IDENTIFYING DYNAMIC NETWORK MODULES WITH TEMPORAL AND SPATIAL CONSTRAINTS

RUOMING JIN [1,*], SCOTT MCCALLEN [1], CHUN-CHI LIU [2], YANG XIANG [1],EIVIND ALMAAS [3], XIANGHONG JASMINE ZHOU [2*]

[1] *Department of Computer Science, Kent State University, Kent, OH, USA*
[2] *Program in Molecular and Computational Biology, University of Southern California, Los Angeles, CA, USA*
[3] *Bioscience and Biotechnology Division, Lawrence Livermore National Laboratory, Livermore, CA, USA*

Despite the rapid accumulation of systems-level biological data, understanding the dynamic nature of cellular activity remains a difficult task. The reason is that most biological data are static, or only correspond to snapshots of cellular activity. In this study, we explicitly attempt to detangle the temporal complexity of biological networks by using compilations of time-series gene expression profiling data. We define a dynamic network module to be a set of proteins satisfying two conditions: (1) they form a connected component in the protein-protein interaction (PPI) network; and (2) their expression profiles form certain structures in the temporal domain. We develop an efficient mining algorithm to discover dynamic modules in a temporal network. Using yeast as a model system, we demonstrate that the majority of the identified dynamic modules are functionally homogeneous. Additionally, many of them provide insight into the sequential ordering of molecular events in cellular systems. Finally, we note that the applicability of our algorithm is not limited to the study of PPI networks, instead it is generally applicable to the combination of any type of network and time-series data.

1. Introduction

Cellular systems are highly dynamic and responsive to cues from the environment. Cellular function and response patterns to external stimuli are regulated by a complex web of diverse molecular interactions, such as protein-protein interactions, protein-DNA interactions, and metabolic processes. Despite the availability of large-scale biological network data[23,15,7,8], gaining a system-level understanding of the *dynamic* nature of cellular activity remains a difficult and, until recently, a much overlooked task. Since there typically is little direct information available on the temporal dynamics of these network interactions, the majority of molecular-interaction network modeling and analysis has been solely focused on static properties. For example, a protein-protein interaction (PPI) network obtained from yeast two-hybrid experiments can be viewed as a comprehensive graph of the edges (interactions) that eventually may occur under the set of tested conditions. However, it is not guaranteed that two adjacent edges would ever occur *simultaneously* or even close in time, and thus, that identified network modules and motifs will correspond to functionally relevant units. While a series of studies

[*]to whom correspondence should be addressed

have been aimed at identifying modules in PPI networks [17,2,19], the networks in their analysis have all been regarded as static.

In this study, we attempt to detangle the *temporal* complexity of biological networks by identifying dynamic modules, consisting of a set of proteins that coexist both spatially and temporally. We will use the PPI network as the spatial constraints. We estimate temporal characteristics of a network component by using compilations of time-series gene expression profiling data, since accurate temporal parameters are not yet available for PPI systems. A similar approach has been employed to study the dynamic protein complex formation during the cell cycle [3,12]. In order to measure the similarity between two time-series gene-expression profiles [1,18], we have implemented time-warping dynamic programming. A significant advantage of this method is that it identifies the best local alignment of segments in two time-series profiles. Consequently, it can identify time-shifted and local similarity patterns which are often overlooked by computing the Pearson's correlation of two entire profiles. We represent the local alignment of two profiles as a range pair (or interval pair), denoted as $[s1, e1]$:$[s2, e2]$, where $s1$ and $s2$ are the starting time points, and $e1$ and $e2$ are the end time points of the alignment. Since two proteins may interact over different time segments in a long time-series and they may show varying level of cooperativity in those time segments, we may also consider suboptimal alignments of two profiles. In this case, the temporal expression similarity of two interacted proteins can be represented as a list of range pairs. Given a PPI network, we add the local alignment information from the time-series gene expression data, and refer to the network as a *temporal network*.

