1

Computational Systems Biology

T. M. Murali
Virginia Polytechnic Institute and State University

Srinivas Aluru
Iowa State University

1.1 Introduction

The functioning of a living cell is governed by an intricate network of interactions among different types of molecules. A collection of long DNA molecules called *chromosomes*, that together constitute the *genome* of the organism, encode for much of the cellular molecular apparatus including various types of RNAs and proteins. Short DNA sequences that are part of chromosomal DNA, called *genes*, can be transcribed repeatedly to result in various types of RNAs. Some of these RNAs act directly, such as micro (miRNA), ribosomal (rRNA), small nuclear (snRNA), and transfer (tRNA) RNAs. Many genes result in messenger RNAs (mRNAs), which are translated to corresponding protein molecules, a diverse and important set of molecules critical for cellular processes. A plethora of small molecules that are outside the hereditarily derived genes-RNAs-proteins system, called *metabolites*, play a crucial role in biological processes as intermediary molecules that are both products and inputs to biochemical enzymatic reactions.

These complex interactions define, regulate, and even initiate and terminate biological processes, and also create the molecules that take part in them. They are pervasive in all aspects of cell function, including transmission of external signals to the interior of the cell, controlling processes that result in protein synthesis, modifying protein activities and their locations in the cell, and driving biochemical reactions. Gene products coordinate to execute cellular processes – sometimes by acting together, such as multiple proteins forming a protein supercomplex (e.g., the ribosome), or by acting in a concerted way to create biochemical pathways and networks (e.g., metabolic pathways that break down food, and photosynthetic pathways that convert sun light to energy in plants). It is the same gene

0-8493-8597-0/01/$0.00+$1.50

products that also regulate the expression of genes, often through binding to cis-regulatory sequences upstream of genes, to calibrate gene expression for different processes and to even decide which pathways are appropriate to trigger based on external stimuli.

The genomic revolution of the past two decades provides the "parts list" for systems biology. Advances in high-throughput experimental techniques are enabling measurements of mRNA, protein, and metabolite levels, and the detection of molecular interactions on a massive scale. In parallel, automated parsing and manual curation have extracted information on molecular interactions that have been deposited in the scientific literature over decades of small-scale experiments. In combination, these efforts have provided us with large-scale publicly-available datasets of molecular interactions and measurements of molecular activity, especially for well-studied model organisms such as *S. cerevisiae* (baker's yeast), *C. elegans* (a nematode), and *D. melanogaster* (the fruitfly), for pathogens such as *P. falciparum* (the microbe that causes malaria), and for *H. sapiens* itself.

These advances are transforming molecular biology from a reductionist, hypothesis-driven experimental field into an increasingly data-driven science, focused on understanding the functioning of the living cell at a systems level. How do the molecules within the cell interact with each other over time and in response to external conditions? What higher-level modules do these interactions form? How have these modules evolved and how do they confer robustness to the cell? How does disease result from the disruption of normal cellular activities? Understanding the complex interactions between these diverse and large body of molecules at various levels, and inferring the complex pathways and intermediaries that govern each biological process, are some of the grand challenges that constitute the field of systems biology. The data deluge has resulted in an ever-increasing importance placed on computational analysis of biological data and computationally-driven experimental design. Research in this area of *computational systems biology* (CSB) spans a continuum of approaches [IL03] that includes simulating systems of differential equations, Boolean networks, Bayesian analysis, and statistical data mining.

Computational systems biology is a young discipline in which the important directions are still in a state of flux and being defined. In this chapter, we focus primarily on introducing and formulating the most well-studied classes of algorithmic problems that arise in the phenomenological and data-driven analysis of large-scale information on the behavior of molecules in the cell. We focus on research where the problem formulations and algorithms developed have actually been applied to biological data sets. Where possible, we refer to theoretical results and tie the work in the CSB literature to research in the algorithms community. The breadth of topics in CSB and the diversity of the connections between CSB and theoretical computer science preclude an exhaustive coverage of topics and literature within the scope of this short chapter. We caution the reader that our treatment of the topics and their depth and citation to relevant literature are by no means exhaustive. Rather, we attempt to provide a self-contained and logically interconnected survey of some of the important problem areas within this discipline, and provide pointers to a reasonable body of literature for further exploration by the reader. By necessity, this chapter introduces a number of biological terms and concepts that a computer scientist may not be familiar with. A glossary at the end of the chapter provides an easy resource for cross-reference.

