ABSTRACT

CHEN, ZHENGZHANG. Discovery of Informative and Predictive Patterns in Dynamic Networks of Complex Systems. (Under the direction of Prof. Nagiza F. Samatova.)

A latent behavior of a dynamic physical system, such as a biological cell or an atmospheric-ocean system is inherently complex. This complexity often arises from the selective, high-dimensional, and nonlinear interconnections of functionally diverse system components to produce coherent behavior. Data-driven prediction or forecast of the system's behavioral states such as those resulting in land-hitting hurricanes, and discovery of state–determining components and their cross-talks are challenging. The scarcity and complexity of the available data limit the applicability of the existing machine learning methods to deal with such underdetermined, or unconstrained problems. This dissertation addresses these challenges through the following theories and advanced algorithms:

(1) System Phase-related Interplaying Components Enumerator (SPICE) that iteratively enumerates statistically significant components that are hypothesized (1) to play an important role in defining the specificity of the target system's state(s) or phrases; (2) to exhibit a functionally coherent behavior, namely, act in a coordinated manner to perform the state-specific function; and (3) to improve the predictive skill of the system's states when used collectively in the ensemble of predictive models. When tested on the three important biological problems—identification of biohydrogen production, motility, and of cancer-related system components—SPICE demonstrated the superior performance in terms of various skill and robustness metrics, including more than 10% accuracy increase on eight real-world data sets.

(2) The network-based community dynamics theory and scalable algorithm to uncover and characterize the community-based dynamics in system networks with multi-functional communities. The underlying theory for representative-based detection of all possible dynamic communities—grown, shrunken, merged, split, born, or vanished—ensures the scalability and practical applicability of the algorithm. The runtime speedup of 11–46 over the baseline algorithm is observed. Significant and informative community-based dynamics are discovered in the Food Web and Enron networks.

(3) A novel direction of contrast-based mining of complex networks is introduced to discover phase-biased, statistically significant, and predictive anomalous communities. When coupled with SPICE–detected system components as seeds, the significant reduction in the search space and increase in informativeness are obtained. When tested on the two important extreme event problems—identification of tropical cyclone-related and of African Sahel rainfall-related climate indices—the algorithm demonstrated the superior performance in terms of various skill

and robustness metrics.

Discovery of Informative and Predictive Patterns in Dynamic Networks of Complex Systems

by
Zhengzhang Chen

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Computer Science

Raleigh, North Carolina

2012

APPROVED BY:

Prof. Steffen Heber	Prof. Anatoli V. Melechko
Prof. Fredrick H. M. Semazzi	Prof. Nagiza F. Samatova Chair of Advisory Committee

BIOGRAPHY

Zhengzhang Chen received a Bachelor of Science (BS) in Mathematics (with Honors) from Central South University, China in Summer 2005, with a minor in Business Administration. He was recommended to the master program of Computer Science Department at Beihang University exempting from the Chinese National Graduate Entrance Test. He completed his Master of Engineering in Computer Science from Beihang University, China in 01/2008. He was enrolled in North Carolina State University's Ph.D. program in 8/2008 with a Teaching Assistantship. From Fall 2008 to Fall 2009, he served as a Teaching Assistant for Design and Analysis of Algorithms and C and Software Tools. He was nominated by Dr. Steffen Heber for an Outstanding Teaching Assistant award in Fall 2009. In 2010, he began working under Dr. Nagiza F. Samatova as a Research Assistant in data mining and machine learning. He passed the Written Qualifier in Spring 2010, and completed the Oral Examination in Summer 2011.

Supporting Publications (Chronologically Ordered)

1. Z. Chen, K. Padmanabhan, A. Rocha, Y. Shpanskaya, J. R. Mihelcic, K. Scott, and N. F. Samatova, "SPICE: Discovery of Phenotype-Determining Component Interplays," *BMC Systems Biology*, vol. 6(1), pp. 40, PMID: 22583800, 2012.

2. Z. Chen, W. Hendrix, G. Han, I. K. Tetteh, A. Choudhary, F. H. M. Semazzi, and N. F. Samatova, "Detecting Predictive and Physically Interpretable Communities in Contrast Groups of Networks: Application to Adverse Spatio-Temporal Extremes," *Data Mining and Knowledge Discovery Journal*, 2012 (2nd Revision).

3. K. Padmanabhan, Z. Chen, S. Lakshminarasimhan, S. S. Ramaswamy, and B. T. Richardson, "Graph-based Anomaly Detection," Practical Data Mining with R, Book Chapter, 2012 (In Press).

4. I. Tetteh, D. Gonzalez, Z. Chen, N. F. Samatova, and F. H. M. Semazzi, "An Application of a Newly Developed Machine Learning Technique for Predicting Climate-Meningitis Seasonal Outlook Over West Africa," *92nd American Meteorological Society Annual Meeting*, 2012.

5. Z. Chen, W. Hendrix, and N. F. Samatova, "Community-based Anomaly Detection in Evolutionary Networks," *Journal of Intelligent Information Systems: Integrating Artificial Intelligence and Database Technologies*, vol. 39(1), pp. 59–85, 2012.

6. H. Sencan*, Z. Chen*, W. Hendrix, T. Pansombut, F. H. M. Semazzi, A. N. Choudhary, V. Kumar, A. V. Melechko, and N. F. Samatova, "Classification of Emerging Extreme

Event Tracks in Multi-Variate Spatio-Temporal Physical Systems Using Dynamic Network Structures: Application to Hurricane Track Prediction," *The 22nd International Joint Conference on Artificial Intelligence (IJCAI) 2011*, pp. 1478–1484, 2011. *: Both authors contribute equally.

7. Z. Chen, T. Pansombut, W. Hendrix, D. Gonzalez, F. H. M. Semazzi, A. Choudhary, V. Kumar, A. V. Melechko, and N. F. Samatova, "Forecaster: Forecast Oriented Feature Elimination-based Classification of Adverse Spatio-Temporal Extremes," *NCSU Technical Report*, 1840.2/2408, 2011.

8. M. Schmidt, A. Rocha, K. Padmanabhan, Z. Chen, K. Scott, J. R. Mihelcic, and N. F. Samatova, "Efficient Alpha, Beta–Motif Finder for Identification of Phenotype-related Functional Modules," *BMC Bioinformatics*, vol. 12, pp. 440, 2011.

9. K. Wilson, A. Rocha, K. Padmanabhan, K. Wang, Z. Chen, Y. Jin, J. R. Mihelcic, and N. F. Samatova, "Detecting Pathway Cross–Talks by Analyzing Conserved Functional Modules across Multiple Phenotype-Expressing Organisms," *IEEE International Conference on Bioinformatics and Biomedicine (BIBM) 2011* pp. 443–449, 2011.

10. Z. Chen, K. Wilson, Y. Jin, W. Hendrix, and N. F. Samatova, "Detecting and Tracking Community Dynamics in Evolutionary Networks," *The 10th IEEE International Conference on Data Mining (ICDM) Workshop on Social Interactions Analysis and Services Providers (SIASP) 2010*, pp. 318–327, 2010.

11. W. Chen, Z. Chen, and N. F. Samatova, "Approximation Algorithms for the Maximum Duo-preservation String Mapping Problem," *IEEE International Conference on Future Information Technology (ICFIT) 2010*, vol. 1, pp. 190–198, 2010.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my special appreciation and thanks to my advisor, Prof. Nagiza F. Samatova, for her tremendous support and guidance throughout my graduate career. I would also like to thank my committee members, Profs. Steffen Heber, Anatoli Melechko, and Fredrick H. M. Semazzi for their helpful suggestions and comments.

I would also like to thank the fellow graduate students at NCSU for their support, especially, William Hendrix, Wenbin Chen, Kanchana Padmanabhan, Andrea Rocha, Tatdow Pansombut, Huseyin Sencan, John Jenkins, Isaac K. Tetteh, Matt Schmidt, Doel Gonzalez, Kevin Wilson, Zhenhuan Gong, and Ye Jin.

Additionally, I would like to thank Prof. Mladen Vouk, Prof. David Thuente, Prof. Douglas Reeves, Prof. Carla D. Savage, Prof. Matthias Stallmann, Prof. Kemafor Anyanwu, Prof. Ben Watson, Margery Page, Carol Allen, and the rest of the faculty and staff of the Department of Computer Science at NCSU for their help over the years.

Finally, I would like to thank my parents, my sister, my brothers and the rest of my family for all of their constant support and encouragement.

This work was supported in part by the U.S. Department of Energy, Office of Science, the Office of Advanced Scientific Computing Research (ASCR) and the Office of Biological and Environmental Research (BER) and the U.S. National Science Foundation (Expeditions in Computing).

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

An emerging field in data mining is detecting and analyzing the key features or functional structures governing the behavior of dynamic physical systems across various domains from atmospheric-ocean systems to biological cells [30, 31, 90]. Mining such informative and predictive patterns or relationships can help scientists reveal underlying simplicity from complexity [3], develop *data-driven* approaches for modeling latent system behavior, and complement the typical *hypothesis–driven* scientific methodologies.

To achieve this goal, researchers often simplify the modeling process of system's behavior by using some key system components or features. In machine learning, feature selection has been successfully employed to increase the prediction accuracy of classifiers; reduce the computation time of the learning algorithms; increase the robustness of classifiers; and facilitate the interpretability of the derived relationships between features by removing irrelevant and redundant features. The development of feature selection techniques [17, 79, 126, 154] has been an important field in many application domains like climate and biology. For example, in the extreme event prediction, researchers often use the correlation method (e.g., Pearson correlation) to calculate the correlation between each feature and tropical cyclone or rainfall activity, choosing the climate features with the best individual correlations [31, 59].

However, physical dynamic systems are inherently complex, and often operate in multiple phases, described as having similar defining characteristics but whose feedbacks behave in a non-linear fashion [53]. And considering the fact that the number of observational samples to build the prediction models of a real-world system is often significantly fewer than the number of available features, the existing machine learning methods easily become hardly suitable for dealing with such *underdetermined*, or *unconstrained* problems.

To complement the machine learning studies in instance–based data, graphic–theoretical approaches for studying dynamic systems has emerged through the concept of complex networks

[45, 155, 123, 20]. Such complex networks model a variety of systems including societies, ecosystems, the Internet, and others [108]. For example, in climate networks [155, 45], the nodes represent the spatial grid points and the edges between pairs of nodes exist depending on the degree of statistical interdependence between the corresponding pairs of anomaly time series taken from the climate data set. Complex networks have enabled hypothesis-driven insights about the intricate interplay between the topology and dynamics of the physical system.

Networks of dynamic systems can be highly clustered [166]. A community, defined as a collection of individual objects that interact unusually frequently, is a very common substructure in many networks [55, 45, 155], including social networks, metabolic and protein interaction networks, financial market networks, and even climate networks. For example, in protein interaction networks, a set of proteins that are strongly related may form a multiprotein complex or perform a function together within a cell [179]. Previous work has been mainly focused on detecting community structures in static graphs [55, 34, 129], or detecting *conserved* communities in evolutionary networks [74, 113, 141].

In spite of the advantages offered by machine learning approaches and graph theoretical approaches to discover some strong patterns in complex systems, there are several challenges that need to be overcome. First, how can we discover the system component interplays in instance-based data? As aforementioned, the system components often form hierarchical functional modules (or communities) like protein complexes. Thus, the traditional approaches that identify individual components that confer a given system state are likely not optimized to detect groups of such interplays between system components. Second, in the networks of dynamic systems, traditional methods can not detect the community dynamics, but only the conserved or stable communities. In networks of biological systems, a small variation in a gene community may indicate an event, such as gene fusion [137], gene fission [137], or gene gain [23]. Finally, conventional community detection methods [55, 34, 129, 74, 113, 141] often fail to detect predictive and phase-biased communities that are conserved within one group of networks but undergo statistically significant structural transformation in the other groups of networks. Such anomalous communities could contribute to our understanding of the system's behavior for a given phase.

In order to tackle these challenges of dynamic systems, we developed three major complementary technologies.

1.1 Discovery of System's State Determining Component Interplays

We first approach the problem of enumerating all the groups of cross–talking system components that could be associated with the system state. In dynamic physical systems, it is often a *coordinated*, *not independent*, action of several system components determines the system's state. The main challenge in enumerating of system state-determining component interplays is how to deal with the enormous number of system components (or features) that could easily reach thousands or even hundreds of thousands. Such enormous feature space could easily lead to the problem, coined by Bellman as "the curse of dimensionality" [8], that is, the number of system components ($n \approx 10,000's$) is significantly larger than the number of observational samples ($m \approx 100's$). Thus, the existing machine learning methods easily become hardly suitable for dealing with such underdetermined, or unconstrained, problems.

We propose an iterative, classification-based approach, called SPICE (*System Phase-related Interplaying Components Enumerator*), that comprehensively enumerates the set of feature subsets that discriminate between different system states (or classes). Given a set of observations about system components (features) with the corresponding assignment of the system's state (class), our method measures the importance of feature subsets to discriminate between system states. Despite combinatorial complexity of the problem, our method almost exhaustively exploits feature subsets by focusing on information-theoretic selection process. Our method rests on a hypothesis that if a subset of system components discriminates between system's functional states when considered altogether but not in any subset, then these components most likely form a cross-talking state-determining feature subset. It also places the contribution of an entire feature subset at the core of the analysis as opposed to the approaches that first evaluate the importance of individual features and then filter those that are associated with a particular system's state. It further filters those feature subsets that are statistically significant and are thus assumed to be relevant to the application domain.

SPICE can effectively handle a large number of features in a relatively small amount of time. This property enables SPICE to work well with underdetermined problems. Additionally, classification problems over climate and biological data are not limited to binary classification problems. SPICE can handle multi-class datasets, which make it suitable for multi-class classification problems. When applied to more than ten microarray and biohydrogen production data sets, SPICE successfully identifies cancer-related genes from various microarray data sets and finds enzymes or COGs associated with biohydrogen production and motility phenotype by microbial organisms. SPICE also improved the predictive skill of the system's state determination by more than 10% relative to individual classifiers and/or other ensemble methods. Further

details on SPICE approach appear in Chapter 2. And this work [174] has been published in the Journal of BMC Systems Biology.

1.2 Discovery of Community Dynamics in Evolutionary Networks

Although *graph-based* anomaly detection has been done on exploring three different types of anomalies including *anomalous nodes*, *novelty edges*, and *abnormal subgraphs*, little work has focused on dynamic communities. Communities in the real networks are changing over time. For example, in biological networks, a small variation in a gene-gene association community may represent an event, such as gene fusion [137], gene fission [137], gene gain [23], gene decay [99], or gene duplication [180], that would change the properties of the gene products (e.g., proteins) and, consequently, affect the phenotype of the organism. Detecting community dynamics is essential for a deeper understanding of the development and self-optimization of the system as a whole.

In contrast to the previous work on *graph-based* anomaly detection and community identification in static graphs or tracking conserved communities in time-varying graphs, we first introduce the concept of community dynamics, and then show that the baseline approach by enumerating all communities in each graph and comparing all pairs of communities between consecutive graphs is infeasible and impractical. We propose an efficient method for detecting and tracking community dynamics in evolutionary networks by introducing graph representatives and community representatives to avoid generating redundant communities and limit the search space. When applied to two real-world evolutionary networks, Food Web and Enron Email, significant and informative community-based anomaly dynamics have been detected in both cases.

Further details on our approach, including a theoretical proof that only six types of community dynamics are possible in simple undirected graphs, the decision rules for detecting the dynamic communities, and the completeness of the algorithm, appear in Chapter 3. The results of this work have been published at the *IEEE ICDM* conference [28] and in *Journal of Intelligent Information Systems* [27].

1.3 Discovery of Anomalous Communities in Contrasting Groups of Networks

The complex networks of a dynamic system can be partitioned into different groups corresponding to different system's states. For example, in a tropical cyclone (TC) prediction system, we

can build two different groups of climate networks, with one corresponding to strong TC years, another with corresponding to low TC years. Different groups of networks may exhibit different properties of the community structure. Detecting the anomalous communities in contrasting groups of networks can help us better interpret the physical relevance of the interplaying features determining the system's phases.

As a third component of our work, we design a contrast-based algorithm to detect anomalous communities in multiple groups of networks. We consider the anomalous communities in contrasting groups of networks as features to determine the system's phases. Because of the expensive computational cost of generating all communities, we use the system components (i.e., features), enumerated by SPICE, as seeds to efficiently generate the communities in the networks. Different groups of networks may exhibit different properties of the community structure. Instead of detecting conserved/stable communities as conventional algorithms, we focus on discovering the abnormal communities that contribute to different system phrases. The abnormal communities are further used to build the ensemble of classifiers for predicting the system states/phases. Further details on our algorithm and experimental results appear in Chapter 4.

Chapter 2

Discovery of System's State Determining Component Interplays

2.1 Introduction

Dynamic physical systems, such as the atmospheric-ocean system or biological cells, are inherently complex. This complexity arises from the selective and nonlinear interconnections of functionally diverse system components to produce coherent behavior. The key challenge is to reveal underlying simplicity from complexity [3]. Unlike the four Maxwell's equations describing all the electro-magnetic phenomena from "first principles," the fundamental rules that quantify the low dimensional behavior of such systems are yet to be discovered.

Complementing approaches based on first principles, where the underlying system model is described by a system of equations, the *data-driven* modeling of system behavior is a promising approach. It aims to interrelate data from disparate and noisy experiments and observations to find informative features and link them to formulate fundamental principles governing a complex behavior. This process frequently begins with a comprehensive *enumeration* of the system "components" (e.g., co-regulated proteins in a cell or climate indices in the atmospheric-ocean system) derived from experimental or observational data. Discovery of putative associations (e.g., teleconnections) between these "components" can then be used to design *in silico* system models (e.g., positive and negative feedbacks, information processing and signal transduction cascades) to better understand real system behavior.

To somewhat simplify this intricate process, data-driven characterization of a complex system behavior often starts with defining a target set of system's distinct states of interest and enumerating only those key system components that could be responsible for or contributing to the given state. For example, if the target state is ethanol production by microbial cells via

biomass degradation, then enumeration of state-related system components would identify all the proteins involved in degradation of cellulose to sugars, transport of these sugars through the membrane, and their fermentation to ethanol. Likewise, if the target state of the atmospheric-ocean system is the intensity of seasonal hurricane activity (i.e., above normal, normal, or below normal), then enumeration of hurricane activity-related system components would produce a set of putative system's parameters (e.g., temperature, precipitable water, pressure) associated with particular spatial regions on Earth that likely affect the magnitude of the system's response. Similarly, if the system's state of interest is cancer-prone cells in the human body, then enumeration of cancer-related cellular components would identify all the genes that are likely related to the expression of cancerous cellular phenotype.

The difficulty in enumerating *all* the state-related system components lies in dealing with the enormous number of system components (or features) that could easily reach thousands or even hundreds of thousands. Such enormous feature space could easily lead to the problem, coined by Bellman as "the curse of dimensionality" [8]. For example, high-resolution ocean-atmospheric models can be defined over the $1.4^\circ \times 1.4^\circ$ (latitude, longitude) spatial grid on the globe, several altitude levels, and a few dozen variables.

Likewise, the interaction between two biomolecules, such as protein-protein interactions, can be described through their set of contacting amino acid residues. A possible set of features to describe this interface is enormous due to a number of chemical identities of the contacting residue pairs (210 features from 20 amino acid types), orientation patterns of the contacting residues, and spatial arrangements of 3-5 contacting residues. One needs to select all those features that would provide clear differentiation between the true interfaces and merely feasible associations of two rigid bodies. In addition, hierarchical nature of most biological systems leads to "short- and long-range" interactions between the features. For example, hydrophobic residue pairs could enhance a propensity for other adjacent hydrophobic pairs ("short-range" feature correlation). On the other hand, highly specific residue interactions may be under selective pressure to fit into an overarching architectural motif (such as helix-turn-helix motif), thus contributing to "long-range" feature dependences.

Moreover, it is often the case that a *coordinated*, *not independent*, action of several system components determines what state a given system is in. A system response represents a complex process, involving a series of (frequently induced) interacting events. Such non-linear cooperative or competing interactions between the system components often form hierarchical functional modules (e.g., communities) that act not only on different spatial and temporal scales but also in response to fluctuations induced by endogenous and exogenous factors. Hence, the approaches that identify individual components that confer a given system state are likely not optimized to detect groups of such interplays between system components. Instead, there

is a need for methods that aim to enumerate all the groups of cross-talking system components that could be associated with the system state. We call this problem the **enumeration of system state-determining component interplays**.

To address this problem, we propose an iterative, classification-based approach that comprehensively enumerates the set of feature subsets that discriminate between different system states (or classes). Given a set of observations about system components (features) with the corresponding assignment of the system's state (class), our method measures the importance of feature subsets to discriminate between system states. Despite combinatorial complexity of the problem, our method almost exhaustively exploits feature subsets by focusing on information-theoretic selection process. Our method rests on a hypothesis that if a subset of system components discriminates between system's functional states when considered altogether but not in any subset, then these components most likely form a cross-talking state-determining feature subset. It also places the contribution of an entire feature subset at the core of the analysis as opposed to the approaches that first evaluate the importance of individual features and then filter those that are associated with a particular system's state. It further filters those feature subsets that are statistically significant and are thus assumed to be relevant to the application domain.

2.2 Related Work

To the best of our knowledge, the proposed problem of **enumerating statistically significant component interplays that are key contributors to the system's states** has not been addressed in literature. The problem resembles, yet with quite apparent distinctions, the problems of feature selection, phylogenetic profiling, network alignment, and frequent subgraph mining.

At a higher level, these problems could be divided into two major categories depending on whether pairwise relationships between system components are known. If they are defined, then the system could be modeled as a complex network, and multiple network alignment approaches [26, 25] that look for subgraphs that co-occur across multiple network instances for the same system's phenotype are putative candidates for the target component interplays. The key limitation of this strategy is that such approaches aim to identify the component groups that are present in all or most of a given set of network instances and would likely miss those that are only common to a subset of the instances. Likewise, they are not equipped with any means to suggest that these groups are specific to the target system phenotype and not common to multiple system phenotypes. While the former limitation is addressed by the approaches based on frequent subgraph mining [92, 109], similar comments would still hold for

the latter comment. In addition, the runtime for these approaches grows exponentially; even the most efficient ones, such as MULE [92] that enumerates maximal frequent edge sets, took almost 57 days for a set of 98 network instances (details available upon request). While efficient heuristics have been reported [128], they are tailored for specific network types (e.g., metabolic networks).

For the second category, the system is often represented by its set of components (i.e., features) that are defined over multiple instances (i.e., observations) for each of the finite set of system's distinct phenotypes. In this case, univariate approaches, such as those that, for the given feature, look for a strong correlation between its profile and the system's phenotype profile across multiple instances identify a set of putative candidates for component interplays. Different correlation measures, such as Pearson correlation, Mutual Information, Student's t-test, ANOVA, Wilcoxon rank sum, Rank products, and other univariate filter feature selection techniques can provide different candidate sets that could be further assessed with set-theoretical approaches to provide either higher specificity (i.e., intersection of sets) or higher sensitivity (i.e., set union).

A particular instance of such a strategy is phylogenetic profiling [136], where different organisms that exhibit various (but finite) phenotypes (e.g., aerobic vs. anaerobic growth) are considered as observations characterized by the the presence or absence of particular genes (or components). The underlying hypothesis behind this approach is that candidate genes are more likely to be present in phenotype-expressing organisms than in phenotype-non-expressing organisms due to an evolutionary pressure to conserve the phenotype-related genes [94]. While simple, fast, and effective [126] in finding *individual components* that are likely associated with the system's phenotype, such methods are quite limited in discovering of the *component interplays*.

Multivariate feature selection approaches could be considered as the closest approximation to the proposed problem. The multivariate feature selection approaches can be broadly divided into the following categories: (1) filter techniques (e.g., fast correlation-based algorithm [93]), (2) wrapper techniques (e.g., GA/KNN method (combining a Genetic Algorithm (GA) and the k-Nearest Neighbor (KNN) method) [96]), and (3) embedded techniques (e.g., random forest [42]). In filter techniques, the relevance of features is evaluated according to some metric, and the features with the top k ranking are then selected for further analysis. Filter feature selection techniques are simple, fast, and effective, but these techniques often ignore the correlations between different features. In biology, these correlations depict protein interactions and should not be ignored. Wrapper methods take the dependencies between the features into account, but suffer from overfitting problem. Additionally, they are often computationally expensive. Embedded methods can be far less computationally expensive than wrapper methods, but these

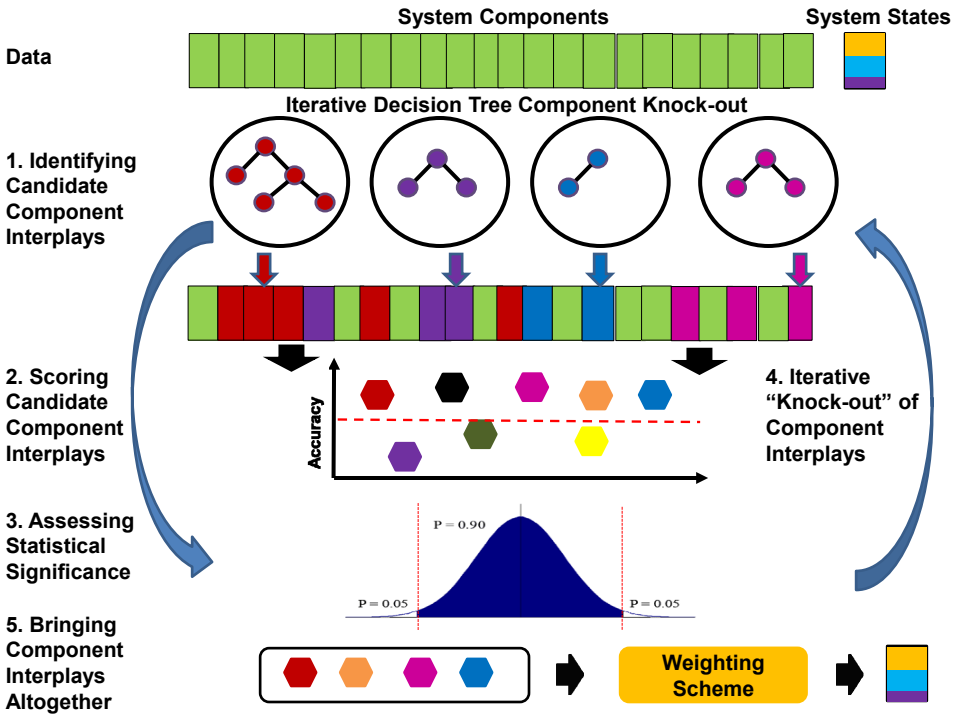


Figure 2.1: The overview of SPICE's key steps.

approaches are very specific to a given classification algorithm.

2.3 Method

The underlying assumption is that the associations between components are unknown. Effectively, the system under study is modeled by its set of components, and each component is characterized by a continuous or categorical attribute. Fig. 4.4 depicts the key steps underlying the proposed method, SPICE (*System Phase-related Interplaying Components Enumerator*). At a higher level, SPICE first identifies a candidate component (feature) set (Section 2.3.1), it then scores its state specificity-determining skill (Section 2.3.2) along with statistical significance assessment (Section 4.3.3). These three steps are repeated in an iterative fashion by "knocking out" the selected candidate component sets until the stopping criterion is met (Section 4.3.4). Finally, the ensemble of classifiers is formed to predict the system's state(s) given the values of all its component-interplay groups (Section 4.3.5). Next, we explain each of these steps in more detail.

2.3.1 Step 1: Identifying Candidate Component Interplays

We hypothesize that if the component is key to defining the system's state then its value distributions will be separable between the observations from different states. If the separation is strong, then such a component, alone, is likely able to discriminate system states. And almost any method, like entropy-based, would likely succeed in detecting those components. However, with real data sets such a strong separation is less likely. There are different reasons for such an assumption. For example, the evolution of system behavior may induce non-functional changes to the system components. For example, natural mutations in a protein sequence happen all the time; and if much time has passed since the functional divergence occurred, then functional state-preserving mutations must have been compensated by correlated mutations at other positions in the sequence to retain the protein function. As a result, one should strive for discovery of separation signals that while being weaker at the individual component level, they—as a group—should be able to discriminate between system states. Although the validity of this assumption is yet to be verified, numerous studies, such as those on correlated mutations [116, 4, 56, 152], provide indirect evidence in support of such a position. Another reason could be attributed to the noise in the data, for example, due to limited sensitivity of experimental devices. For instance, the chance of observing a strong signal about transient or transmembrane protein interactions from mass spectrometry experiments is low.

Thus, the effective analysis should not only include an individual component with a strong discriminatory signal, but also extend to a group(s) of interplaying components out of a set of thousands of components. This creates a multiplicity of possible combinatorial interplays to search for and excludes a possibility for a brute-force enumeration. Therefore, our goal is to provide a framework for automatic exploration of such combinatorial interplays that could offer both the computational efficiency and the application domain relevance.

In some cases, the domain knowledge may assist with constraining the search space of possible interplays. For example, functionally important amino acid residues, such as substrate binding sites, in a protein are likely located on the surface (i.e., clefts and cavities) of the protein 3-dimensional structure and not in the core. For a more general and domain-independent solution, however, the issue of properly constraining the search space still remains.

To address this issue, we propose to employ the multilevel paradigm via *divide-and-conquer* strategy. The multilevel paradigm is known for its effectiveness when solving very large-scale scientific problems. In the context of linear systems of equations, for instance, algebraic multi-grid methods, have been devised to solve linear systems by essentially resorting to divide-and-conquer strategies that utilize the relationship between the mesh and the eigen-functions of the operator. In the data analysis field, however, methods that take advantage of the multi-level paradigm are less explored. A few recent studies include [66, 67] as well as the top-down divisive

clustering (e.g., [68, 69, 72]) or spectral graph partitioning techniques (e.g., [73, 78]).