On a temporal network, we can discover how two proteins' activities correlate with each other over time. More importantly, for multiple proteins we can identify when a dynamic module is activated in the PPI network. We define a dynamic network module to be a set of proteins which satisfy two conditions: (1) they form a connected component in the PPI network; and (2) their expression profiles form certain structures (see below) in the temporal domain. An example dynamic module satisfying these constraints could consist of all proteins, for which time-series local alignments overlap over a common time period. That is, all protein interactions in this example module are synchronized to perform their functions. Another interesting example corresponds to the case of activity cascades. Here, a dynamic module is triggered by one or a small number of proteins that activate their network neighbors. These in turn, may also propagate the signal further through neighbor activation. Given this, we use a directed edge to represent how the activity of a protein may invoke that of another protein, and the direction is determined by the range-pair of their time series alignment, i.e. the beginning aligned time point of the triggering protein is likely to be earlier than that of the triggered protein.

Although many graph algorithms are available to perform network analysis, including those recent work [21,22,4,14] for mining temporal and evolutionary networks, to our knowledge, no algorithm has been developed to identify network patterns while accounting for the interval-based temporal constraints. In this study, we take the first step in this direction by developing efficient mining algorithms to discover dynamic modules in a temporal biological network. We have performed detailed experimental testing using yeast as a model system. In addition, many of those modules provide insight into the sequential order of molecular events in cellular systems. Furthermore, a majority of the discovered modules are not densely connected, yet they comprise functional units. Such modules are very difficult to identify based on the PPI network alone. We are well aware of the limitation of microarray data in measuring protein activities, and the incompleteness of protein-protein interaction data. While using those data as an example, we want to emphasize that our algorithm is generally applicable to combining any types of network and time-series data.

2. Problem Definition

We represent the PPI network as a graph $G = (V, E)$, where proteins correspond to the vertex set V, and the protein-protein interactions are recorded as the edge set E. Each vertex (protein) is associated with a gene expression time-series, formally $T_i = (x_1^i, x_2^i, \cdots, x_n^i)$.

We use the time-warping dynamic programming algorithm [18] to align two time-series expression profiles, and use a range pair to represent the resulting best local alignment for each interacted protein pair. The range pair for protein pair $e = (v_1, v_2) \in E$ is denoted as $r(e) = ([i, i'] : [j, j'])$, where i, j are the starting time points of the aligned interval pair of an interacting protein pair, and i', j' are the end points. We denote $r(e)[v_1] = [i, i']$ and $r(e)[v_2] = [j, j']$. From the PPI network we eliminate those edges (the interacted protein pairs) that do not have any local similar range pairs, and term the resulting range-pair network as the *temporal network*.

We also assign directions to the edges in the temporal network. Such assignment is based on the observation that the range pairs in the temporal network can provide the sequential ordering of activities between edge-connected vertices. Specifically, if one interval in the range pair appears earlier than the other one, it could be possible that the activity of the vertex (protein) associated with the first interval induces the activity of the other vertex. That is, the sequential order obtained could facilitate causal inference, although causal inference requires much more information and is beyond the scope of this work. In the following, we will use a directed edge to represent the sequential order between two vertices in the temporal network. Specifically, let $[t_{A1}, t_{A2}] : [t_{B1}, t_{B2}]$ be the interval pair between protein A and B. If $t_{A1} - t_{B1} \geq k$, we build a directed edge from protein A to B; if $|t_{A1} - t_{B1}| < k$, we build an undirected edge between A and B, where k serves as a smoothing parameter against noise.

As previously mentioned, the module discovery approach in this study is bound by both *spatial* and *temporal* constraints. Intuitively, the spatial constraints refer to the physical interactions between proteins, and the temporal constraints corresponds to the closeness of range pairs, which can be formalized based on the *second order subgraph*.

Definition 1. (Second Order Graph) The second order graph of the temporal biological network $G = (V, E)$ is a graph $G' = (V', E')$, where V' records each edge e in $E(G)$ as a unique vertex v'_e, and for any two vertices v'_{e1} and v'_{e2} in $V(G')$ there exists an edge in $E' = E(G')$ between them if and only if 1) the two edges e_1 and e_2 share a common vertex y, i.e., $e_1 = (x, y)$ and $e_2 = (y, z)$, or vice versa; and 2) the temporal constraint: $|r(e_1)[y] \cap r(e_2)[y]| \geq d$.

To facilitate our discussion, we refer to a vertex in the second order graph as an *edge-vertex*. Note that, in the second order graph, two edge-vertices with a common vertex y (in the original temporal network G) are neighbors if the two intervals of y specified by the corresponding interval-pairs in G overlap an interval with length no less than d. Further, for a subgraph G_s of G ($G_s \subseteq G$), its corresponding second order graph, G'_s, is a subgraph of G' ($G'_s \subseteq G'$).