1.2 An Illustrative Example

To elucidate how a typical biological process may unfold, and help explain some of the models used in systems biology, consider a generic process by which a eukaryotic cell responds to an external signal, e.g., a growth factor. See Figure 1.1 for a specific illustration of such

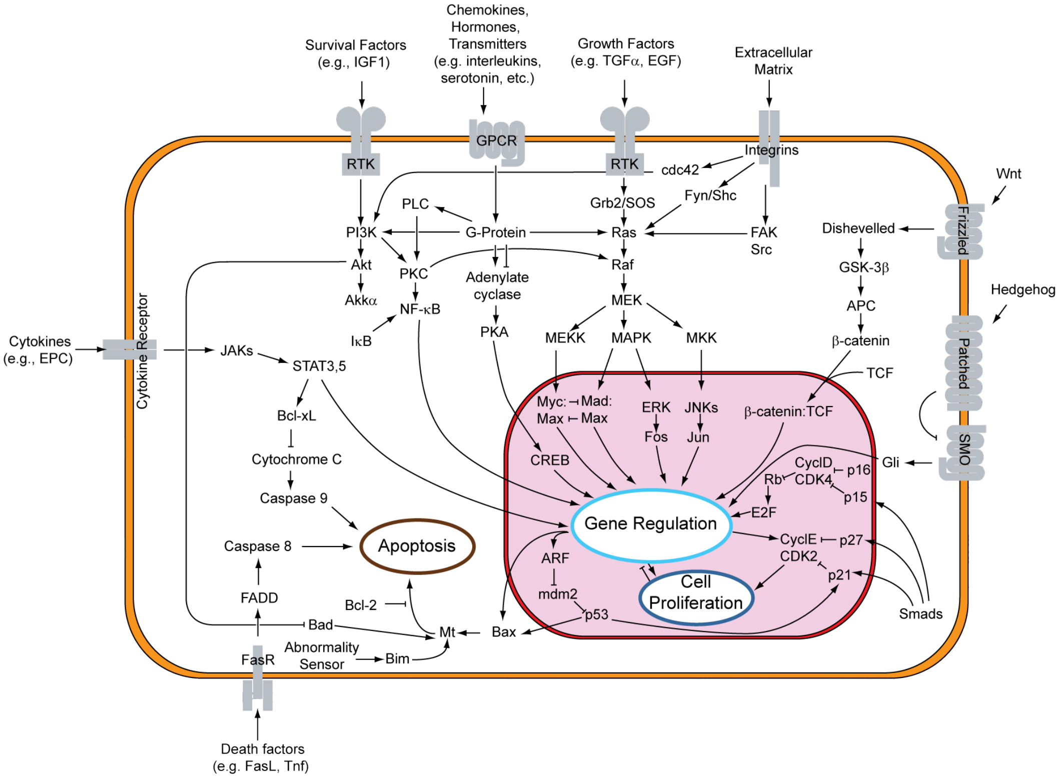

FIGURE 1.1: An illustration of a cellular signaling network (a Wikipedia image released by the author into the public domain).

a process. The growth factor binds to a specific receptor protein on the cell surface. The receptor protein dimerizes (i.e., protein molecules with bound growth factors themselves bind to each other). The dimerized form of the receptor is active; it phosphorylates (adds phosphate groups to) other proteins in the cytoplasm of the cell, which in turn physically interact with or phosphorylate other proteins. Multiple such signaling cascades may be activated. The cascade culminates in the activation of a transcription factor; the transcription factor (TF) moves to the nucleus, where it binds to target sites on genomic DNA that recognize the TF. The TF recruits the cellular apparatus for transcription, resulting in the expression of numerous genes. These genes are converted to proteins in the ribosome of the cell, and then transported to various locations within the cell to perform their activities. Some proteins may be TFs themselves and cause the expression of other genes. Others may catalyze enzymatic reactions that produce or consume metabolites. Synthesized proteins may activate other signaling or reaction cascades. Ultimately, the initial binding of the growth factor with its receptor changes the levels and activities of numerous genes, proteins, metabolites, and other compounds, and modulates global responses such as cell migration, adhesion, and proliferation.

