BIOINFORMATICS

Vol. 23 ISMB/ECCB 2007, pages i577–i586
doi:10.1093/bioinformatics/btm227

A graph-based approach to systematically reconstruct human transcriptional regulatory modules

Xifeng Yan[1], Michael R. Mehan[2,†], Yu Huang[2], Michael S. Waterman[2], Philip S. Yu[1] and Xianghong Jasmine Zhou[2,*]

[1]IBM T. J. Watson Research Center, Hawthorne NY and [2]Program in Molecular and Computational Biology, University of Southern California, Los Angeles CA, USA

ABSTRACT

Motivation: A major challenge in studying gene regulation is to systematically reconstruct transcription regulatory modules, which are defined as sets of genes that are regulated by a common set of transcription factors. A commonly used approach for transcription module reconstruction is to derive coexpression clusters from a microarray dataset. However, such results often contain false positives because genes from many transcription modules may be simultaneously perturbed upon a given type of conditions. In this study, we propose and validate that genes, which form a coexpression cluster in multiple microarray datasets across diverse conditions, are more likely to form a transcription module. However, identifying genes coexpressed in a subset of many microarray datasets is not a trivial computational problem.

Results: We propose a graph-based data-mining approach to efficiently and systematically identify frequent coexpression clusters. Given m microarray datasets, we model each microarray dataset as a coexpression graph, and search for vertex sets which are frequently densely connected across $\lceil \theta m \rceil$ datasets ($0 \leq \theta \leq 1$). For this novel graph-mining problem, we designed two techniques to narrow down the search space: (1) partition the input graphs into (overlapping) groups sharing common properties; (2) summarize the vertex neighbor information from the partitioned datasets onto the 'Neighbor Association Summary Graph's for effective mining. We applied our method to 105 human microarray datasets, and identified a large number of potential transcription modules, activated under different subsets of conditions. Validation by ChIP-chip data demonstrated that the likelihood of a coexpression cluster being a transcription module increases significantly with its recurrence. Our method opens a new way to exploit the vast amount of existing microarray data accumulation for gene regulation study. Furthermore, the algorithm is applicable to other biological networks for approximate network module mining.

Availability: http://zhoulab.usc.edu/NeMo/

Contact: xjzhou@usc.edu

*To whom correspondence should be addressed.
†The authors wish it to be known that, in this opinion, the first two authors should be regarded as joint first authors.

1 INTRODUCTION

Reverse-engineering transcriptional regulatory networks is one of the key challenges for computational biology (Conlon *et al*., 2003; Luscombe *et al*., 2004; Pilpel *et al*., 2001; Segal *et al*., 2003; Wang *et al*., 2005). Microarray technology, with its ability to simultaneously measure the expression of thousands of genes, has revolutionized the way of studying gene transcription. A commonly used analytical approach is to derive coexpression clusters, which are presumably likely to be controlled by the same transcription factors (Banerjee and Zhang, 2002; Liu *et al*., 2001; Roth *et al*., 1998; Zhou *et al*., 2003). However, this assumption is not always true, because (1) one type of experimental condition may simultaneously perturb multiple regulatory programs, such that genes from these different regulatory programs may show similar and indistinguishable expression patterns; (2) even if the regulation of those genes can be traced to the same transcription factors, they may be located in different positions of transcription cascades, and thus not share the same direct regulators and (3) experimental noise and outliers may lead to biased and erroneously high estimates of coexpression similarity.

The rapid accumulation of microarray data has offered new promises in addressing the above problems; however, the potential is so far not well recognized and vastly under-utilized. Intuitively, if a set of genes form a coexpression cluster in multiple datasets generated under different conditions, they are more likely to represent a transcription module than a single-occurrence cluster does (Zhou et al., 2005). Here, we define a transcription module to be a set of genes regulated by the same transcription factor(s). The challenge is how to efficiently identify such gene sets. Although a variety of approaches have been developed to cluster a microarray dataset (Eisen *et al*., 1998; Tamayo *et al*., 1999; Tavazoie and Church, 1998) they cannot be easily extended to identify gene sets coexpressed across a subset of given microarray datasets. The difficulty is that two factors must be simultaneously determined: first, which set of genes can recurrently form a cluster; second, in which subset of microarrays does this set of genes form clusters. It is even harder if (1) we consider that not all genes within a coexpression cluster will strictly exhibit high expression correlation due to measurement noise; and (2) both the number of genes and the number of datasets are large. Since a set of genes may form coexpression clusters only under a small subset of conditions due to the highly dynamic nature of

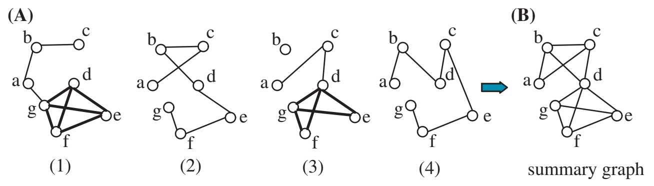

Fig. 1. (**A**) Schematic illustration of a frequent dense vertexset. Four graphs with the same nodes but different edges are shown. The vertex set $\{d, e, f, g\}$ is a frequent dense vertexset because $>80\%$ of the vertex pairs are connected in at least 2 out of the 4 graphs (thick lines). (**B**) We construct a summary graph by adding these four graphs together and by deleting edges that occur less than two times in the graphs. One of the two dense subgraphs in the summary graph, $\{a, b, c, d\}$, is not dense in any original graph.

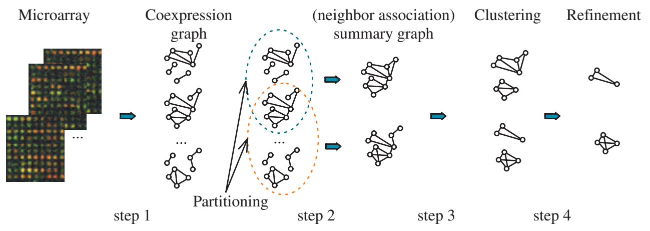

Fig. 2. The pipeline of our frequent dense vertexset mining algorithm (called NeMo). Step 1: extract coexpression graphs from multiple microarray datasets with insignificant edges removed. Step 2: partition coexpression graphs into groups and construct a (neighbor association) summary graph for each group. Step 3: cluster each summary graph for dense subgraphs and Step 4: refine/extract frequent dense vertexsets from the dense subgraphs discovered in Step 3.

human transcriptome, randomly selecting a subset of microarray datasets to search might miss many modules.