Formally, we define the *dynamic module* as follows.

Definition 2. (Dynamic Module) Given a temporal network $G = (V, E)$ and its second order network $G' = (V', E')$, a dynamic module is a subgraph G_s of G such that 1) G_s is connected (we refer to this condition as the spatial constraint); and 2) its corresponding second order subgraph G'_s is also connected (we refer to this condition as the temporal constraint).

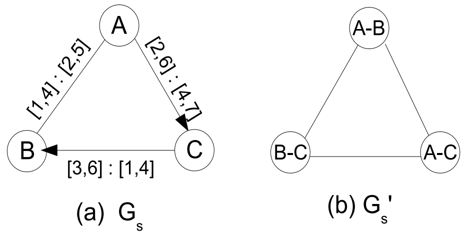

Figure 1. A dynamic module G_s and its corresponding second order graph G'_s

Figure 1(a) lists an example of a dynamic module with the choice of temporal constraint parameter $d = 2$. Note that the range pairs on the outside of the subgraphs represent the local similarity alignment. The direction of edges for the dynamic module is added with the smoothing parameter $k = 2$. Figure 1 (b) shows the second order graph of the 3-vertex dynamic module in Figure 1(a).

However, to enumerate all dynamic modules is computationally very expensive. In this paper, in order to reduce the enumeration complexity, we enumerate only the *edge-maximal dynamic modules*.

Definition 3. (Edge-Maximal Dynamic Module) Given a temporal network $G = (V, E)$ and its second order network $G' = (V', E')$, an edge-maximal dynamic module is a dynamic module $G_s \subseteq G$, and there is no any other dynamic module $G_{s'} \subseteq G$, such that $V(G_s) = V(G_{s'})$ and $E(G_s) \subset E(G_{s'})$.

3. Algorithm for Edge-Maximal Dynamic Module Discovery

The definition of dynamic modules requires to find the subgraph G_s which is connected (the spatial constraint) and its corresponding second order subgraph G'_s is also connected (the temporal constraint). An efficient enumeration algorithm thus should aggressively employ these two constraints to prune the search space. In particular, these two constraints have an interesting relationship, which will form the basis of our enumeration algorithm.

Lemma 1. *Let G_s and G'_s be the corresponding subgraph and second order subgraph. If G'_s is connected (the temporal constraint holds), then its corresponding subgraph G_s is also connected (the spatial constraint is true). However, if G_s is connected, its second order subgraph may not be connected.*

For simplicity, the proof is omitted. What this lemma suggests is that these two constraints are not *symmetric*, and one of the constraints can be easily satisfied without explicit tests. This lemma also relates to an important property, the *anti-monotone* property, which is the key for many efficient mining and enumeration algorithms.

Lemma 2. *Let G_s and its corresponding second order subgraph G'_s form a dynamic module. All the connected subgraphs of the second order subgraph G'_s always form dynamic modules.*

From the enumeration purpose, G'_s has the follow properties. Let $G'_s \cup \{v'\}$ be a new connected second order subgraph by adding a new vertex v' and associated edges to G'_s. Then, the new subgraph corresponding to the new second order subgraph will always form a *dynamic module*. However, this property does not hold for G_s. Adding a new vertex v and associated edges to G_s may produce a new second order subgraph which is not connected. Thus, if we try to enumerate all the (edge-maximal) dynamic modules from the first order graph, we do not have the anti-monotone property.

Given this, we will try to enumerate edge-maximal dynamic module directly by utilizing the second order graph. Simply speaking, we will try to recursively enumerate all the connected subsets of edge-vertices in the second order graph, and each of such subsets corresponds to a unique dynamic module. However, the issue is that how we can identify the edge-maximal ones efficiently. Clearly it is too computationally expensive to first enumerate *all* dynamic modules and then identify the edge-maximal ones. Note that the number of dynamic modules can be exponentially larger than the number of edge-maximal modules.