High-throughput experiments shed light on many of these interaction types. Interactions between signaling proteins (e.g., kinases and phosphatases) and their substrates constitute directed protein phosphorylation networks. Undirected protein-protein interaction (PPI) networks represent physical interactions between proteins. Directed transcriptional regulatory networks connect TFs to genes they regulate. Biochemical networks describe metabolic reactions with information on the enzymes that catalyze each reaction. Taken together, the known molecular interactions for an organism constitute its *wiring diagram*, a graph where each node is a molecule and each edge is a directed or undirected interaction between two molecules. As generally conceived, a wiring diagram contains molecules and interactions of many types.

Wiring diagrams usually contain direct interactions between molecules. Sometimes they are augmented with indirect interactions. A prominent example is a genetic interaction: two genes have a *genetic interaction* if the action of one gene is modified by the other. An extreme case of a genetic interaction is a *synthetically lethal* interaction, where knocking out each of two genes does not kill the cell but knocking out both genes results in cell death. For other examples of indirect or "conceptual" interactions, see functional linkage networks in Section 1.3 and reverse-engineered gene networks in Section 1.6.2.

1.3 Gene Function Prediction

The genomes of more than 600 organisms (including more than 70 eukaryotes) have been completely sequenced [LMTK08]. However, a fundamental roadblock to progress in systems biology is the poor state of knowledge about the biological functions of the genes in sequenced genomes [Kar04, RKKe04, Rob04]. Many genes of unknown function might support important cellular functions. Discovering the functions of these genes will provide critical insights into the biology of many organisms. In addition, discovering these functions will improve our ability to annotate genomes that are sequenced in the future.

The phrase "gene function" has a variety of meanings. It often refers to the "molecular function" of the protein the gene codes for, e.g., whether the protein catalyzes a reaction or binds DNA to regulate the expression of a target gene. More generally, the phrase refers to the context in which the protein acts in the cell, e.g., the component of the cell it is localized to; the pathway or biological process it is a member of; the cell type the gene is expressed in (in the case of multi-cellular organisms); or the developmental stage during which the gene is active. In this section, we will restrict our attention to three structured controlled vocabularies (ontologies) developed by the Gene Ontology (GO) Consortium [ABBB00]: molecular function, cellular component, and biological process. Each ontology is a Directed Acyclic Graph (DAG) where each function is connected to parent functions by relationships such as "is_a" or "part_of". By design, a function represents a more specific biological concept than any of its parents. GO annotations follow the *true path rule*: if a gene is annotated with a function, then the gene must be annotated with all parents of that function.

A powerful method for predicting gene function relies on the evolutionary conservation of gene and protein sequences. Thus, if a gene in an organism has a nucleotide sequence, amino-acid sequence, or protein structure very similar to that of a gene with a known function [GJF07], then the function can be transferred to the first gene. These methods are primarily useful for determining the molecular function of a gene, which often depends directly on the structure of the protein encoded by the gene. Further, these methods do not provide annotations for the more than 40% of eukaryotic genes that do not have high sequence or structural similarity to any genes in other organisms [EKO03].