In this article, we develop a graph-based approach to efficiently and systematically identify gene sets which form coexpression clusters across multiple datasets. Given m microarray datasets, we model each dataset with a coexpression graph, where each node represents one gene. If two genes show high and significant correlation in their expression profiles, they are connected with an edge. We then search for a set of genes which form dense subgraphs in at least θm out of the m datasets ($0 \leq \theta \leq 1$). Since edges indicate coexpression, a dense subgraph represents a coexpression cluster. Importantly, instead of requiring the recurrence of the exact dense subgraph, here we only require the connectivity among the gene set to be higher than a threshold. That is, as long as a large percentage (e.g. $\geq 60\%$) of gene pairs in a gene set are coexpressed, we consider the gene set to be coregulated (see an example in Fig. 1A). We relax the criteria based on our experiences that pursuing exact match would overlook some coexpressed clusters due to the noisy data and the unavoidable cutoff selection for edge construction.

Our problem can be formulated as mining frequent dense vertexset (FDVS) across multiple graphs. This is a previously unaddressed problem, of which the frequent dense subgraph-mining problem can be regarded as a special case where the same edge set must recur. We have recently designed two algorithms to identify frequent dense subgraphs, SPLAT (Yan *et al.*, 2005) and CODENSE (Hu *et al.*, 2005). The former algorithm searches for exact recurrences of dense subgraphs, and the latter allows approximation of edge recurrence but requires coherency of edge recurrence, i.e. the edge set shall show highly correlated recurrence across the given graph set. The requirements in both algorithms are too stringent to identify many potential transcription modules. On the other hand, to relax the requirement, one may take a summary-graph-based approach, i.e. aggregating the input graphs together and identifying dense subgraphs in the aggregated graph, as proposed by Lee e*t al*. 2004. However, it could result in false FDVS's that may not be dense in any of the original graphs (see an example in Figure 1B). Other network motif discovery tools such as Kelley *et al.* 2004 are designed for sparse biological networks, not appropriate for large-scale coexpression graphs.

The essential problems with the summary graph approach are: (1) noise may aggregate and become indistinguishable with the signals that occur only in a small subset of the graphs; and (2) the edges in a dense summary graph may never occur together in individual original graphs. Guided by our biological understanding, we devised the following two techniques to overcome these problems: (1) since similar biological conditions are likely to activate similar sets of transcription modules, we partition the input graphs (can be regarded as transcriptomes under different conditions) into subsets of graphs sharing certain topological properties, thus are more likely to contain frequent dense vertex sets. In such a subset of graphs, the aggregation of signal shall be greater than that of noise, thus improving signal/noise ratio; (2) for each subset of graphs, we construct a neighbor association summary graph, which measures the association of two vertices based on their connection strength with their neighbors across multiple graphs. For example, given two vertices u and v, if many small frequent dense vertex sets include them, they are more likely from the same FDVS. Biologically speaking, such two genes are more likely to be in the same transcription module. Thus, if we increase their edge weight in the summary graph, it may help separating subtle clusters from dominant clusters in a summary graph. Figure 2 depicts the pipeline of this graph-mining methodology.

We applied our algorithm to 105 human microarray datasets, and identified a large number of potential transcription modules. We demonstrated the power of our approach based on ChIP-chip data of 20 transcription factors and genome-wide evolutionarily conserved transcription factor binding sites. We show that the likelihood for a coexpression cluster to form a transcription module increases significantly with the cluster recurrence, validating the principle of this integrative approach. In addition, we demonstrate the potential of our approach in revealing condition-specific regulatory activation.

2 PROBLEM FORMULATION

Given a graph $G=(V, E)$ and a subgraph induced by vertex set $V' \subseteq V$, written $G[V']$, the *density* of V' is defined as $\sigma(G[V']) = \frac{2|E'|}{|V'|(|V'|-1)}$, where E' is the edge set of $G[V']$.

Definition 1 (Frequent Dense Vertexset). *Consider a set of undirected unweighted graphs, $D = \{G_i = (V, E_i)\}$, where a common vertex set V is shared by G_i. Given a density threshold δ and a frequency threshold θ, a set of vertices, $V' \subseteq V$, is a frequent dense vertexset if, among all induced graphs $\{G_i[V']\}$, at least $\theta|D|$ graphs have density $\geq \delta$.*

Let $D_\delta(V')$ be the graphs in D where the density of V' is at least δ. We call $D_\delta(V')$ the *supporting graphs* of a dense vertexset V'. The number of graphs in $D_\delta(V')$ is called the *support* of V'. According to the above definition, a frequent dense vertexset is a set of vertices, rather than a classical graph with vertices and edges. This definition is able to support the concept of approximate graph patterns, which do not always have the same edge set in the supporting dataset. In addition to density, we could also add other constraint such as minimum degree ratio, $\min_{v \in V'}(degree(v)\ /(|V'| - 1))$, to avoid small degree vertices.

From a computational point of view, given a graph dataset, it could be hard to enumerate all of the frequent graphs that satisfy the density constraint. For the complexity of mining FDVS's, observe that mining all maximal dense subgraphs of D implies finding the largest ones, the maximum dense subgraphs, which is NP-hard. Therefore, we resort to an approximate solution that aggregates the graphs together to form a summary graph and then identifies dense subgraphs from it in a top–down manner. In the next section, we are going to examine this solution in detail.

Definition 2 (Summary Graph). *Given a set of graphs, $D = \{G_i = (V, E_i)\}$, the summary graph S is a weighted graph with vertex set V and each edge (u, v) assigned a weight equal to the number of graphs that contain* (u, v).

3 MINING FREQUENT DENSE VERTEXSETS

Given m graphs, a frequent dense vertexset with density δ and frequency θ must form a subgraph with density $\geq \delta\theta m$ in the summary graph. According to this observation, we can start from the summary graph and mine its dense subgraphs first. Once it is done, those dense subgraphs are post-processed for extraction of true frequent dense vertexsets. The overview of this summary-graph-based method is outlined as follows:

1. Construct a summary graph: Given m graphs, remove infrequent edges, aggregate all of m graphs to form a summary graph S.
2. Mine dense subgraphs from the summary graph: Apply overlapping clustering algorithms such as MODES (Hu *et al.*, 2005) to decompose the summary graph S to a set of dense subgraphs $\widehat{M}$, such that $\widehat{M}$ satisfies the density constraint, e.g. $\geq \delta\theta m$.
3. Refine: extract true frequent dense vertexsets from the vertex set of $\widehat{M}$.