Specifically, the intuition behind our approach stems from the well-known concept of modularity, introduced by Hartwell *et al.* [63], as a generic principle of complex system's organization and function. These functionally associated modules often combine in a hierarchical manner into larger, less cohesive subsystems, thus revealing yet another the essential design principles of system organization and function–*hierarchical modularity* [122, 138]. Thus, our method first identifies modules of system components with putatively stronger associations within the modules than between the modules. This process *divides* all system components into modules that likely function together to define what state the system is in. The process further *conquers* each of these modules in order to refine the specificity of the inter-component relationships within the module. Fig. 2.2 shows an illustration of this divide-and-conquer approach to multilevel dimen-

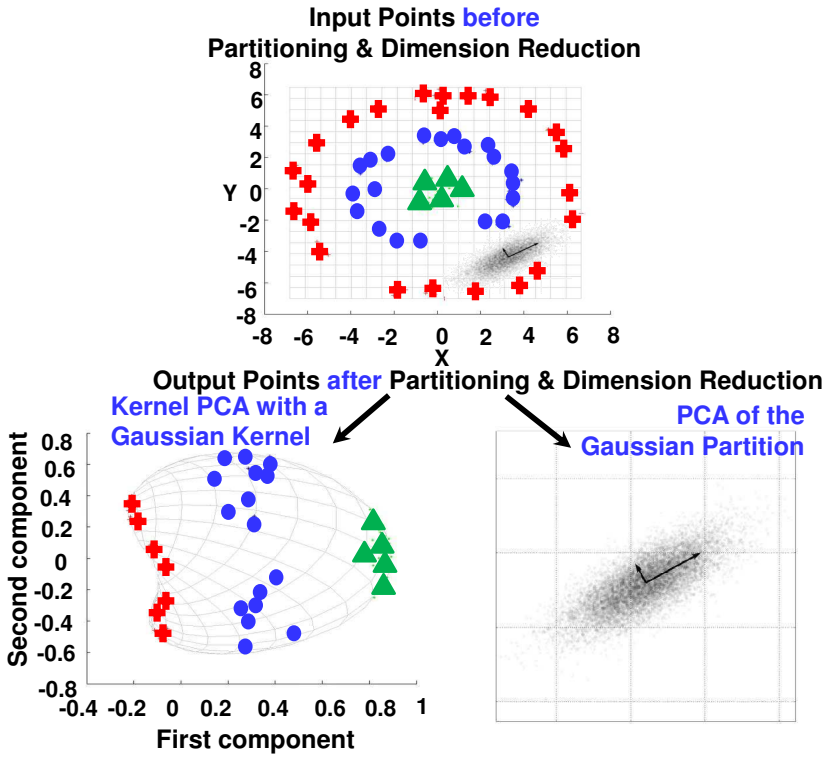


Figure 2.2: An illustration of divide-and-conquer strategy for multi-level dimension reduction.

sion reduction. The sample artificial input set shown contains two substructures: points from a multivariate Gaussian distribution (grey) and the three groups of colored points arranged into nested rings (top). (Note that the color of the points is only there to show how the data groups

together before and after the partition followed by dimension reduction). The standard PCA result performed on the monolithic set is mediocre, i.e., distinguishing the four different groups is impossible using only linear PCA. After partitioning the set, the "appropriate" technique is applied to each partition (bottom): the kernel PCA to the nested ring points (left partition) and the linear PCA to the Gaussian cluster (right partition). As a result, not only is the size of the data reduced for each partition, but also the four groups become distinguishable using only the first principal component.

Unlike the example in Fig. 2.2, in the context of our problem—*enumeration of statistically significant and application-relevant component interplays that are key contributors to the system's state*—we deploy decision tree based procedure for identifying the right partitions of the system's features and then apply the "appropriate" classification technique to each partition. The reason is that due to highly underdetermined nature of our problem, subsampling of the input data sample could possibly lead to an unreliable inference methodology. Likewise, due to a possibly non-linear interplay between the system's features, it would be more desirable to divide the system components into "blocks" with possibly stronger interconnects within the blocks and weaker inter-connects between the blocks. This strategy is inspired by the modularity principle of complex systems. Thus, a higher-level supervised separation of the high dimensional feature space into the rectangular shape hyperspaces is achieved via information-theory driven decision boundaries with a subsequent refinement of decision boundaries within the identified subspaces (see Step 2).

We propose a decision tree-based methodology for our feature space partitioning. The features in a decision tree are considered as one feature subset, and each feature is a system component. There are multiple reasons for why we choose decision tree based methodology, including (a) efficiency to process many features (unlike BBNs that are exponential in the number of features), (b) inherently multiclass by nature, and (c) the ability to handle continuous and multi-variate types of features (unlike NNs for which distance metrics are poorly defined for mixed data types), among others. We use the CART-decision tree algorithm [16] to select a set of discriminatory features from the available feature space. Basically, CART builds a decision tree by choosing the locally best discriminatory feature at each split step based on the Gini Index Impurity Function. To avoid overfitting, CART employs backward pruning to build smaller, more general decision trees. CART chooses features in a multivariate fashion, which allows the feature selection process to find a set of discriminatory features instead of considering one feature at a time.

More importantly, especially, in the context of underdetermined or unconstrained problems, CART's inherent feature pruning capability often leads to a fewer number of components, or smaller size modules. This is a desirable property for building a more robust classifier

downstream of our analysis pipeline (Step 2 and Step 5). Also, decision boundaries themselves could result in rules that are more interpretable and could provide additional insights to domain scientists on the magnitude of the feature attributes that affect a system's phenotype. The reason is that not only is it important to know what group of features is contributing to the system's phenotypic state but to what extent the feature values could change the system's phenotypic state. For example, if the expression of a particular gene becomes above a certain threshold, then this causes a "knock-out" of a particular metabolic pathway. With decision trees, the full feature space gets partitioned into hypersubspaces by the decision rules of the form of $a_i \leq f_i \leq b_i$. Once this high-level factors contributing to the system's states are learned, more complex (e.g., non-linear or conditional) relationships between the components in the group could be learned by more sophisticated classifiers, such as BBNs or kernel SVMs (see Step 2).

2.3.2 Step 2: Scoring Candidate Component Interplays

Candidate system's components identified in Step 1 are next assessed in terms of their *collective* ability to contribute to the system's states. Basically, the goal is to define a scoring function that could measure how well this group of components (features) discriminates between system phenotypic states. On the one hand, mutual information (MI) for an individual component could be used with its proper generalization to a group of components. However, robust probability estimation—an essential step in MI definition—requires a large sample size, which is often unavailable for underdetermined systems. Moreover, the generalized MI is biased toward the presence of a component in the group with high information content.

Due to these limitations, we define a scoring function in terms of classification accuracy provided by multivariate discriminant methods, such as SVMs, BBNs, neural networks, or decision trees. Specifically, we ask a question: if only a candidate component set were used to determine the system's phenotypic state, how much predictive skill this set could have. Since individual components within the candidate group could be related to each other in a complex manner, we first let a proper classifier (e.g., kernel SVM or BBN) learn this complex relationships from the entire group of features and choose the accuracy of the best performing classifier as the scoring measure of the putative components' interplay (see Line 5–6 in Algorithm 4). Note that different candidate groups may require different classifiers—the best performing classifier model is chosen both for Step 3 and for Step 5. [For our experiments, we use training accuracy.]

2.3.3 Step 3: Assessing Statistical Significance

Given a candidate feature set (Step 1) and its predictive skill score (Step 2), we next assess statistical significance of this score, namely, how likely a similar skill score could be observed at random. Specifically, we want to use the confidence level for the classification accuracy to sift phenotype-specificity determining component groups. It is expected that the statistically significant, highly scored component groups are application-significant. For example, a group of candidate genes could be biologically significant for biohydrogen production or cancer phenotype expression (see Sections 2.4.1).

It is worth observing that, generally, sample instances within the same system phenotype tend to be more similar than those from the other phenotypes. Hence, separation of feature value distributions between the samples from different states will be relatively clearer, and thus classification accuracy—as a measure of feature set's discriminatory power—can be biased. This implies that standard statistical testing like shuffling the phenotype (class) labels is not acceptable.

Thus, to provide a robust assessment of statistical significance, we measure an empirical $p-$value of each candidate feature set using the Monte Carlo procedure described in [177]. Specifically, for each feature subset, we randomly sample N feature subsets ($N = 1,000$) from the entire feature set of the same size as our candidate set, and compute the corresponding accuracies of the classifiers built from these feature sets. Then, we estimate an empirical $p-$value of the target feature subset as $p = (R + 1)/(N + 1)$, where N is the total number of random samples ($N \sim 1,000$) and R is the number of these samples that produce a test statistic greater than or equal to the value for the target feature subset. This corresponds to the percentile where our target score falls onto within the accuracy distribution for N samples. In our experiments, the selected $p-$value meets 95% confidence level. Algorithm 1 presents the detailed pseudo-code for the statistical significance assessment.

2.3.4 Step 4: Iterative "Knock-out" of Component Interplays

The candidate component-interplay group identified in Steps 1-3 is probably not the only group of system components that is responsible for a system's behavioral phenotypic state. For example, such a group of enzymes could contribute to a direct conversion of a particular type of sugar to ethanol, but there could still be other groups of genes required for ethanol production, such as regulators of these enzymes' expression in the cell, transporters of different sugars from the environment into the cell, or stress response regulators that detect toxin (i.e., ethanol) concentration level in the cell. In addition, if a subsystem is critical for a specific system's function, then it often gets replicated (e.g., multiple gene copy numbers in the genome) in the

Algorithm 1: Statistical significance assessment

Input:
F : entire feature set
F_c : candidate set of features
M_c: classifier model learned from F_c
D : entire training data set over F and system phenotypic states S
A : the best performing classifier
α : the required confidence level (e.g., 95%)
N : the number of samples for Monte Carlo estimation

Output:
An indicator of the quality of F_c

```
1  Let ε_c be the training accuracy of M_c
2  Let ε be an empty set
3  for N iterations do
4  |  Let F_r be a random sample of |F_c| features from F
5  |  Let D_r be the restriction of D to the features in F_r
6  |  Train a classifier M_r by running A on D_r
7  |  Calculate the training accuracy of M_r and add it to ε
8  Calculate a p-value for ε and ε_c
9  if p-value ≤ (1 − α/100) then
10 |  return PASS
11 else
12 |  return FAIL
```

complex system; this redundancy contributes to system's robustness. Therefore, our task is not simply to identify a single "best" group but, ideally, to enumerate them all.

The combinatorial nature of this task necessitates heuristic approaches. Our strategy is inspired by the way biologists often conduct their mutagenesis studies. Namely, they *knock-out* a group of genes (e.g., via gene deletion) and observe the *mutant* system's response. By analogy, our methodology *knocks-out* the selected candidate feature sets and proceeds with Steps 1-3 on the *mutant* system in an *iterative* fashion until some *stopping criterion* is met (see Line 2 in Algorithm 4). Under this approach, each iteration produces a subset of features out of the current feature set (see Line 4 in Algorithm 4), then removes these features from the set so that they can't be selected again (see Line 9 in Algorithm 4).

There are several different criteria that could be used to decide when to stop the iterative process. Ideally, one would observe a monotonically decreasing scoring value with the number of iterations and will stop once the score falls bellow a certain threshold. However, no theoretical grounds could be provided for such a monotonic behavior of the scoring function under the scenario of iterative feature set knock-outs. In fact, we empirically observed a fluctuating behavior of the scoring function with the number of iterations. Therefore, due to inherently high dimensional data, we set the threshold on the maximum number of iterations as our

stopping criterion. Line 2–2 in Algorithm 4 summarizes the aforementioned iterative knock-out procedure.

Algorithm 2: SPICE: System's state determining interplaying components enumerator

Input:
F : a set of components (features)
D : a set of training data over F
D': a set of test data over F
Y : a set of system states over D
A : basic classification algorithms
: (e.g., decision tree, SVM, Naïve Bayes, etc.)
Output:
Y' : predicted states for the test set D'
CIG: identified component-interplay groups

1 $CIG \leftarrow \emptyset$
`/* E: save the prediction results of candidate models */`
2 $E \leftarrow \emptyset$
3 **while** stopping criterion is not met **do**
`/* Run CART-decision tree to get a candidate component group */`
4 A pruned decision tree $T \leftarrow$ CART(D, Y)
5 Let F_c be a set of all components that belong to the internal nodes of T
6 $D_{F_c} \leftarrow$ Extract the data from D only with the components in F_c
7 Prediction skill score $\epsilon_c \leftarrow$ applying A to D_{F_c}
8 Let M_c be the classifier model learned from F_c
9 **if** ϵ_c meets the statistical significance criterion (see Algorithm 1) **then**
10 Let D'_{F_c} be the restriction of D' to the features in F_c
11 Predicted system states $Y'_c \leftarrow$ Apply M_c to D'_{F_c}
12 Add Y'_c to E
13 Add F_c to CIG
14 Remove features in F_c from F
15 Remove the data over feature F_c from D
16 Predict the class labels Y' based on a majority vote of the results in E
17 **return** Y' and CIG

2.3.5 Step 5: Bringing Component Interplays Altogether

While the enumerated set of putative system's component interplays is important in its own right (as illustrated in Section 4.4), here we combine them altogether by building an ensemble of classifier models from Step 3. Thus, unlike traditional classification methods that aim to find the single subset of features that offer the most optimum classifier performance, our goal is to enumerate suboptimal feature sets that could provide insights on what factors and their inter-

factor relationships could determine the specificity of the system's state. We then combine these subsystems through the framework of the ensemble methods in order to construct a system-level predictor of system's behavioral states.

In the last step (Step 5 in Figure 4.4), we need to combine the predictions of all the classifiers that pass statistical significance criterion (Step 3) to come up with the final prediction value. In order for the ensemble to make a prediction, each classifier is given a weighted vote, and the class with the most votes is the prediction of the ensemble (see Line 16 in Algorithm 4). We tested three possible weighting schemes: a simple majority voting scheme, in which every classifier is given equal weight; a training accuracy-based method, in which every classifier is weighted based on its training accuracy; and an internal cross-validation-based voting, in which each classifier is weighted by that model's cross-validation accuracy on the original training data.

Two of the key characteristics for building a robust classifier ensemble include (a) the diversity among the classifier models in the ensemble [105] and (b) the reasonably high accuracy of the individual members in the ensemble. In our case, the former is ensured due to our feature set knock-out strategy (Step 4) and the latter is guaranteed by a combination of decision-tree based feature enumeration (Step 1), the scoring function (Step 2), and the statistical significance assessment (Step 3) that, in combination, also reduce possible redundancy among the models and thus reduce the possible bias (e.g., due to a significantly large portion of highly similar models). By bringing the enumerated component interplays altogether (Step 5) a good ensemble of classifiers can be achieved (as illustrated in Section 4.4).

2.4 Results

The nature of the proposed methodology, SPICE, suggests that detected component interplays (Steps 1-4) (1) could play an important role in defining the specificity of the system's state(s); (2) would likely exhibit stronger inter-component relationships within the same group than between the groups and are functionally coherent, namely, act in a coordintaed manner to perform the state-specific function; and (3) collectively, could improve the predictive skill of the system's states (Step 5).

2.4.1 State-Specificity Determining Components

Groups of Enzymes Associated with Biohydrogen Production

Biological hydrogen is a promising renewable energy source [85], which can be generated by utilizing one of three metabolic processes: light fermentation, dark fermentation, or photosyn-

thesis [107]. To date, a number of phylogenetically diverse microorganisms have been identified as hydrogen producing. Such organisms include photosynthetic bacteria, nitrogen-fixers, and heterotrophic microorganisms [125]. In order to generate hydrogen, these organisms may rely upon one or more metabolic routes. As such, the biohydrogen production phenotype provides an opportunity to evaluate the capabilities of SPICE to handle a relatively complex phenotype. Identification of phenotype-related components was based on the assumption that if a component (i.e., a group of enzymes in a metabolic process) is specific to biohydrogen production, then it is likely evolutionarily conserved across H_2-producing organisms, and it is absent in most H_2-non-producing ones.

Our first experiment includes the data about 17 H_2-producing and 11 H_2-non-producing microorganisms and compares SPICE's performance against the two commonly used statistical methods: Mutual Information (MI) and Student's t-test, and one multivariate feature selection approach: SVM recursive feature elimination (SVM-RFE). Among 17 H_2-producing microorganisms, four microorganisms utilize bio-photolysis, five microorganisms utilize light fermentation, and eight microorganisms utilize dark fermentation. 11 microorganisms are listed as non-hydrogen producing because they are not associated with hydrogen production based on literature review, or they lack hydrogenase [76], one of the key enzymes involved in hydrogen production. All microorganisms used in this experiment were verified as completely sequenced using the NCBI database. The input to SPICE is a matrix, with the enzyme EC numbers along the rows, 28 organisms (hydrogen producing and non-producing) along the columns, and the entry in each cell (i, j) is the copy number for enzyme i in organism j. The last row of the matrix includes information about the organism's ability to express the hydrogen production phenotype.

The mutual information method [81] assesses correlation between the enzyme's phylogenetic profile and the organism's H_2-production profile across multiple organisms. In addition, it reports a significance threshold by shuffling the enzyme profile vectors and calculating the mutual information with the organism's phenotype profile. Only those enzymes, whose mutual information values lie above the confidence cutoff are reported.

The Student's t-test is another statistical method to identify phenotype related enzymes, where we utilize the enzyme phylogenetic profiles alone to measures statistical bias of enzyme copy numbers in one phenotypic group of organisms vs. the other. The test results are filtered so that only enzymes with the p-value less than 0.05 are considered significant.

Guyon *et al.* [60] proposed the SVM-RFE algorithm to rank the features (enzymes) based on the value of the decision hyperplane given by the SVM. The features with small ranking scores are removed. The top 240 enzymes (out of 1,229 enzymes) are considered significant.

Figure 2.3 and 2.4 show the pathway and key enzymes for hydrogen production from the

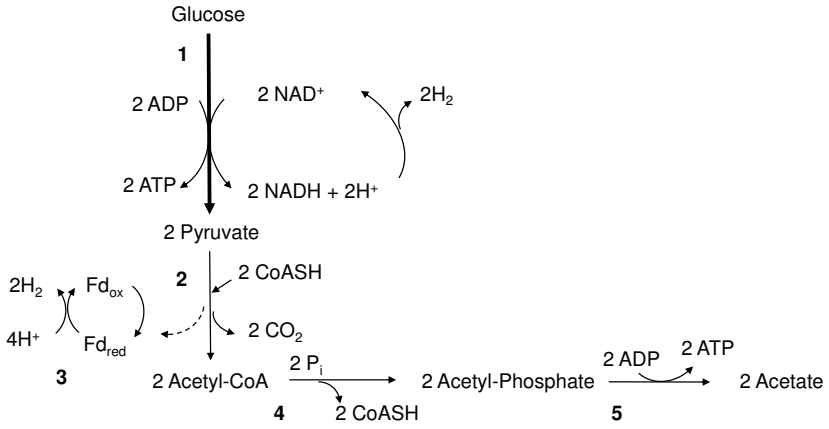


Figure 2.3: Fermentation of glucose to generate acetate. Schematic of key metabolic pathways for hydrogen production in *Clostridium acetobutylicum*. Arrows with larger width indicate a series of reactions. Arrows with narrow width indicate individual reactions. Enzymes: 1, glycolytic enzymes; 2, pyruvate ferredoxin oxidoreductase (E.C. 1.2.7.1); 3, hydrogenase (E.C.1.12.7.2); 4, phosphotransacetylase (E.C. 2.3.1.8); 5, acetate kinase (E.C. 2.7.2.1).

fermentation of glucose to acetate (Figure 2.3) and butyrate (Figure 2.4) in *Clostridium acetobutylicum*. Within this process, glucose is broken down through a series of glycolytic enzymes to generate pyruvate. Pyruvate is then converted to acetyl-CoA through the action of pyruvate ferredoxin oxidoreductase. During this step, hydrogen gas is produced when pyruvate is oxidized, thus resulting in the formation of CO_2 plus H_2. Production of hydrogen via this route is mediated through two enzymes—pyruvate ferredoxin oxidoreductase and hydrogenase. Acetyl-CoA generated produced from pyruvate can then enter a number of pathways, including the acetate and butyrate formation pathways.

While production of hydrogen occurs predominately during formation of Acetyl-CoA and not in the secondary pathway (e.g., conversion of Acetyl-CoA to acetate), acetate and butyrate fermentation pathways play an important role in the overall yield of hydrogen by microorganisms. In metabolic engineering studies, the goal is to generate the highest theoretical yield of hydrogen through alteration of metabolic routes or key enzymes related to hydrogen production.

For enhanced hydrogen production, acetate is the desired end product because of its higher hydrogen yield compared to other by-products, such as butyrate [65, 103]. Specific differences in conversion efficiencies can be observed by comparing the two chemical reactions below:

$C_6H_{12}O_6 + 2H_2O \rightarrow 2CH_3COOH + 2CO_2 + 4H_2$: glucose into acetate

$C_6H_{12}O_6 \rightarrow CH_3CH_2CH_2COOH + 2CO_2 + 2H_2$: glucose into butyrate

The first reaction shows that the maximum theoretical hydrogen yield is 4 H_2 per mol of glucose produced when acetate is the end product [95, 88], compared to a maximum theoret-

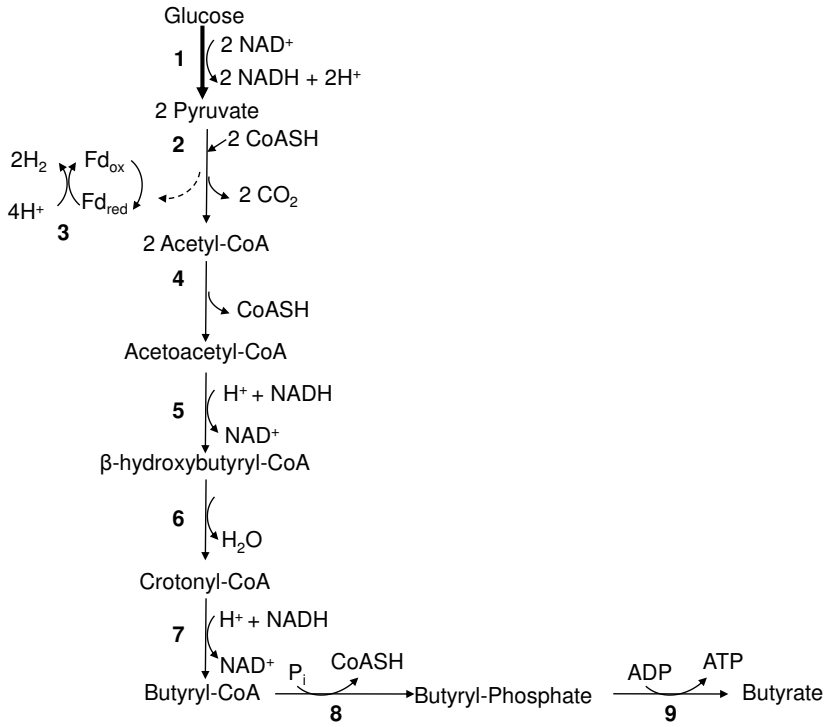


Figure 2.4: Fermentation of glucose to generate butyrate. Schematic of key metabolic pathways for hydrogen production in *Clostridium acetobutylicum*. Arrows with larger width indicate a series of reactions. Arrows with narrow width indicate individual reactions. Enzymes: 1, glycolytic enzymes; 2, pyruvate ferredoxin oxidoreductase (E.C. 1.2.7.1); 3, hydrogenase (E.C.1.12.7.2); 4, acetyl-CoA acetyltransferase (thiolase) (E.C. 2.3.1.9); 5, β-hydroxybutyryl-CoA dehydrogenase (E.C. 1.1.1.157); 6, crotonase (E.C. 4.2.1.55); 7, butyryl-CoA dehydrogenase (E.C. 1.3.99.2); 8, phosphotransbutyrylase (E.C.2.3.1.19); 9, butyrate kinase (E.C. 2.7.2.7). Abbreviations: Ferredoxin (Fd); Coenzyme A (CoASH).

ical hydrogen yield of 2 H_2 with butyrate as the end product [65, 97, 168]. During acetate and butyrate formation, 2 mols of hydrogen are generated during reaction 3 when pyruvate ferredoxin oxidoreductase reduces ferredoxin (Fd) and hydrogenase immediately oxidizes it to generate H_2 (Figure 2.3 and 2.4). When acetate is the only end product as depicted in 2.3, then additional hydrogen is produced when $2NAD^+$ is reduced to form $2NADH + 2H^+$ (reaction 3). An illustration of the two reactions is shown in Figure 2.3 (acetate) and Figure 2.4 (butyrate).

Due to the importance of acetate and butyrate production in the generation of hydrogen production, we evaluated the ability of SPICE to identify these two pathways. Results show that SPICE identified all of the acetate pathway's constituent enzymes, including acetate kinase (E.C. 2.7.2.1), as being significant. In contrast, the Student's t-test and the MI method did not find any of the enzymes, and SVM-RFE detected acetate kinase. Additionally, all five enzymes active in the butyrate pathway [103] were found by the SPICE method. Among these, only three were discovered by the SVM-RFE, two were found by the Student's t-test and none by the MI method.

Hydrogen Production in Association with Formate: Within facultative anaerobes like *Escherichia coli*, hydrogen gas may be produced directly through the production of formate. In this pathway, pyruvate is converted to formate and acetyl-CoA with the use of pyruvate formate lyase (E.C. 2.3.1.54) [61]. The formate hydrogen lyase complex made up of formate dehydrogenase and ferredoxin hydrogenase breaks down the formate into hydrogen gas and carbon dioxide [103]. In this study, pyruvate formate lyase was found by the SPICE method to be significant.

Table 2.1: H_2-related enzymes detected by different methods

Pathway	Enzyme	Enzyme Name	t	MI	SVM-RFE	SPICE
Acetate	2.7.2.1	acetate kinase			+	+
Butyrate	1.3.99.2	butyryl-CoA dehydrogenase			+	+
	2.7.2.7	butyrate kinase	+		+	+
	1.1.1.157	3-hydroxybutyryl-CoA dehydrogenase				+
	2.3.1.19	phosphate butyryltransferase	+			+
	2.3.1.9	acetyl-CoA C-acetyl-transferase			+	+
Formate	2.3.1.54	pyruvate formate lyase				+

Note: t: Students' t-test; MI: Mutual Information.

Table 2.1 shows that SPICE detected all the enzymes specific to the three pathways in facultative anaerobes, such as *Escherichia coli*, while mutual information could not even discover a single enzyme, Student's t-test could only detect 2 enzymes, and SVM-RFE could find four out of 7 enzymes. Thus, SPICE outperformed, in terms of sensitivity, the existing state-of-the-

art methods based on Student's t-test, MI, and SVM-RFE. The enzymes identified by SPICE are next described in the context of their corresponding metabolic pathways.

COG Modules Corresponding to Biohydrogen Production

To expand our study beyond metabolic subsystems to include possible regulators, transporters, and others, in our next experiment, we replace enzymes in the matrix with the clusters of orthologous groups (COGs) [151]. We obtain COG–organism association information from the STRING database.

SPICE was able to identify COG modules that are known to be associated with hydrogen production based on our literature review and prior knowledge. Next, we will briefly summarize some of these modules.

COG Modules Related to Nitrogenase

In addition to the metabolic pathways described above, other key enzymes are known to be associated with hydrogen production in a number of microorganisms [162, 18, 104]. Examples of such enzymes include nitrogenase and hydrogenase enzyme complexes. Hydrogen producing organisms capable of fixing nitrogen contain enzyme complexes, termed nitrogenases. Within nitrogenase complexes, nitrogen gas is converted to ammonia, inadvertently resulting in the production of hydrogen gas as a byproduct [125, 18].

Evaluation of the COG modules generated by SPICE indicated the presence of two modules, each containing an essential component of enzyme complex nitrogenase. In the first module, two COGs (COG2710 and COG0120) were identified. COG2710 is associated with expression of the molybdenum–iron protein (NifD) [125] and COG0120 is associated with the protein—Ribose 5-phosphate isomerase (RpiA). NifD protein is one essential component of nitrogenase, serving as the binding site for substrates during nitrogen-fixation [125, 124]. RpiA takes a vital part in carbohydrate anabolism and catabolism through its participation in the Pentose Phosphate Pathway (PPP) and Calvin Cycle [181]. In addition, studies of central metabolism indicate that RpiA is a protein highly conserved across many microorganisms [181]. However, in this study, RpiA was paired with NifD, suggesting that both proteins may be associated with nitrogen-fixation, hence biological hydrogen production. In terms of hydrogen production, metabolism of and the ability to metabolize specific carbohydrates play an indirect role in the over-production of hydrogen. One example is the *C. butyricum*. Metabolic studies of the *C. butyricum* demonstrate the ability of this bacterium to digest a variety of carbohydrates and to produce hydrogen via degradation of carbohydrates [39].

Another role RpiA may play is the production of NADPH required for fixing nitrogen [9]. In nitrogen fixers, the oxidative pentose phosphate cycle has been reported as active. During oxidative PPP, Riboluse-5-phosphate is converted to ribose-5-phosphate by Rpi. During this

reaction, NADPH is generated, thus allowing for N assimilation, N-fixation, and production of hydrogen.

The second nitrogenase-related module identified by SPICE contains COG1348 (NifH) and COG3883 (Uncharacterized). Similar to NifD, NifH is also considered to be an essential component of nitrogenase. It is responsible for assisting with the biosynthesis of co-factors for NifD [140]. COG3883 is uncharacterized. While we cannot predict the role of the protein from this module, its presence suggests that it is either associated with the nitrogen fixation or hydrogen production phenotype.

COG Modules Corresponding to Hydrogenase

Hydrogenase enzyme complexes are key enzymes involved in the uptake and production of biological hydrogen [162]. Analysis of hydrogenase enzymes have identified three different types, each associated with a number of accessory proteins necessary for activation [162, 161]. These include the [NiFe]-hydrogenase, [FeFe]-hydrogenase, and non-metal containing hydrogenase enzyme [162]. Due to the importance of hydrogenase in both hydrogen production and hydrogen uptake, several studies have examined the role of hydrogenase enzymes in a number of different hydrogen-producing organisms [2, 62]. These studies have found many microorganisms, including *Clostridium acetobutylicum*, capable of having both hydrogen uptake (e.g., [FeFe]-hydrogenase) and hydrogen evolving enzymes (e.g., [NiFe]-hydrogenase). In this study, SPICE predicted the presence of both hydrogen uptake and hydrogen evolving enzymes as related to the hydrogen production phenotype. Categorization of hydrogen uptake hydrogenases may be due to the absence of hydrogenase in microorganisms present in our data set.

In this study, SPICE identified one module containing a hydrogen evolving hydrogenase. Within this module two COGs, COG4624 (iron only hydrogenase) and COG3541 (predicted nucleotidyltransferase) were present. The protein ID for COG4624 was not identified in the literature review; however, [Fe]-hydrogenases are responsible for producing hydrogen [163]. Nucleotidyltransferases are proteins involved in a number of biological processes ranging from DNA repair to transcription [102]. Since these proteins are generally involved in DNA and RNA-related processes, it is unclear why a predicted nucleotidyltransferase was paired with hydrogenase. To understand the interaction between these two proteins, experimental molecular analysis is necessary.