In the following, we introduce a novel approach which uncovers all the edge-maximal dynamic modules by coupling the original temporal network and its second order graph. It directly generates edge-maximal dynamic modules without any redundancy. Algorithm 1 provides the backbone of our approach. Conceptually, this algorithm enumerates all the edge-maximal dynamic modules in two stages. In the first stage, it tries to find a new dynamic module, which corresponds to a connected edge-vertex set in the second order graph. Then in the second

Algorithm 1 $DyModSearch(Set\ P, Set\ N, Set\ Ex)$

```
Parameter: $P$ {the set of edge-vertices in $G'$ currently in the module}
Parameter: $N$ {the set of edge-vertices in $G'$ that can join to the current module}
Parameter: $Ex$ {the set of edge-vertices in $G'$ that have been previously expanded}
1: for each $n \in N$ do
2:   $P' \leftarrow Reach(n, G'_s[cover(P \cup \{n\})])$ {the edge-vertices that can be reached from vertex $n$ in the second order subgraph $G'_s[cover(P \cup \{n\})]$}
3:   if $(P' \backslash P) \cap Ex = \emptyset$ then {no edge-vertex in $P' \backslash P$ has been visited earlier}
4:     Output the maximal dynamic module $G_{P'}$ {$G_{P'}$ is the subgraph corresponding to the vertex-set $P'$ in the second order graph}
5:     if $|cover(P')| < T$ then {$T$: user-specified size for maximal dynamic modules}
6:       $Ex' \leftarrow Ex \cup P'$
7:       $N' \leftarrow (N \cup Neighbor(P' \backslash P)) \backslash Ex$ {Except for initialization ($|P'| = 1$): $N \leftarrow Neighbor(P') \backslash Ex$}
8:       $DyModSearch(P', N', Ex')$
9:     end if
10:  end if
11:  $v \leftarrow cover(\{n\}) \backslash cover(P)$ {the new vertex in the original network being introduced by the new edge $n$}
12:  $P_v \leftarrow \{v' | v' \in (P' \backslash P) \wedge v'\ covers\ v\}$
13:  $Ex \leftarrow Ex \cup P_v$ {add $P_v$ to the exclusion set $E$}
14:  $N \leftarrow N \backslash Ex$
15: end for
```

stage, it tries to quickly expand this dynamic module into an edge-maximal dynamic module.

Algorithm 1 performs a DFS (depth-first search) style enumeration utilizing three sets P, N and Ex. Set P records all the edge-vertices of the second order graph which represents the current edge-maximal dynamic module. Set N records all the edge-vertices that are neighbors of P, and can be appended to produce new dynamic modules. Set Ex records all the edge-vertices which can no longer be used to generate any new edge-maximal dynamic modules. At each invocation of the algorithm, we will produce an edge-maximal dynamic module, represented by set P.

To make the algorithm efficient, we need specifically address the following three issues. 1) How can we transform a dynamic module into an edge-maximal dynamic module? Specifically, how can we pick up edge-maximal dynamic modules quickly without enumerating other non-edge-maximal ones? 2) How can we avoid generating an edge-maximal dynamic module more than once? In other words, how can we ensure that each edge-maximal dynamic module is only generated once? 3) How can we aggressively prune the search space to ensure that we remove as early as possible the dynamic modules that will not grow into edge-maixmal dynamic modules?

The first issue is taken care of by Line 2: simply speaking, after we generate a new dynamic module $P \cup \{n\}$ by combining the edge-maximal module P with a new edge-vertex n, we will immediately expand it to make it edge-maximal. This can be done quickly by utilizing the following property: *each newly added edge-vertex n will cover a new vertex v in the original temporal network G*, except that the first edge-vertex in P will cover two new vertices in G. This property comes from the fact that P is an edge-maximal dynamic module. Basically, it already contains all the edges which can be connected for the existing vertices to form a dynamic module. Thus, any new dynamic module $P \cup \{n\}$ ($P \neq \emptyset$) must involve a new vertex v. However, we need to note that those additional edge-vertices which can help make the new dynamic module $P \cup \{n\}$ *edge-maximal* do not necessarily links to v directly, since they may be connected through other newly introduced edge- vertices. Given this, we will apply $cover(P \cup \{n\})$ to map the edge-vertices in set $P \cup \{n\}$ to their corresponding vertices in the original network G, and then extract its induced subgraph $G_s[cover(P \cup \{n\})]$, and eventually the corresponding second order subgraph $G'_s[cover(P \cup \{n\})]$. Finally, the connected component in this induced subgraph containing n is our desired edge-maximal dynamic module for $P \cup \{n\}$ (this is achieved by Line 2: searching all the edge-vertices that can be reached from edge-vertex n in the second order subgraph $G'[cover(P \cup \{n\})]$).