For dense subgraph mining, we apply MODES (Hu *et al.*, 2005) to cluster the summary graph. MODES is an overlapping hierarchical clustering method, which is built on HCS (mining highly connected subgraphs) (Hartuv and Shamir, 2000). Given a graph, MODES first selects the minimum normalized cut, (Shi and Malik, 2000), to break the summary graph into two subgraphs. Such a decomposition process continues until the resulting subgraph either satisfies the density constraint or reaches a minimum size threshold set by users. After that, all of discovered dense subgraphs are condensed into a node to form a new graph. The decomposition is applied again to the newly formed graph. For detailed information, the readers are referred to our previous work (Hu *et al.*, 2005).

For the refinement step, we adopt a heuristic refinement process. Given a dense summary subgraph $\widehat{M}$ with n' vertices, we first calculate the weight sum of adjacent edges of each vertex and sort these n' vertices in increasing order of edge weight sum. Then the vertices are dropped in that order one by one until the rest of vertices in $\widehat{M}$ form a frequent dense vertexset. More advanced search methods using simulated annealing are in development.

Figure 1 shows an example of the summary graph-based approach for mining frequent dense vertexsets ($\theta = 0.5, \delta = 0.8$). All of the graphs in Figure 1A are first aggregated to form a summary-graph, shown in Figure 1B. Through clustering, two clusters $\{a, b, c, d\}$ and $\{d, e, f, g\}$ are found. Each cluster is then examined for their density and frequency. The vertextset $\{d, e, f, g\}$ satisfies the requirement, while $\{a, b, c, d\}$ does not.

As one can see, the dense subgraphs discovered in a summary graph could significantly shrink the search space and provide a good starting point for the refinement process. Unfortunately, the above framework might create three kinds of artifacts. First, it could generate false patterns, e.g. $\{a, b, c, d\}$ in Figure 1B. Second, it could fail in splitting large infrequent dense vertexsets (see Fig. 3). Third, a clustering algorithm might even break a true dense vertexset in half. If this situation takes place, it could become hard for the refinement process to rediscover it.

One question is how to reduce false FDVS's in summary graph so that true FDVS's can be easily recognized. It was observed that when more and more graphs are integrated, the summary graph will become denser and denser, eventually saturated as a clique. In that case, the possibility of generating false patterns will increase significantly.

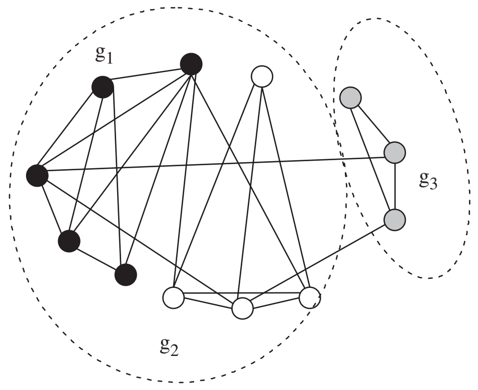

Fig. 3. A summary graph derived from a set of coexpression graphs, where three FDVS's exist, g_1, g_2 and g_3. It is observed that two subgraphs g_1 and g_2 might not be separated easily.

Suppose an observed coexpression graph from a microarray dataset is a real coexpression graph, which includes power-law, transitivity, and all complex dependencies among genes, overlaid with a random graph $G'(n, q)$ resulting from noise, where n is the size of V and each noise edge occurs independently with probability q. q is called noise edge ratio. (Koyuturk *et al.*, 2006) discussed a similar formulation in the context of protein interaction networks.

Let G be the observed graph, G' be the noise graph and G^* be the 'real' graph. $G = G' + G^*$. Multiple observed graphs $\{G_i\}$ are aggregated to formulate a summary graph S. In the following discussion, we are going to examine the noise edge ratio in the summary graph and the number of dense subgraphs formed by noise edges.

Assume noise edge occurs independently with probability q in an observed coexpression graph, the chance for a noise edge to have weight $\geq \theta m$ in a summary graph is as follows (light weight edges are deleted directly),

$$b(m, \theta, q) = \sum_{l=\lceil \theta m \rceil}^{m} \binom{m}{l} q^l (1-q)^{m-l}. \tag{1}$$

If $m = 100$, $q = 2\%$, $\theta = 5\%$ (this setting will be explained in Section 6), $b(m, \theta, q)$ the noise edge ratio in summary graph, is around 5%.

Then, what is the expected number of k-vertex dense subgraphs that could be formed by noise edges? Let $p = b(m, \theta, q)$ and $s = k(k-1)/2$. The expected number of k-vertex subgraphs with minimum density δ and minimum degree d is given below,

$$N = \binom{n}{k} \cdot \sum_{l=\lceil \delta s \rceil}^{s} \binom{s}{l} p^l (1-p)^{s-l} \cdot P(k, l, d), \tag{2}$$

where $P(k, l, d)$ is the probability that a k-vertex l-edge graph has minimum degree d. For small k, $P(k, l, d)$ can be derived through simulation. For example, $P(11, 28, 5) \approx 3.18E{-}5$. For a typical setting in human gene coexpression datasets, $n = 9000$, $\delta = 0.5$, $k = 11$, $d = 5$, $N << 1$ when $p = 2\%$. When $p = 3\%$, $N >> 1$. A slight growth of p might significantly increase the chance of false dense subgraphs.

According to the above analysis, when the noise edge ratio is high in individual coexpression graphs, many false dense subgraphs might exist in summary graph, i.e. dense subgraphs formed by noise edges. In addition to this problem, noise edges in a summary graph might interfere with true patterns and fool clustering algorithms.

There are two approaches to remedy the false dense subgraph problem: (1) divide the coexpression graphs into small groups and formulate summary graphs based on a subset of coexpression graphs [with θm fixed, when m decreases, $b(m, \theta, q)$ will decrease significantly according to Equation (1). (2) Re-weight summary graph to reduce the weights of noise edges [reduce p in Equation (2)].

4 UNSUPERVISED PARTITIONING

According to the previous discussion, it is useful to group coexpression graphs into subsets so that the summary-graph-based approach could be carried out effectively on each subset. From a computational point of view, it is infeasible to try all of combinations of m coexpression graphs when m is large. On the other hand, random partitioning will not work too since a frequent dense vertexset could be infrequent in a random subset of graphs.