Another COG module found by SPICE contains COG0068 and COG0025, which are associated with expression of two hydrogenase uptake proteins—hydrogenase maturation factor (HypF) and NhaP-type Na+/H+ and K+/H+ antiporters (Nhap). HypF has been found as a carbamoyl phosphate converting enzyme (or an auxiliary protein) involved in the synthesis of active [NiFe]–hydrogenases in *Escherichia coli* and other bacteria [115]. NT01CX_0020, an orthologous group of COG0025, is associated with expression of sodium/hydrogen exchanger

protein (NHE3). NHE3 has been found to play an important role in hydrogen production of *Acidaminococcus fermentans*, *Escherichia coli* and bacterial communities within a dark fermentation fluidised-bed bioreactor [75, 91, 5].

SPICE also identified three other types of hydrogenase maturation proteins—HypC, HypD, and HypE. COGs corresponding to these proteins are COG0298 (HypC), COG0409 (HypD), and COG0309 (HypE). Understanding complexes, such as uptake hydrogenase enzymes, is important for deciphering regulatory mechanisms and activity of these key enzymes. For example, in studies evaluating accessory proteins present in [NiFe]-hydrogenase complexes, HypCDEF proteins are described as regulators for maturation of uptake hydrogenase through participation in development of the active center [162, 1]. If one of the Hyp proteins is missing, the entire complex is inactivated.

In H_2–producing microorganisms such as *Escherichia coli*, hydrogenase maturation proteins act as regulators for maturation of uptake hydrogenase in development of the active center [162, 18]. Regulation is conducted by inserting Fe, Ni, and diatomic ligands of HypA–F proteins into the hydrogenase center for activation and maturation [133]. To carry out this process, HypE and HypF are in charge of synthesis and insertion of Fe cyanide ligands into the hydrogenase's metal center, and HypC and HypD are responsible for construction of the cyanide ligands [18, 11].

In addition, SPICE identified two hydrogenase proteins associated with anaerobiosis [162]. They are COG0374 (HyaB) and COG0680 (HyaD). Unlike the Hyp proteins, which are accessory proteins involved in the assembly of the metallocenters, Hya proteins are responsible for the maturation of hydrogenase-1 [163].

Other COG Modules Related to Biohydrogen

Other biohydrogen production-related COGs, such as COG0374, COG0375, COG3261, COG0680, COG4624 and others, shown under the hydrogenase category in STRING database are detected as part of other modules by SPICE. As mentioned earlier, hydrogenase is one of the key proteins (or enzymes) involved in hydrogen production and uptake [76].

Motility-related COG Modules

For a large-scale experiment, we set up another experiment on a different phenotype—motility. A total of 141 organisms including 56 non-motile organisms and 85 motile organisms were chosen from Slonim *et al.* [136]. For p-value of less than 0.01, SPICE detected 96 modules.

One of the motility phenotype-related COG modules contained COG1338, COG0265, COG1484, and COG3420. Among the four COGs, COG1338, whose function is associate with the expression of flagellar biosynthetic protein (Flip), has a high correlation with flagellar assembly pathway [98]. Flagellar assembly pathway, which enables the movement of microorganisms, is

well-known to be important for bacterial motility [98, 120]. Proteins associated with the other three COGs include uncharacterized serine protease (YyxA) and two hypothetical proteins. YyxA in a motile organism, *Bacillus amyloliquefaciens*, has a similar phylogenetic profile to chemotaxis-related proteins [134]. Chemotaxis pathway, which is also important for bacterial motility, determines how the microorganism moves according to its environment [136]. Chemotaxis pathway and flagellar assembly pathway function together to guide bacteria's direction of movement [136]. The phylogenetic profile of the other two hypothetical proteins (associate with COG1484 and COG3420) are shown to be correlated with the pattern of motility across many bacterial genomes [136].

Additionally, SPICE enumerated other COG modules that contained other known flagellar-related COGs like COG1516, COG1345, and COG1815 and other known chemotaxis-related COGs such as COG0840, COG0643, and COG0835, supported by literature [136, 98, 120]. Besides flagellar-related and chemotaxis-related COGs, type III secretion system-related COGs, such as COG1766, COG1684, COG1987, and COG1338, were also found in some of our enumerated modules. The type III secretion system is found to be highly correlated with bacterial motility, because some of its protein structure is very similar in structure, function, and gene sequence to the flagellar assembly system [10, 98].

Cancer-related Genes

Identifying *all* the genes that could discriminate tumor cells from normal cells in microarray gene expression data is non-trivial [148]. Again, the task is *not* to find a *single* "best"-discriminating gene set, but enumerate as many cancer-related genes and groups of genes as possible provided they are associated with cancer expression phenotype; this task is becoming particularly important in the context of personalized medicine.

Leukemia data was selected to show the effectiveness of our method to detect phenotype-related gene modules in biological networks. Leukemia data can be downloaded from Broad Institute Cancer Program Data (http://www.broadinstitute.org/cgi-bin/cancer/datasets.cgi). It contains 72 measurements for the expression of 7,129 genes, corresponding to the samples taken from bone marrow and peripheral blood. Out of these samples, 47 samples are classified as ALL (Acute Lymphoblastic Leukemia), and 25 samples are classified as AML (Acute Myeloid Leukemia).

The first 5 models built by SPICE identified a total of 11 genes supported by available literature on Leukemia cancer and information from NCBI database (Table 2.2). Specifically, KIAA0016 (Zyxin) gene is highly (hardly) correlated with anti-cancer agents [19]. Other genes (e.g., ID's of 1834, 2288, 2, and 1882) are informative for Leukemia cancer diagnostics [30]. These gene groups would be difficult to detect with a single iteration step.

Table 2.2: Cancer-related genes found by SPICE

Model ID	Gene ID	Gene description
Model 1	210	KIAA0016
	4847	Zyxin
Model 2	4	AFFX-BioC-5_at
	760	CYSTATIN A
Model 3	96	WUGSC
	1834	CD33 CD33 antigen
Model 4	129	Niemann-Pick C disease protein mRNA
	2288	DF D component of complement
Model 5	2	AFFX-BioB-M_at
	3	AFFX-BioB-3_at
	1882	CST3 Cystatin C

Note: More cancer-related genes are found by other models.

2.4.2 Topological Connectivity of Components

We analyzed topological connectivity of the components via *cliquishness* value. Given a component group C with n enzymes and an underlying biological network O, the cliquishness is the ratio of the number of edges present between the enzymes to the total number of possible edges, $\frac{n*(n-1)}{2}$.

The underlying biological network O is the organism specific functional association network from STRING [80]. Each enzyme in the component group C can be mapped to one or more genes in $V(O)$, and so the component group C is represented as a set of genes G. The induced subgraph over G from O is used to calculate the cliquishness of C. Our assumption is that a high cliquishness value indicates a possible interplay.

We used the biological networks of two *dark fermentative hydrogen producing* organisms, *Clostridium perfringens ATCC 13124* (cpf) and *Clostridium acetobutylicum ATCC 824* (cac). Out of the 65 statistically significant components that were enumerated for the dark fermentative hydrogen producing phenotype, we only considered those component group C with the corresponding gene set G of *size* > 1. Using the *Clostridium perfringens* network, nearly 50% of the components were statistically significant (p-value ≤ 0.05) in terms of connectivity (cliquishness). Using the *Clostridium acetobutylicum* network, nearly 56% were statistically significant.

2.4.3 Functional Enrichment of Component Interplays

We also performed a functional enrichment analysis on the discovered components, i.e, to test if the enzymes identified as part of a component group C are also functionally related. As

a first step, we mapped the enzymes in C to organism specific gene set G from *Clostridium perfringens*. Each G and the *Clostridium perfringens* functional annotation from the JCVI comprehensive microbial resource [119] were given as input to the GO TERM FINDER [13], a functional enrichment analysis tool. Nearly 54% of the components were functionally coherent (p-value ≤ 0.05).

Some components had zero cliquishness but were found to be significant via functional enrichment analysis. Also, there were components that had statistically significant connectivity but poor functional enrichment. Hence, topological connectivity and functional enrichment analysis are complementary evidences. Thus, we could provide evidence for a possible interplay if one of these clues predicts the component to be significant. Under this assumption nearly 65% of the components were significant (p-value ≤ 0.05).

We did not perform functional enrichment using *Clostridium acetobutylicum*, since only a small percentage of genes from this organism had any annotation.

2.4.4 Predictive Skill of System's States

Data: Eight publicly available multi-phenotype-genotype datasets are used in this study. Table 2.3 and Table 2.4 summarize some characteristics of these datasets, their sources, and the best-to-date performance reported in literature. For comparison purposes, the last column indicates SPICE's performance.

Table 2.3: Microarray data sets

Dataset	Features	Samples	Classes
Leukemia	7129	72	2
Colon cancer	2000	62	2
B-cell lymphoma	4026	96	2
Prostate	6033	102	2
Lymphoma_3class	4026	62	3
SRBCT	2308	63	4
CNS*	74	60	2
Prostate outcome*	208	21	2

Notes: *: Discretized data.

Evaluation Methodology: For two-class, 10-fold cross-validation are employed. 10-fold cross validation has been proved by Witten and Frank [169] to be a good way to evaluate the performance of a classifier. In 10-fold cross-validation, the original data is partitioned into 10

Table 2.4: Performance comparison on microarray data sets

Data	Dataset Source	CV	Acc.^{r} (%)	Acc.$^\flat$ (%)	SPICE (%)
Leukemia	[148]	10-fold	91.2	97.14 [114]	98.6
Colon cancer	[41]	2:1 RP	87.14	87 [182]	89
B-cell lymphoma	[167]	5:3 RP	92.1	93.55 [165]	94.7
Prostate	[148]	10-fold	73.5	87 [77]	93.1
Lymphoma_3class	[41]	2:1 RP	99.05	97.36 [147]	100
SRBCT	[41]	2:1 RP	98.7	98.7 [182]	98.7
CNS*	[148]	10-fold	88.3	75 [36]	96.7
Prostate outcome*	[148]	10-fold	85.7	90 [38]	100

Notes: CV: Cross-validation; RP: Random partition;
^{r}: Accuracy from source reference; $^\flat$: Accuracy reported in a recent literature.

different subsets. Each of the 10 subsets is used as the test set, and nine other subsets are used as training set. For multi-class datasets, 3-fold cross validation is used to ensure that each subset can have all different classes of samples.

Bootstrapping validation, via commonly used bootstrap estimators, e0 bootstrap and .632 bootstrap [48], is also applied. In e0 bootstrap, the training data consists of n instances by re-sampling with replacement from the original data of the same size of n. And the test data is the set difference between original data and training data. Thus, if the training data has j unique instances, then the test data will be the other $n - j$ instances on the original data. The error rate on the test data is treated as the e0 estimator, while the .632 bootstrap also takes the training error into consideration, and uses the linear combination of $0.368 * \epsilon + 0.632 * e0$ as the estimated error rate, where ϵ is the training error. For good error estimation, we use ≈ 200 iterations [48] and report the average error rate.

Bagging [14], boosting [51], random forest [15], nearest shrunken centroid method (PAM) [165], and random forest variable selection (varSelRF) [43] ensemble learning techniques are employed as benchmark methods. The ensemble size used for these methods is the same as the one used for SPICE.

We utilize different skill metrics including accuracy, sensitivity, specificity, precision, F_1-measure, variance, Heidke Skill Score (HSS) [83], Peirce Skill Score (PSS) [83], and Gerrity Skill Score (GSS) [83]. Accuracy is defined as the ratio of the number of correctly classified data points to the total number of data points in the test set. The HSS measures how well a forecast did as to a randomly selected forecast. PSS, also called "true skill statistic," is another popularly skill score computed by the difference between the hit rate and the false alarm rate. GSS, also known as "threat" score or critical success index, is a particular useful measure of skill

for situations where the occurrences of the event to be forecast are substantially less frequent than the non-occurences [83].

Skill Metrics Evaluation: Fig. 2.5 shows cross validation accuracy of SPICE compared

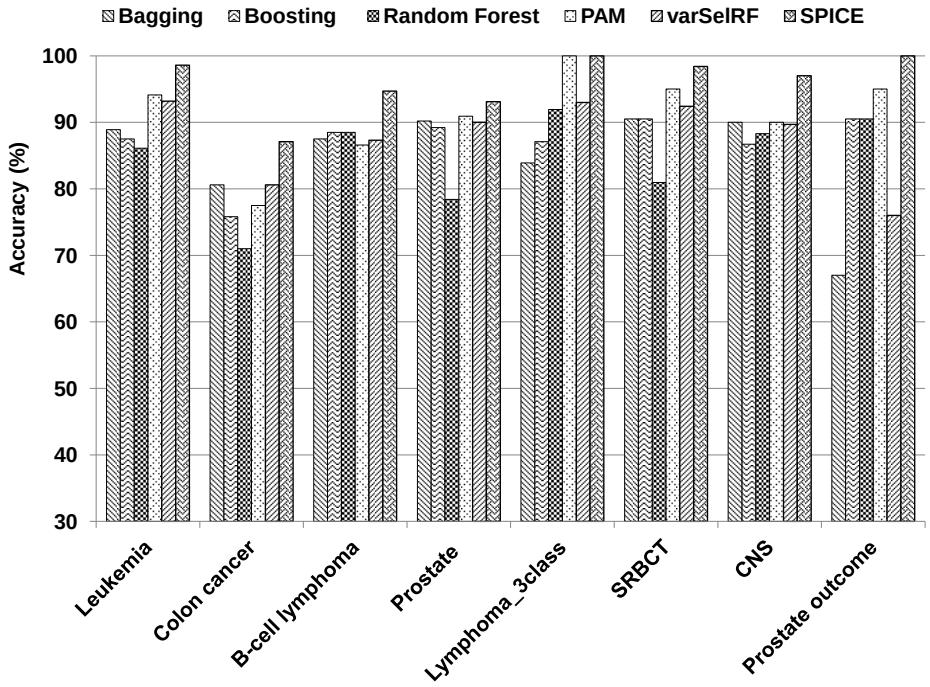


Figure 2.5: Comparison of prediction accuracy of SPICE to other ensemble classifiers on ten datasets

to bagging, boosting, random forest, PAM, and varSelRF ensemble methods. We report the accurate results of bagging, boosting, random forest, PAM, and varSelRF by using the default parameters. CART decision tree is used as the base classifier for bagging, boosting, and SPICE. To be consistent, we use 11 iterations as the stopping criterion (or the maximum ensemble size) for all the methods. SPICE outperforms bagging, boosting, random forest, PAM and varSelRF by up to 33%, 13%, 18%, 10%, and 24%, respectively. Table 2.5 summarizes SPICE's skill on two-class microarray data using five metrics: accuracy and its variance, sensitivity, specificity, precision, and F_1-measure; it also reports an average number of features per model. Table 5 summarizes SPICE's skill on multi-class microarray data using five metrics: accuracy and its variance, HSS, PSS, and GSS.

Different Weighting Schemes' Test: One factor that may influence the results of SPICE

Table 2.5: Performance on two-class data sets

Metric	Leukemia	Colon	B-cell lymphoma	Prostate
Accuracy	0.99	0.87	0.95	0.93
Variance	0.001	0.001	0.000	0.000
Sensitivity	0.98	0.90	1	0.9
Specificity	1	0.82	0.85	0.96
Precision	1	0.90	0.92	0.95
F_1-measure	0.99	0.90	0.96	0.93
Features	2.23	2.61	2.52	3.33

Table 2.6: Performance on multi-class data sets

Metric	Lymphoma_3class	SRBCT
Accuracy	1.0	0.98
Variance	0.000	0.005
HSS	1	0.98
PSS	1	0.981
GSS	1	0.98

method is the weights assigned to different candidate classifiers in the ensemble for determining the phenotype. Here, we test three different weighting schemes described in Section 4.3.5: majority voting, training accuracy-based voting, and internal cross-validation-based voting. The experimental results show that there is no bearing on prediction accuracy by choosing different weighting schemes for a majority of microarray datasets, although the training accuracy-based voting and internal cross-validation-based voting performed slightly better (3–5%) than the majority voting scheme on few datasets like the B-cell lymphoma dataset. However, all weighting schemes highly outperformed any single classifier in the ensemble.

Robustness Assessment: To assess robustness, we applied bootstrapping using both e0 and .632 bootstrap estimators with 200 bootstrapping trials. Bootstrapping is applied to all three categories of data sets. Leukemia data is the original 2-class data without any preprocessing, CNS data is the discretized data, and Lymphoma_3class data is multi-class data with logarithmic transformation and standardization. Table 6 shows that SPICE provides bootstrap error rates comparable with cross-validation results.

Generalization: SPICE can be considered one of meta-learning ensemble algorithms [105], because SPICE can employ an arbitrary base classifier. Table 2.8 shows its effectiveness compared to a single classifier using different base classifiers on the Colon cancer dataset with the 10-fold cross-validation. SPICE improves the prediction accuracy of a single classifier, namely by about 30%, 14%, and 7% for Naïve Bayes, CART decision tree, and linear SVM, respectively.

Table 2.7: Bootstrapping performance

Data	e0	ϵ	.632	10-fold cross validation
Leukemia	0.037	0	0.024	0.014
CNS	0.044	0.031	0.007	0.030
Lymphoma_3class	0.027	0	0.017	0.000

Thus, SPICE can be applied to improve some base classifiers other than decision tree, which makes SPICE more useful.

Table 2.8: Accuracy improvement over a single base classifier

Classifier	Single classifier	SPICE
Decision Tree (CART)	0.73	0.87
Naïve Bayes	0.57	0.87
Linear SVM	0.82	0.89

2.5 Conclusion

In this chapter, we addressed the important and challenging problem of enumerating statistically significant and application-relevant component interplays that are key contributors to the system's phases or states. We presented SPICE, an effective, iterative feature subsets enumeration method that discriminates between different systems' states. SPICE successfully identified cancer-related genes from various microarray data sets and found enzymes or COGs associated with biohydrogen production and motility phenotype by microbial organisms. SPICE also improved the predictive skill of the system's state determination by up to 10% relative to individual classifiers and/or other ensemble methods, such as bagging, boosting, random forest, nearest shrunken centroid, and random forest variable selection method.

Chapter 3

Discovery of Community Dynamics in Evolutionary Networks

3.1 Introduction

Networks of dynamic systems can be highly clustered [166]. A community, defined as a collection of individual objects that interact unusually frequently, is a very common substructure in many networks [55], including social networks, metabolic and protein interaction networks, financial market networks, and even climate networks. In social networks, a community is a real social grouping sharing the same interests or background [55]. In biological networks, a community might represent a set of proteins that perform a distinct function together. Communities in financial market networks might denote groups of investors that own the same stocks, and communities in climate networks might indicate regions with a similar climate or climate indices.

Many algorithms have been developed for detecting community structures in static graphs. Girvan and Newman [55] proposed a community discovery algorithm based on the iterative removal of edges with high *betweenness* scores. To reduce the computational cost of the betweenness-based algorithm, Clauset *et al.* [34] proposed a modularity-based algorithm. In contrast, Palla and Derenyi [112] did not focus on detecting separate communities, but on finding overlapping communities. Defining communities as maximal cliques, Schmidt *et al.* [129] proposed a parallel, scalable, and memory-efficient algorithm for their enumeration.

In addition, some work has been done on detecting *conserved* or *stable* communities in evolutionary networks. Hopcroft and Khan [74] proposed a method that utilizes a "nature community" to track stable clusters over time. A framework for identifying communities in dynamic social networks, proposed by Tantipathananandh *et al.* [149], makes explicit use of temporal changes. Using the Clique Percolation Method to locate communities, Palla *et al.* [113]

defined auto-correlation and stationarity to characterize a community. From an application perspective, Steinhaeuser *et al.* [141] provided a method to identify climate regions by detecting communities in time-varying climate networks.

Communities in real networks change over time, and being able to detect small community deviations can help us understand and exploit these networks more effectively. For example, in biological networks, a small variation in a gene-gene association community may represent an event, such as gene fusion [137], gene fission [137], gene gain [23], gene decay [99], or gene duplication [180], that would change the properties of the gene products (e.g., proteins) and, consequently, affect the phenotype of the organism. Interesting community deviation patterns in Food Web and social networks are discussed in Section 3.3.

Thus, in contrast to the previous work on identifying communities or tracking *conserved* communities, we focus on detecting *community-based dynamics*, a new type of "*in-disguise*" anomalies, in time-evolving networks. Specifically, our work proposes the novel problem of detecting these "*in-disguise*" anomalies across *multiple* dynamically evolving graphs, or *evolutionary networks*, for short. Our approach follows from the need to address the following four challenges:

- How do we define community dynamics, and how many types of community dynamics are possible in evolutionary networks? Community dynamics would reveal latent behaviors of the network, as opposed to conserved communities or communities in a single snapshot. For example, is there any community in snapshot t that splits into smaller communities or merges with others in snapshot k? Does any community in snapshot t disappear in snapshot k, or does any new community appear in snapshot k? Do the sizes of the communities change over time?

- Most real networks are dynamic and characterized by overlapping communities [112]. Detecting community dynamics from networks characterized by overlapping communities is more challenging than discovering communities in static networks.

- How do we detect community dynamics across multiple dynamically evolving networks? As we mentioned earlier, real-world networks change over time, requiring us to adopt evolutionary analysis techniques to detect such dynamics.

- Since there may be hundreds or even thousands of communities in each real-world network, how to scale a community dynamic detection algorithm to large graphs?

In this chapter, we propose solutions to all four of these problems. Our algorithm is based on the proposed notion of *graph representatives* and *community representatives*. *Graph representatives* helps us reduce the expensive computational cost of enumerating communities,

which we model as maximal cliques, whereas *community representatives* is utilized to identify community dynamics.

The contributions of our work are:

1. Our work tackles the unexplored question of detecting community dynamics across *multiple* graphs.
2. We prove that there are only six possible types of community dynamics in dynamic simple undirected graphs.
3. We develop a community dynamic detection algorithm based on *graph representatives* and *community representatives*.
4. We evaluate our method on real datasets to confirm its applicability in practice.

The rest of the chapter is organized as follows: Section 3.2 introduces some necessary definitions and formally defines the problem. In Section 3.3, we show application of community dynamic detection to two real-world dynamic networks, Food Web and Enron Email. Section 3.4 presents the community dynamic detection algorithm. In Section 3.5, we evaluate the algorithm with synthetic data. Finally, Section 3.7 concludes the chapter.

3.2 Problem Statement

In this chapter, the ultimate goal is to find community dynamics in dynamic graphs, and our algorithm is based on *graph representatives* and *community representatives*. Thus, the following terms and problems need to be addressed. The symbols used in the chapter are listed in Table 3.1.

Problem 1 (Community dynamic detection)**.** *Given a time-varying sequence of undirected simple graphs $\mathcal{G} = \{G_1, G_2, G_3, ...\}$, where the nodes in each graph can belong to different communities, detect the community dynamics between consecutive graphs, including grown, shrunken, merged, split, born, and vanished communities (see Definition 7).*

Definition 1 (Community)**.** *Communities are the maximal cliques in a graph.*

There is no formal definition for the community structure in a network [55]. The simplest and the most conservative definition of a community is a *clique*, a set of vertices that are pairwise adjacent to one another. Another definition used by Newman [55] is a dense subgraph, a group of vertices within which the connections are denser than between different groups [55].

Table 3.1: Symbol table

Symbol	Description
G_i	A simple undirected labeled graph
$\mathcal{G}$	A sequence of graphs
C_t^i	The community of index i in graph G_t
$Rep(G_i)$	The representative node set of graph G_i
$C_t^i \to C_{t+1}^j$	C_t^i is a predecessor of C_{t+1}^j, or C_{t+1}^j is a successor of C_t^i
$V(C_t^i)$	The node set of community C_t^i
$SV(G_i)$	Common nodes between graphs G_i and G_{i+1}
$\|C\|$	The size of community C
T	The number of timestamps in the sequence
$\|V(C)\|$	The number of vertices in community C
v_j	A vertex j in a graph
CG_i	The list of communities in G_i
$VC_i^{v_j}$	The list of communities that contain node v_j in graph G_i
$Checked(G_i)$	The list of nodes in G_i that have been checked
$\emptyset$	The empty set

As our goal is to detect abnormal, changing communities, we propose to use the more specific community definition, namely clique. From an application perspective, we could lose important information if we shifted to dense subgraphs as communities. For example, consider protein functional modules (biological communities) in protein-protein interaction networks. Across different organisms, such evolutionary networks might have undergone small changes, or perturbations, due to evolutionary events such as gene fusion, gene fission, gene gain, gene decay, or gene duplication. Relatively small perturbations to the network structure due to the genotype variation may induce phenotype variations, such as organism's capability to produce hydrogen or ethanol, to resist high temperature, to fix nitrogen, etc. Since network perturbations could be infinitesimal, considering communities as dense subgraphs with respect to some density parameter, may arguably be insufficient for capturing such fine-grain changes to the network structure. Therefore, we propose to use the simplest, the most stringent, and parameter-free definition of a community—a clique. We use the maximal clique, i.e., a clique that can't be extended by adding any more vertices in order to decrease the space of putative community-based dynamics to evaluate and thus to reduce the overall computational cost.

Definition 2 (Community size)**.** *The community size, $|C|$, is the number of vertices in the community, so $|C| = |V(C)|$.*

Definition 3 (Graph representative)**.** *Representatives of graph G_i are the nodes that also appear in G_{i-1}, G_{i+1}, or both. Thus, $Rep(G_i) = \{v_i \mid v_i \in V(G_i) \cap (V(G_{i-1}) \cup V(G_{i+1}))\}$.*

Nodes that only appear in one graph are called graph-specific nodes or vertices.

Definition 4 (Graph-specific community)**.** *A graph-specific community is a community that does not contain any graph representative.*

Since our goal is to detect community dynamics, we do not try to discover graph-specific communities. Thus, by using graph representatives as seeds, we do not need to enumerate all communities in the graphs, only those communities that contain graph representatives, and thus potentially reducing computational time (see Section 3.4 for details).

Definition 5 (Community predecessor and successor)**.** *If community C_t^i at snapshot t is a subset or superset of community C_{t+1}^j at snapshot $t+1$, then the community C_t^i is a predecessor of C_{t+1}^j, and C_{t+1}^j is a successor of C_t^i. This relationship is denoted by $C_t^i \rightarrow C_{t+1}^j$.*

Definition 6 (Community representative)**.** *A community representative of C_t^i is a node in C_t^i that has the minimum number of appearances in other communities of the same graph. If there is more than one node that satisfies this condition, we choose one at random.*

The rationale for our definition of a community representative follows from the observation that the community C_t^i can be represented by a node that only appears in community C_t^i. However, since the communities in our networks may be highly overlapping, we cannot guarantee that such a node exists, so we look for a node in C_t^i that has the minimum number of appearances in other communities to use as its representative. In this way, we limit the nodes that belong to more than one community from being a community representative, which helps to establish the relationships between the communities (see Section 3.4 for details).

Definition 7 (Community dynamics)**.** *In contrast to [113], which focuses on the stability/stationarity of the communities, our goal is to detect community dynamics. As there are six basic events that may occur to a community [113], we can define six possible types of community dynamics in evolutionary networks (see Figure 3.1).*

1. *Grown community*
   In real-world networks, some "big" communities, like community 2, can be grown from previous "smaller" communities by adding some new members. These "big" communities are called grown communities.

2. *Shrunken community*
   On the other hand, shrunken communities, like community 4, are communities caused by previous "bigger" communities losing some members.

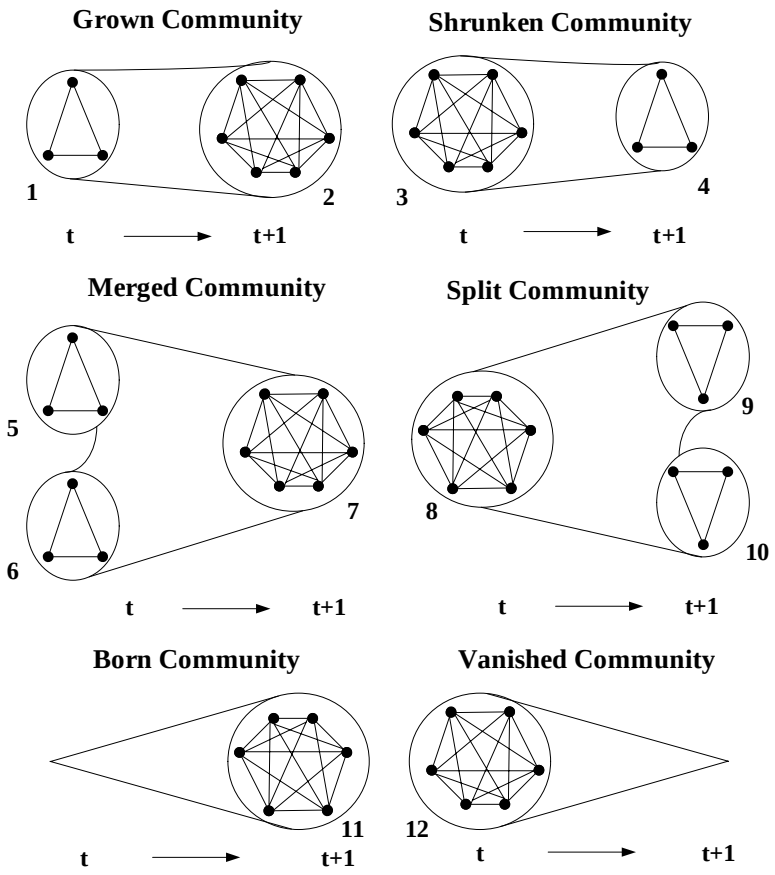


Figure 3.1: Possible types of community dynamics in evolutionary networks

3. *Merged community*
   In addition, two or more "small" communities at snapshot t often join together to form one merged community, like community 7*, at snapshot* $t+1$.

4. *Split community*
   Meanwhile, a split community at snapshot t, like community 8*, may break up into multiple communities at snapshot* $t+1$.

5. *Born community*
   What's more, some "new" communities, like community 11*, may appear in some snapshots, but born communities should contain at least one graph representative in order to avoid considering graph-specific communities as dynamic communities.*

6. *Vanished community*
   Alternatively, some "old" communities, like community 12*, may disappear. Similar to born communities, vanished communities should contain at least one graph representative to exclude graph-specific communities.*

Evolutionary network conservation, which is often manifested with stable communities that do not change over time, is a well-recognized property of many real-world complex dynamic networks. For example, in climate networks, such communities may correspond to well-known climate indicies. Likewise, in biological networks, such stable communities may correspond to protein complexes, such as ATP synthase or ribosomal machinery, and metabolic pathways, such as TCA cycle. In contrast to stable communities, an anomalous community is often highly hidden among an enormous number of stable communities in evolutionary networks. In real-world networks, like social networks, a majority of people's friendship communities tend to be stable despite frequently occuring changes in individuals' activities and communication patterns [113].