The second issue is resolved by a simple test in Line 3. Basically, if the newly added edge-vertices had been previously visited, i.e., $(P' \backslash P) \cap Ex \neq \emptyset$ (Line 3), then, we would know that the newly generated edge-maximal dynamic module had already been enumerated. In other words, we will only select the modules when $(P' \backslash P) \cap Ex = \emptyset$ (Line 4).

Finally, the third issue is addressed by exploiting the coupling between the original graph and the second order graph, and by maintaining the sets N and Ex (Lines 6-7 and Lines 11-14). The set N contains edge-vertices that could be the valid extension of the current dynamic module, and the set Ex contains the edge-vertices that are invalid for extensions. Here, we will again apply the property that each newly added edge-vertex n will cover a new vertex v in the original temporal network G (when $P \neq \emptyset$). For the newly added vertex v, we identify all of its adjacent edges in the original temporal network, and further identify the subset that are connected to P in the second-order graph (P_v, Line 12). This subset (P_v) is then appended to Ex and removed from N (Line $13 - 14$). Such coupling between the two graphs helps to narrow down the search space, and the removal of edge-vertices allows the effective pruning of the search space.

4. Results

4.1. *Data source and alignment of time-series expression data*

We collected 36066 distinct protein-protein interactions of *S. cerevisiae* from The BioGrid databases [20], the MIPS database [9], and other sources [25,10]. As is customary, self interactions representing autoregulation or protein homodimerization were not included in the analysis. We collected a total of 11 microarray gene expression time-series for *S. cerevisiae*. These datasets include from 10 to 25 time points of various temporal scales (*e.g.* minutes and hours) and measure gene-regulation under very different biological conditions, such as the cell cycle, filamentous-form growth, fermentation, carbon source perturbation, and thiolutin treatment.

To analyze these time-series, we implemented local similarity analysis [18], which is a variant of the time-warping algorithm, to identify the local alignment between two time series. Since most of the collected time-series expression datasets are short, here we only identify the best local alignment for each protein pair and do not consider suboptimal alignments. For each gene pair with an identified protein-protein interaction, we performed a normal score transformation [11] for their expression profiles before analyzing their local alignment by dynamic programming. We define the local similarity score as the maximal sum of the product of the corresponding entries of all the subsequences of the two time-series within a predefined time delay D. D can be chosen to fit to individual situations. Without losing generality, we choose $D = 5$ in our analysis. To determine the significance threshold ($P < 0.025$) of the computed local similarity score, we carried out the local similarity analysis on 1,000,000 pairs of random expression profiles with a normal distribution. We assigned an edge between all gene pairs with a known protein interaction if we also uncovered a significant local expression similarity score between the two genes.

From protein interaction data and the 11 expression time series, we obtain 11 graphs, ranging from 1872 to 13298 edges. Interestingly, the terminal genes of 32% of those edges have Pearson's correlations with P-value higher than 0.05. Consequently, the local alignments are either shifted or are short for all of these edges and, thus, can not be captured by standard correlation analysis.

4.2. *Dynamic network modules are more biologically meaningful than static modules*

We implemented Algorithm 1 and applied it to obtain edge-maximal dynamic modules with different time interval overlap parameter $d = 3, 4, 5, 6, 7, 8$ and with node size from 5 to 8 [a] This constraints collects the set of edge-maximal dynamic modules Using the cell cycle dataset (GDS2347) as an example, at $d = 8$, we identified 7574 modules of node size 5 to 8. To assess whether the temporal information improves our ability to discover biologically meaningful modules, we compared the functional homogeneity of the dynamic modules with that of the static modules determined based only on the protein interaction data. That is, for each of the edge-maximal dynamic modules, we identify the connected component of the PPI network with the same size, and compare the homogeneity of their biological functions. We used the Gene Ontology (GO) biological process annotation and defined specific functions to be those associated with GO nodes containing less than 200 genes. A module is considered functionally homogeneous if its variation in functional classification, as modeled by the hypergeometric distribution, has a $P < 10^{-6}$ (P stands for P-value). Our results showed that 38%