An optimal solution should group together graphs that likely contain at least one FDVS. For example, apply known functional categories to finding a set of seed FDVS's and use them to group coexpression graphs. However, this supervised strategy might bias the partitioning to known modules, while leaving unknown modules undiscovered forever. We found that for any frequent dense vertexset, it must be a dense vertexset in at least one graph. Therefore, one can actually mine dense subgraphs in individual graphs separately, extract frequent ones and take them as seed vertexsets to bootstrap the mining process. This bootstrap process is outlined as follows (it is also illustrated in Fig. 2).

1. Extract dense subgraphs $\widehat{M}$ from each individual graph; refine these subgraphs for true frequent dense vertexset M, using the greedy refinement process introduced in Section 3.
2. For each frequent dense vertexset M, calculate its supporting graph set $D_\delta(M)$. Take $D_\delta(M)$ as one subset.
3. Remove duplicate subsets.
4. For each subset $D_\delta(M)$, call the summary-graph-based approach (see Section 3) to find frequent dense vertexsets in $D_\delta(M)$.

Furthermore, the discovered frequent dense vertexsets from the above process could be taken as new seeds to repeat. An interesting question is why we are not satisfied with the clusters discovered in the first step. The reason is that some subtle clusters can only be discovered from multiple graphs rather than from single graphs. In the experimental section, we will demonstrate that our method can find more frequent dense vertexsets.

Partitioning is one way to reduce false dense vertexsets. Starting from the next section, we are going to examine another approach that re-weights edges in summary graph.

5 NEIGHBOR ASSOCIATION

Summary graph is a way to measure the association strength of two vertices. For the traditional summary graph, it is measured by the number of edges shared by the two vertices across m graphs. We could adopt a stronger measure, for example, the number of small frequent dense subgraphs that two vertices belong to. Under this measure, if two vertices share many small frequent dense subgraphs, likely these two vertices come from the same dense vertexset. This idea leads to the discovery of 'neighbor association' summary graph.

Let us first examine the concept of neighbor association in a single graph (Section 5.1) and then extend it to multiple graphs (Section 5.2). Given two vertices u and v in a graph, if a lot of small frequent dense subgraphs contain both u and v, it is likely that u and v come from the same cluster. Figure 3 shows this intuition: the two clusters, black nodes and white nodes, share a lot of triangles within clusters, but not many across clusters. If the number of shared triangles is used to weigh the edges in a graph, it could make clustering much easier. In the next subsection, we are going to show that if a graph/cluster is dense, its vertices will share many dense subgraphs.

5.1 Graphlets

DEFINITION 3 (Graphlet). *Given a graph G, a k-graphlet is an induced subgraph of G with k vertices.*

Graphlets have been introduced to measure local structural similarity between two networks (Pržulj *et al.*, 2004). This study is focused on the property of their density.

LEMMA 1. *Given a graph G with density δ, written G_δ, the average density of a k-graphlet g of G_δ is δ.*

PROOF. see our Supplementary Material.

Lemma 1 is true for any graph, showing that there is a connection between the density of a graph and its k-graphlets. According to this lemma, it is concluded that there is at least one k-graphlet in G_δ whose density is at least δ. Consider a dense random graph $G'(n, \delta)$. What is the density distribution of its k-graphlets? Let $s = n(n-1)/2$ and $t = k(k-1)/2$. The probability that a k-graphlet has l edges follows binomial distribution with parameter t and δ.

$$\binom{t}{l}\delta^l(1-\delta)^{(t-l)}. \tag{3}$$

Since the median of binomial distribution is around $\lfloor \delta t \rfloor$, it implies that when a graph is dense, half of its k-graphlets are likely dense. The above discussion suggests that if we randomly draw a k-graphlet g from a dense graph G_δ, g could be dense with a good chance, and the chance does not depend on the size of graph G_δ very much.

In order to identify dense subgraphs in a large unweighted graph, it could be beneficial to weight its edges based on the number of dense graphlets shared by vertices. This kind of weighted graphs, called *neighbor association graphs*, relies on more than one neighbor to determine the weight between two vertices. This weighting method could increase signal/noise ratio for identifying subtle dense subgraphs.

For the graph shown in Figure 3, the two dense subgraphs g_1 and g_2 might not be separated accurately by a clustering algorithm. However, it can be done easily in the neighbor association graph since the dashed edges receive less weight (Fig. 4). A neighbor association graph attempts to reduce the weight of edges between different dense subgraphs, while conserving edges within each dense subgraph.

There are two issues left. First, the larger dense subgraph the two vertices belong to, the larger number of dense graphlets they will have, which might be unfair for small dense subgraphs. Hence, a normalization is needed. Second, since we are mining frequent dense vertexsets in multiple graphs, the weight of edge (u, v) in a summary graph should be the number of small frequent dense vertex subsets shared by u and v. We are

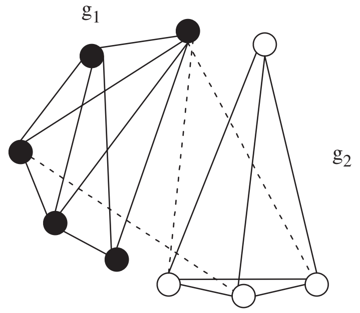

Fig. 4. A neighbor association graph derived in part from the graph shown in Figure 3. The solid edge means the pair of nodes share at least two triangles in the summary graph, while the dashed line shows they share at most one triangle.

going to examine the first issue in this subsection, and the second issue in the next subsection.

Let $score(u, v)$ be the weight of edge (u, v) in a neighbor association graph. Intuitively, if u and v are in the same dense subgraph, $score(u, v)$ should be close to 1.0. If u and v are not in the same dense subgraph e.g. u might also be a member of another dense subgraph, the score should be smaller. Certainly, if u and v do not share any dense k-graphlet, the score should be set to 0. Furthermore, $score(u, v)$ should not be too much relevant to the size of the dense subgraph that u and v belong to, which could cause bias against small dense subgraphs.