It is often the case, especially if Δt is small, that only very few communities might slighly change due to some anomalous events. For example, resignation of the CEO in a company may induce changes to community composition, if community membership is defined by email communication traffic between a sender and a receiver. Likewise, in climate networks, the seasons of unusually high hurricane activity are likely induced by changes in climate communities found in the climate networks for the seasons with low hurricane activity. Thus, rare and anomolous events are likely caused by or induce structural changes in the communities, and result in the appearance of anomalous communities. Such anomalous communities often overlap with other "normal" (or stable) communities, which makes it even more difficult to distinguish between normal and anomolous communities. Thus, dynamic community can be seen as a new type of "in-disguise" anomaly.

3.3 Application of Community Dynamic Detection to Real-world Dynamic Networks

In addition to the more controlled experiments using synthetic graph data sets, as described in Section 3.5, we applied our algorithm to two real-world dynamic networks, Food Web and Enron Email. In this section, we consider only communities of size three or more.

Food Web dataset: The Food Web dataset, which was originally compiled by Baird and Ulanowicz [6], consists of marine organisms living in the Chesapeake Bay, containing 33 vertices that represent the ecosystem's most prominent taxa. Edges between taxa denote trophic relationships—one taxon feeding on another. Here, we ignore directionality and consider the network as an undirected graph. Newman [55] has used this dataset as a static graph to detect the communities, while we construct the networks on a seasonal basis from spring to winter to discover community dynamics in the dynamic network.

Table 3.2: Food Web communities

Season	Number of Communities	Abnormal Communities
Spring	15	None
Summer	15	Four grown communities, one born community, four communities will split in fall, one vanished community, and one merged community
Fall	19	Two shrunken communities and four vanished communities (will disappear in winter)
Winter	9	Four merged communities

By applying our algorithm to the Food Web networks, we find instances of all six types of community dynamics (see Table 3.2). Summer is the most active community changing season: four communities grow because microzooplankton, which cannot be found in Spring, become involved in the energy flow network. Four communities split because bacteria do not feed on microzooplankton in Fall. The disappearance of sea nettle in the Fall results in a vanished community (zooplankton, ctenophore, and sea nettle). This community was a born community in Summer, indicating that it is unstable. Due to a lack of food in Winter, four communities merge in order to benefit from more members with food energy. Typical dynamic community examples discovered in Food Web are shown in Figures 3.2–3.4. Note that the

different circles represent different communities at the same time stamp—we can see that Food Web is characterized by overlapping communities.

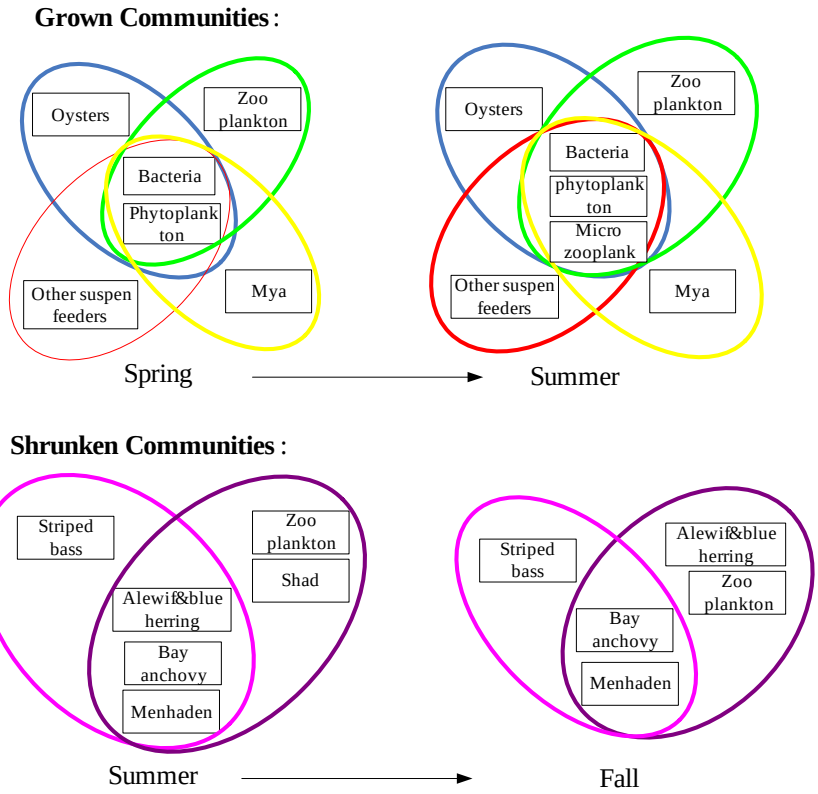


Figure 3.2: Example of a grown community and a shrunken community in Food Web.

ENRON dataset: This data set consists of approximately 1.5 million email communications sent or received by employees in Enron, Inc. It is much more complex than the Food Web dataset. We take a subset containing only messages between Enron employees from January to December of 2001 and construct sender-to-recipient undirected graphs on a monthly basis. The graphs have 151 nodes (Enron employees), with low edge density and short average distance between vertices, which shows a "small-world" effect and indicates that the graphs have community structure. The properties of each graph are shown in Table 3.3.

The community dynamics in each month discovered by our algorithm are shown in Table 3.4. We can see that there are more abnormal communities in October than in any other month. The most likely trigger of this event is the fact that Enron announced a third quarter loss of $618 million on October 16 of 2001, which is also thought to be the trigger of the Enron scandal.

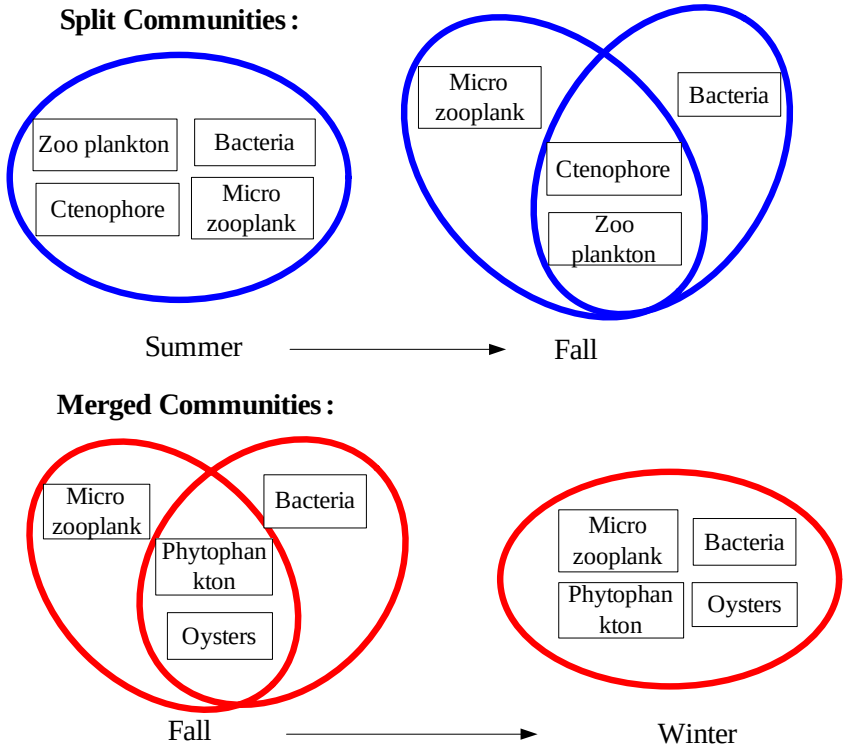


Figure 3.3: Example of a split community and a merged community in Food Web.

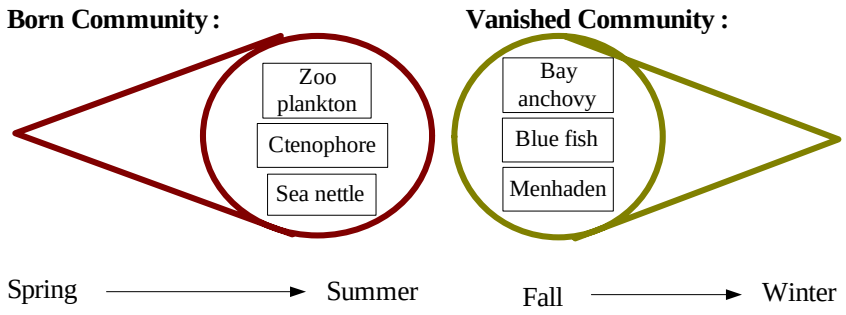


Figure 3.4: Example of a born community and a vanished community in Food Web.

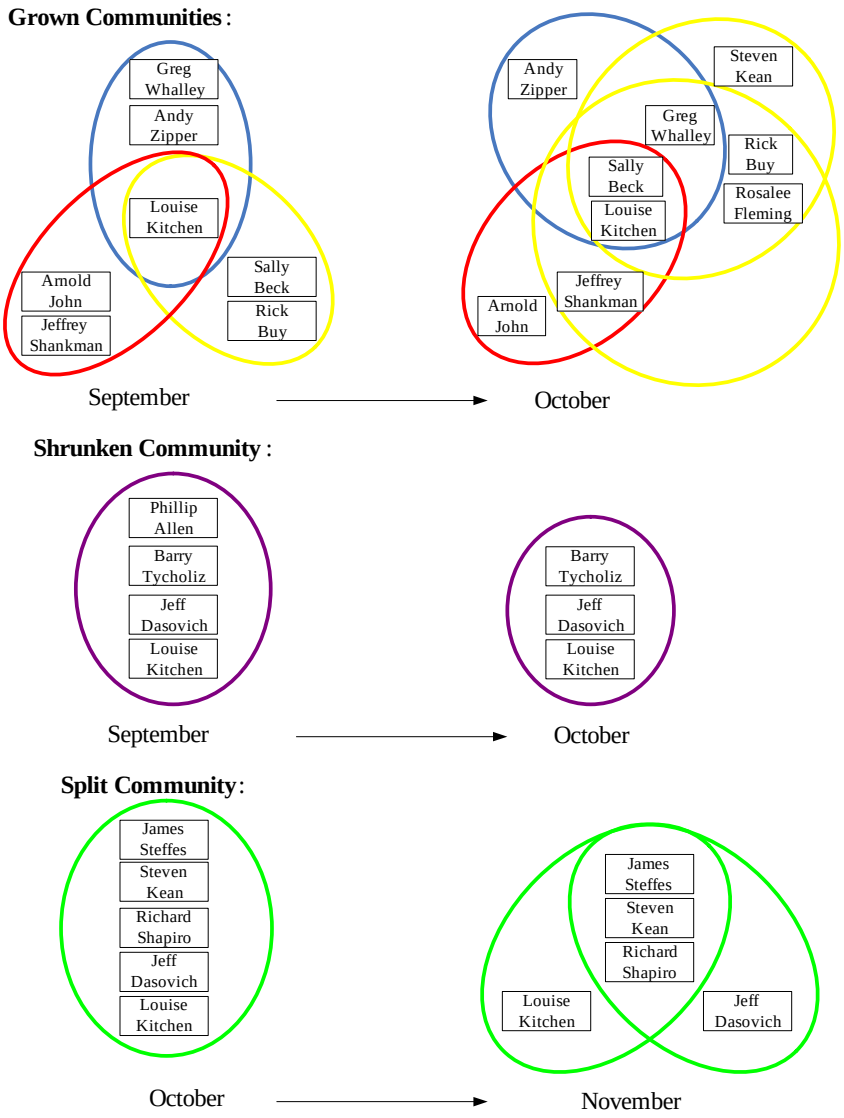


Figure 3.5: Abnormal communities containing Louise Kitchen in October.

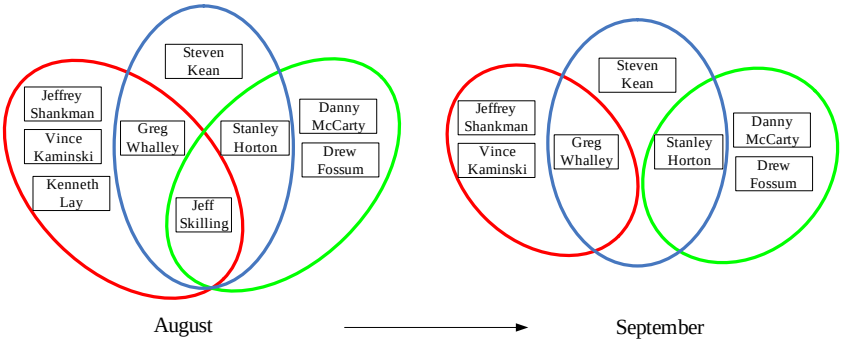


Figure 3.6: Shrunken communities due to Jeff Skillng resigning as CEO in August.

Table 3.3: Enron email dataset properties

Month	Number of Edges	Number of Communities
Jan.	126	21
Feb.	190	56
Mar.	199	54
Apr.	240	66
May	273	90
Jun.	218	49
Jul.	240	68
Aug.	371	120
Sep.	343	110
Oct.	531	196
Nov.	438	143
Dec.	290	93

In order to see the details of the community dynamics in October, let us consider one of the most important nodes—Louise Kitchen, the former President of Enron. There are 20 abnormal communities containing Louise Kitchen in October: 16 born communities, 4 grown communities, 1 split community, and 1 shrunken community. From Figure 3.5, we can see that some employees like Sally Beck, Chief Operating Officer, joined the senior management communication groups, probably to discuss the serious issues or suggest strategies, while only one person—Phillip Allen—left the groups. Confusion among the Enron employees may be why Louise Kitchen's communication groups grew rather than shrank during the turbulent times. As a second example, take Jeff Skilling, the former CEO of Enron. There are 19 email communities contain Jeff Skilling in August, but among these, 3 communities shrank in September (see Figure 3.6), while the other 16 communities disappeared after Jeff Skilling resigned as CEO in August, perhaps because many employees quit or joined other work groups after Skilling's resignation.

3.4 Community Dynamic Detection Algorithm

In this section, we discuss the proposed algorithm for solving the problem presented above. We prove some necessary lemmas and theorems in Section 3.4.1. Then, based on the abnormal community decision rules described in Section 3.4.2, we illustrate how to detect community dynamics in Section 3.4.3.

Table 3.4: Community dynamics in Enron email dataset

Month	Grown	Shrunken	Merged	Split	Born	Vanished
Jan.	0	0	0	0	0	14
Feb.	3	1	0	1	48	32
Mar.	3	2	4	1	33	39
Apr.	6	1	0	2	42	43
May	3	4	0	1	75	76
Jun.	3	3	0	1	34	38
Jul.	1	0	2	0	58	54
Aug.	9	4	2	2	97	89
Sep.	8	9	3	1	79	56
Oct.	30	5	10	8	136	160
Nov.	7	13	0	9	102	97
Dec.	2	17	2	0	54	14

3.4.1 Lemmas and Theorems

We present the following theorems and lemmas to provide a sound theoretical basis for our community dynamic detection.

Lemma 3.4.1. *If community C_t^i has more than one predecessor (or successor), the sizes of its predecessors (or successors) are either all larger than $\left|C_t^i\right|$ or all smaller than $\left|C_t^i\right|$.*

Proof. Suppose otherwise, that C_t^1 has a predecessor with smaller size, as well as one with a larger size. Let C_{t-1}^1, C_{t-1}^2,..., C_{t-1}^n (where $n \geq 2$) be all predecessors of C_t^i, and suppose that $\left|C_{t-1}^j\right| < \left|C_t^i\right|$ and $\left|C_{t-1}^k\right| > \left|C_t^i\right|$ for some $1 \leq j, k \leq n$, $j \neq k$. From Definition 5 and the sizes of the three communities, we know that $C_{t-1}^j \subset C_t^i$ and $C_t^i \subset C_{t-1}^k$, so $C_{t-1}^j \subset C_{t-1}^k$. However, C_{t-1}^j and C_{t-1}^k are both maximal cliques in the same graph, and $C_{t-1}^j \subset C_{t-1}^k$ contradicts the definition of a maximal clique. Therefore, it is impossible to have the size of one predecessor be larger than the size of the community and the size of another predecessor be smaller than the size of the community. □

Similarly, if a community C_t^i has more than one successor, then the sizes of its successors are either all larger than $\left|C_t^i\right|$ or all smaller than $\left|C_t^i\right|$. This lemma is used to prove the following completeness result:

Theorem 3.4.1. *Let G_t and G_{t+1} both be simple, undirected graphs, where communities are defined as maximal cliques. If G_{t+1} is the perturbed graph formed by either adding edges/nodes to or removing edges/nodes from the baseline graph G_t, then there are only six possible types of*

community dynamics between G_t and G_{t+1}: grown communities, shrunken communities, merged communities, split communities, born communities, and vanished communities, as defined in Definition 7.

Proof. Assume that $C_t^1, C_t^2, \ldots, C_t^m$ are all communities in G_t and that $V_t^1, V_t^2, \ldots, V_t^m$ are the node sets of the communities, respectively. Also assume that $C_{t+1}^1, C_{t+1}^2, \ldots, C_t^n$ are all communities in G_{t+1} and that $V_{t+1}^1, V_{t+1}^2, \ldots, V_{t+1}^n$ are the node sets of the communities, respectively. Here, we define $V_t^i = V_{t+1}^j$ to mean that V_t^i only contains all the nodes in V_{t+1}^j.

To determine the type of a specific community, we only need to compare the node sets of communities in G_{t+1} with the node sets of communities in G_t. If $V_{t+1}^j = V_t^i$, where $1 \leq i \leq m$ and $1 \leq j \leq n$, then community C_{t+1}^j contains exactly those nodes in community C_t^i, which means that C_{t+1}^j is a conserved community and not a dynamic community.

In the following, we consider all possible community dynamics by analyzing all possible mappings between predecessors and successors. In particular, when deciding if community C_t^i is a dynamic community, we do not need to consider the situation where C_t^i has a single successor as long as we have covered all cases for the predecessors of C_{t+1}^j. If the community C_t^i has only one successor C_{t+1}^j, then the community C_{t+1}^j should have either one predecessor or more than one predecessor, both of which can be covered by using predecessor conditions. The same reasoning applies for not considering the case where a community has more than one successor of larger size. In other words, we need to consider all cases for predecessors, but only two cases for successors: when a community has no successor and when a community has more than one successor of smaller size.

1. For a specific j (where $1 \leq j \leq n$), there is at least one i (where $1 \leq i \leq m$) that satisfies $V_{t+1}^j \subset V_t^i$. Then, by Definition 5, community C_{t+1}^j has at least one predecessor, including C_t^i, with larger size than C_{t+1}^j. Let $I = \{i \mid V_{t+1}^j \subset V_t^i\}$. There are two non-exclusive sub-cases here:

   (a) For $\ell \in I$, if there is some k (where $1 \leq k \leq n$) other than j that satisfies $V_{t+1}^k \subset V_t^\ell$, then C_t^ℓ has more than one smaller-size successor (C_{t+1}^j and C_{t+1}^k). Additionally, by Lemma 3.4.1, we know that C_t^ℓ cannot have a successor with larger size than C_t^j. Thus, C_t^ℓ is a split community, and C_{t+1}^j is one of its products.

   (b) For $\ell \in I$, if there is no k (where $1 \leq k \leq n$) other than j that satisfies $V_{t+1}^k \subset V_t^\ell$, then C_t^ℓ has only one smaller-size successor C_{t+1}^j, and C_{t+1}^j has at least one predecessor, including C_t^ℓ, with larger size. Also, by Lemma 3.4.1, we know that C_{t+1}^j cannot have a predecessor with smaller size than C_{t+1}^j. Thus, C_{t+1}^j is a shrunken community.

2. For a specific j (where $1 \leq j \leq n$), there is only one i (where $1 \leq i \leq m$) that satisfies $V_{t+1}^j \supset V_t^i$. Then, community C_{t+1}^j has one predecessor C_t^i with smaller size than C_{t+1}^j. Additionally, by Lemma 3.4.1, we know that C_{t+1}^j cannot have a predecessor with larger size than C_{t+1}^j. Thus, community C_{t+1}^j is a grown community.

3. For a specific j (where $1 \leq j \leq n$), there is more than one i (where $1 \leq i \leq m$) that satisfies $V_{t+1}^j \supset V_t^i$. Then, community C_{t+1}^j has more than one predecessor with smaller size. Also, by Lemma 3.4.1, we know that C_{t+1}^j cannot have a predecessor with larger size than C_{t+1}^j. Thus, community C_{t+1}^j is a merged community.

4. For a specific j (where $1 \leq j \leq n$), there is no i (where $1 \leq i \leq m$) that satisfies $V_{t+1}^j \supset V_t^i$ or $V_{t+1}^j \subset V_t^i$, which means that community C_{t+1}^j has no predecessor. Thus, C_{t+1}^j is a born community.

5. For a specific i (where $1 \leq i \leq m$), there is at least one j (where $1 \leq j \leq n$) that satisfies $V_{t+1}^j \subset V_t^i$. Let $J = \{j \mid V_{t+1}^j \subset V_t^i\}$. Then, for each $k \in J$, there is at least one i (where $1 \leq i \leq m$) that satisfies $V_{t+1}^k \subset V_t^i$, which is case 1. Thus, this case can be converted to case 1.

6. For a specific i (where $1 \leq i \leq m$), there is at least one j (where $1 \leq j \leq n$) that satisfies $V_{t+1}^j \supset V_t^i$. Let $J = \{j \mid V_{t+1}^j \supset V_t^i\}$. Then, for each $k \in J$, there is at least one i (where $1 \leq i \leq m$) that satisfies $V_{t+1}^k \supset V_t^i$, which is case 2 or 3. Thus, this case can be converted to case 2 or 3.

7. For a specific i (where $1 \leq i \leq m$), there is no j (where $1 \leq j \leq n$) that satisfies $V_{t+1}^j \supset V_t^i$ or $V_{t+1}^j \subset V_t^i$, which means that community C_t^i has no successor. Thus, C_t^i is a vanished community.

Since all relationships between V_{t+1}^j (where $1 \leq j \leq n$) and V_t^i (where $1 \leq i \leq m$) have been covered, there are only six possible different types of community dynamics. □

Lastly, we present a theorem that will allow us to reduce the computational complexity of identifying the community dynamics.

Theorem 3.4.2. *If community C_t^i is represented by vertex $v_i \in C_t^i$, and community C_{t+1}^j is represented by vertex $v_j \in C_{t+1}^j$, where $C_t^i \to C_{t+1}^j$, then $v_i \in C_{t+1}^j$ or $v_j \in C_t^i$.*

Proof. By Definition 5, $C_t^i \to C_{t+1}^j$ implies that $C_t^i \subseteq C_{t+1}^j$ or $C_{t+1}^j \subseteq C_t^i$. If $C_t^i \subseteq C_{t+1}^j$, then $v_i \in C_{t+1}^j$, and if $C_{t+1}^j \subseteq C_t^i$, then $v_j \in C_t^i$. □

From Theorem 3.4.2, if there is more than one node in community C_t^m and C_{t+1}^n that satisfies the condition of community representative, then we can randomly choose v_m to represent C_t^m and v_n to represent C_{t+1}^n to help check the relationship between C_t^m and C_{t+1}^n as follows:

1. If $v_m \in C_{t+1}^n$, then C_{t+1}^n can be detected as a potential successor to C_t^i. By Definition 5, the relationship $C_t^m \to C_{t+1}^n$ would be established if C_{t+1}^n also satisfies $C_t^m \subseteq C_{t+1}^n$ or $C_{t+1}^n \subseteq C_t^m$.

2. If $v_m \notin C_{t+1}^n$, then by Theorem 3.4.2, $v_n \in C_t^m$ if $C_t^m \to C_{t+1}^n$. In this case, we can detect the relationship $C_t^m \to C_{t+1}^n$ through v_n and check whether $C_t^m \supset C_{t+1}^n$.

Thus, random selection of the community representative will not affect our detection results.

3.4.2 Decision Rules for Community Dynamic Detection

Based on our result from Theorem 3.4.1, we can identify community dynamics using the following rules:

1. If community C_t^i has only one predecessor C_{t-1}^j:
   (a) If the size of the predecessor is smaller than $\left|C_t^i\right|$, then C_t^i is a grown community.
   (b) If the size of the predecessor is larger than $\left|C_t^i\right|$ and C_t^i is the only successor of C_{t-1}^j, then C_t^i is a shrunken community.
   (c) If the size of the predecessor is larger than $\left|C_t^i\right|$ and C_t^i is not the only successor of C_{t-1}^j, then C_t^i is a product of the split community C_{t-1}^j.

2. If community C_t^i has more than one predecessor:
   (a) If the sizes of the predecessors are all smaller than $\left|C_t^i\right|$, then C_t^i is a merged community.
   (b) If the sizes of the predecessors are all larger than $\left|C_t^i\right|$ and C_t^i is the only successor of one of its predecessors, then C_t^i is a shrunken community.
   (c) If the sizes of the predecessors are all larger than $\left|C_t^i\right|$ and C_t^i is not the only successor of one of its predecessors, then that community is a split community and C_t^i is one of its products.

3. If community C_t^i has no predecessor, then C_t^i is a born community.

4. If community C_t^i has no successor, then C_t^i is a vanished community.

3.4.3 Algorithm Description

In this section, we describe our method for detecting and tracking anomalous communities based on the proposed notion of *graph representatives* and *community representatives*.

To the best of our knowledge, the proposed problem of *detecting and tracking community dynamics in evolutionary networks* has not been addressed in literature. Thus, for comparison purposes, we first briefly describe a brute-force solution that does not use graph representatives and community representatives. Then, we provide details on how *graph representatives* can help reduce the expensive computational cost caused by community enumeration, and how *community representatives* can be utilized to effectively identify community dynamics.

Non-representative-based Method:

A brute-force solution that does not use graph or community representatives is to first enumerate all communities in each graph, and then compare all possible pairs of communities belonging to consecutive timestamps. For example, to find the successors of community A in Figure 3.7, we need to compare community A with communities D, E, F, and K at snapshot $t+1$; that is, we compare community A with all communities at snapshot $t+1$, although only community F is the successor of A. This two-stage approach is infeasible and impractical, because of a possibly enormous number of communities to search. Among those, there are many redundant communities (e.g., graph-specific communities (see Definition 4)) in each graph, and it does not make much sense to compare pairs of communities that contain no common members or few members.

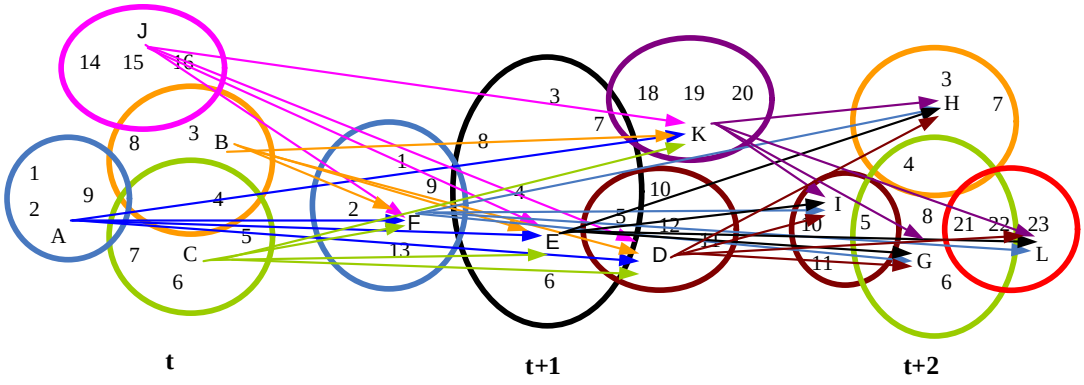


Figure 3.7: Example for tracking community dynamics using the non–representative-based method.

Representative-based Method:

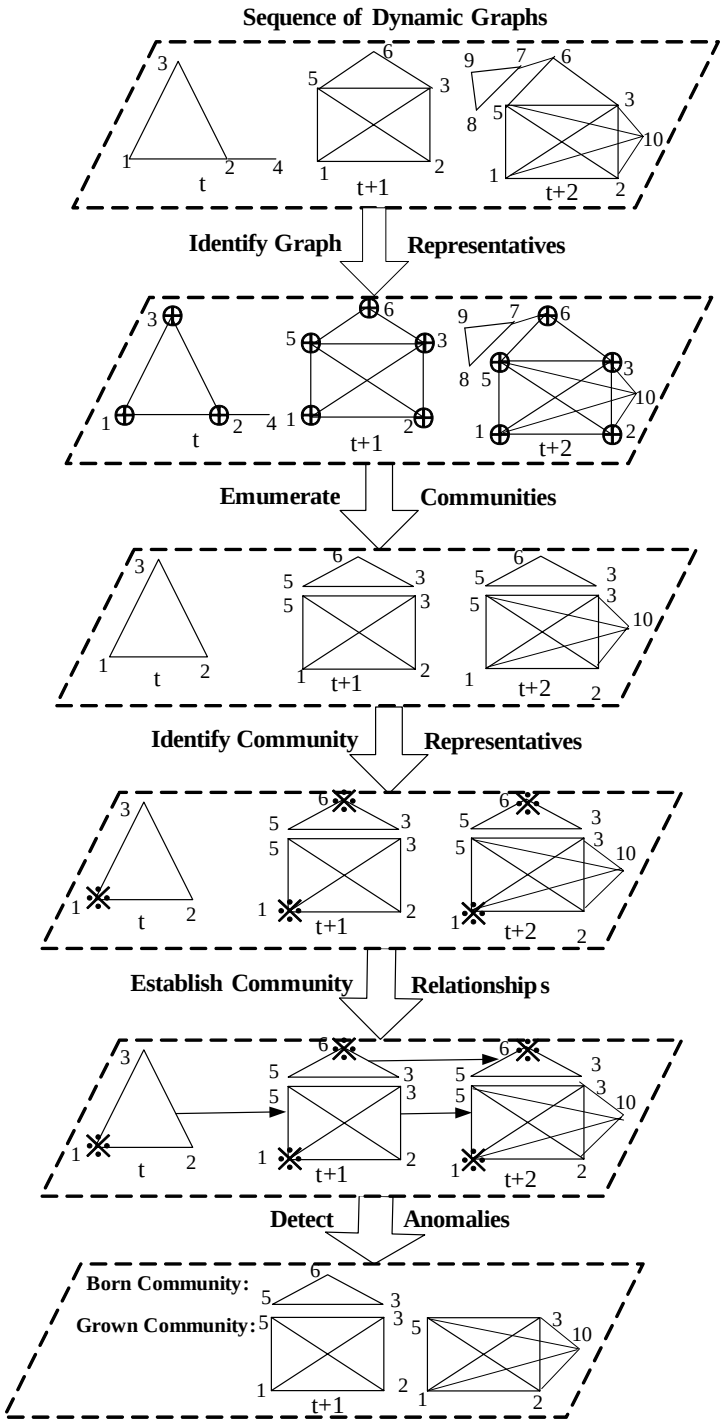


Figure 3.8: Workflow of the community dynamic detection algorithm.