[a] Due to the very large number of dynamic modules within each temporal network, we apply an *overlap* constraint in Algorithm 2 to prune the edge-maximal dynamic modules which share more than k nodes with an already identified edge-maximal dynamic module in the resulting set. Here, we choose $k = 2$.

of dynamic modules are functionally homogenous, while on average (from 100 randomizations), only 27% of static modules are functionally homogenous. This highlights the usage of temporal information in uncovering biologically meaningful gene constellations that are hidden by a static network representation. In general, the functional homogeneity of modules increases with increasing d.

Interestingly, a majority of the dynamic module are very sparse. For example, 81% of the dynamic modules from the dataset GDS2347 have a connectivity γ less than 0.5, where γ is defined as $\gamma = 2m/(n(n-1))$, with m being the number of edges and n the number of nodes in a module. We have illustrated two such examples in Figure 2. The non-dense modules are very hard to extract from the static protein-protein interaction network solely based on their topology. Here, we demonstrate that the temporal expression information can facilitate the identification of likely important signals that are otherwise easily overlooked.

Dataset	# modules	GO[a]
GDS124	13729	40.96%
GDS1611-1	32532	36.43%
GDS1611-2	43024	31.21%
GDS1752-1	26306	37.14%
GDS1752-2	25412	37.68%
GDS18	26662	31.33%
GDS2318	12405	41.93%
GDS2347	7574	48.26%
GDS2350	3038	26.56%
GDS39	23983	36.17%
GDS608	2814	46.55%

[a] The percentage of modules with significant enrichment in proteins from a GO term ($P < 10^{-6}$ is required and at least 2 genes in the module are annotated with the same GO term).

4.3. *Dynamic network motifs reveal temporal events in cellular systems*

The edge-maximal dynamic modules can aid in the characterization of signal propagation and the elucidation of temporal organizational patterns in cellular activities. Note that, there is little information on the detailed sequential ordering of molecular events due to the limited power and resolution of current experimental techniques. This leaves an important opportunity for the community of computational scientists, as *in silico* predictions can complement experimental approaches to provide novel insights. In the following, we will discuss two cases from our analysis in detail, for which the identified dynamic events are supported by current knowledge.

Figure 2(a) illustrates an edge-maximal dynamic module activated during the cell cycle progression (obtained using dataset GDS2350). The module contains 5 genes, all of which are involved in the biological process of “microtubule cytoskeleton organization and biogenesis”. Among those, SPC105 and SPC110 are components of the spindle pole body; NUF2 is implicated in connecting the centromere to the spindle pole body during chromosome segregation [13]; Cin8 clusters

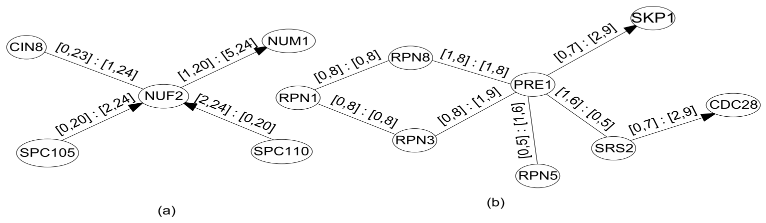

Figure 2. Two examples of edge-maximal dynamic modules.

kinetochores into their characteristic bi-lobed metaphase configuration [24]; while NUM1 is required for nuclear migration and localizes to the mother cell cortex as well as the bud tip, forming cytoplasmic microtubule capture-sites during the late anaphase [5,6]. Although all of these genes participate in microtubule dynamics, their interactions do not occur at the same time. The directed edges from SPC105/SPC110 to NUF2 indicates that the assembly of the spindle pole body takes place ahead of its interaction with the centrosome, and the directed edge from NUF2 to NUM1 implies that the connection from the centromere to the spindle pole occurs before nuclear migration. The identified edge-maximal dynamic module correctly captures the sequential order of protein-protein interactions along the time axis. It is noteworthy that all of the 4 connected gene pairs have the very low Pearson's expression correlations of -0.34, -0.08, -0.12, and 0.35. Consequently, this module can only be identified by taking the temporal characteristics into account.