Let us first examine an extreme case. Suppose there is a large clique with n vertices. Given two vertices, u and v in the clique, the maximum number of k-graphlets (cliques) they share is $\binom{n-2}{k-2}$. Obviously, this number is highly sensitive to n and should be normalized. Since n is not known in advance, we use the number of dense $k-1$-graphlets (cliques) that contain vertex u to normalize it,

$$\frac{\binom{n-2}{k-2}}{\binom{n-1}{k-2}} = \frac{n-k+1}{n-1}. \tag{4}$$

When $n \gg k$, Equation (4) is close to 1.

Let π_u be the set of frequent dense $(k-1)$-vertexlets that contain vertex u and $\pi_{u,v}$ be the set of frequent dense k-vertexlets that contain vertices u and v. We define $score(u, v)$ as follows,

$$score(u, v) = \frac{|\pi_{u,v}|}{|\pi_u|}. \tag{5}$$

Given a graph G_δ, the probability of a k-graphlet whose density is at least δ is not very relevant to the size of G_δ. Let this probability be p. The score function in Equation (5) is roughly normalized,

$$score(u, v) \approx \frac{p * \binom{n-2}{k-2}}{p * \binom{n-1}{k-2}} = \frac{n-k+1}{n-1}.$$

The score function discussed so far is not symmetric: *score* (v, u) is not equal to *score*(u, v). We could take the average as the final weight between two vertices u and v. One advantage of this score function is that it need not enumerate all of the dense k-graphlets. In fact, sufficient sampling in k-graphlets is good enough to estimate the number of k-graphlets containing u or v.

5.2 Neighbor association summary graph

Definition 4 (Vertexlet). *Given a vertex set V, a k-vertexlet is a subset of V with k vertices.*

Since we are working on multiple graphs, it is important to count small frequent dense k-vertex sets (called vertexlets) to construct the neighbor association summary graph from a set of graphs. That is, given two vertices u and v, we calculate *score*(u, v) as the number of frequent dense k-vertexlets that contain u and v, and normalize it using frequent dense $(k-1)$-vertexlets. A frequent dense k-vertexlet is a frequent dense vertexset with k vertices. We call such summary graph a *neighbor association summary graph*. In contrast, the original summary graph is also called *first-order summary graph*.

As to the neighbor association summary graph, one can imagine, given two vertices u and v, if k is chosen large enough, the neighbor association method is equivalent to an exhaustive search method, where it enumerates all frequent dense vertexsets that include u and v. From this point of view, by setting k to a small number, our design of a neighbor association summary graph is positioned between the first-order summary graph and the exhaustive search method.

An interesting question is that given a frequent dense vertexset V', what is the percentage of its k-vertexlets that are frequently dense? We use random graph as an example to give analytical result. Suppose there is an n-vertex dense vertexset V' whose support is r in a graph dataset D. Assume the supporting graphs in $D_\delta(V')$ are r independent random graphs whose density is at least δ. What is the probability of a k-vertexlet g $(g \subseteq V')$ whose density is at least δ' in all of these r graphs?

Let $t = k(k-1)/2$, the probability that the density of g is at least δ' in the *ith* graph $(1 \le i \le r)$ is

$$p = \sum_{l=\lceil \delta' t \rceil}^{t} \binom{t}{l} \delta^l (1-\delta)^{(t-l)} = I_\delta(\lceil \delta' t \rceil, t - \lceil \delta' t \rceil + 1), \qquad (6)$$

where

$$I_\delta(a, b) = \frac{B(\delta, a, b)}{B(1, a, b)}, \; B(\delta, a, b) = \int_0^\delta u^{(a-1)}(1-u)^{(b-1)} du.$$

Equation (6) is the probability that g is dense in the the *ith* random graph. The probability that g is dense in all of the r independent graphs is p^r. For a typical value $p = 0.5$, the value of p^r could be very low when r is large. The number of k-vertexlets in V' whose density is at least δ' is $p^r \cdot \binom{n}{k}$ on average. It seems that when r increases, no matter what the value δ' has, p^r drops quickly. Fortunately, in our problem setting, the difficulty of mining FDVS's emerges when r is very low $(r = \theta m)$. When the support threshold (θ) is set high, the first-order summary graph becomes good enough because most of the noise edges will be removed due to their low weights in the summary graph. Furthermore, since most of the connections in a frequent dense vertexset are correlated across multiple coexpression graphs, the chance to have frequent dense vertexlets should be much higher than p^r.

5.3 Implementation

In this section, we put together our design of the neighbor association summary graph. Algorithm 1 outlines the workflow of building a neighbor association summary graph. In the first step, the first-order summary graph is constructed with infrequent edges removed. This first-order summary graph is then taken as a template for the neighbor association summary graph and its edge weight will be recalculated. In practice, since the number of frequent edges should be significantly smaller than that of infrequent ones, the template is relatively sparse. In order to estimate the cardinality of π_u, one could sample t subsets of V, each of which draws $k-2$ vertices from the vertexset V except u. Each of t sets forms a $(k-1)$-vertexlet with u. Let α be the percentage of frequent dense ones among these t $(k-1)$-vertexlets. Hence, $|\pi_u|$ can be estimated via $\alpha \cdot \binom{n-1}{k-2}$. When π_u is small, we might have to sample a very large number of vertex subsets from V. In order to speed up this process, a trade-off is used to restrict the sampling to those vertices that have frequent edges connecting to u. The number of such vertices should be smaller than n. If the cost is still high, we could first cluster V into subsets and calculate $|\pi_u|$ within each subset. The same technique can also be applied to the estimation of $|\pi_{u,v}|$. Lines 8–11 re-assign the edge weights and finish the construction of a neighbor association summary graph.

Algorithm 1 Neighbor Association Summary Graph Construction

```
Input: Graph dataset $D = \{G_1, G_2, \ldots, G_m\}$,
       Vertex set $V$,
       Density threshold $\delta$, minimum degree ratio $d$,
       Minimum support threshold $\theta$;
       Vertexlet size $k$;
Output: A neighbor association summary graph $S$;

1: build the first-order summary graph $S_0$ over $D$;
2: remove edges in $S_0$ whose weight is below $\delta\theta m$;
3: for each node $u$ in $S_0$ do
4:     estimate $|\pi_u|$;
5: for each edge $(u, v)$ in $S_0$ do
6:     estimate $|\pi_{u,v}|$;
7: $S \leftarrow S_0$;
8: for each edge $(u, v)$ in $S$ do
9:     if $|\pi_u|, |\pi_v| > 0$ then
10:        re-assign weight $w(u, v) = (\frac{|\pi_{u,v}|}{|\pi_u|} + \frac{|\pi_{u,v}|}{|\pi_v|})/2$;
11:    else set weight $w(u, v) = 0$;
12: return;
```

Once the neighbor association summary graph is built, we apply the same mining routine illustrated in Section 3: use clustering to decompose the neighbor association summary graph into (overlapping) dense subgraphs and then refine those discovered dense subgraphs if their corresponding vertexsets are not frequently dense enough.