To reduce the computational cost, we designed an algorithm based on the graph representatives and community representatives (see Definition 3 and 6 in Section 3.2). The workflow of the algorithm is shown in Figure 3.8. For each graph, we first find graph representatives (see Step 1 in Figure 3.8) and enumerate the communities that are seeded by the graph representatives to avoid generating graph-specific communities (see Step 2 in Figure 3.8). We call these communities seed-communities. In every seed-community, we select only one node as a community representative (see Step 3 in Figure 3.8) and use community representatives to establish predecessor–successor relationships between a pair of seed-communities from two consecutive graphs (see Step 4 in Figure 3.8). Once all the predecessors and successors of the community C_t^i have been found, we apply the abnormal community decision rules in Section 3.4.2 to determine the type of dynamic community present, if any (see Step 5 in Figure 3.8).

Let us apply the representative-based algorithm to the same example in Figure 3.7. Instead of enumerating all communities, the algorithm first identifies the set of graph representatives, which are the filled triangle or rectangle nodes highlighted in Figure 3.9. Using graph representatives as seeds to generate communities, graph-specific communities (see Definition 4), like communities K and L in Figure 3.7, now disappear (see Figure 3.9). This strategy could possibly save a lot of computational cost on community enumeration. Once, we generate the seed-communities in each graph, the algorithm searches for community representatives (triangular nodes in Figure 3.9) by selecting the vertices that appear in the fewest number of communities.

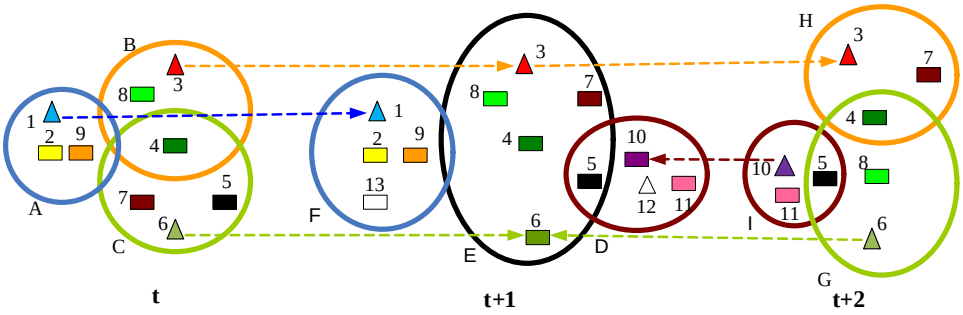


Figure 3.9: Example for tracking community dynamics using the representative-based method. Triangles: community representatives; Filled shapes: graph representatives; Empty shapes: graph-specific vertices; Circles: communities; Dashed lines: predecessor-successor community relationships.

Taking advantage of community representatives, the algorithm can establish the predecessor–successor community relationships much more efficiently. Let us take community A at times-

tamp t, for example. To find the successor(s) of community A, the algorithm first finds all the communities that contain the community representative of A at timestamp $t+1$. In this case, only community F contains the community representative. Then, the algorithm checks whether community F is a subset or superset of A (see Definition 5). Only if one of these two conditions holds true does the algorithm establish the predecessor–successor relationship between A and F. When there are only grown, merged, born, or vanished communities, the algorithm does not need to consider commmunities earlier in the sequence of graphs. For example, community A grows into community F, communities B and C merge into E, community D emerges, and community F disappears. However, in cases of shrunken or split communities, the algorithm may need to "backtrack" by using the representative of community $C^(i)_t$ to look for its predecessors at timestamp $t-1$. For example, community D shrinks into I with the disappearance of representative 12 at timestamp $t+2$, and community E splits into H and G with representative $3 \in H$ but $3 \notin G$. To detect these community dynamics, the algorithm needs to connect communities G and I at timestamp $t+2$ to communities E and D, respectively, at timestamp $t+1$ by "backtracking" the community representatives of G and I (6 and 10).

From Figure 3.9, we can also see that if there is more than one node in the same community with the minimum number of appearances in other communities, then randomly choosing one node as a community representative would not affect the detection results. For example, if we choose 9 instead of 1 as representative of community A or 8 instead of 3 as representative of B, we will still identify F as successor of A and E as the successor of B, since all nodes in the predecessors of the grown (or merged) community will also appear in the grown (or merged) community. Namely, if the representative of C^i_t appears in the successors of C^i_t, then we will find all such successors for C^i_t, because we check all communities that contain the representative of C^i_t at timestamp $t+1$. Even if the representative of C^i_t disappears in some successors of C^i_t, we can still establish the relationship between C^i_t and its successors by "backtracking" from the representatives of its successors, like the example of communities G, I, E, and D shown previously.

Once the community relationship is established, the algorithm uses the decision rules (see Section 3.4.2) to determine whether a community is a dynamic community, based on the numbers and sizes of its predecessors and successors:

- A grown community, like community F in Figure 3.9, can be detected by comparing the communities at the prior timestamp that contain the community representative 1 of F (community A, in this case). Because community F is larger than its predecessor A and has no other predecessors, based on decision rule 1a, community F is a grown community.

- A shrunken community, like community J in Figure 3.9, can be detected by decision rule

1b, since it has only one larger predecessor.

- A community, such as community E that has two smaller successors–community H and G is identified as a split community (see decision rule 1c).
- A community, such as community E, is also detected as a merged community by the algorithm, because it has two smaller predecessors, community B and C, at timestamp t (see decision rule 2a).
- A community, like community M that has no predecessor (see decision rule 3), is detected as a born community.
- A community, like community J that has no successor (see decision rule 4), is identified as a vanished community.

We give a pseudocode description for our representative-based community dynamic detection algorithm in Algorithm 3. The input to the Algorithm 3 is a sequence of undirected graphs. Lines 1–1 are concerned with finding the graph representatives for each graph in the sequence and enumerating all the communities in each graph using the graph representatives as seeds. In lines 10–10, the algorithm calculates the number of times each node in each seedcommunity appears in each graph. It chooses one node with the fewest occurence in each community as the community representative (line 16). In line 17 through line 18, the algorithm establishes predecessor–successor community relationships. Since some community representatives may disappear in the successors of the community, lines 21 through 21 backtrack to establish community relationships between communities in the preceding timestep. This way, the algorithm can establish all community relationships. Finally, lines 25 through 25 apply the abnormal community decision rules to these relationships to identify the community dynamics. Thus, if any community in each graph belongs to one of six possible types of community dynamics, the algorithm will detect this dynamic community and return its type.

3.5 Effectiveness of Representative-based Methodology

In this section, we evaluate the community dynamic detection algorithm on synthetic graph datasets to have a more controlled settings for assessing alogirthmic performance. These experiments complement the discoveries and insights offered by our algorithm when applied to real-world network data, Food Web and Enron Email datasets, as described in Section 3.3. Specifically, we focus on answering the following two questions:

1. What is the performance of the community dynamic detection algorithm using our representative-based technique?

Algorithm 3: Community dynamic detection algorithm

Input : A sequence of undirected graphs: $\{G_1, G_2, \ldots G_T\}$
Output: Community dynamics and the discovery timestamps

for every graph G_i in the sequence **do**
  `/* Detect graph representatives */`
  $Rep(G_i) = SV(G_{i-1})$
  $SV(G_i) = \emptyset$
  for every node $v_j \in G_i$ **do**
    if $v_j \in G_{i+1}$ **then**
      add v_j to $Rep(G_i)$
      add v_j to $SV(G_i)$
  `/* Enumerate communities */`
  CommunityEnumeration($Rep(G_i)$)
  Create community list CG_i
`/* Detect community representatives */`
for every graph G_i in the sequence **do**
  for every community C_t^i **do**
    if $v_j \in C_t^i$ **then**
      Add i to the list $VC_t^{v_j}$
      $NC_t^{v_j} = NC_t^{v_j} + 1$
`/* Establish community relationship */`
for every community $C_t^i \in CG_t$ **do**
  Choose one node $v_j \in C_t^i$ with minimum $NC_t^{v_j}$ value
  Add v_j to $Checked(G_t)$
  for every k, where $k \in VC_{t+1}^{v_j}$ **do**
    if $(V(C_t^i) \subseteq V(C_{t+1}^k))$ OR $(V(C_t^i) \supset V(C_{t+1}^k))$ **then**
      Establish the relationship $C_t^i \rightarrow C_{t-1}^k$
  for every k, where $k \in VC_{t-1}^{v_j}$ **do**
    if $((C_{t-1}^k \rightarrow C_t^i) = FALSE)$ AND $(|C_{t-1}^k| > |C_t^i|)$ AND $(v_j \notin Checked(G_{t-1}))$ **then**
      if $V(C_t^i) \subset V(C_{t-1}^k)$ **then**
        Establish the relationship $C_{t-1}^k \rightarrow C_t^i$
`/* Use decision rules to detect the community dynamics */`
for every community C_t^i in graph sequence **do**
  if C_t^i is a dynamic community based on decision rules **then**
    Output the community C_t^i with its type and discovery time t

2. Is our algorithm scalable to large graphs?

We study the performance of the community dynamic detection algorithm relative to the non–representative-based algorithm on synthetic networks of increasing size. Our experiments were conducted on a PC with an Intel Core 2 Duo CPU (2.1GHz) and 4GB of RAM. Our algorithm was implemented in the C programming language, and is available upon request. We measure the improvement in the runtime of our algorithm versus the non-representative-based algorithm in terms of speedup, which we calculate by dividing the runtime of non–representative-based algorithm to the runtime of our algorithm.

In this experiment, we study the effectiveness of the proposed representative-based technique. All the graphs in the synthetic datasets are generated by GTgraph [101] and follow the Recursive Matrix Graph model (R-MAT) [37] so that they have a small-world nature. The parameters for the synthetic graphs, which appear in Table 3.5, are defined as follows: $|V|$ is the number of vertices in a graph, Num_{gv} is the number of graph-specific vertices in a graph, and E_i is the number of edges in a graph G_i. On all graphs, we use default values of 0.45, 0.15, 0.15 and 0.25 for the R-MAT parameters a, b, c, d, with $a : b$ and $a : c$ ratios of 3:1, as in many real world graphs [37]. After graph enumeration, we use a program to re-label some of the vertices according to the parameter Num_{gv}, so that we can have some graph-specific vertices in each graph when we build the sequence of graphs. For example, in the dataset *syn_500*, we can relabel the vertices $v_j \in [451, 500]$ in graph G_i as $v_j + 50 * (i - 1)$. Other graph-specific vertices in other datasets can be similarly re-labeled.

Table 3.5: Summary of synthetic datasets

Dataset	$\|V\|$	Num_{gv}	E_1	E_2	E_3	E_4	E_5
syn_500	500	50	8,000	11,000	9,000	12,000	10,000
syn_1000_1	1,000	100	400,000	550,000	45,0000	60,0000	50,0000
syn_1000_2	1,000	200					
syn_1000_3	1,000	300					
syn_1500	1,500	150	64,0000	880,000	720,000	96,0000	800,000
syn_2000	2,000	200	80,0000	1,100,000	900,000	1,200,000	1,000,000
syn_3000	3,000	300	160,0000	2,200,000	1,800,000	2,400,000	200,0000

In the first experiment, we try to test the collective effectiveness of graph representatives and community representatives in our algorithm. We measure the entire runtime of the representative-based algorithm and the non–representative-based algorithm in each of the

synthetic datasets. The result of the experiments on the datasets *syn_500*, *syn_1000_1*, *syn_-1500*, *syn_2000*, and *syn_3000* are shown in Table 3.6, where T_{non} is the runtime of the non–representative algorithm, T_{rep} is the runtime of representative algorithm, and the last six columns are the counts of the six types of community dynamics detected by the algorithm. From Figure 3.10, we can see that the representative-based algorithm achieves a speedup of 11–46 times with respect to the non–representative-based algorithm. Additionally, the experimental results show that our algorithm is scalable to large graphs.

Table 3.6: Performance comparison on synthetic data

Dataset	T_{non}(ms)	T_{rep}(ms)	Born	Vanished	Grown	Shrunken	Merged	Split
syn_500	265	25	3,425	3,482	87	79	18	23
syn_1000_1	1132	87	8,702	8,569	154	119	42	33
syn_1500	6,442	329	23,482	23,621	401	295	71	86
syn_2000	16,182	489	23,111	23,178	333	278	71	83
syn_3000	61,912	1,354	59,261	59,220	813	718	465	313

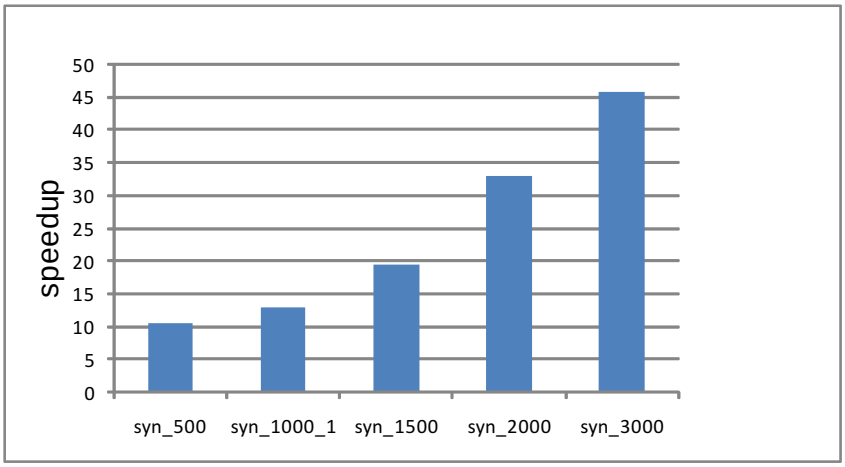


Figure 3.10: Runtime speedup of the representative-based algorithm over the non–representative-based algorithm. The time to perform I/O operations is excluded.

In the second experiment, we try to test the sole effectiveness of graph representatives in the community enumeration step. We use our in-house parallel MCE algorithm [129] (available upon request) to enumerate the communities in each graph for both algorithms. However, as discussed in Section 3.4.3, we enumerate all the communities in each graph for the non–representative-based method, but in the representative-based algorithm, we use the graph representatives as seeds to avoid graph-specific community enumeration.

The results are shown in Table 3.7, where NC_{non} is the number of cliques enumerated by the non–representative-based method, NC_{rep} is the number of cliques enumerated by the representative-based method, TC_{non} is the runtime of community enumeration using the non–representative-based method, and TC_{rep} is the runtime of community enumeration using representative-based method.

Table 3.7: Effectiveness of graph representatives

Dataset	E_i	NC_{non}	NC_{rep}	$TC_{non}(ms)$	$TC_{rep}(ms)$	Speedup
syn_1000_1	400,000	2,505	2,214	9	7.9	1.14
	550,000	2,541	2,299	9.4	8.4	1.12
	450,000	2,721	2,336	9.8	8.4	1.17
	600,000	2,564	2,303	9.3	8.3	1.12
	500,000	2,661	2,341	9.7	8.4	1.15
syn_1000_2	400,000	2,505	1,773	9	6.3	1.43
	550,000	2,541	1,878	9.4	6.9	1.36
	450,000	2,721	1,882	9.8	6.7	1.46
	600,000	2,564	1,880	9.3	6.7	1.39
	500,000	2,661	1,871	9.7	6.7	1.45
syn_1000_3	400,000	2,505	1,197	9	4.3	2.09
	550,000	2,541	1,382	9.4	5	1.83
	450,000	2,721	1,158	9.8	4.1	2.39
	600,000	2,564	1,221	9.3	4.4	2.11
	500,000	2,661	1,278	9.7	4.6	2.11

As shown in Table 3.7, the representative-based method achieves speedups of more than 1.1 in community enumeration on the dataset *syn_1000_1*, in which 10 percent of the vertices are graph-specific vertices; speedups of around 1.4 on the dataset *syn_1000_2*, in which 20 percent of the vertices are graph-specific vertices; and speedups of around 2 on the dataset *syn_1000_3*, in which 30 percent of the vertices are graph-specific vertices. The experiments on the datasets *syn_500*, *syn_1500*, *syn_2000*, and *syn_3000* also show that the representative-based method can

achieve a speedup of at least 1.1 in the community enumeration step, when the dataset has 10 percent graph-specific vertices.

3.6 Related Work

Our work is related to the *graph-based* anomaly detection. As opposed to most research in anomaly detection, which is based on strings or attribute-value data as the medium, *graph-based* anomaly detection focuses on data that can be represented as a graph [110]. It has provided new approaches for handling data that can't be easily analyzed with traditional non–graph-based data mining approaches [110] and has found applications in several domains. One of the most important of these areas is intrusion detection. GrIDS, a graph-based intrusion detection system, was developed by Cheung *et al.* [139]. Padmanabh *et al.* [111] proposed a random walk-based approach to detect outliers in Wireless Sensor Networks. Eberle and Holder [46] focused on detecting anomalies in cargo shipments. Noble and Cook [110] used anomaly detection techniques to discover incidents of credit card fraud [47].

Graph-based anomaly detection has been studied from two major perspectives: "*white crow*" and "*in-disguise*" anomalies. Intuitively, a "*white crow*" anomaly (also called an "outlier" in many papers) is an observation that deviates substantially from the other observations [106], while an "*in-disguise*" anomaly is only a minor deviation from the normal pattern [47], as shown in Figure 3.11. For example, if we are analyzing the voters list and we come across a person whose age is 322, then we can take that as a "*white crow*" anomaly, because the age of a voter will typically lie between 18 and 100. On the other hand, anyone who is attempting to commit credit card fraud would not want to be caught—a criminal would want his or her activities to look as normal as possible, which represents an "*in-disguise*" anomaly. Anomalies classified as "*white crow*" are usually detected as nodes, edges, or subgraphs, while "*in-disguise*" anomalies are now only identified through unusual patterns, including uncommon nodes or entity alterations. A summary of the various research directions in this area is shown in Figure 3.12.

Research on *"white crow"* anomaly detection has traditionally focused on exploring three different types of anomalies. Aiming to identify *anomalous nodes*, Moonesinghe and Tan [106] proposed a random walk–based approach that represents the dataset as a *weighted undirected* graph. Similarly, an algorithm based on random walks with restarts was used by Sun and Faloutsos [144] for relevance search in an *unweighted bipartite* graph, in which vertices with low normality scores were treated as anomalies. Hautamaki *et al.* [64] took a different approach and applied two density-based outlier detection methods to discover *novelty vertices* in a *k-nearest neighbour* graph. To identify *unusual edges*, Chakrabarti [21] used the minimum description length principle to identify outlier edges, or the edges whose removal would best compress the

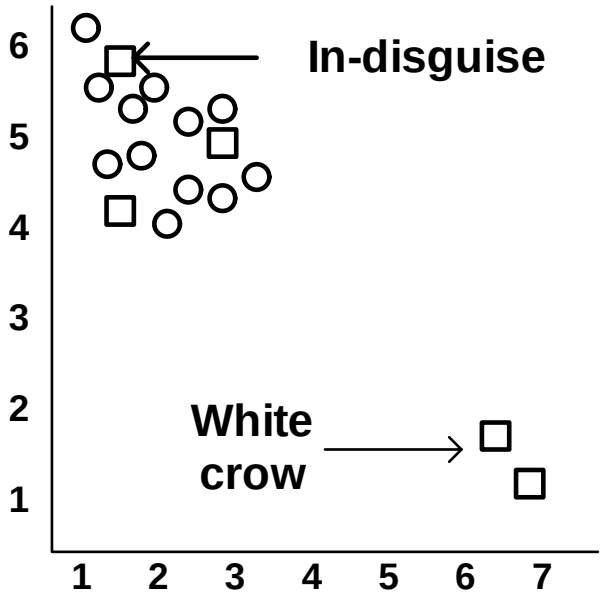


Figure 3.11: "White crow" and "in-disguise" anomalies.

graph. With the purpose of finding *abnormal patterns*, Noble and Cook [110] used a variant of the minimum description length principle to deal with both anomalous substructures and anomalous subgraphs, based on their Subdue system. In contrast to Noble and Cook [110], Lin and Chalupsk [40] applied rarity measurements to discover *unusually linked entities* within a labeled directed graph.

Unlike the previously mentioned *single graph* algorithms, Cheng and Tan [29] provided a robust algorithm for discovery of anomalies in noisy multivariate time series data. To deal with higher order data, Sun and Faloutsos [145] introduced a tensor-based approach. Other related work on "*white crow*" anomaly detection can be found in Chan *et al.* [22], Sun *et al.* [143], Keogh *et al.* [87], and others.

"*In-disguise*" anomalies are more difficult to detect because they are highly hidden in the graph, and less work has been reported on detecting such anomalies. Eberle and Holder [47] introduced three algorithms based on the minimum description length principle for the purpose of detecting three categories of anomalies that closely resemble normal behavior, including label modifications, vertex/edge insertions, and vertex/edge deletions. In addition, Shetty and Adibi [132] exploited an event-based entropy model that combines information theory with statistical techniques to discover hidden prominent people in an Enron e-mail dataset. However, none of these work is focused on detecting "*in-disguise*" anomalies in *multiple graphs*. Meanwhile, neither "*white crow*" nor "*in-disguise*" anomaly detection approaches have considered one of the

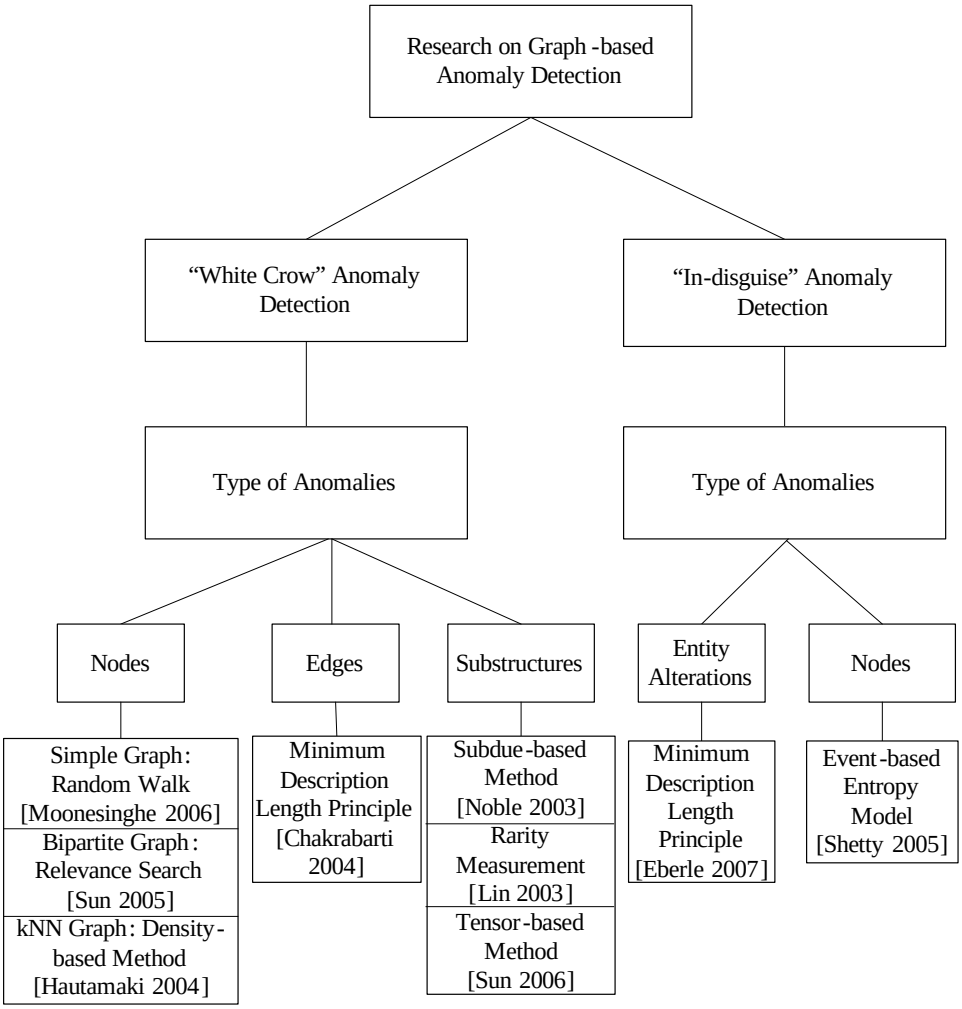


Figure 3.12: A summary of the various research directions in graph-based anomaly detection.

important properties of evolutionary networks: their community structure, which is sometimes referred to as clustering [55].

3.7 Conclusion

In this chapter, we have defined a new type of "in-disguise" anomaly, the *dynamic community*. In addition, we have proven that there are only six possible types of community dynamics in evolutionary networks: *grown*, *shrunken*, *merged*, *split*, *born*, and *vanished* communities. We have proposed a new method based on graph representatives and community representatives to reduce the computational cost. Based on the abnormal community decision rules, our algorithm

can discover meaningful results in evolutionary networks that cannot be detected by other graph-based anomaly detection algorithms. The main properties of our algorithm are as follows:

- it is parameter-free and automatic by nature;
- it is applicable to evolutionary networks characterized by overlapping communities; and
- it is scalable to large networks.

We have demonstrated the effectiveness of our algorithm over a number of synthetic as well as practical examples. Experimental results on real-world networks show that our algorithm can detect meaningful community abnormalities.

Chapter 4

Discovery of Anomalous Communities in Contrasting Groups of Networks

4.1 Introduction

Recent studies of the structure, dynamics, and function of complex networks have witnessed a growing interest. Such complex networks model a variety of systems including societies, ecosystems, the Internet, and others [108]. In particular, climate networks have lately emerged as a promising approach for modeling spatio-temporal dynamics of the climate system [58, 142, 157, 160]. In these climate networks, nodes (or oscillators) represent spatial grid points, and the edges between pairs of nodes exist depending on the degree of statistical interdependence between the corresponding pairs of time series taken from the climate data set [155].

Complex networks have enabled hypothesis-driven insights about the intricate interplay between the topology and dynamics of the physical system at different scales. For example, on the global scale, climate networks exhibit "small-world" properties due to teleconnections (i.e., edges linking geographically distant nodes), such as those in El Niño and La Niña climate networks [157, 58], that stabilize the climate system and enhance the energy and information transfer within the system [158, 159]. Likewise, the collective behavior of interacting subsystems in a network of different climate indices has explained the great climate shifts of the 20th century as synchronized transitions between different equilibria of oscillators representing the earth system [156].

To complement these fruitful hypothesis-driven studies, data-driven approaches to discovery of predictive insights from complex networks have emerged [141, 52]. A representative example

of such approaches focuses on detecting and characterizing the community structure, in which nodes are grouped into communities with more interactions (i.e., edges) within communities and fewer interactions between communities. A community is a common structure in many real-world networks [55, 160], including social networks, biological networks, and climate networks. However, the enormous size and the intrinsic complexity of the system data used for network construction challenge existing graph-based approaches and call for a paradigm shift in how the networks are analyzed.

Comparative analysis of multiple networks is a promising strategy. It can be performed at multiple levels for the purpose of (a) understanding climate dynamics over different time periods, (b) comparing multiple climate simulation models, (c) quantifying the agreement between climate simulation and observation data, or (d) correlating networks derived for different climate variables. Such analyses could translate to different problems on graphs, such as finding conserved network motifs to detect and track climate regions of similar behavior, or communities, over subsequent time windows [141], or graph-based anomaly detection to identify which communities have grown/contracted, merged/split, or born/vanished [27].

It is often the case that such multiple networks could be partitioned into different groups, such as those corresponding to different system phases; it is a known fact that a dynamic physical system often undergoes phase transitions in response to fluctuations induced on system parameters [71]. For example, in a tropical cyclone (TC) prediction system, one can build three different groups of climate networks, with one corresponding to high TC years, and another corresponding to medium TC years, and the other one for low TC years. Different groups of networks may exhibit different properties of the community structure. The question is how one could discover network motifs that could contribute to our understanding of the system's behavior for a given phase.

In this chapter, we hypothesize that anomalous communities, or dense subnetworks that are conserved within one group of networks but undergo statistically significant structural transformation in the other groups of networks, could be candidate structures for explaining physical basis underlying the group-related extreme events. For example, if an anomalous community corresponding to the El Niño/La Niña–Southern Oscillation (ENSO) climate index is identified, then the changes in such a community structure would explain why a particular season would enjoy low tropical cyclone activity or would be affected by the severity of the abnormally high number of hurricanes [135]. It is thus important to find effective means for detecting anomalous communities in contrasting (or system phase-related) groups of networks. To the best of our knowledge, such a problem has not been addressed before in literature.

It is worth noticing that performing such analyses for larger-scale, high-resolution physical models and over multiple heterogeneous data sources is a challenging problem not only compu-

tationally but also methodologically. For example, current algorithms for identifying conserved network motifs are limited in either the size [12, 171] or the number [84, 131] of networks they can effectively compare; plus they are not particularly designed for contrasting groups of networks. To detect the differences, one may want to find those communities that are conserved across dynamic networks derived from one data source but not conserved for the other data source. However, most algorithms do not support such contrast-based detection and tend to require that the motif be conserved in every one of the input networks [84, 131]. While some comparative techniques have been designed for the biological networks [54, 178], they only consider the structural or topological differences between pairs of networks. Similarly, previous work has been done on finding dense subgraphs that are present in a majority [176] or every member of a set of graphs [117], but neither of these are applicable to contrasting groups of networks, nor can they identify anomalous communities. Likewise, graph-based anomaly detection has been mainly focused on identifying anomalous nodes [106, 144], unusual edges [21], or small abnormal patterns [47] in a single graph, with few exceptions focusing on graph-based discovery of anomalies in noisy multivariate time series data [29], for multiple data sources [145], and across multiple graphs [22, 27, 143]. However, none of these approaches provides a means for detecting anomalous communities in contrasting groups of graphs.

Our approach follows from the need to address the graph classification problem of detecting predictive and phase-biased anomalous communities in contrasting groups of networks. We build groups of networks corresponding to different system phases, detect system phase-related components as seeds to help prune the search space in community generation, and use the proposed contrast-based techniques to discover abnormal communities that are further used to build the ensemble of classifiers for predicting the system states/phases.

4.2 Problem Statement

In this chapter, the ultimate goal is to detect and track phase-biased communities in contrasting groups of networks. Thus, in this section, we provide some formal definitions related to the community structure of a network. A weighted undirected graph is used to represent a complex network.

Definition 8 (Community)**.** *A community is a dense subgraph or a group of vertices within which the connections are denser than between different groups [55].*

In other words, a community is a "fuzzy cluster," or a quasi-clique, but not necessarily a "formal clique" with a set of vertices that are all adjacent to one another.