Our second example, Figure 2(b), shows a dynamic module activated in the dataset GDS608 [16]. This dataset resulted from studies of the temporal expression patterns in wild-type diploid cells shifted from the typical yeast-form growth to that of a filamentous-form growth. Filamentous-form cells were collected and profiled hourly for 10 hours. The identified dynamic module contains 8 genes, in which RPN1, RPN3, RPN8, PRE1, PRE5 are involved in ubiquitin-dependent protein catabolic process. In particular, PRE5 belongs to the proteasome alpha-subunit complex, RPN3 and RPN8 are part of a proteasome regulatory particle, the lid subcomplex, while SRS, SKP1, and CDC28 are involved in the cell cycle. Most edges in this module are undirected (that is, they contain at most a time shift of 1 unit), and only two edges are directed, one pointing from PRE1 to SKP1, and the other pointing from SRS2 to CDC28. This is in agreement with the hypothesis that the ubiquitin-dependent protein degradation by the 26S proteasome is controlling the filamentous-form growth [16]. More specifically, the 26S proteasome regulates the activity of the Cdc28 kinase, which in turn controls cell-cycle progression and morphogenesis during filamentous-form growth [16].

5. Conclusions

We have presented a graph-based algorithm to identify dynamic network building-blocks by combining information from multiple temporal networks. To this end, we have designed a set of efficient algorithms to identify *dynamic* network modules. Our approach provides an alternative to traditional graph-based module finding algorithms that assume all links are simultaneously present (static). Consequently, traditional algorithms are incapable of addressing questions such as the time-ordering of link-based events.

In contrast, our approach facilitates the use of temporal information to detangle the complex wiring diagrams of cellular systems. As an example, we have applied our algorithm to a combination of microarray time-series expression data and the yeast protein-interaction network. We have demonstrated that our method uncovers functionally homogenous network modules, and more importantly, that it has the ability to identify the *sequential ordering* of molecular events taking place in biochemical systems. While the results from our approach have no direct implication on the causality of events, the sequential activity information embedded in the derived modules may serve as a starting point for causal-inference analyses.

Microarray gene-expression data and protein interaction data have their specific limitations, and as such, may not provide a measurement of all functional activities in a cell. However, we want to emphasize that as a general data integration framework, our algorithm can be applied to any type of time-series activity measurement (not limited to gene-expression data) and any type of network data (not limited to protein-interaction data). Thus, our approach provides an important new class of tools for the systems-level analysis of biological data.

6. Acknowledgments

This work of CCL and XJZ was supported by the grants NIH R01GM074163, NIH U54CA112952, NSF 0515936, and NSF 0747475. EA's work was performed under the auspices of DoE by LLNL under Contract DE-AC52-07NA27344 and funded by LDRD project 06-ERD-061.

References

1. J Aach and G M Church. Aligning gene expression time series with time warping algorithms. *Bioinformatics*, 17(6):495–508, Jun 2001. Comparative Study.
2. Dongbo Bu, Yi Zhao, Lun Cai, Hong Xue, Xiaopeng Zhu, Hongchao Lu, Jingfen Zhang, Shiwei Sun, Lunjiang Ling, Nan Zhang, Guojie Li, and Runsheng Chen. Topological structure analysis of the protein-protein interaction network in budding yeast. *Nucleic Acids Res*, 31(9):2443–2450, May 2003. Comparative Study.
3. Ulrik de Lichtenberg, Lars Juhl Jensen, Soren Brunak, and Peer Bork. Dynamic complex formation during the yeast cell cycle. *Science*, 307(5710):724–727, Feb 2005.
4. Prasanna Desikan and Jaideep Srivastava. Mining temporally evolving graphs. In *WebKDD*, 2004.
5. M Farkasovsky and H Kuntzel. Yeast Num1p associates with the mother cell cortex during S/G2 phase and affects microtubular functions. *J Cell Biol*, 131(4):1003–1014, Nov 1995.
6. M Farkasovsky and H Kuntzel. Cortical Num1p interacts with the dynein intermediate chain Pac11p and cytoplasmic microtubules in budding yeast. *J Cell Biol*, 152(2):251–262, Jan 2001.
7. D A Fell and A Wagner. The small world of metabolism. *Nat Biotechnol*, 18(11):1121–1122, Nov 2000.