A promising approach to gene function prediction starts by constructing a *functional linkage network* (FLN) connecting genes of interest. In such a network, each node is a gene and each edge connects two genes that may share the same function, based on some experimental or computational evidence. For instance, two genes may be linked if they have similar expression profiles in some experiment; if the proteins they code for inter-

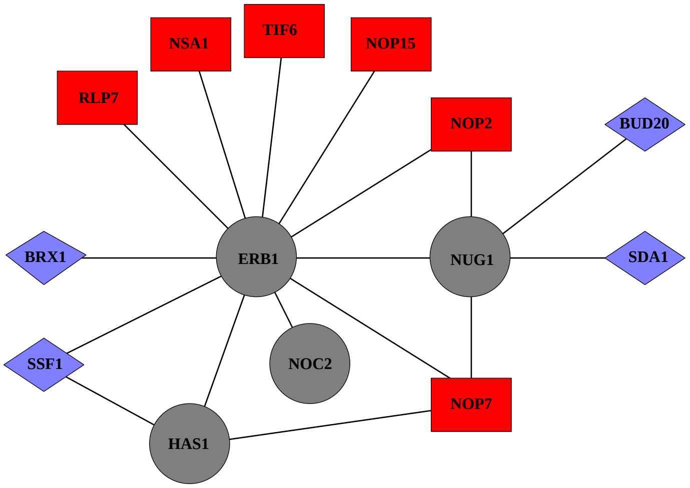

FIGURE 1.2: A subgraph of an FLN in *S. cerevisiae* for the biological process "ribosome biogenesis." Each node is a gene. Each edge corresponds to two genes whose protein products interact. Using the notation in the text, the rectangles are members of V_f^+ (where f is "ribosome biogenesis"), the circles are elements of V_f^0, and the diamonds are in V_f^-. To improve readability, we display only interactions involving genes in V_f^0. (Figure taken from Karaoz *et al.* [KMLe04].)

act physically or catalyze reactions involving the same metabolite; or, if knocking-out or silencing the expression of both genes produces the same phenotype. Constructing biologically meaningful FLNs from functional genomic data is an active area of research. Most existing methods proceed by estimating the probability that a given pair of genes should be functionally linked based on a given type of data and then integrating signals from multiple data sets [MT07, LLC$^+$08]. Many such FLNs are available in publicly-accessible databases [BCS06].

In this section, we focus on the problem of predicting gene functions assuming that an FLN is given as input. We denote the FLN by an undirected graph $G = (V, E)$, where V is the set of nodes and E is the set of edges. For an edge $(u, v) \in E$, let $0 \leq w_{uv} \leq 1$ denote the weight of the edge in E between u and v; we interpret w_{uv} as a measure of confidence that u and v should be annotated with the same function. Note that the edge (u, v) suggests that u and v could perform the same function in the cell but does not specify what that function is. Let f be a function of interest in the Gene Ontology. We cast the problem of predicting which nodes (genes) in G have the function f as a *semi-supervised learning* problem [CSZ06]. We partition V into three subsets V_f^+, V_f^0, and V_f^-, corresponding to positive examples, unknown examples, and negative examples, respectively. A node v is in V_f^+ if v is annotated either with f or with a descendant of f in the GO DAG; v is in V_f^0 if $v \notin V_f^+$ and there is a function f' that is an ancestor of f that annotates v; and v is a member of V_f^- otherwise. See Figure 1.2 for an example.

For each gene in V_f^0, our goal is to predict whether that gene should be an element of V_f^+ or V_f^-. We formulate the problem in general terms as computing a (mathematical) function $r : V \rightarrow \mathbb{R}$ that is "smooth" over the nodes of G, i.e., for every edge $(u, v) \in E$, the larger w_{uv} is, the closer $r(u)$ and $r(v)$ are. After computing such a function, we predict every node $v \in V_f^0$ such that $r(v) \geq t$, for some input threshold t, as being annotated with

f. Note that $r(v)$ is directly compared with the threshold, and not with the $r(u)$ values for each node u connected to v. This is because edge weights are used to ensure that r is smooth - i.e, for a highly weighted edge (u, v), it is likely both u and v are classified the same way due to the closeness of $r(u)$ and $r(v)$. There are a number of ways to ensure that r is smooth. A popular technique [ZGL03] is to fix $r(u) = 1$ for each $u \in V_f^+$, $r(u) = -1$ for every $u \in V_f^-$, and to compute r so that it minimizes the "energy"

$$E(G, r) = \sum_{\substack{(u,v)\in E \\ u\in V_f^0 \text{ or } v\in V_f^0}} w_{uv}(r(u) - r(v))^2. \tag{1.1}$$