To be more specific, the community structure can be defined:

Definition 9 (γ-dense Community)**.** *Given a labeled graph G and a real value $\gamma \in [0.5, 1]$, a subgraph S of G is a γ-dense community, if and only if every vertex of S is adjacent to at least $\gamma(|S| - 1)$ of the other vertices of S [118, 175].*

The advantage of this community definition is two-fold. First, it corresponds nicely with the typical use of the term "density" in that it forces a certain fraction of the possible edges in the subgraph to exist. The second advantage is that our definition must be satisfied by every vertex of the community, ensuring that each vertex "belongs" to the community. One disadvantage of this definition is that it is not monotone; that is, a superset or subset of a γ-dense community does not need to be γ-dense, though basing our definition on the density of the subgraph rather than a maximum number of disconnections (as in a k-plex [7, 130]) gives us more flexibility in finding large subgraphs.

If a γ-dense-community contains a number of vertices in a seed or query set, we call it μ-enriched γ-dense community:

Definition 10 ((μ, γ)-community)**.** *Given a labeled graph G, a "seed" set of vertices Q, a real value $\gamma \in [0.5, 1]$, and a real value $\mu \in [0, 1]$, a γ-dense community S is μ-enriched with respect to Q, if and only if at least $\mu|S|$ vertices of S are contained in Q.*

The "seed" set Q can be used to incorporate the domain scientists' knowledge. For example, we can take in a biologist's prior knowledge as a set of "seed" proteins and identify all the communities in a biological network that contain some part of the "seed" proteins.

Fig. 4.1 shows an example of (μ, γ)-communities. If we set $\mu = 0.2$ and $\gamma = 0.75$, then only C_1 and C_2 in Fig. 4.1 can be considered $(0.2, 0.75)$-communities. Subgraph C_3 is not a $(0.2, 0.75)$-community, because it does not contain any "seed" node. Although subgraph C_4 has two "seed" nodes, not all of the vertices in C_4 are adjacent to at least three (*i.e.* $0.75 * (5-1)$) of the other nodes. Thus, it is also not a $(0.2, 0.75)$-community. But if we relax the requirements to be $\mu = 0$ and $\gamma = 0.5$, then all four subgraphs can be considered as communities.

One of the main ways in which (μ, γ)-communities differ from traditional communities, such as those produced by modularity-based clustering algorithms (e.g., [34, 164]) is that (μ, γ)-communities are allowed to overlap. As climatological factors in a particular region may contribute to multiple system events, this is a very desirable feature for a community detection algorithm to have in the climate domain, as well as other scientific domains like biological networks, where pathways or gene modules work in a cross-talking manner. While such algorithms may have other advantages, such as the parameter-free nature of clustering algorithms that maximizes modularity, and might work better for other domains like social networks, these algorithms are partitional by nature and typically heuristic, giving no guarantees of global optimality or the quality of individual communities.

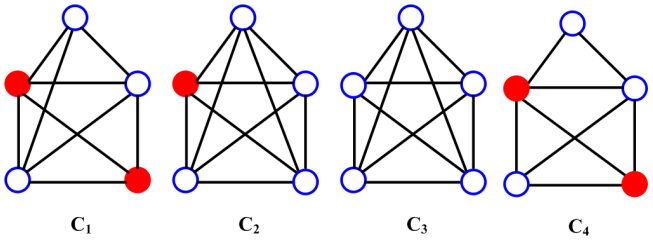


Figure 4.1: An example of (μ, γ)-communities. Filled nodes: seed nodes; Empty nodes: normal nodes.

Definition 11 (Corresponding Community)**.** *Given two communities $C_{i,m}$ and $C_{j,n}$ belong to networks G_m and G_n, $C_{j,n}$ is a corresponding community to $C_{i,m}$ if and only if $\frac{|C_{i,m} \bigcap C_{j,n}|}{|C_{i,m} \bigcup C_{j,n}|} > \alpha$, where $\alpha \in (0, 1]$ and $m \neq n$, and $|C|$ is the number of vertices in the community.*

For example, in Fig. 4.2, community $\{V_1, V_2, V_3, V_4, V_6\}$ of graph G_2 and community $\{V_2, V_3, V_4\}$ of graph G_4 are both corresponding communities to community $\{V_1, V_2, V_3, V_4\}$ in graph G_1, if we set $\alpha = 0.6$, $\mu = 0.1$, and $\gamma = 0.75$. Thus, each community can have multiple corresponding communities, and each corresponding community can correspond to several communities.

Definition 12 (Conserved Community)**.** *Given a set of k different networks $\{G_1, G_2, ..., G_k\}$, a community $C_{i,m}$ of graph G_m, where $1 \leq m \leq k$, is an (α, β)-conserved community, if and only if $C_{i,m}$ has an α-corresponding community in more than $\beta \times k$ networks, where $\alpha \in (0, 1]$ and $\beta \in [0.5, 1]$. If both α and β are larger than or equal to 0.5, we call this community a stable community in a group of networks.*

For example, community $\{V_1, V_2, V_3, V_4\}$ in graph G_1 (Fig. 4.2) can be considered as a conserved community, if α, β, μ and γ are set to be 0.6, 0.75, 0.1, and 0.75, respectively. Although community $\{V_1, V_2, V_3, V_4\}$ does not have any corresponding community in graph G_3, it still meets the requirement of a conserved community, because it has corresponding communities in the other two graphs.

Definition 13 (Anomalous Community)**.** *Given τ different groups of networks $\{U_1, U_2, ..., U_\tau\}$, a community C is an anomalous community if and only if C is an (α, β)-conserved community in one group of networks U_j, where $1 \leq j \leq \tau$, $\alpha \in [0.5, 1]$ and $\beta \in [0.5, 1]$, but C has no ω-corresponding community among the (α, β)-conserved communities of all the other groups of networks, where $\omega \in (0, \alpha)$.*

Fig. 4.3 shows an example of anomalous communities, where ω is set to be 0.4, and we assume that C_{11}, C_{12}, C_{21}, and C_{22} are conserved communities detected from two different

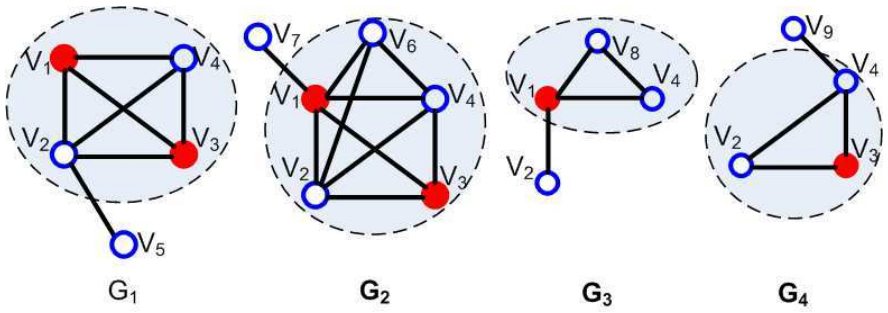


Figure 4.2: An example of corresponding communities and conserved communities. Filled nodes: seed nodes; Empty nodes: normal nodes; Dashed circles: communities.

groups of networks, U_1 and U_2 with $\alpha = 0.6$ and $\beta = 0.75$. C_{12} and C_{22} are anomalous communities, because they do not have any ω-corresponding community among the conserved communities of the other group.

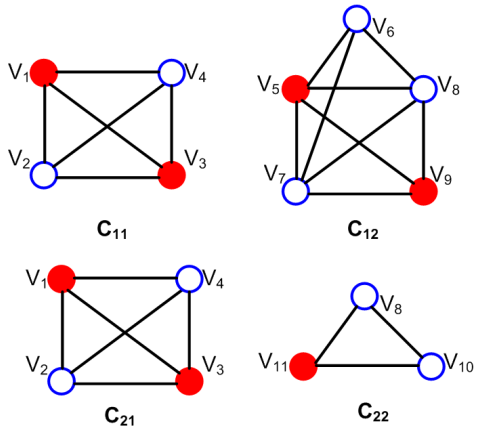


Figure 4.3: An example of anomalous communities. C_{11} and C_{12} are conserved communities from the network group U_1, and C_{21} and C_{22} are conserved communities from the network group U_2. Filled nodes: seed nodes; Empty nodes: normal nodes.

Problem 2 (Detecting Predictive and Phase-biased Communities in Contrasting Groups of Complex Networks)**.** *Given a* multi-phase *system that can be characterized by different groups of networks, the problem is to detect all the anomalous communities that are biased toward a target system phase from the training networks, and utilize all the detected phase-biased communities as the features to build an ensemble of classifiers to predict the unknown system phases on the*

testing data.

According to the statement of Problem 2, the main goal of our technique is to create an ensemble classifier for determining the phase-state of a network based on the phase-biased communities detected in the training set. Given a set of networks, we form this ensemble by: (1) identifying phase-related system components, (2) enumerating the (μ, γ)-communities enriched by these phase-related components, (3) identifying phase-biased communities, and (4) forming a classifier ensemble, where each member predicts the phase-state of a network based on the features in these phase-biased communities.

4.3 Method

Given the definitions and theorems, in this section we address the aforementioned technical challenges through some key innovative steps underlying the methodology. The methodology is summarized in Fig. 4.4.

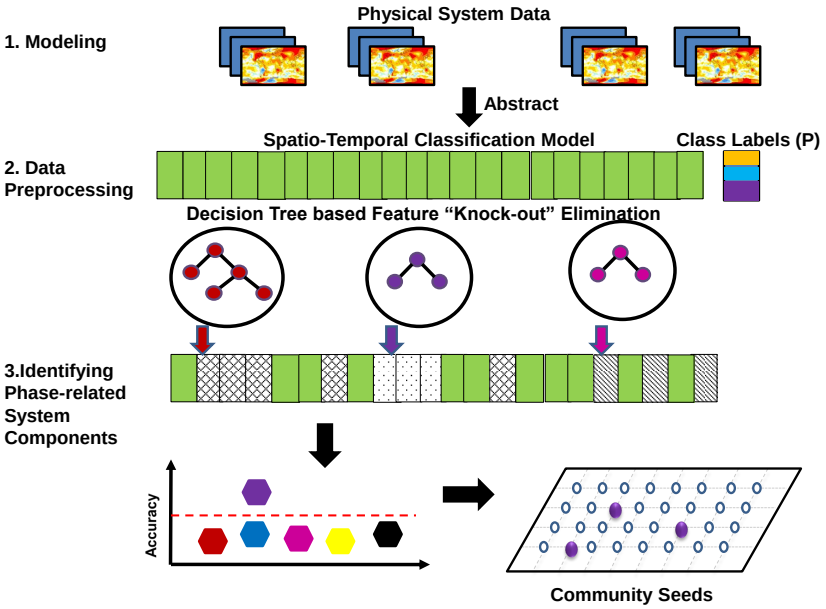
4.3.1 Step 1: Abstracting the Dynamic System

We first define the mathematical form for the dynamic system using climate spatio-temporal data as an example. Formally, let F be a set of variables (or factors) that characterize the system over spatial locations L over time period T. For example, the climate system could be characterized by its climatological factors, such as Sea Surface Temperature (SST), Sea Level Pressure (SLP), and Vertical Wind Shear (VWS) defined over spatial (latitude, longitude, altitude) grid points over a time period of 1950-2010 with monthly mean values.

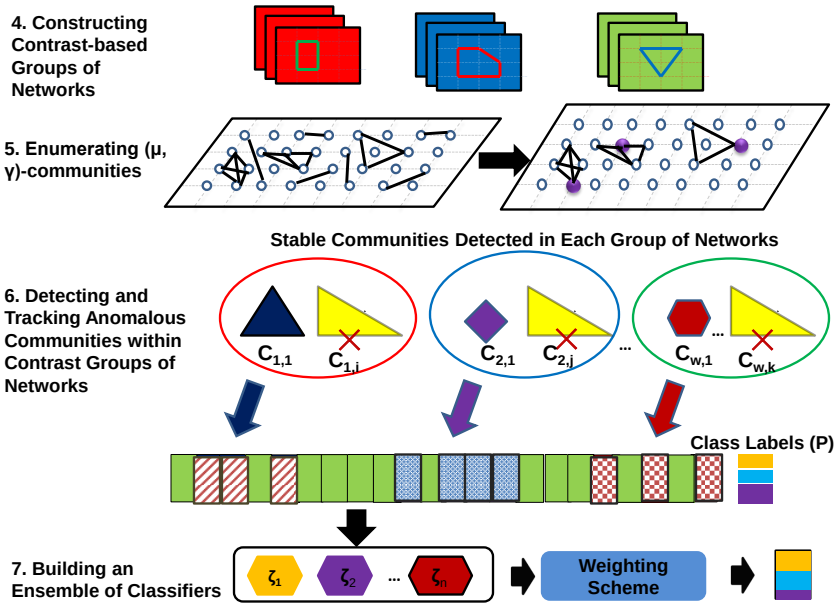
We divide T into disjoint segments $T_1, T_2, \ldots, T_m$ (say, calendar years), where each T_j can be further split into an *observable* time period $T_{j,o}$ and a *forecasting* time period $T_{j,f}$, according to time frame of the extreme event.

In the context of hurricane extreme events, for example, each time interval T_j may correspond to a calendar year that is further divided into a hurricane season $T_{j,f} = \{\text{July-November}\}$, for which hurricane activity, say in the North Atlantic region, is being forecasted based on the observed or simulated monthly means for climatological factors defined over the entire globe L during the hurricane pre-season, $T_{j,o} = \{\text{November-June}\}$.

We consider the problem of classifying the climate system's state P over these time intervals according to some event-specific taxonomy. For example, according to paper [31], seasonal hurricane activity of Taiwan region could be broadly categorized as "above normal" (say, more than four hurricanes during the hurricane season), "normal," or "below normal" (say, less than three hurricanes in a season).



(a) Step1-Step3



(b) Step4-Step7

Figure 4.4: The overview of our methodology.

Based on the aforementioned notations, the mathematical form can then be defined as follows (Step **1**, Fig. 4.4). Let each row of the matrix correspond to each time interval T_j, $j \in \{1, 2, ..., m\}$, and let each column of the matrix correspond to a 3-tuple defined over $\mathcal{F} = F \times L \times T_{*,o}$, where $T_{*,o}$ is replaced with $T_{j,o}$ for the corresponding row T_j. Thus, each (row, col) cell of the matrix is filled in with the value of the corresponding variable in F for column col defined at the corresponding spatial point in L and the corresponding time $T_{row,o}$.

Furthermore, let us assume that a set of known extreme events E is defined over some spatial region L_e, and the class label from P is assigned to each time interval T_j based on the accumulative statistics of the observed events over $T_{j,f}$ time period in region L_e.

Fig. 4.5 illustrates this mathematical abstraction using SST as variable, or predictand, defined over $T = (1970 - 1972)$ during the months of $T_{*,o} = \{\text{May, June}\}$ over (latitude, longitude) spatial grid points for the sea-level altitude. The class label is inferred based on the historical record of observed hurricanes in North America during $T_{*,f} =$ (July-November) hurricane season.

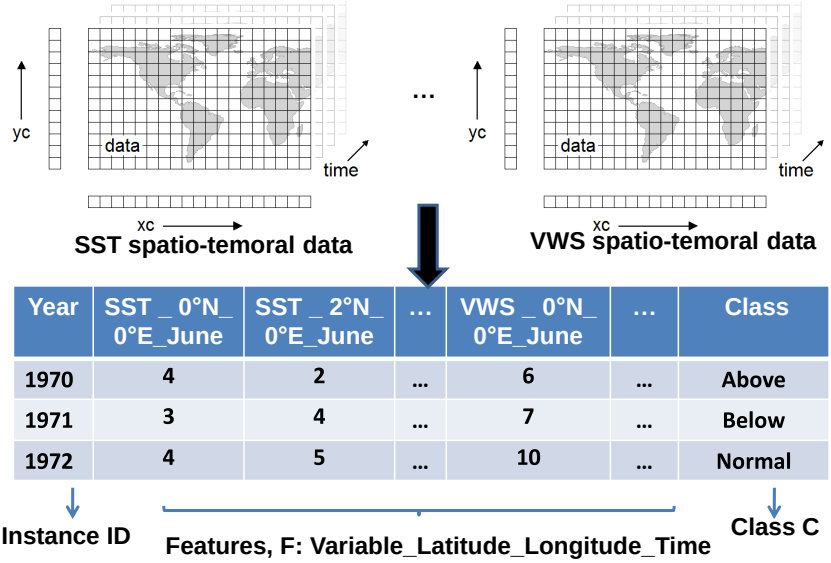


Figure 4.5: Our proposed mathematical form for classification of spatio-temporal data.

4.3.2 Step 2: Data Preprocessing

Given the aforementioned mathematical form of the original system data, the next step of our algorithm is data preprocessing designed to help us identify phase-related community seeds

in Step 3 (see Section 4.3.3). While the choice of which data preprocessing techniques to employ may be dependent on the type of data under consideration, for preprocessing spatio-temporal data, we use two techniques including *spatio-temporal deseasoning* and *discretization-based denoising*.

Spatio-temporal deseasoning: If temporal data can exhibit seasonality, such as winter, spring, summer, and fall, each variable's time series at each spatial location is first transformed into the time series with zero mean and unit variance per season. This technique avoids learning a strong seasonality signal and also enables multiple variables with different scales of measurement to be combined into different columns of the same matrix.

Discretization-based denoising: Dynamic system data like the climate data contains a lot of "noise" or irrelevant signals, so another important preprocessing step is to perform data cleaning or data denoising. We use a discretization method by Fayyad and Irani [50] to filter out noise or irrelevant features in the data. This technique has been found to be effective in some domains like microarray analysis [148], where non-discriminatory genes are filtered out before performing actual learning process on the gene expression data.

4.3.3 Step 3: Identifying Phase-related System Components

In this section, we aim to detect the phase-related system components or features, which can be used as seeds to generate (μ, γ)-communities in a network (see Step 5).

Given the mathematical classification form (Step 1) and the preprocessed spatio-temporal data (Step 2), we deploy decision tree based procedure for identifying the candidate system phase-related components or features.

There are multiple reasons for why we use a methodology based on decision trees for our feature space partitioning, including (a) their efficiency in processing many features (unlike Bayesian Belief Networks (BBNs), which have exponential complexity relative to the number of features), (b) support for multi-class data sets (unlike Support Vector Machines (SVMs), which are inherently binary classifiers), (c) the ability to handle continuous and multi-variate types of features (unlike Neural Networks (NNs), for which distance metrics are poorly defined for mixed data types), among others. We use the Classification and Regression Trees (CART)-decision tree algorithm [16] to select a set of discriminatory features from the available feature space. Basically, CART builds a decision tree by choosing the locally best discriminatory feature at each split step based on the Gini Index Impurity Function. To avoid overfitting, CART employs backward pruning to build smaller, more general decision trees. CART chooses features in a multivariate fashion, which allows the feature selection process to find a set of discriminatory features instead of considering one feature at a time.

More importantly, especially in the context of underdetermined or unconstrained problems,

CART's inherent feature pruning capability often leads to a smaller number of features. Also, decision boundaries themselves could result in rules that are more interpretable and could provide additional insights to domain scientists on how much the identified features affect the system's state. Not only is it important to know what group of features contributes to the system's state, but also to what extent the feature values influence the system's state.

Algorithm 4: Phase-related component enumerator

Input:
$\mathcal{F}$: a set of features
D: a set of training data over $\mathcal{F}$
P: a set of system states over D
A: basic classification algorithms
: (e.g., decision tree, SVM, Naïve Bayes, etc.)
Output:
CIG: identified community seeds

```
1  CIG ← ∅
2  while stopping criterion is not met do
       /* Run CART-decision tree to get a set of candidate features   */
3      Run decision tree algorithm on D with feature set F to get a pruned decision tree M
4      Let F_M be a set of all features that belong to the internal nodes of M
5      D_{F_M} ← Extract the data from D only with the features in F_M
6      Predictive skill score ε_M ← applying A to train D_{F_M}
7      if ε_M meets the training accuracy criterion then
8          Add F_M to CIG
9      Remove features in F_M from F
10 return CIG
```

Specifically, we identify a candidate set of discriminatory features by building a decision tree model M using CART and extract the features that correspond to the internal nodes of M (Lines 3–5 in Algorithm 4). The candidate system's features are then assessed in terms of their ability to contribute to the system's states. Basically, the goal is to define a scoring function that measures how well each group of features discriminates between system states. We define a scoring function in terms of classification accuracy (training accuracy in our experiments) provided by multivariate discriminant methods, such as SVMs, BBNs, neural networks, or decision trees. Specifically, we ask a question: if we used only the given set of candidate features to determine the system's state, how much predictive ability would this set have? Since individual features within the candidate group could be related to each other in a complex manner, we first let a proper classifier (e.g., kernel SVM or BBN) learn these complex relationships from the candidate features and predict the state of the system by using the candidate features only

(see Lines 5–6 in Algorithm 4). If the training accuracy of the candidate feature set is above the threshold we set, the features are added to the community seed set.

The combinatorial nature of this task necessitates heuristic approaches. Our strategy is inspired by the way biologists often conduct their mutagenesis studies. Namely, they *knock-out* a group of genes (e.g., via gene deletion) and observe the *mutant* system's response. By analogy, our methodology *knocks-out* the selected candidate feature sets and proceeds in an *iterative* fashion until some *stopping criterion* is met (see Line 2 in Algorithm 4). Under this approach, each iteration produces a subset of features out of the current feature set (see Line 4 in Algorithm 4), then removes these features from the set so that they can't be selected again (see Line 9 in Algorithm 4). The maximum number of iterations is set as our stopping criterion. A set of phase-related features or components is output, when the stopping criterion is met.

4.3.4 Step 4: Constructing Contrast-based Groups of Networks

There are several steps to construct climate networks, including constructing nodes of a network, calculating anomaly value, building edges of a network, and partitioning the networks into different groups.

The nodes (or oscillators [155]) of a climate network are identified with the physical locations or spatial grid points, which correspond to the time series of gridded climate data (see Fig. 4.6).

Year	Month	Day	(0°N, 0°E)	(0°N, 2°E)	...	(90° N, 180° E)
1970	1	1	3	6	...	7
1970	1	2	3	7	...	6
...	...	...	...	...	...	...
1970	12	31	12	10	...	22

Time Series

Figure 4.6: A table-view of spatio-temporal data.

At each grid point, we calculate for each month $m = 1, ..., 12$ (i.e., separately for all Januaries, Februaries, Marches, etc.) the mean $\theta_m = \frac{1}{Y}\sum z_{m,y}$ and standard deviation $\sigma_m = \sqrt{\frac{1}{Y-1}\sum(z_{m,y} - \theta_m)^2}$, where y is the year, Y is the total number of years in the dataset, and $z_{m,y}$ is the value of series Z at month m and year y. Each data point is then transformed by using z-score transformation, that is each data point is ($\overline{z_{m,y}} = \frac{z_{m,y} - \theta_m}{\sigma_m}$) subtracted the mean and divided by the standard deviation of the corresponding month.

The edges between pairs of nodes exist depending on the degree of statistical interdepen-

dence between the corresponding pairs of time series taken from the climate data set. The Pearson correlation coefficient is chosen as a measure of link strength [155]. For two series Z and X the correlation r is computed as $r(Z,X) = \frac{\sum(z_i-\bar{z})(x_i-\bar{x})}{\sqrt{\sum(z_i-\bar{z})^2 \sum(x_i-\bar{x})^2}}$, where z_i is the i^{th} value in Z and $\bar{z}$ is the mean of all values in the series. Note that the correlation coefficient has a range of [-1, 1], where 1 denotes perfect agreement and -1 perfect disagreement, with values near 0 indicating no correlation. Since an inverse relationship is equally relevant in the present application, we set the correlation score to $|r|$, the absolute value of the correlation coefficient. Although nonlinear relationships are known to exist in climatological systems, the observed similarity of Pearson correlation still can be considered statistically significant, as concluded by Donges *et al.* [44]. Thus, we use Pearson correlation to measure the similarity between a pair of nodes in this work.

A correlation-based pruning is applied to the networks to prune the edges, that is only the pairs of nodes with the correlation scores above some threshold would be considered connected. To avoid the multiple comparison problem, the Monte Carlo method is used. Specifically, for each network, we randomly sample N sets (say, $N = 1,000$) from the entire edge set of the tenth size as the original network, and compute the corresponding correlation threshold with p-value $= 0.05$ from each sample set. The selected threshold for the target network is the one that meets 95% confidence level within the threshold distribution for N samples.

Because the networks change over time, we build a network according to a calendar year per climate variable. For example, for a time period over 1950-2009 with two climate variables (e.g., SLP, SST), up to 120 different networks can be built with one network per year for each variable.

The complex networks of a dynamic system can be partitioned into different groups corresponding to different system's states (i.e., class P in Fig. 4.5). For example, in a tropical cyclone (TC) prediction system, we can build three different groups of climate networks, with one corresponding to strong TC years, one with normal TC years, and another with low TC years, based on the distribution of historical data.Different groups of networks may exhibit different properties of the community structure.

4.3.5 Step 5: Enumerating (μ, γ)-communities

We hypothesize that if the system feature or component is key to defining the system's state then its value distributions will be separable between the observations from different states. If the separation is strong, then such a feature, alone, is likely able to discriminate system states. And almost any method, like entropy-based, would likely succeed in detecting those features. However, with real data sets such a strong separation is less likely. There are different reasons for such an assumption. For example, the evolution of system behavior may induce non-

functional changes to the system features. Thus, the effective analysis should not only include an individual feature with a strong discriminatory signal, but also extend to a group(s) of interplaying features out of a set of thousands of features. This creates a multiplicity of possible combinatorial interplays to search for and excludes a possibility for a brute-force enumeration.

In some cases, the domain knowledge may assist with constraining the search space of possible interplays. For example, climate index El Niño/La Niña–Southern Oscillation (ENSO) has been found to be highly correlated with hurricane activities [135]. For a more general and domain-independent solution, however, the issue of properly constraining the search space still remains.

Standard algorithms would attempt to find all dense subgraphs throughout the networks. However, in real-world dynamic system data, there are a lot of irrelevant features or "noises." Including all features including the "noises" to generate the dense subgraphs would retrieve a huge number of results irrelevant to the system phases or states. We hope to reduce the problems of high algorithmic complexity and the number of irrelevant results by integrating the system phase-related components or features into the search in the form of a "seed set" of vertices. For example, given a phenotype-expressing organism, a biologist might have known a set of proteins that are related to the target phenotype. By using those proteins as the "seed" set, we can identify all the dense functional modules in a biological network that contain some part of the "seed" vertices.

Thus, Dense and Enriched Subgraph Enumeration algorithm (*DENSE*) [70] is used in this work to generate (μ, γ)-communities. Specifically, given the set of phase-related system components as seeds (Step 3) and a constructed network (Step 4), the basic premise of *DENSE* algorithm is that we will build the (μ, γ)-communities one vertex at a time, starting with a single query vertex v_0 and backtracking as we find maximal (μ, γ)-communities or subgraphs that cannot be contained in a (μ, γ)-community. The details of the *DENSE* algorithm can be found in paper [70].

4.3.6 Step 6: Detecting and Tracking Anomalous Communities in Contrasting Groups of Networks

The anomalous communities in the contrasting groups of networks are more "biased" towards the target system phases than the communities in a single network, or conserved (or stable) communities in a group of time-varying networks.Thus, in this section, our goal is to extract only the anomalous communities from all communities generated from different groups of networks.

Based on the Definition 13, in order to identify anomalous communities, we first need to detect all (α, β)-conserved communities in each group of networks, where $1 \leq j \leq i$, $\alpha \in [0.5, 1]$ and $\beta \in [0.5, 1]$. A stable community should have at least one α-corresponding community in

Algorithm 5: Anomalous community detection algorithm, continued in Algorithm 6.

Input: C: All communities generated from all graphs
in contrasting groups $\{U_1, U_2, ..., U_\tau\}$
β, ω, α: Parameters
Output: χ: A set of anomalous communities
`/* Detecting stable communities in each group of networks */`
1 **for** $i = 1 : \tau$ **do**
2 | anomaly_indicator = 0
3 | SC_i= **call** `Detecting`
`/* Using the` τ `sets of` SC `as inputs for detecting anomalous communities */`
4 anomaly_indicator = 1
5 $\alpha = \omega$
6 χ= **call** `Detecting`

majority of the networks of the same group. That is the size of overlapping parts between the stable community and its "strict" corresponding community should be larger than half (at a minimum) the size of any of them. Algorithm 6 summarizes the aforementioned stable community detection procedure. After detecting stable communities from all groups of networks, each stable community is examined to see if it has any "looser" corresponding community (with minimum intersection factor ω, where $\omega \in (0, \alpha]$) in the set of stable communities of all the other groups. Only those communities that do not have any "looser" corresponding community will be considered as anomalous communities.

The anomalous community detection between the different sets of stable communities (with each set generated from each group of networks) only requires a little change with regard to the input variables (see Lines 1 to 6 in Algorithm 5) and the output process (see Lines 14 to 16 in Algorithm 6).

4.3.7 Step 7: Building an Ensemble of Classifiers from Anomalous Communities

While the enumerated set of anomalous communities is important in its own right (as illustrated in Section 4.4), here we combine them altogether by building an ensemble of classifier models.

For each of the anomalous communities χ identified, we specifically distinguish between treating it as a *binary* feature (i.e., the community is present or absent in a graph) or *continuous* features, that is we form a new data set D_χ by restricting the original data to include only the features (or spatial grid points) F_χ in χ. We then train a separate base classification algorithm A (e.g., decision tree, SVM, Naïve Bayes, etc.) on the binary data set or the restricted data set to construct a candidate classifier model ζ. The candidate classifier model ζ will only be included into the ensemble of classifiers if it meets the *model selection criterion*. The resulting

Algorithm 6: `Detecting` function

Input: α, β: Parameters for conserved community
C: All communities in k graphs,
or stable communities from k groups
anomaly_indicator: An indicator for anomalous community detection
Output:
η: A set of detected communities

```
Initialize count
for snapshot s = 1 : (k − 1) do
    for snapshot n = s + 1 : k do
        indicator =0
        for each community C_{s,i} in G_s do
            for each community C_{n,j} in G_n do
                overlap_part = |C_{s,i} ⋂ C_{n,j}|
                if overlap_part/|C_{s,i} ⋃ C_{n,j}| > α then
                    count_{s,i} = count_{s,i} + 1
                    if indicator=0 then
                        count_{n,j} = count_{n,j} + 1
                        indicator =1
                    break
if anomaly_indicator = 0 then
[I J]=find(count > β ∗ k)
else
    [I J]=find(count < β ∗ k)
Add C_{I,J} to η for each pair of I and J at the same row
Delete duplicate communities in η
Output η
```

class prediction for the event with the unknown class label is based on the majority voting of the selected classifiers ζ's.

Some of the key characteristics for building a robust classifier ensemble include (a) the diversity among the classifier models in the ensemble and (b) the reasonably high accuracy of the individual members in the ensemble. In our case, the former is ensured due to our feature set knock-out strategy (Step 4) and the latter is guaranteed by a combination of the scoring function (Step 2) and the statistical significance assessment (Step 3) that, in combination, also reduce possible redundancy among the models and thus reduce the possible bias (e.g., due to a significantly large portion of highly similar models).

