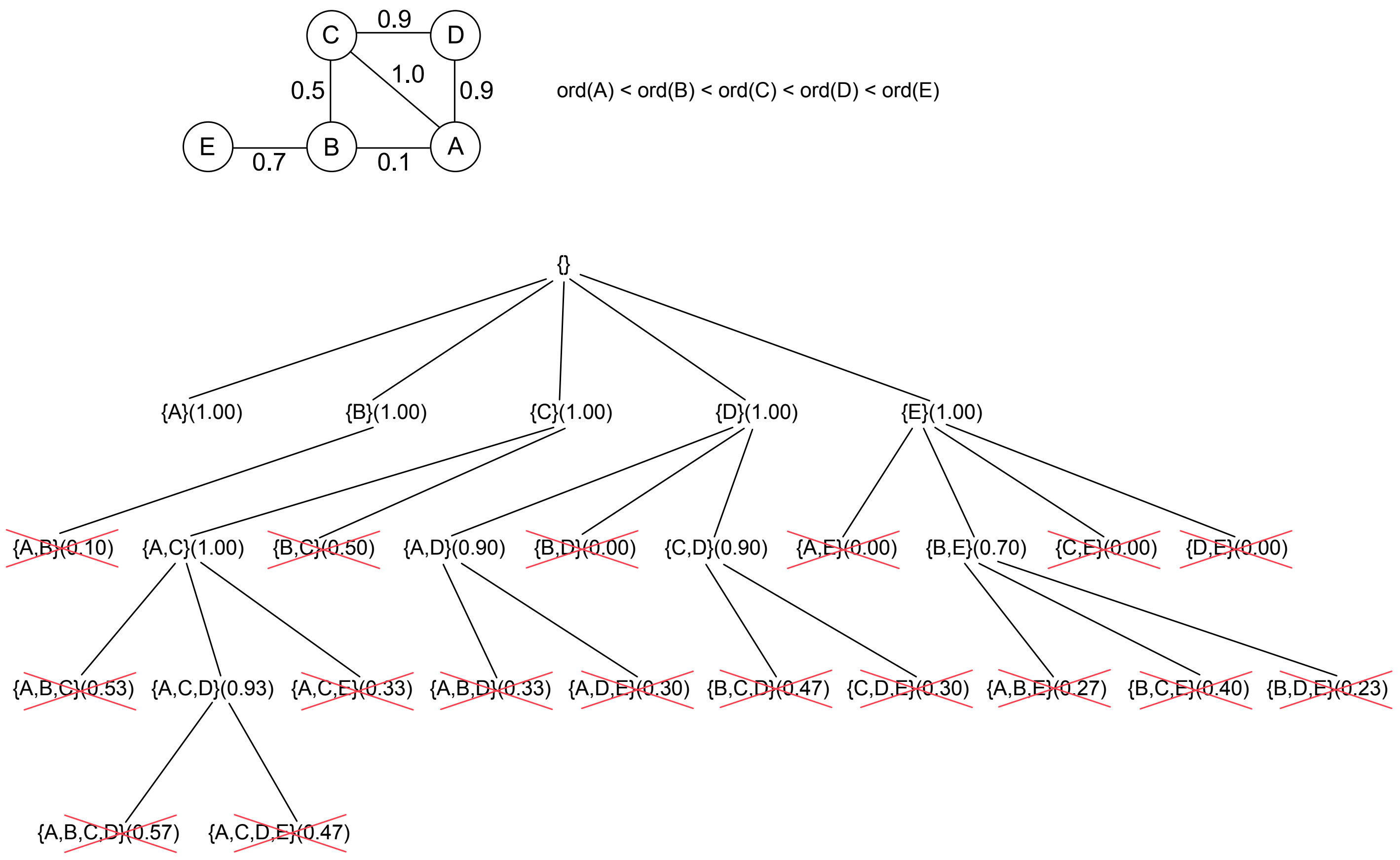


Figure 2: Module tree traversed by DME for the given input graph and density threshold 0.6. Density values are shown in parentheses. The red crosses mark the nodes where the tree is pruned.

Module	f_1	f_0
ABCDE	2	2
ABCDEF	1	2

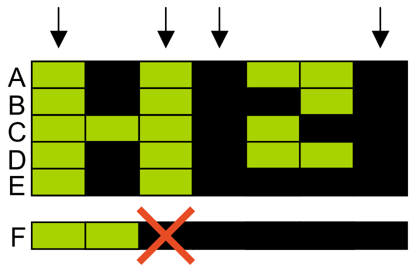


Figure 3: Frequency of consistent columns before and after module extension (f_1 is the frequency of consistently green columns, f_0 the frequency of consistently black columns).

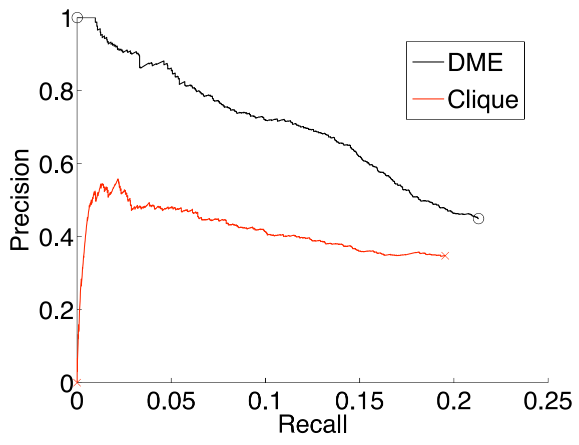


Figure 4: Precision-recall curves for overlapping interactions.

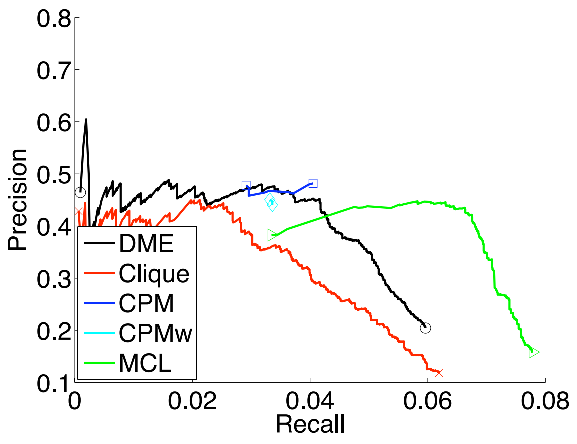


Figure 5: Comparative precision-recall analysis for the human interaction network.