The mining algorithm, including coexpression graph partitioning, summary graph and neighbor association summary graph construction, clustering and refinement, is named NeMo for **N**etwork **M**odule Mining. Note that NeMo will merge the results generated by the first order and neighbor association summary graphs to maximize its potential.

6 EXPERIMENTAL STUDIES

6.1 Data Source, modeling and parameter setting

We selected 105 human microarray datasets, generated by Affymetrix U133 and U95Av2 platforms (details on Supplementary Material). Each microarray dataset is modeled as a coexpression graph where each node is a unique gene and an edge exists between two nodes/genes if their expression correlation with a p-value less than 0.01 is significant. The expression correlation between each pair of genes, denoted as r, is the correlation coefficient of minimum absolute value using the leave-one-out Pearson correlation, which is robust against single experiment outliers and sensitive to overall similarities in expression patterns (Zhou *et al.*, 2002). In our study, the top 2% most significant correlations with a p-value less than 0.01 are included in each graph. The threshold of 2% can be justified by Equation 2 under which noise edges are unlikely to form a random dense subgraph.

6.2 Validation of transcriptional module discovery

We applied NeMo to discover frequent approximate dense vertexsets in the 105 coexpression networks, and identified 4,727 recurrent coexpression clusters, which satisfy the following criteria: the cluster density is greater than 0.7 in at least 10 supporting datasets. The average cluster size is 10.7 with a SD of 2.9. A histogram of the cluster size distribution is provided on the Supplementary Material. To assess the clustering quality we tested the member genes of each cluster for enrichment of the same bound transcription factor. The transcription factor to target gene relationships were ascertained through ChIP-Chip experiments, which contain 9,176 target genes for 20 TFs covering the entire human genome (details on our Supplementary Material). We define a recurrent cluster to be a potential transcriptional module if (1) >75% of the cluster genes are bound by the same transcription factor; and (2) the enrichment of the particular TF in the cluster is statistically significant with a hypergeometric P-value <0.01 relative to its genome-wide occurrences. Among the identified clusters, 15.4% satisfied our definition of potential transcription modules, which is a high hit rate considering we only tested 1% of the approximately 2000 transcription factors estimated to exist in the human genome. One possible explanation for this high coincidence with the ChIP-Chip data is some of the transcription factors, such as E2F4, play a major role in regulating cell proliferation; and nearly half of our datasets study neoplastic transcriptomes. To approximate the false positive rate of our potential transcription module

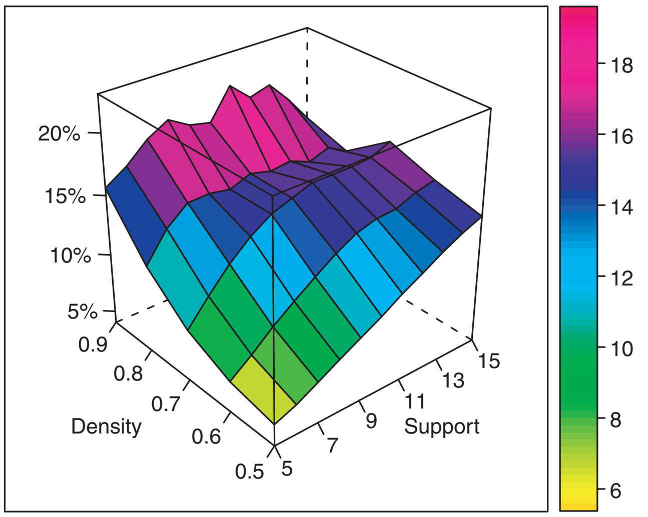

Figure 5. Percentage of potential transcription modules validated by ChIP-Chip data increases with cluster density and recurrence.

identification, we permuted the cluster memberships 100 times and tested for TF-binding enrichment. On average, the permuted set of clusters was only 0.2% enriched for a common transcription factor. This demonstrates the potential power of our approach to reliably reconstruct regulatory modules.

The integrity of the clusters is further validated by varying the cluster density and recurrence. As the criteria for cluster identification increases in stringency, the relative proportion of clusters that share a common bound TF also increases (Fig. 5). At a high recurrence rate, even clusters that are not tightly coexpressed are likely to represent transcription modules. For example, 16.1% of clusters with a density of 0.5 at support 20 are enriched for a bound transcription factor whereas clusters with a density 0.8 and support 5 has a hit rate of only 11.3%. This reveals the compensatory effect of recurrence versus density in capturing transcription modules. Therefore, relaxing the density criterion allows us to identify loosely coexpressed transcription modules, caused by either subtle regulatory mechanisms or measurement noise, which analysis of an individual dataset would not identify. This confirms the foundation of this study to integrate multiple microarray datasets for transcription module discovery.

In an attempt to scale up the transcription module identification, we attempted to validate our clusters based on putative transcription factor binding sites (TFBS) which are conserved in human, mouse and rat (Kuhn *et al.*, 2007) (download version Feb/06). The data contains 407 TFs with 7,720 distinct binding targets within our dataset. Among the identified clusters that satisfied our criteria, 12.5% showed enrichment for the same TFBS with a hypergeometric P-value <0.001 as compared to the permutation test of which only 3.3% were enriched.

The high quality of the clusters identified by NeMo is also supported by functional homogeneity analysis. We define a cluster to be functionally homogeneous if >75% of the member genes belong to the same Gene Ontology biological process with a hypergeometric P-value <0.01. As the cluster density and recurrence increase, the functional homogeneity of the clusters increases as well (Fig. 6). Using our cluster criteria of a minimum density of 0.7 and a support of 10, 65.3% of our clusters are functionally homogeneous compared to the 2.2% in the permuted clusters. This high degree of functional homogeneity provides further support for the potential co-regulation of our clusters. The clusters participate in a broad range of biological processes such as cell cycle, immune response and electron transport. Interestingly, most of the modules with evolutionarily conserved TFBSs are functionally annotated as immune response genes, indicating the important role of immune-related regulatory mechanisms in evolutionary history.