Finally, in the last step (Step 7 in Fig. 4.4), we need to combine the predictions of all the classifiers that pass statistical significance criterion (Step 3) to come up with the final prediction value. In order for the ensemble to make a prediction, each classifier is given a weighted vote, and the class with the most votes is the prediction of the ensemble. We tested three possible weighting schemes [150]: a simple majority voting scheme, in which every classifier is given equal weight; a training error-based method, in which every classifier is weighted based on its training error; and a confidence-based method, in which each classifier is weighted by that model's associated confidence value. Due to space limitations, we present results for a simple case, majority voting.

4.4 Experimental Results

The nature of the proposed methodology suggests that detected anomalous communities from contrasting groups of networks (Steps 1-7) (1) could play an important role in defining the system's state(s) and (2) collectively, could improve the predictive skill of the system's states (Step 7). We also demonstrate the efficiency of our algorithm by applying it to the synthetic datasets.

4.4.1 Data and Tasks

Two real-world extreme event prediction tasks are considered:

1. Seasonal tropical cyclone prediction: The first task is to predict the seasonal tropical cyclone (TC) count in some spatial region [57, 89]. TCs, especially hurricanes, have become a serious issue of our era because they result in enormous loss of life and property.

2. African Sahel rainfall prediction: The second task is to predict the seasonal rainfall in North Africa, especially, in the Sahel area [172]. Rainfall in this area is highly related

to meningitis epidemics that affects more than 200,000 people throughout the region annually.

We use the North Atlantic tropical cyclone (TC) count series from 1950 to 2009 from the seasonal (July through November) Atlantic hurricane database (HURDAT) at the National Climatic Data Center to form the class labels. We also utilize the North Pacific seasonal (June through October) TC count series from 1970 to 2006 provided by the Central Weather Bureau [31]. Monthly rainfall data is obtained from the Climate Research Unit at a $0.5^\circ \times 0.5^\circ$ latitude and longitude resolution for the period of 1950–1998. East Sahel rainfall indices are obtained by averaging seasonal (July through September) mean precipitation data over (10-20°N, 15–30°E).

The monthly mean sea level pressure (SLP), precipitable water (PW), sea surface temperature (SST), and tropospheric vertical wind shear (VWS) data are used for the North Atlantic TC, North Pacific TC and Sahel rainfall class prediction. SLP and PW are NCEP/NCAR reanalysis datasets. They are available at a $2.5^\circ \times 2.5^\circ$ latitude and longitude resolution. SST is from the NOAA Climate Diagnostic Center in Boulder, Colorado, at a resolution of $2^\circ \times 2^\circ$ latitude and longitude. VWS is calculated by computing the square root of the sum of the square of the difference in zonal wind component between 850 and 200 hPa levels and the square of the difference in meridional wind component between 850 and 200 hPa levels [33] from NCEP/NCAR reanalysis data.

The observed extreme event count series of the target system are classified into three classes: below normal, normal, and above normal, with a distribution of 40% as normal and 30% each as below normal and above normal. For instance, in the case of Taiwan region TC prediction [31], years with fewer than three seasonal TCs are classified as below normal, and years with at least five TCs are classified as above normal.

We use parameters $\gamma = 0.75$ and $\mu = 0.001$, which correspond to searching for dense but not necessarily complete subgraphs as communities that contain at least one of system phase-related components. We use parameters $\alpha = 0.6$, $\omega = 0.4$, and $\beta = 0.6$ for defining the anomalous communities.

4.4.2 State Determining Communities

Climate Indices Associated with Hurricane Activities:

Table 4.1 shows four different anomalous communities, representing functionally associated or synchronized groups of oscillators (or spatial grid points), detected by our algorithm for North Atlantic tropical cyclone prediction. In each community, our algorithm is able to identify at least one oscillator corresponding to a known climate index related to tropical cyclone activity.

For example, for the first anomalous community detected from the SST networks, we can see that one oscillator is located in the Niño 3 region. Niño 3 SST has a strong correlation with Atlantic hurricane activity [57, 89]. Another oscillator belongs to the El Niño/La Niña-Southern Oscillation (ENSO) region, which has been found to modulate the tropical systems and strongly influences North Atlantic tropical cyclones [135].

The second anomalous community identified oscillators in the hurricane main development region (MDR) and North Atlantic Oscillation (NAO). The MDR index has been shown to contribute to the hurricanes generated in the MDR region [127, 170]. And the NAO index, especially the June NAO, has been found to be correlated with North Atlantic hurricane tracks of the incoming hurricane season [49, 170]. The Pacific Decadal Oscillation (PDO) index was identified in our third community. Shifts in the PDO phase can have significant implications for Atlantic hurricane activity, and significant differences are shown in hurricane intensity between El Niño and La Niño years when the PDO is in the warm phase [153]. The PDO index is also identified in the fourth anomalous community. Our algorithm also finds some other anomalous communities, which correspond to other climate indices like Atlantic multidecadal Oscillation (AMO) and Artic Oscillation (AO) that might affect the North Atlantic tropical cyclone activities too, though this has not been reported in the literature. There are other 342 anomalous communities detected by our algorithm.

Table 4.1: Identified climate indices related to hurricane activities

Community ID	Variable	Spatial location	Climate indices
1	SST	(4°N, 114°W)	Niño 3
		(2°S, 168°W)	ENSO
		(42°N, 30°W)	
		(32°S, 16°W)	
2	VWS	(27.5°N, 65°W)	MDR
		(52.5°N, 37.5°W)	NAO
		(7.5°N, 122.5°W)	Niño 3
		(10°S, 60°W)	
		(27.5°N, 55°E)	
3	PW	(52.5°N, 135°E)	PDO
		(82.5°N, 15°W)	AO
		(37.5°N, 40°E)	
4	SLP	(57.5°N, 22.5°W)	NAO
		(60°N, 155°E)	PDO
		(37.5°N, 162.5°W)	
		(12.5°N, 122.5°E)	

African Sahel Rainfall-related Teleconnection Patterns: For the African Sahel region rainfall

prediction case, our algorithm also detected some anomalous communities with one shown in Fig. 4.7.

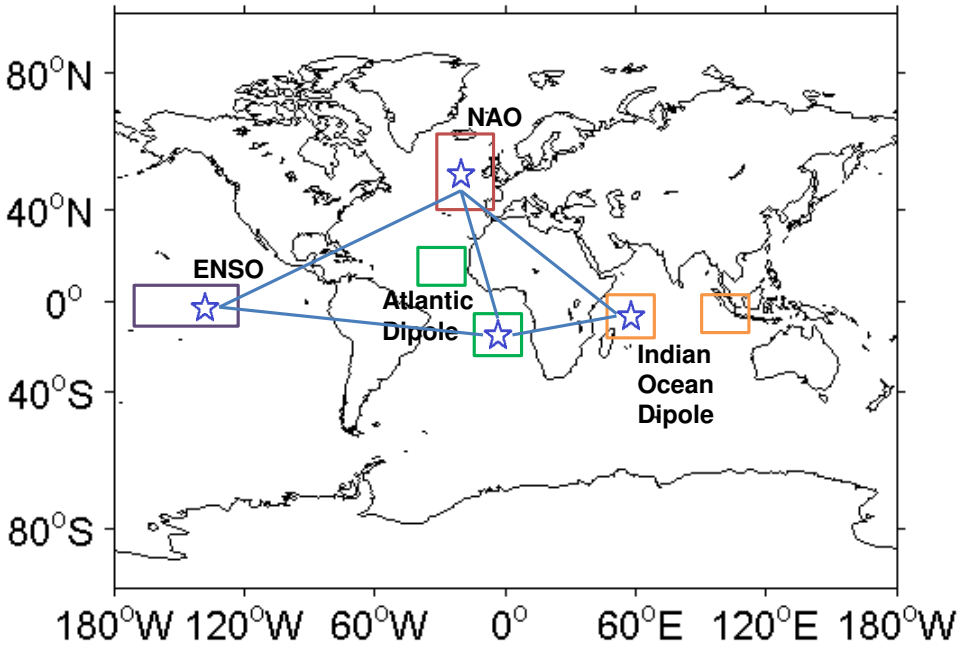


Figure 4.7: One anomalous community detected for African Sahel rainfall prediction.

Climate variability in the tropical Atlantic involves complex but interacting processes that actively or passively exert their influences on rainfall and relative humidity variability over West Africa [146]. Moisture supply over West Africa primarily emanates from the eastern equatorial and South Atlantic, determined from the strength of the meridional and the zonal modes. However, other teleconnection patterns such as ENSO, NAO, and Indian Ocean dipole are competitively engaged to dictate the rainfall and relative humidity variability at different scales. The equatorward extension of the extratropical NAO pattern influences the West African climate by weakening the northeasterly trades, whose presence is a prerequisite to the formation of large-scale convergence over the continent to reinforce convective development. NAO also influences the region's climate through a modification of the northern lobe of the meridional mode. Thus, the detected anomaly community shown in Fig. 4.7 appears to support the hypothesis proposed by our climate scientists (our co-authors), which is being further investigated, that the NAO modulates meridional moisture transport over the tropical Atlantic, mediated mainly through the zonal equatorial trades.

4.4.3 Predictive Skill of System's States

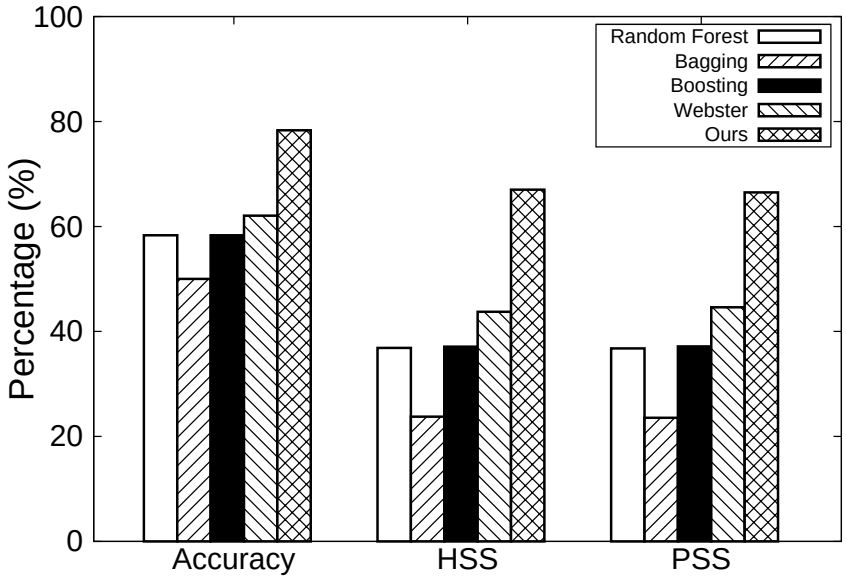
Performance Evaluation Method: Because of the small sample size of the spatio-temporal data, leave-one-out cross validation (LOOCV) is employed to evaluate the robustness of our methodology. We utilize several metrics to evaluate the performances: accuracy, Heidke Skill Score (HSS) [83], and Peirce Skill Score (PSS) [83]. Accuracy is defined as the ratio of the number of correctly classified data points to the total number of data points in the test set. The HSS measures how well a forecast performs compared to a randomly selected forecast [83]. And PSS, also called "true skill statistic," is another popular skill score computed by the difference between the hit rate and the false alarm rate [83].

Performance Comparison:

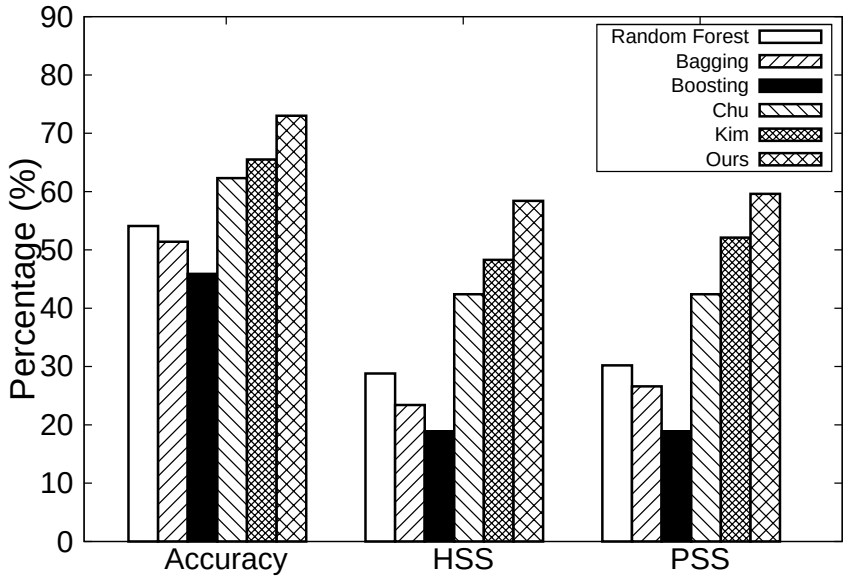
Figure 4.8 compares our algorithm performance to seasonal tropical cyclone predictions by Chu *et al.* [31], Kim *et al.* [90], Kim and Webster [89], and three benchmark ensemble classification methods: random forest, bagging, and boosting. The same basic classifier–CART decision tree, and the same data including four variables (SST, SLP, VWS, and PW) with all features are used for all methods. For the North Pacific region, there is a roughly 8% increase over the 65.5% reported by Kim *et al.* [90]. For the North Atlantic region, our method achieves an increase of at least 16% in accuracy and 20% in HSS and PSS over the four benchmark methods.

To estimate the contributions of each module in our algorithm to the performance improvement, we implemented different versions of our algorithm: the *original-continuous* version (OC) includes all the algorithm modules by using the *continuous* community features (see Section 4.3.7); the *original-binary* version (OB) also includes all the algorithm modules but uses the *binary* community features; the *brute-force* (BF) version uses all original features without detecting the anomalous communities, but it builds the classifiers by using our ensemble method (see Step 7 in Fig. 4.4); the *all-community* (AC) version enumerates all γ-dense communities without using the phase-related components as the query set (see Step 3 in Fig. 4.4), while keeping the other steps in the *original-continuous* algorithm unchanged; and the *random forest with anomalous community detection* (RFC) version changes only one step in the *original-continuous* algorithm by using the random forest instead of our ensemble method to build the ensemble of classifiers. Among those, AC is the most time-consuming version because it generates all possible γ-dense communities without using any query vertex. Irrelevant communities containing all "noises" can be generated as well, which would affect the prediction performance.

Table 4.2 compares the performances of different versions on seasonal North Atlantic tropical cyclone prediction. The *original-continuous* version outperforms the *original-binary* version by 2% using the *binary* community features. The accuracy decreases by 10% if we did not use the anomalous communities as the features, and decreases by 7% if we used γ-dense communi-



(a) North Atlantic TC prediction



(b) North Pacific TC prediction

Figure 4.8: LOOCV performance for seasonal TC prediction.

ties instead of (γ, μ)-communities. And our ensemble method outperforms the random forest method by 9% using the same selected anomalous community features.

Table 4.2: Different modules' contributions on performance

Metric	OC	OB	BF	AC	RFC
Accuracy	0.82	0.8	0.72	0.75	0.73
HSS	0.72	0.69	0.58	0.60	0.58
PSS	0.72	0.68	0.59	0.62	0.60
GSS	0.71	0.68	0.55	0.63	0.60

4.5 Discussion

4.5.1 Parameter Selection

Our algorithm requires five parameters: the enrichment (μ) and the density (γ) for defining the communities, and parameters ω, α, and β for defining the anomalous communities. The description of these parameters (in Section 4.2) suggests that higher values of γ will produce more connected (clique-like) subgraphs. Similarly, higher values of the enrichment ($\mu \geq 0.5$) will produce subgraphs that are primarily composed of the "query" vertices, whereas a very low value ($\mu \leq 0.001$) will result in enumeration of all the subgraphs that satisfy the γ threshold and contain at least one query vertex. And higher values of α and β will produce fewer conserved communities in each group of networks, whereas higher ω will result in more anomalous communities.

Parameter thresholds depend on the application. In this work, we are interested in identifying phase-biased communities in contrasting groups of climate networks, given a set of extreme event-related climatological oscillators as a "seed" set. Setting μ value to 0.001 will result in finding all the communities containing at least one "seed" vertex that could potentially be related to the spatio-temporal extreme events. Since climate networks are prone to missing information (edges), the value of $\gamma = 1$ could be too stringent, and the algorithm may miss some of the extreme event-related communities. Hence, we chose a γ value of 0.75 (midpoint of 0.5 and 1) to identify highly connected (but not fully connected) subgraphs as most probable communities that are teleconnected (i.e., edges linking geographically distant nodes) with extreme event-related "seed" oscillators. And due to the dynamics of climatological systems, we set the value of $\alpha = 0.6$ and $\beta = 0.6$ to find all possible but highly phase-related conserved

communities in each group of networks. Finally, a relatively small value of $\omega = 0.4$ (smaller than α) is chosen to make sure that the anomalous communities are only conserved within one group of networks, not in the other groups of networks.

Fig. 4.9 shows the sensitivity analysis results of the five parameters on North Atlantic TC prediction. The default values for the five parameters are: $\mu \leq 0.001$, $\gamma = 0.75$, $\alpha = \beta = 0.6$, and $\omega = 0.4$. We only change the value of one parameter at a time to test the sensitivity. The results shown in Fig. 4.9 agree with the aforementioned parameter analysis.

4.5.2 Generalization: Detecting Biologically Relevant Functional Modules through Biological Networks

Thus far, we have presented how to detect phase-biased communities from climate networks. But our algorithm can be applied to other domains as well. Here, we provide a general idea on how our algorithm can be used to detect functional modules through biological networks.

The biological networks like gene functional association networks can be obtained from the STRING database [80]. The nodes in the networks are genes. And a pair of nodes is connected with an edge if the corresponding genes are considered to be functionally associated by some evidence. The edge weights are assigned by the STRING database based on the evidence that support the functional association [80].

For a set of networks corresponding to phenotype-expressing organisms, we hypothesize that the conserved α-corresponding communities across the group of networks are the phenotype-associated functional modules. After generating all communities from each biological network, we first detect the α-corresponding communities across two networks, and then check if the α-corresponding communities detected in the previous two networks are conserved in the third network. This procedure is continued until all networks in the group are examined.

We can take it one step further and use a group of contrast biological networks (i.e., networks of organisms that do not express the phenotype) to filter and obtain communities that are not only identified as conserved in the previous step but are also "biased" towards the target phenotype. Here, by biased, we mean occurring in phenotype expressing organisms but not occurring in the phenotype non-expressing organisms. To achieve this goal, first, the networks are partitioned into different groups according to the phenotype(s), and then the conserved community detection algorithm is applied to each group of networks. After getting all the conserved communities from all groups, we remove all the common conserved communities appearing in at least two groups of networks. The remaining anomalous communities are the phenotype-associate functional modules, which can be used to improve the predictive skill of the system's phenotypes.

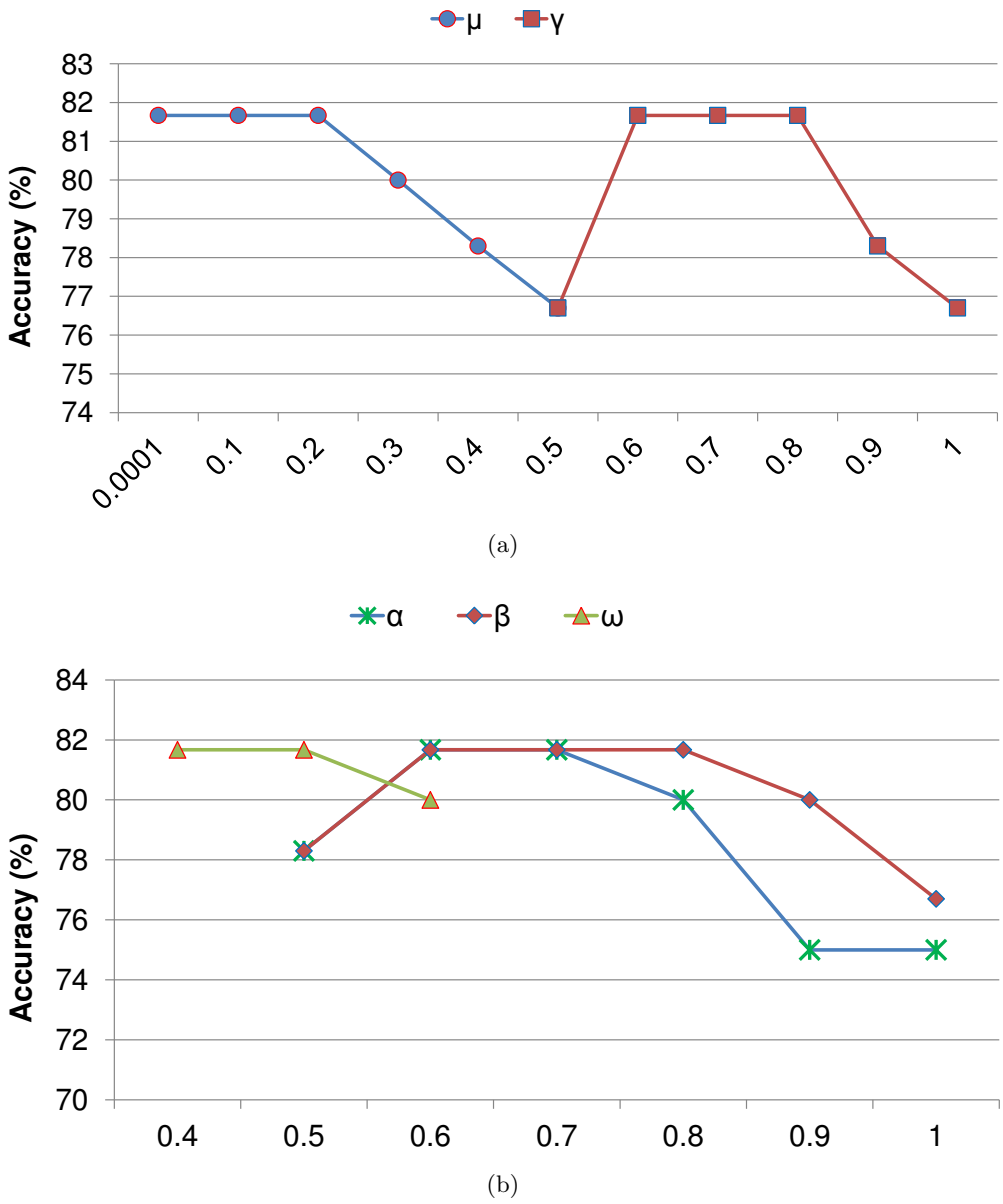


Figure 4.9: Sensitivity analysis for seasonal North Atlantic TC prediction.

4.5.3 Comparison to the Modularity-based Community Detection

Since there is no existing algorithm that is specifically designed for solving our problem (see Problem 2), here we only compare the community detection module in our algorithm with the modularity-based approach [34]. Both algorithms are applied on the SLP network of the year 1950. The known pressure dipoles shown in paper [86] were used as a validation set.

Dipoles are one class of teleconnection phenomena that are characterized by recurring patterns of climate anomalies related to each other at long distances. Such teleconnections are important for understanding and interpreting climate variabilities.

Table 4.3 shows the dipole detection results by the modularity-based method and our (μ, γ)-community generation algorithm. Only if the opposite polarities of a dipole appearing at two different locations were both detected in a single community, the dipole was marked as "found" by the algorithm. Among the five known dipoles, only AO dipole was found by the modularity-based method, while all five dipoles were found by our algorithm. Thus, although modularity-based method might work better for some application domains like social networks, we may lose important teleconnection information by using the modularity-based community definition. Also, our algorithm detected many overlapping communities, which are not shown in the table, while the modularity-based method could only generate the non-overlapping communities. As mentioned earlier, climate communities (or biological functional modules) often work in a cross–talking manner. Ignoring the correlation and interaction between communities is not a good modeling for some complex systems like climatological ocean-atmosphere system.

Another advantage of our (μ, γ)-community generation algorithm is that a set of query nodes can be directly incorporated into the community search to improve the complexity and the quality of the results. For example, a climatologist might wish to search an El Niño or La Niña climate network for those communities associated with El Niño or La Niña events using some of his/her known climate indices as "prior knowledge."

Table 4.3: Dipole detection results

Dipole	Modularity	Our method
North Atlantic Oscillation (NAO)		Found
Southern Oscillation Index (SOI)		Found
Pacific/North American Index (PNA)		Found
Arctic Oscillation (AO)	Found	Found
Western Pacific (WP)		Found

4.6 Conclusions

In this chapter, we introduced the important and challenging problem of detecting predictive and phase-biased communities in contrasting groups of networks. We presented an efficient and effective method that partitions physical system networks into different groups according to

the system's phases, discovers phase-related system components, and uses these components as seeds to identify the phase-biased communities across different groups. Our method successfully identified climate indices associated with hurricane activities and found teleconnection patterns related to rainfall in the Africa Sahel region. Our method also improved the predictive skill of the system's state by 8-16% relative to state-of-the-art approaches and other ensemble methods, such as bagging, boosting, and random forest.

Chapter 5

Conclusion and Future Work

This thesis has proposed four computational approaches to address several novel yet challenging problems in mining informative and predictive patterns within complex data of dynamic systems. In Chapter 2, an iterative, classification-based algorithm, called SPICE, has been developed for enumerating statistically significant and application-relevant component interplays that are key contributors to the system's state. SPICE is inspired by the modularity principle of complex systems, and utilizes information–theoretic selection process and knock–out strategy to ensure the predictability and diversity of the members in the classifier ensemble. SPICE successfully solves the *highly underdetermined* problem in high–dimensional instance–based data. In Chapter 3, a novel method based on graph representatives and community representatives has been proposed for detecting and tracking six types of community dynamics in evolutionary networks. We have presented empirical and theoretical results demonstrating the efficiency and applicability of community dynamic detection algorithm. Additionally, in Chapter 4, we have proposed and implemented an algorithm to detect predictive and phase–biased communities in contrasting groups of networks. We use the phase–related components detected by SPICE as "seeds" to generate the communities and thus reducing the computational cost of the algorithm. We empirically show that the detected anomalous community patterns can be used to improve the predictive skill of the dynamic system's state. To continue this work, we would like to point out a few possible directions of research.

Although SPICE is able to address the underdetermined and non-linear relationship problems, SPICE relies on the decision tree algorithm to select the candidate set of discriminatory features from the available feature space. Thus, SPICE inherits limitations of decision tree algorithms. One of major disadvantages of the decision tree algorithms is its inadequacy to apply regression and predict continuous values. Therefore, SPICE is limited to the classification tasks. One interesting idea for future work is to extend SPICE to regression tasks. We could

first use SPICE to detect the particular phase the system is in, and then based on that specific state, build an ensemble of regression models tuned for this state to predict the magnitude of the system's response. The classifier-regressor ensemble would be one of the possible ways to solve the non-linear dynamic system response prediction problem.

The community dynamic detection algorithm is an efficient and parameter free algorithm for detecting changing communities in time-evolving networks. However, our method considers only the dynamic communities between two consecutive networks, while it ignores the dynamic communities between two networks within a time-window larger than one. Therefore, a time window-based algorithm could be considered in the future work.

Further, the definition of community structure is not limited to maximal cliques used in our work. Although modeling a community as a maximal clique has some advantages in domains like biology, other community definitions such as quasi-clique or modularity might work better for some other application domains like social networks. Thus, another idea for future work is to extend this algorithm by using other community definitions.

In contrast to the community dynamic detection algorithm, the anomalous community detection algorithm for contrasting groups of networks requires several parameters including the enrichment (μ) and the density (γ) for defining the communities, and parameters ω, α, and β for defining the anomalous communities. There are some disadvantages of the non–parameter–free algorithm, such as sensitivity of the parameters. One important area for future work is to design a parameter-free algorithm.

REFERENCES

[1] K. Akhtar and P. Jones. Engineering of a synthetic *ydF-hydE-hydG-hydA* operon for biohydrogen production. *Anal Biochem*, 373:170–172, 2008.

[2] D. Antoni, V. Zverlov, and W. Schwarz. Biofuels from microbes. *Appl Microbiol Biotechnol.*, 77:23–35, 2007.

[3] C. Ash. From simplicity to complexity. *Science*, 329:1125, September 2010.

[4] W. R. Atchley, K. R. Wollenberg, W. M. Fitch, W Terhalle, and A. W. Dress. Correlations among amino acid sites in bHLH protein domains: An information theoretic analysis. *Molecular Biology and Evolution*, 17(1):164–178, January 2000.

[5] K. Bagramyan and A. Trchounian. Structural and functional features of formate hydrogen lyase, an enzyme of mixed-acid fermentation from *escherichia coli*. *Biochemistry*, 68:1159–1170, 2003.

[6] D. Baird and R. E. Ulanowicz. The seasonal dynamics of the chesapeake bay ecosystem. *Ecological Monographs*, 59:329–364, 1989.

[7] B. Balasundaram, S. Butenko, and I. V. Hicks. Clique relaxations in social network analysis: The maximum k-plex problem. *Operations Research*, 59(1):133–142, 2011.

[8] R. E. Bellman. *Adaptive control processes: A guided tour*. Princeton University Press, 1961.

[9] K. Black, R. Parsons, and B. Osborne. Uptake and metabolism of glucose in the nostoc-gunnera symbiosis. *New Phytol.*, 153:297–305, 2002.

[10] A. Blocker, K. Komoriya, and S. Aizawa. Type III secretion systems and bacterial flagella: Insights into their function from structural similarities. *Proceedings of the National Academy of Sciences of the United States of America*, 100(6):3027–3030, March 2003.

[11] M. Blokesch, S. P. J. Albracht, B. F. Matzanke, N. M. Drapal, A. Jacobi, and A. Bock. The complex between hydrogenase-maturation proteins hypc and hypd is an intermediate in the supply of cyanide to the active site iron of [nife]-hydrogenases. *J Mol Biol.*, 344(1):155–167, 2004.

[12] C. Borgelt and M. R. Berthold. Mining molecular fragments: Finding relevant substructures of molecules. In *Proceedings of the 2002 IEEE International Conference on Data Mining*, page 51. IEEE Computer Society, 2002.

[13] E. I. Boyle, S. Weng, J. Gollub, H. Jin, D. Botstein, J. M. Cherry, and G. Sherlock. Go: Termfinder-open source software for accessing gene ontology information and finding significantly enriched gene ontology terms associated with a list of genes. *Bioinformatics*, 20(18):3710–3715, December 2004.

[14] L. Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.

[15] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[16] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and regression trees*. Wadsworth and Brooks, Monterey, CA, 1984.