For a node $v \in V$, let N_v denote the neighbors of v in G. Karaoz *et al.* [KMLe04] restrict $r(v)$ to be either -1 or 1; in this case, they can equivalently minimize

$$-\sum_{\substack{(u,v)\in E \\ u\in V_f^0 \text{ or } v\in V_f^0}} w_{uv} r(u) r(v).$$

An algorithm that iterates over all the nodes in V_f^0 and for each node v sets

$$r(v) = \operatorname{sgn}\left(\sum_{u\in N_v} w_{uv} r(u)\right) \tag{1.2}$$

until the $r(v)$ values converge, will yield a value of energy that is at most half as large as the smallest value possible [KBDS93]. Nabieva *et al.* [NJAe05] and Murali, Wu, and Kasif [MWK06] note that this problem can also be solved by computing minimum cuts in an appropriately transformed version of G. Nabieva *et al.* solve the problem as a special case of the NP-hard minimum multiway k-cut problem using integer linear programming. However, their approach allows a gene to have only one among a set of k functions. The approach adopted by Murali, Wu, and Kasif transforms G into a flow network; they compute the minimum s-t cut in this graph using standard approaches [GT88].

Other approaches to gene function prediction based on FLNs include the use of frameworks such as Markov Random Fields, support vector machines (SVMs), and decision trees. We refer the reader to two recent surveys for discussions of these and other approaches [NBH07, SUS07, PCTMe08].

A thorny issue in gene function prediction is that biological experiments rarely report that a gene *does not* perform a particular function. Hence, the set V_f^- is hard to define accurately. A few approaches attempt to predict gene functions only from V_f^+ and V_f^0 [CX04, NJAe05]. How best to exploit the hierarchical dependence between functions in GO is an active research problem [BST06, PCTMe08]. There may be other types of dependencies between functions, e.g., genes annotated with function f_1 may have a surprisingly large number of edges in G to genes annotated with function f_2. What is the best way to detect such dependencies and utilize them to predict gene function? Finally, the question of systematically using FLNs to transfer function between organisms has received surprisingly little attention [NK07, SSKe05].

1.4 Gene Expression Analysis

Gene expression is the process by which a gene is first transcribed to messenger RNA (mRNA) and then translated into a protein. The *expression level* of a gene is the number of copies of its mRNA that are present in a cell. Genome sequencing and the concomitant advent of DNA microarrays have allowed biologists to simultaneously measure the expression levels of all the genes in a sample of cells; the expression level so measured is an average over all the cells in the sample. DNA microarrays have revolutionized biological research since they capture a snapshot of the activity of all genes in the cells in the sample. A typical gene expression dataset usually consists of measurements from multiple samples under a particular experimental condition; the samples can correspond to multiple time-points after exposing cells to a particular treatment or stimulus or to multiple patients diagnosed with a particular disease.

1.4.1 Gene expression clustering

Let V denote the set of genes in an organism. The gene expression data set for a condition consists of a set of samples S, each with an expression level for each gene in V; we denote by g_S the vector of expression levels for gene g in the samples in S. Let $d = |S|$ and $n = |V|$. Typically, $d \ll n$.