8. Nabil Guelzim, Samuele Bottani, Paul Bourgine, and Francois Kepes. Topological and causal structure of the yeast transcriptional regulatory network. *Nat Genet*, 31(1):60–63, May 2002.
9. Ulrich Guldener, Martin Munsterkotter, Matthias Oesterheld, Philipp Pagel, Andreas Ruepp, Hans-Werner Mewes, and Volker Stumpflen. MPact: the MIPS protein interaction resource on yeast. *Nucleic Acids Res*, 34(Database issue):436–441, Jan 2006.
10. T Ideker, V Thorsson, J A Ranish, R Christmas, J Buhler, J K Eng, R Bumgarner, D R Goodlett, R Aebersold, and L Hood. Integrated genomic and proteomic analyses of a systematically perturbed metabolic network. *Science*, 292(5518):929–934, May 2001.
11. Ker-Chau Li. Genome-wide coexpression dynamics: theory and application. *Proc Natl Acad Sci U S A*, 99(26):16875–16880, Dec 2002.
12. Nicholas M Luscombe, M Madan Babu, Haiyuan Yu, Michael Snyder, Sarah A Teichmann, and Mark Gerstein. Genomic analysis of regulatory network dynamics reveals large topological changes. *Nature*, 431(7006):308–312, Sep 2004.
13. A Nabetani, T Koujin, C Tsutsumi, T Haraguchi, and Y Hiraoka. A conserved protein, Nuf2, is implicated in connecting the centromere to the spindle during chromosome segregation: a link between the kinetochore function and the spindle checkpoint. *Chromosoma*, 110(5):322–334, Sep 2001.
14. Joshua O'Madadhain, Jon Hutchins, and Padhraic Smyth. Prediction and ranking algorithms for event-based network data. *SIGKDD Explorations*, 7(2):23–30, 2005.
15. Christos A. Ouzounis and Peter D. Karp. Global Properties of the Metabolic Map of Escherichia coli. *Genome Res.*, 10(4):568–576, 2000.
16. Susanne Prinz, Iliana Avila-Campillo, Christine Aldridge, Ajitha Srinivasan, Krassen Dimitrov, Andrew F Siegel, and Timothy Galitski. Control of yeast filamentous-form growth by modules in an integrated molecular network. *Genome Res*, 14(3):380–390, Mar 2004. Comparative Study.
17. Alexander W. Rives and Timothy Galitski. Modular organization of cellular networks. *Proceedings of the National Academy of Sciences*, 100(3):1128–1133, 2003.
18. Quansong Ruan, Debojyoti Dutta, Michael S Schwalbach, Joshua A Steele, Jed A Fuhrman, and Fengzhu Sun. Local similarity analysis reveals unique associations among marine bacterioplankton species and environmental factors. *Bioinformatics*, 22(20):2532–2538, Oct 2006.
19. Victor Spirin and Leonid A. Mirny. Protein complexes and functional modules in molecular networks. *Proceedings of the National Academy of Sciences*, 100(21):12123–12128, 2003.
20. Chris Stark, Bobby-Joe Breitkreutz, Teresa Reguly, Lorrie Boucher, Ashton Breitkreutz, and Mike Tyers. BioGRID: a general repository for interaction datasets. *Nucleic Acids Res*, 34(Database issue):535–539, Jan 2006.
21. Jimeng Sun, Christos Faloutsos, Spiros Papadimitriou, and Philip S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *KDD*, pages 687–696, 2007.
22. Chayant Tantipathananandh, Tanya Berger-Wolf, and David Kempe. A framework for community identification in dynamic social networks. In *KDD*, pages 717–726, 2007.
23. D. Thieffry, A. Huerta, E. Perez-Rueda, and J. Collado-Vides. From specific gene regulation to genomic networks: A global analysis of transcriptional regulation in escherichia coli, 1998.
24. Jessica D Tytell and Peter K Sorger. Analysis of kinesin motor function at budding yeast kinetochores. *J Cell Biol*, 172(6):861–874, Mar 2006.
25. Christian von Mering, Roland Krause, Berend Snel, Michael Cornell, Stephen G Oliver, Stanley Fields, and Peer Bork. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, 417(6887):399–403, May 2002. Comparative Study.