[17] R. Breitling, P. Armengaud, A. Amtmann, and P. Herzyk. Rank products: a simple, yet powerful, new method to detect differentially regulated genes in replicated microarray experiments. *FEBS Letters*, 573(1-3):83–92, 2004.

[18] G. Butland, J. W. Zhang, W. Yang, A. Sheung, P. Wong, J. F. Greenbalt, A. Emili, and D. B. Zamble. Interactions of the *escherichia coli* hydrogenase biosynthetic proteins: Hybg complex formation. *FEBS Letters*, 580:677–681, 2006.

[19] A. J. Butte, P. Tamayo, D. Slonim, T. R. Golub, and I. S. Kohane. Discovering functional relationships between RNA expression and chemotherapeutic susceptibility using relevance networks. *Proc. Natl. Acad. Sci.*, 97(22):12182–12186, October 2000.

[20] J. Camacho, R. Guimer, and L. A. N. Amaral. Robust patterns in food web structure. *Physical Review Letters*, 88(22):228102, 2002.

[21] D. Chakrabarti. AutoPart: Parameter-free graph partitioning and outlier detection. In *PKDD*, pages 112–124, 2004.

[22] P. K. Chan and M. V. Mahoney. Modeling multiple time series for anomaly detection. In *ICDM*, pages 90–97, 2005.

[23] DeVries A. L. Chen, L. and C. H. Cheng. Convergent evolution of antifreeze glycoproteins in antarctic notothenioid fish and arctic cod. *Proc Natl Acad Sci USA*, pages 3817–3822, 1997.

[24] L. Chen, J. Xuan, R. Riggins, R. Clarke, and Y. Wang. Identifying cancer biomarkers by network-constrained support vector machines. *BMC Syst Biol*, 5(1):161, 2011.

[25] W. Chen, A. Rocha, W. Hendrix, M. Schmidt, and N. F. Samatova. The multiple alignment algorithm for metabolic pathways without abstraction. In *Proceedings of IEEE International Conference on Data Mining Workshops*, pages 669–678.

[26] W. Chen, M. Schmidt, W. Tian, and N. F. Samatova. A fast, accurate algorithm for identifying functional modules through pairwise local alignment of protein interaction networks. In *Proceedings of the International Conference on Bioinformatics & Computational Biology*, pages 816–821, 2009.

[27] Z. Chen, W. Hendrix, and N. F. Samatova. Community-based anomaly detection in evolutionary networks. *Journal of Intelligent Information Systems*, vol.39(1), pages 59–85, 2012.

[28] Z. Chen, K. A. Wilson, Y. Jin, W. Hendrix, and N. F. Samatova. Detecting and tracking community dynamics in evolutionary networks. In *ICDM Workshops*, pages 318–327, 2010.

[29] H. Cheng, P. Tan, C. Potter, and S. Klooster. A robust graph-based algorithm for detection and characterization of anomalies in noisy multivariate time series. In *ICDM Workshops*, pages 349 – 358, 2008.

[30] S. Cho and J. Ryu. Classifying gene expression data of cancer using classifier ensemble with mutually exclusive features. *Proceedings of the IEEE*, 90(11):1744–1753, 2002.

[31] P. Chu, X. Zhao, C. Lee, and M. Lu. Climate prediction of tropical cyclone activity in the vicinity of Taiwan using the multivariate least absolute deviation regression method. *Terr. Atmos. Ocean. Sci.*, 18:805–825, 2007.

[32] H. Chuang, Y. Lee, E.and Liu, D. Lee, and T. Ideker. Network-based classification of breast cancer metastasis. *Mol. Syst. Biol.*, 3:140, 2007.

[33] J. D. Clark and P. S. Chu. Interannual variation of tropical cyclone activity over the Central North Pacific. *JMSJ*, 80(3):403–418, 2002.

[34] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review E*, pages 1– 6, 2004.

[35] R. Curtis, M. Oresic, and A. Vidal-Puig. Pathways to the analysis of microarray data. *Trends Biotechnol.*, 23:429–435, 2005.

[36] M. Czajkowski and M. Krętowski. Top scoring pair decision tree for gene expression data analysis. *Advances in experimental medicine and biology*, 696:27–35, 2011.

[37] Y. Zhan D. Chakrabarti and C. Faloutsos. R-mat: A recursive model for graph mining. In *SDM*, 2004.

[38] O. Dagliyan, F. Uney-Yuksektepe, I. H. Kavakli, and M. Turkay. Optimization based tumor classification from microarray gene expression data. *PLoS ONE*, 6(2):e14579, 02 2011.

[39] D. Das and N. Veziroglu. Hydrogen production by biological processes: A survey of literature. *Int J Hydrogen Energy.*, 26:13 – 28, 2001.

[40] S. Lin and H. Chalupsky. Unsupervised link discovery in multi-relational data via rarity analysis. In *ICDM*, pages 171–178, 2003.

[41] M. Dettling. BagBoosting for tumor classification with gene expression data. *Bioinformatics*, 20(18):3583+, 2004.

[42] R. Diaz-Uriarte and S. A. Andres. Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, 7:3, 2006.

[43] R. Díaz-Uriarte. Variable selection from random forests: application to gene expression data. In *Spanish Bioinformatics Conference*, pages 47–52, 2004.

[44] J. F. Donges, Y. Zou, N. Marwan, and J. Kurths. Complex networks in climate dynamics. *The European Physical Journal - Special Topics*, 174(1):157–179, July 2009.

[45] J. F. Donges, Y. Zou, N. Marwan, and J. Kurths. The backbone of the climate network. *EPL (Europhysics Letters)*, 87(4):48007+, February 2010.

[46] W. Eberle and L. Holder. Detecting anomalies in cargo shipments using graph properties. In *Proceedings of the IEEE Intelligence and Security Informatics Conference*, 2006.

[47] W. Eberle and L. Holder. Discovering structural anomalies in graph-based data. In *IEEE ICDM*, pages 393 – 398, 2007.

[48] B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, New York, 1993.

[49] J. B. Elsner. Tracking hurricanes. *AMS*, 84:353–356, 2001.

[50] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In *IJCAI*, pages 1022–1027, 1993.

[51] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *International Conference on Machine Learning*, pages 148–156, 1996.

[52] A. R. Ganguly, K. Steinhaeuser, D. J. Erickson, M. Branstetter, E. S. Parish, N. Singh, J. B. Drake, and L. Buja. Higher trends but larger uncertainty and geographic variability in 21st century temperature and heat waves. *Proceedings of the National Academy of Sciences*, 106(37):15555–15559, September 2009.

[53] J. W. Gibbs. On the equilibrium of heterogeneous substances. *Transactions of the Connecticut Academy of Arts and Sciences*, 3:108–248, 343–534, 1874-1878.

[54] R. Gill, S. Datta, and S. Datta. A statistical framework for differential network analysis from microarray data. *BMC bioinformatics*, 11(1):95+, February 2010.

[55] M. Girvan and M. E. Newman. Community structure in social and biological networks. *Natl Acad Sci U S A*, 99:7821–7826, 2002.

[56] U. Göbel, C. Sander, R. Schneider, and A. Valencia. Correlated mutations and residue contacts in proteins. *Proteins*, 18(4):309–317, April 1994.

[57] S. B. Goldenberg and L. J. Shapiro. Physical mechanisms for the association of El Niño and West African rainfall with Atlantic major hurricane activity. *Journal of Climate*, 9(6):1169–1187, June 1996.

[58] A. Gozolchiani, K. Yamasaki, O. Gazit, and S. Havlin. Pattern of climate network blinking links follows el niño events. *EPL*, 83:28005, 2008.

[59] W. M. Gray, C. W. Landsea, P. W. Mielke, Jr., and K. J. Berry. Predicting Atlantic basin seasonal tropical cyclone activity by 1 august. *Weather Forecast*, 8:73–86, March 1993.

[60] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vectormachines. *Machine Learning*, 46:389–422, 2002.

[61] P. Hallenbeck and D. Ghosh. Improvements in fermentative biological hydrogen production through metabolic engineering. *J Environ Manage.*, pages 1–5, 2010.

[62] D. Hart. *Hydrogen power: The commercial future of 'the ultimate fuel'*. Financial Times Energy Publishing, 1997.

[63] L. H. Hartwell, J. J. Hopfield, S. Leibler, and A. W. Murray. From molecular to modular cell biology. *Nature*, 402(6761):47–52, 1999.

[64] V. Hautamäki, I. Kärkkäinen, and P. Fränti. Outlier detection using k-nearest neighbour graph. In *ICPR (3)*, pages 430–433, 2004.

[65] F. R. Hawkes, R. Dinsdale, D. L. Hawkes, and I. Hussy. Sustainable fermentative hydrogen production: Challenges for process optimisation. *International Journal of Hydrogen Energy*, 27:1339–1347, 2002.

[66] X. He and P. Niyogi. Locality preserving projections. Cambridge, MA, 2004. MIT Press.

[67] X. He, S. Yan, Y. Hu, P. Niyogi, and H. J. Zhang. Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:328–340, 2005.

[68] B. Heisele, T. Serre, S. Mukherjee, and T. Poggio. Feature reduction and hierarchy of classifiers for fast object detection in video images. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 2:18, 2001.

[69] B. Hendrickson and R. Leland. An improved spectral graph partitioning algorithm for mapping parallel computations. *SIAM J. Sci. Comput.*, 16:452–469, March 1995.

[70] W. Hendrix, A. M. Rocha, K. Padmanabhan, A. Choudhary, K. Scott, J. R. Mihelcic and N. F. Samatova. DENSE: efficient and prior knowledge-driven discovery of phenotype-associated protein functional modules. *BMC Systems Biology*, 5(1):172+, 2011.

[71] T. Hey, S. Tansley, and K. Tolle. *The fourth paradigm: data-intensive scientific discovery.* Microsoft Research, Redmond, WA, 2009.

[72] N. J. Higham. A survey of componentwise perturbation theory in numerical linear algebra. In *in Mathematics of Computation 1943–1993: A Half Century of Computational Mathematics*, pages 49–77, 1994.

[73] S. A. Hofmeyr, S. Forrest, and A. Somayaji. Intrusion detection using sequences of system calls. *Journal of Computer Security*, 6:151–180, 1998.

[74] J. Hopcroft, O. Khan, B. Kulis, and B. Selman. Tracking evolving communities in large linked networks. *PNAS*, 101:5249–5253, 2004.

[75] U. Hrtel and W. Buckel. Sodium ion-dependent hydrogen production in *acidaminococcus fermentans*. *Archives of Microbiology*, 166:350–356, 1996.

[76] Y. Huang, W. Zong, X. Yan, R Wang, C. Hemme, J. Zhou, and Z. Zhou. Succession of the bacterial community and dynamics of hydrogen producers in a hydrogen-producing bioreactor. *Appl Environ Microbiol.*, 76:3387–3390, 2010.

[77] T. Hwang, C. Sun, T. Yun, and G. Yi. Figs: a filter-based gene selection workbench for microarray data. *BMC Bioinformatics*, 11:50, 2010.

[78] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *STOC*, pages 604–613, 1998.

[79] P. Jafari and F. Azuaje. An assessment of recently published gene expression data analyses: reporting experimental design and statistical factors. *BMC Medical Informatics and Decision Making*, 6(1):27, 2006.

[80] L. J. Jensen, M. Kuhn, M. Stark, S. Chaffron, C. Creevey, J. Muller, T. Doerks, P. Julien, A. Roth, M. Simonovic, P. Bork, and C. von Mering. STRING 8–a global view on proteins and their functional interactions in 630 organisms. *Nucleic Acids Research*, 37(Database):D412–D416, 2009.

[81] K. Jim, K. Parmar, M. Singh, and S. Tavazoie. A Cross-Genomic approach for systematic mapping of phenotypic traits to genes. *Genome Research*, 14(1):109–115, 2004.

[82] M. Johannes, J. Brase, H. Fröhlich, S. Gade, M. Gehrmann, M. Fälth, H. Sültmann, and T. Beißbarth. Integration of pathway knowledge into a reweighted recursive feature elimination approach for risk stratification of cancer patients. *Bioinformatics*, 26:2136–2144, 2010.

[83] I. T. Jolliffe and D. B. Stephenson. *Forecast verification: a practitioner's guide in atmospheric science*. Wiley and Sons, 2003.

[84] M. Kalaev, V. Bafna, and R. Sharan. Fast and accurate alignment of multiple protein networks. In *RECOMB 2008*, pages 246–256, 2008.

[85] I. Kapdan and F. Kargi. Bio-hydrogen production from waste materials. *Enzyme Microb Technol.*, 38:569–582, 2006.

[86] J. Kawale, S. Chatterjee, A. Kumar, S.n Liess, M. Steinbach, and V. Kumar. Anomaly construction in climate data: Issues and challenges. In *CIDU*, pages 189–203, 2011.

[87] E. J. Keogh, J. Lin, and A. W. Fu. Hot sax: Efficiently finding the most unusual time series subsequence. In *ICDM*, pages 226–233, 2005.

[88] S. Khanal. Biohydrogen production: Fundamentals, challenges, and operation strategies for enhanced yield. In S. Khanals, editor, *Anaerobic biotechnology for bioenergy production: Principles and applications*, volume 161-187. Wiley-Blackwell, 2008.

[89] H. M. Kim and P. J. Webster. Extended-range seasonal hurricane forecasts for the North Atlantic with a hybrid dynamical-statistical model. *Geophys. Res. Lett.*, 37(21):L21705, 2010.

[90] H. S. Kim, C. H. Ho, P. S. Chu, and J. H. Kim. Seasonal prediction of summertime tropical cyclone activity over the East China Sea using the least absolute deviation regression and the Poisson regression. *Int. J. Climato.*, 30(2):210–219, 2010.

[91] P. Koskinen, A. Kaksonen, and J. Puhakka. The relationship between instability of H2 production and compositions of bacterial communities within a dark fermentation fluidised-bed bioreactor. *Biotechnol Bioeng*, 97:742–758, 2007.

[92] M. Koyutürk, Y. Kim, S. Subramaniam, W. Szpankowski, and A. Grama. Detecting conserved interaction patterns in biological networks. *J Comput Biol.*, 13(7):1299–1322, 2006.

[93] Y. Lei and L. Huan. Efficient feature selection via analysis of relevance and redundancy. *J Mach Learn Res.*, 5:1205–1224, 2004.

[94] M. Levesque, D. Shasha, W. Kim, M. Surette, and P. Benfey. Trait-to-Gene: A computational method for predicting the function of uncharacterized genes. *Curr Biol.*, 13:129–133, 2003.

[95] C. Li and H. P. Fang. Fermentative hydrogen production from wastewater and solid wastes by mixed cultures. *Crit. Rev. Environ. Sci. Technol.*, 37(1):1–39, 2007.

[96] L. Li, C. Weinberg, T. Darden, and L. Pedersen. Gene selection for sample classification based on gene expression data: Study of sensitivity to choice of parameters of the GA/KNN method. *Bioinformatics*, 17:1131–1142, 2001.

[97] R. Li and H. Fang. Heterotrophic photo fermentative hydrogen production. *Critical Reviews in Environmental Science and Technology*, 39(12):1081–1108, 2009.

[98] X. Liu, Y. Zhu, and S. Yang. Construction and characterization of ack deleted mutant of *clostridium tyrobutyricum* for enhanced butyric acid and hydrogen production. *Biotechnol Prog*, 22:1265–75, 2006.

[99] M. Long, E. Betran, K. Thornton, and W. Wang. The origin of new genes: glimpses from the young and old. *Nature Reviews Genetics*, 4(11):865–875, November 2003.

[100] S. Ma, M. Shi, Y. Li, D. Yi, and B. Shia. Incorporating gene co-expression network in identification of cancer prognosis markers. *BMC Bioinformatics*, 11:271, 2010.

[101] K. Madduri and D.A. Bader. Gtgraph: A synthetic graph generator suite. *Georgia Institute of Technology College of Computing*, (3):2–5, 2006.

[102] A. Martins and S. Shuman. An end-healing enzyme from *clostridium thermocellum* with 5' kinase, 2',3' phosphatase, and adenylyltransferase activities. *RNA*, 11:1271–80, 2005.

[103] J. Mathews and G. Wang. Metabolic pathway engineering for enhanced biohydrogen production. *INT J HYDROGEN ENERG*, 34:7404–7416, 2009.

[104] J. B. McKinlay and C. S. Harwood. *Clostridium ljungdahlii* represents a microbial production platform based on syngas. *Proc Natl Acad Sci.*, 107(26):11669–75, 2010.

[105] P. Melville and R. J. Mooney. Diverse ensembles for active learning. In *In Proceedings of 21st International Conference on Machine Learning (ICML-2004*, pages 584–591. ACM Press, 2004.

[106] H. D. K. Moonesinghe and P. Tan. Outlier detection using random walks. In *ICTAI*, pages 532 – 539, 2006.

[107] K. Nath and D. Das. Improvement of fermentative hydrogen production: Various approaches. *Appl Microbiol Biotechnol.*, 65(5):520–9, 2004.

[108] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[109] S. Nijssen and J. Kok. The gaston tool for frequent subgraph mining. *Electron. Notes Theor. Comput. Sci.*, 127:77–87, 2005.

[110] C. C. Noble and D. J. Cook. Graph-based anomaly detection. In *KDD '03*, pages 631–636, New York, NY, USA, 2003. ACM.

[111] K. Padmanabh, A. M. R. Vanteddu, S. Sen, and P. Gupta. Random walk on random graph based outlier detection in wireless sensor networks. In *Wireless Communication and Sensor Networks, 2007. WCSN '07. Third International Conference on*, pages 45–49, Dec. 2007.

[112] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, June 2005.

[113] G. Palla, A. Barabasi, and T. Vicsek. Quantifying social group evolution. *Nature*, 446:2007, 2007.

[114] T. Pansombut, W. Hendrix, Z. J. Gao, B. E. Harrison, and N. F. Samatova. Biclustering-driven ensemble of bayesian belief network classifiers for underdetermined problems. In *IJCAI*, pages 1439–1445, 2011.

[115] A. Paschos, A. Bauer, A. Zimmermann, E. Zehelein, and A. Böck. *HypF*, a carbamoyl phosphate-converting enzyme involved in [*nife*] hydrogenase maturation. *J Biol Chem*, 277:49945–51, 2002.

[116] F. Pazos, M. Helmer-Citterich, G. Ausiello, and A. Valencia. Correlated mutations contain information about protein-protein interaction. *J Mol Biol*, 271(4):511–523, August 1997.

[117] J. Pei, D. Jiang, and A. Zhang. Mining cross-graph quasi-cliques in gene expression and protein interaction data. In *Proceedings, 21st International Conference on Data Engineering (ICDE 2005)*, pages 353–356, April 2005.

[118] J. Pei, D. Jiang, and A. Zhang. On mining cross-graph quasi-cliques. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, pages 228–238, New York, NY, USA, 2005. ACM.

[119] J. D. Peterson, L. A. Umayam, T. Dickinson, E. K. Hickey, and O. White. The comprehensive microbial resource. *Nucleic Acids Research*, 29(1):123–125, 2001.

[120] S. V. Rajagopala, B. Titz, J. Goll, J. R. Parrish, K. Wohlbold, M. T. McKevitt, T. Palzkill, H. Mori, R. L. Finley, and P. Uetz. The protein network of bacterial motility. *Molecular Systems Biology*, 3, July 2007.

[121] F. Rapaport, A. Zinovyev, M. Dutreix, E. Barillot, and J. Vert. Classification of microarray data using gene networks. *BMC bioinformatics*, 8:35, 2007.

[122] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A. L. Barabási. Hierarchical organization of modularity in metabolic networks. *Science*, 297(5586):1551–1555, 2002.

[123] R. Albert and A. Barabási. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74:47–97, June 2002.

[124] F. Rey, Y. Oda, and C. Harwood. Regulation of uptake hydrogenase and effects of hydrogen utilization on gene expression in *rhodopseudomonas palustris*. *J Bacteriol.*, 188(17):6143–6152, 2006.

[125] F. E. Rey, E. K. Heiniger, and C. S. Harwood. Redirection of metabolism for biological hydrogen production. *Appl Environ Microbiol.*, 73(5):1665–1671, 2007.

[126] Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics (Oxford, England)*, 23(19):2507–2517, 2007.

[127] M. A. Saunders and A. R. Harris. Statistical evidence links exceptional 1995 Atlantic hurricane season to record sea warming. *JGRL*, 24:1255–1258, 1997.

[128] M. C. Schmidt and N. F. Samatova. An algorithm for the discovery of phenotype related metabolic pathways. In *BIBM*, pages 60–65, 2009.

[129] M. C. Schmidt, N. F. Samatova, K. Thomas, and B. Park. A scalable, parallel algorithm for maximal clique enumeration. *J. Parallel Distrib. Comput.*, 69(4):417–428, 2009.

[130] S. B. Seidman and B. L. Foster. A graph-theoretic generalization of the clique concept. *Journal of Mathematical Sociology*, 6:139–154, 1978.

[131] R. Sharan, T. Ideker, B. Kelley, R. Shamir, R. M. Karp 2004, and R. M. Karp. Identification of protein complexes by comparative analysis of yeast and bacterial protein interaction data. *Journal of computational biology*, 12(6):835–846, July 2005.

[132] J. Shetty and J. Adibi. Discovering important nodes through graph entropy the case of enron email database. In *LinkKDD '05: Proceedings of the 3rd international workshop on Link discovery*, pages 74–81, New York, NY, USA, 2005. ACM.

[133] Y. Shomura, H. Komori, N. Miyabe, M. Tomiyama, N. Shibata, and Y. Higuchi. Crystal structures of hydrogenase maturation protein *hype* in the apo and atp-bound forms. *J Mol Biol.*, 372:1045–1054, 2007.

[134] A. H. Singh, D. M. Wolf, P. Wang, and A. P. Arkin. Modularity of stress response evolution. *Proceedings of the National Academy of Sciences*, 105(21):7500–7505, May 2008.

[135] J. P. Kossin S. J. Camargo and M. Sitkowski. Climate modulation of North Atlantic hurricane tracks. *Journal of Climate*, 23:3057–3076, 2010.

[136] N. Slonim, O. Elemento, and S. Tavazoie. Ab initio genotype-phenotype association reveals intrinsic modularity in genetic networks. *Mol. Syst. Biol.*, 2, 2006.

[137] B. Snel, P. Bork, and M. Huynen. Genome evolution. Gene fusion versus gene fission. *Trends Genet*, 16(1):9–11, 2000.

[138] V. Spirin and L. A. Mirny. Protein complexes and functional modules in molecular networks. *Proceedings of the National Academy of Sciences of the United States of America*, 100(21):12123 –12128, October 2003.

[139] S. Staniford-chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagl, K. Levitt, C. Wee, R. Yip, and D. Zerkle. Grids-a graph based intrusion detection system for large networks. In *In Proceedings of the 19th National Information Systems Security Conference*, pages 361–370, 1996.

[140] C. Steffes, J. Ellis, J. Wu, and B. Rosen. The *lysp* gene encodes the lysine-specific permease. *J. Bacteriol.*, 174:3242–3249, 1992.

[141] K. Steinhaeuser, N. V. Chawla, and A. R. Ganguly. An exploration of climate data using complex networks. In *SensorKDD*, pages 23–31, 2009.

[142] K. Steinhaeuser, N. V. Chawla, and A. R. Ganguly. Complex networks as a unified framework for descriptive analysis and predictive modeling in climate science. *Statistical Analysis and Data Mining*, 4(5):497–511, 2011.

[143] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu. GraphScope: parameter-free mining of large time-evolving graphs. In *KDD*, pages 687–696, 2007.

[144] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos. Neighborhood formation and anomaly detection in bipartite graphs. In *The Fifth IEEE ICDM*, pages 418–425, 2005.

[145] J. Sun, D. Tao, and C. Faloutsos. Beyond streams and graphs: dynamic tensor analysis. In *KDD '06*, pages 374–383, 2006.

[146] R. T. Sutton, S. P. Jewson, and D. P. Rowell. The elements of climate variability in the tropical atlantic region. *J. Climate*, 13:3261– 3284, 2000.

[147] N. Tajunisha, and V. Saravanan. An improved method of unsupervised sample clustering based on information genes for microarray cancer data sets. *IJCB*, 2(1):24–31, 2011.

[148] A. C. Tan and D. Gilbert. Ensemble machine learning on gene expression data for cancer classification. *Applied bioinformatics*, 2(3 Suppl), 2003.

[149] C. Tantipathananandh, T. B. Wolf, and D. Kempe. A framework for community identification in dynamic social networks. In *KDD '07*, pages 717–726. ACM, 2007.

[150] D. Tao, X. Tang, X. Li, and X. Wu. Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval. *IEEE T. Pattern Anal.*, 28(7):1088–1099, July 2006.

[151] R. L. Tatusov, N. D. Fedorova, J. D. Jackson, A. R. Jacobs, B. Kiryutin, E. V. Koonin, D. M. Krylov, R. Mazumder, S. L. Mekhedov, A. N. Nikolskaya, B. S. Rao, S. Smirnov, A. V. Sverdlov, S. Vasudevan, Y. I. Wolf, J. J. Yin, and D. A. Natale. The COG database: an updated version includes eukaryotes. *BMC bioinformatics*, 4(1):41+, September 2003.

[152] W. R. Taylor and K. Hatrick. Compensating changes in protein multiple sequence alignments. *Protein Eng.*, 7:341–348, 1994.

[153] C. J. Melick, A. R. Lupo, T. K. Latham, T. H. Magill, J. V. Christopher, and P. S. Market. The interannual variability of hurricane activity in the art. *National Weather Digest*, 32:1–15, 2008.

[154] J. G. Thomas, J. M. Olson, S. J. Tapscott, and L. P. Zhao. An efficient and robust statistical modeling approach to discover differentially expressed genes using genomic expression profiles. *Genome Research*, 11(7):1227–1236, 2001.

[155] A. A. Tsonis and P. Roebber. The architecture of the climate network. *Physica A*, 333:497–504, February 2004.

[156] A. A. Tsonis, K. Swanson, and S. Kravtsov. A new dynamical mechanism for major climate shifts. *GRL*, 34:L13705+, 2007.

[157] A. A. Tsonis and K. L. Swanson. Topology and predictability of el niño and la niña networks. *Physical Review Letters*, 100(22), 2008.

[158] A. A. Tsonis, K. L. Swanson, and P. J. Roebber. What do networks have to do with climate? *BAMS*, 87(5):585–595, May 2006.

[159] A. A. Tsonis, K. L. Swanson, and G. Wang. On the role of atmospheric teleconnections in climate. *J. Climate*, 21:2990–3001, 2008.

[160] A. Tsonis, G. Wang, K. Swanson, F. Rodrigues, and L. Costa. Community structure and dynamics in climate networks. *Climate Dynamics*, pages 1–8, July 2010.

[161] A. Veit, M. Akhtar, T. Mizutani, and P. Jones. Constructing and testing the thermodynamic limits of synthetic NAD(P)H:H2 pathways. *Microb Biotechnol*, 1(5):382–94, 2008.

[162] P. M. Vignais, B. Billoud, and J. Meyer. Classification and phylogeny of hydrogenases. *FEMS Microbiol Rev*, 25:455–501, 2001.

[163] P. M. Vignais and A. Colbeau. Molecular biology of microbial hydrogenases. *Curr Issues Mol Biol.*, 6:159–188, 2004.

[164] K. Wakita and T. Tsurumi. Finding community structure in mega-scale social networks. *CoRR*, abs/cs/0702048, 2007.

[165] L. Wang, F. Chu, and W. Xie. Accurate cancer classification using expressions of very few genes. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 4:40–53, January 2007.

[166] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, June 1998.

[167] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

[168] D. White. *The physiology and biochemistry of prokaryotes*. Oxford University Press, New York, 2007.

[169] I. H. Witten and E. Frank. Data mining: practical machine learning tools and techniques with Java implementations. *ACM SIGMOD Record*, 31(1):76–77, 2002.

[170] L. Xie, T. Yan, and L. Pietrafesa. The effect of Atlantic sea surface temperature dipole mode on hurricanes: Implications for the 2004 Atlantic hurricane season. *JGRL*, 32:3701+, February 2005.

[171] J. Peng, L. Yang, J. Wang, Z. Liu, and M. Li. An efficient algorithm for detecting closed frequent subgraphs in biological networks. In *BMEI*, pages 677–681, 2008.

[172] A. Yeshanew and M. R. Jury North african climate variability. part 3: Resource prediction. *Theoretical and Applied Climatology*, 89(1-2):51–62, 2007.

[173] M. Yousef, M. Ketany, L. M. Manevitz, L. C. Showe, and M. K. Showe. Classification and biomarker identification using gene network modules and support vector machines. *BMC Bioinformatics*, 10:337, 2009.

[174] Z. Chen, K. Padmanabhan, A. Rocha, Y. Shpanskaya, J. R. Mihelcic, K. Scott, and N. F. Samatova. SPICE: Discovery of phenotype-determining component interplays. *BMC Systems Biology*, 6(1):40, 2012.

[175] Z. Zeng, J. Wang, L. Zhou, and G. Karypis. Coherent closed quasi-clique discovery from large dense graph databases. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '06, pages 797–802, New York, NY, USA, 2006. ACM.

[176] Z. Zeng, J. Wang, L. Zhou, and G. Karypis. Out-of-core coherent closed quasi-clique mining from large dense graph databases. *ACM Trans. Database Syst.*, 32(2):13, 2007.

[177] B. Zhang, B. Park, T. Karpinets, and N. F. Samatova. From pull-down data to protein interaction networks and complexes with biological relevance. *Bioinformatics*, 24:979–986, April 2008.

[178] B. Zhang, H. Li, R. B. Riggins, M. Zhan, J. Xuan, Z. Zhang, E. P. Hoffman, R. Clarke, and Y. Wang. Differential dependency network analysis to identify condition-specific topological changes in biological networks. *Bioinformatics*, 25(4):526–532, February 2009.

[179] B. Zhang, B. Park, T. Karpinets, and N. F. Samatova. From pull-down data to protein interaction networks and complexes with biological relevance. *Bioinformatics (Oxford, England)*, 24(7):979–86, 2008.

[180] J. Zhang. Evolution by gene duplication: an update. *Trends in Ecology & Evolution*, 18(6):292–298, June 2003.

[181] R. Zhang, C. E. Andersson, A. Savchenko, T. Skarina, E. Evdokimova, S. Beasley, C. H. Arrowsmith, A. M. Edwards, A. Joachimiak, and S. L. Mowbray. Structure of *escherichia coli* ribose-5-phosphate isomerase: A ubiquitous enzyme of the pentose phosphate pathway and the calvin cycle. *Structure*, 11(1):31–42, 2003.

[182] Q. Zhou, W. Hong, L. Luo, and F. Yang. Gene selection using random forest and proximity differences criterion on dna microarray data. *JCIT*, 5(6):161–170, 2010.