6.3 Context-specific transcriptional control

Integrating multiple microarray datasets not only allows us to identify robust transcription modules, but also to determine the context information in which those modules are activated. We determine the phenotypic context of a microarray dataset based on the MeSH headings of its corresponding PubMed record (Butte and Chen, 2006), which are then mapped to UMLS concepts (Bodenreider, 2004). In total, 143 UMLS concepts were mapped to at least 8 datasets. A total of 109 of the clusters that satisfied our criteria are statistically significant with a hypergeometric *P*-value <0.01 in a specific condition such as malignant neoplasms, respiratory diseases or nervous system disorders. Figure 7 shows a cluster of size 8 which is densely connected in 12 datasets, among which 5 are leukemia datasets as shown. Relative to the 12 leukemia datasets out of the total 105 datasets, the condition-specific enriched activation of this module is statistically significant at 0.0042 level. The module is potentially regulated by E2F4, and majority of its member genes are involved in cell cycle which agrees with the nature of leukemia. The power of NeMo to detect clusters with non-persistent edges is demonstrated here, as only 7 of the 8 possible edges are present in all five datasets.

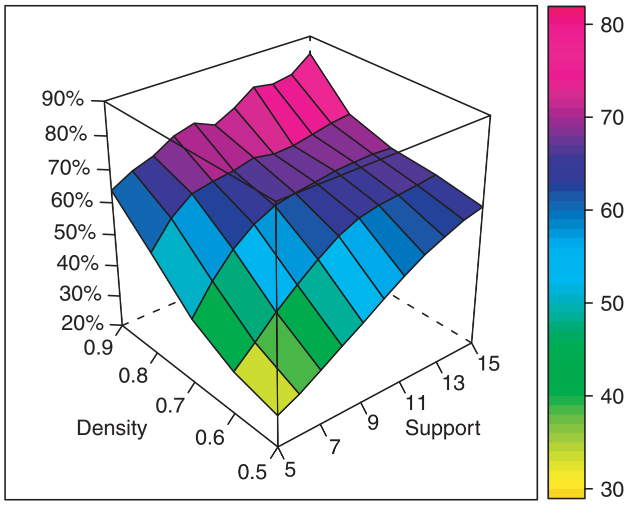

Fig. 6. Percentage of functionally homogeneous clusters increases with cluster density and recurrence.

6.4 Comparison with other approaches

In this section, we are going to compare several approaches to demonstrate the effectiveness of graph partitioning and neighbor association techniques. Four algorithms will be compared: (1) modules discovered from individual graphs (Independent), (2) summary graph over all coexpression graphs (Summary), (3) summary graphs over partitioned graph sets (Partitioning), (4) neighbor association summary graphs with partitioning (Neighbors). In this experiment, we set the cluster frequency, $\theta = 5\%$, and the cluster density $\delta = 50\%$ (minimum degree ratio $= 0.5$). Figure 8 shows the comparison of these four algorithms. The *Y* axis is the percentage of homogeneous modules discovered by each algorithm.

As shown in Figure 8, both Partitioning and Neighbors techniques are effective to find dense modules with much higher homogeneity than individual graphs and summary graph without partitioning. We also experimented increasing the support threshold. Both Partitioning and Neighbors perform better since noise edges have smaller weight in summary graph. When the support reaches $\theta m = 16$, the homogeneity percentage is 78.3 and 84.6% for Partitioning and Neighbors, respectively.

Once frequent dense subgraphs in (neighbor association) summary graphs are identified, they are post-processed for generating frequent dense vertexsets that strictly satisfy the density and frequency constraint. After post-processing, the percentage of functionally homogeneous modules discovered becomes similar for all the methods, which is above 36%. This is reasonable because frequent dense vertexsets with similar density and frequency should have similar characteristics. However, the total number of genes identified by these four methods could be different. Figure 9 shows the number of genes

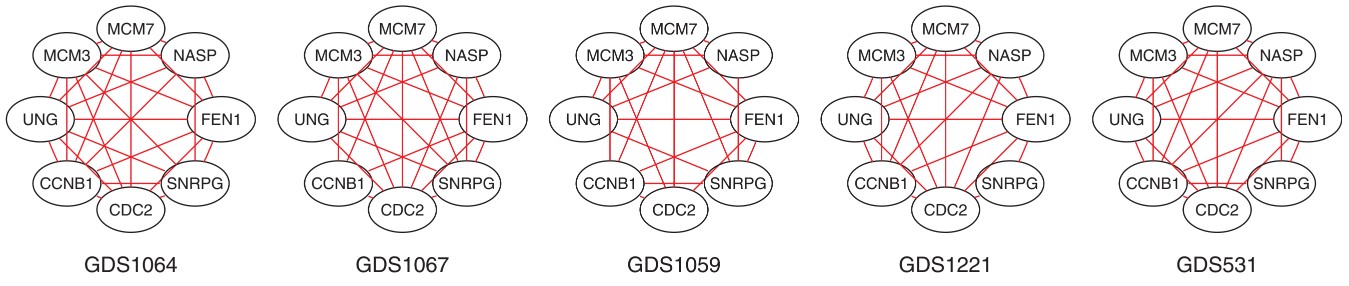

Fig. 7. A potential transcription module is tightly coexpressed in five datasets studying leukemia.

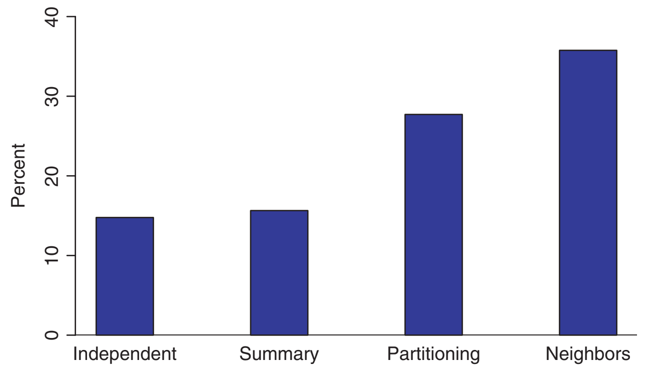

Fig. 8. Percentage of homogeneous modules.