A natural problem that arises now is to cluster the vectors $V_S = \{g_S \mid g \in V\}$.[1] Clustering allows the grouping of genes based on the similarity of their response in the condition; since similarly-expressed genes may perform the same function in the cell, such clusters can be the basis for constructing FLNs for gene function prediction (see Section 1.3). Two popular methods are k-means clustering and hierarchical clustering. In k-means clustering, the goal is to partition V into k sets such that the sum of squared distances from each gene to the centroid of the partition it belongs to is minimized. This problem is known to be NP-hard even for $k = 2$ when d and n are part of the input [DFKe04]. Feldman, Monemizadeh, and Sohler developed a polynomial-time approximation scheme (PTAS) for this problem [FMS07]. Given parameters $\varepsilon, \lambda > 0$, their algorithm computes a $(1 + \varepsilon)$-approximate solution in time $O(nkd + d(k/\varepsilon)^{O(1)} + 2^{\tilde{O}(k/\varepsilon)})$ with probability at least $1 - \lambda$. In practice, most applications of k-means clustering use Lloyd's heuristic [Llo82] – start with a random set of k centers and repeatedly apply the following two steps until convergence: (i) associate each gene with the center closest to it, and (ii) move each center to the centroid of the genes associated with it. Har-Peled and Sadri [HPS05] prove that the number of iterations taken by variants of this algorithm is polynomial in $|V|$, k, and the *spread* of V_S, which is defined to be the diameter of V_S divided by the distance between the two closest genes. Denoting by $\Delta_k^2(V_S)$ the optimal solution to the k-means problem for V_S, Ostrovsky *et al.* [ORSS06] developed a linear-time constant-factor approximation algorithm and a PTAS that returns a $(1 + \varepsilon)$-optimal solution with constant probability in time $O(2^{O(k(1+\omega^2)/\varepsilon)}dn)$, when V_S is ω-separated, i.e., if $\Delta_k^2(V_S)/\Delta_{k-1}^2(V_S) \leq \omega^2$.

The agglomerative version of hierarchical clustering is typically used to analyze gene expression data [ESBB98]. It starts by putting each gene in a separate cluster and repeatedly merging the closest pair of clusters. Typically, these algorithms continue until only one cluster remains. To specify this algorithm completely, it suffices to define the measure of distance between two genes and between two sets of genes. Let $\delta(a, b)$ denote the distance between two genes a and b and let A and B be two sets of genes. In *single-linkage clustering*, we define the distance $\delta(A, B) = \min_{a \in A, b \in B} \delta(a, b)$; hierarchical clustering under this model is equivalent to computing the minimum spanning tree of the complete graph whose nodes

[1] Clustering the vectors corresponding to the samples is also useful. For the sake of concreteness, we focus on clustering the vectors corresponding to the genes.

are genes and an edge between two genes has weight equal to the distance between them. For the frequently used Pearson correlation metric, Seal, Komarina, and Aluru [SKA05] provide an $O(n \log n)$ algorithm for single-linkage clustering by exploiting the geometric transformation that the Pearson correlation coefficient between two gene expression vectors is equal to the cosine of the angle between the corresponding vectors. In *complete-linkage clustering*, $\delta(A, B) = \max_{a \in A, b \in B} \delta(a, b)$, whereas in *average-linkage clustering*, $\delta(A, B)$ is the distance between the centroids of A and B. Naive algorithms usually run in $O(dn^2)$ time and may require $O(n^2)$ space (for storing all pair-wise distances between genes). Krznaric and Levcopoulos [KL02] present an $O(n \log n)$ time and $O(n)$ space algorithm for complete linkage clustering under the L_1 and L_∞ metrics; for every other fixed L_t metric, their algorithm approximates the complete linkage clustering to an arbitrarily-small factor with the same bounds. Borodin, Ostrovsky, and Rabani present sub-quadratic algorithms for approximate versions of the agglomerative clustering problem [BOR04].

Displaying a hierarchical clustering is problematic since the order in which the leaves of the underlying tree should be laid out is unclear. This issue is important since practitioners still use visualizations of the clustering to detect important or interesting patterns in the data. Bar-Joseph *et al.* [BJDGe03] compute an ordering of the leaves that minimizes the sum of the similarity between adjacent leaves in the ordering in $O(4^t n^3)$ time. They also present a method that allows up to t (a user-specified number) clusters to be merged at any step; the method runs in $O(n^3)$ time.