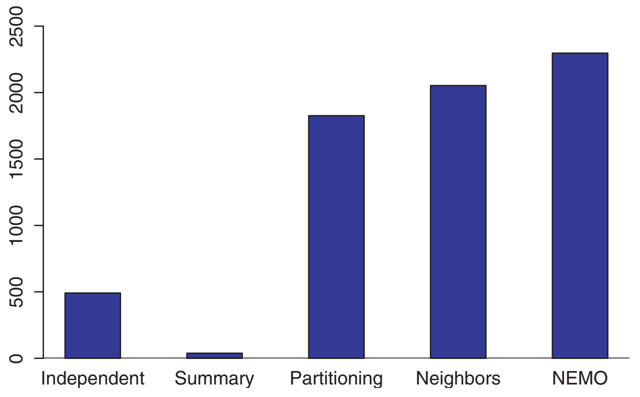

Fig. 9. Frequent dense vertexsets: number of genes in homogeneous modules.

in homogeneous modules covered by frequent dense vertexsets. Five numbers are included for Independent, Summary, Partitioning, Neighbors and NeMO which merges the results from Partitioning and Neighbors. From the result of NeMo, we can calculate the genes exclusively discovered by Partitioning and Neighbors. It shows their results are complementary to each other. Figure 9 also shows summary graph without partitioning does not work well for generating frequent dense vertexsets, in part because it only looked at dense subgraphs in one summary graph rather than multiple summary graphs.

7 CONCLUSIONS

We developed a novel graph-based algorithm, NeMo, to efficiently mine the frequent dense vertexsets in a set of coexpression graphs. We demonstrated its application in identifying frequent coexpression clusters across many microarray datasets. The identified clusters possess high functional homogeneity, and are likely to be regulated by the same transcription factor(s), thus forming potential transcription modules. Depending on the microarray datasets in which the modules occur, we can further infer the conditions and contexts in which they are activated. Given the vast amount of microarray data accumulation in public repositories, NeMo shall serve as a timely tool to reconstruct transcription modules in a context-dependent way. In this study, we focused on human transcriptional module reconstruction. However, the proposed method is generally applicable to any organism for which large amount of microarray data exists. It can also be applied to other biological relational graphs for finding approximate network modules.

ACKNOWLEDGEMENTS

We thank Mrinal Kalakrishnan for helping with the UMLS concept mapping, Juan Nunez-Iglesias for proof reading and Chun-Chi Liu for discussion. This work is supported by the NIH grants R01GM074163, P50HG002790, U54CA112952, and the NSF grant 0515936.

Conflict of Interest: none declared.

REFERENCES

Banerjee,N. and Zhang,M.Q. (2002) Functional genomics as applied to mapping transcription regulatory networks. *Curr. Opin. Microbiol.*, **5**, 313–317.
Bodenreider,O. (2004) The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Res.*, **32**, D267–D270.
Butte,A.J. and Chen,R. (2006) Finding disease-related genomic experiments within an international repository: first steps in translational bioinformatics. In *AMIA Annu. Symp. Proc.*, pp. 106–110.
Conlon,E.M. *et al.* (2003) Integrating regulatory motif discovery and genome-wide expression analysis. *Proc. Natl Acad. Sci. USA*, **100**, 3339–3344.
Eisen,M.B. *et al.* (1998) Cluster analysis and display of genome-wide expression patterns. *Proc. Natl Acad. Sci. USA*, **95**, 14863–14868.
Hartuv,E. and Shamir,R. (2000) A clustering algorithm based on graph connectivity. *Information Processing Lett.*, **76**, 175–181.
Hu,H. *et al.* (2005) Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics*, **21** (Suppl. 1), i213–i221.
Kelley,B.P. *et al.* (2004) PathBLAST: a tool for alignment of protein interaction networks. *Nucleic Acids Res.*, **32**, W83–W88.
Koyutürk,M. *et al.* (2006) Assessing significance of connectivity and conservation in protein interaction networks. In *RECOMB*, pp. 45-–59.
Kuhn,R.M. *et al.* (2007) The UCSC genome browser database: update 2007. *Nucleic Acids Res.*, **35**, D668–D673.
Lee,H.K. *et al.* (2004) Coexpression analysis of human genes across many microarray data sets. *Genome Res.*, **14**, 1085–1094.
Liu,X. *et al.* (2001) BioProspector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. *Pac. Symp. Biocomput.*, 127–138.
Luscombe,N.M. *et al.* (2004) Genomic analysis of regulatory network dynamics reveals large topological changes. *Nature*, **431**, 308–312.
Pilpel,Y. *et al.* (2001) Identifying regulatory networks by combinatorial analysis of promoter elements. *Nat. Genet.*, **29**, 153–159.
Pržulj,N. *et al.* (2004) Modeling interactome: scale-free or geometric? *Bioinformatics*, **20**, 3508–3515.
Roth,F.P. *et al.* (1998) Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nat. Biotechnol.*, **16**, 939–945.
Segal,E. *et al.* (2003) Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat. Genet.*, **34**, 166–176.
Shi,J. and Malik,J. (2000) Normalized Cuts and Image Segmentation. *IEEE Trans. on Pat. Analy. and Mach. Int.*, **22**, 888–905.
Tamayo,P. *et al.* (1999) Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation. *Proc. Natl Acad. Sci. USA*, **96**, 2907–2912.
Tavazoie,S. and Church,G.M. (1998) Quantitative whole-genome analysis of DNA-protein interactions by in vivo methylase protection in E. coli. *Nat. Biotechnol.*, **16**, 566–571.
Wang,W. *et al.* (2005) Inference of combinatorial regulation in yeast transcriptional networks: a case study of sporulation. *Proc. Natl Acad. Sci. USA*, **102**, 1998–2003.

Yan,X. *et al*. (2005) Mining closed relational graphs with connectivity constraints. In *Proceedings of the Int. Conf. on Knowledge Discovery and Data Mining*, pp. 324–333.

Zhou,X. *et al*. (2002) Transitive functional annotation by shortest-path analysis of gene expression data. *Proc. Natl Acad. Sci. USA*, **99(20)**, 12783–12788.

Zhou,X. *et al*. (2003) Novel mechanisms of T-cell and dendritic cell activation revealed by profiling of psoriasis on the 63,100-element oligonucleotide array. *Physiol. Genomics*, **13**, 69–78.

Zhou,X.J. *et al*. (2005) Functional annotation and network reconstruction through cross-platform integration of microarray data. *Nat. Biotechnol*., **23**, 238–243.