Given a hierarchical clustering of the genes, for every $k > 1$, it is possible to obtain an *induced k-clustering* of the genes, i.e., a partition of the genes into k clusters, by stopping the clustering algorithm when only k clusters remain. Dasgupta and Long [DL05] consider whether there is a hierarchical clustering such that for every $k > 1$, there is an induced k-clustering that is close to the optimal k-clustering of the genes. Defining the cost of a clustering to be the largest radius of one of its clusters, they modify Gonzalez's approximation algorithm for the k-center problem [Gon85] to produce a hierarchical clustering such that for every k, the induced k-clustering has cost at most eight times the optimal k-clustering. They also present a randomized algorithm that achieves an approximation factor of $2e \approx 5.44$.

1.4.2 Gene expression biclustering

The clustering algorithms discussed in the previous section suffer from two primary drawbacks. First, they operate in the space spanned by *all* the samples; thus, they may not detect patterns of clustering that are apparent only in a sub-space of $\mathbb{R}^d$. Second, since many algorithms partition the set of genes into clusters, they are unable to correctly deal with genes that perform multiple functions; such genes should participate in multiple clusters but will be placed in at most one cluster.

Biclustering (also known as projective or subspace clustering) has emerged as a powerful algorithmic tool for tackling these problems. A typical definition of a bicluster is a pair (U, T), where $U \subset V$ and $T \subset S$ such that the genes in U are clustered well in the samples in T but are not clustered well in the samples in $S - T$. In this formulation, a bicluster includes only a subset of genes and samples. Hence, algorithms that compute biclusters capture condition-specific patterns of co-expression. Biclustering algorithms allow a gene or a sample to participate in multiple biclusters, each of which may correspond to a different pathway or biological process. Different biclusters may contain different numbers of genes and/or samples. A number of different methods have emerged for computing biclusters in gene expression data; two papers provide excellent surveys [MO04, TSS06].

A powerful approach to computing biclusters rests on representing gene expression data as a bipartite graph connecting genes to samples. Algorithms use different criteria to decide which gene-sample pairs to connect in such a graph. A bicluster is usually modeled as a bipartite clique (biclique) (U, T), where $U \subset V$ and $T \subset S$. The goal is to compute one or more bicliques of large size in the graph, where size of a biclique is usually defined as $|U||T|$. Finding the biclique with the largest number of edges in an unweighted bipartite graph is known to be NP-hard [Pee03]. Ambühl, Mastrolili and Svensson [AMS07] extend results by Khot [Kho06] to prove that this problem is hard to approximate, i.e., it does not have a PTAS, under the assumption that NP does not have randomized algorithms that run in sub-exponential time. Lonardi, Szpankowski and Yang [LSY06] propose a random-sampling algorithm to compute the largest bicluster (formalized as a biclique in a bipartite graph). We describe a related approach by Mishra, Ron, and Swaminathan [MRS04] in more detail. These authors consider ε-bicliques: each gene in such a biclique is connected to at least a $(1-\varepsilon)$ fraction of the samples in the biclique. They pose the problem of computing an ε-biclique that has at least a $(1-2b)$ fraction of the number of edges in the maximum biclique, for a small constant $b \geq 0$. They present a random-sampling algorithm that is efficient if the largest biclique has at least some fraction ρ_G of the genes and some fraction ρ_S of the samples. Under this assumption, their algorithm runs in time linear in d, logarithmic in n, quasi-polynomial in ρ_G and ρ_S, and exponential in poly$(1/\varepsilon)$.

Mishra, Ron, and Swaminathan also propose a strategy for computing multiple bicliques. Simply computing the k largest bicliques for some value of k may be unsatisfactory: these bicliques may have considerable overlap in their edge sets, and highly overlapping bicliques may not capture the diversity of biclusters present in the data. To preclude this possibility, the authors introduce the notion of δ-*domination*: one biclique δ-dominates another if the number of edges in the second biclique that do not belong to the first is at most a δ fraction of the size of the union of the two edge sets. Next, they introduce the notion of when a collection $\mathcal{C}$ of k ε-bicliques is *diverse*: when for every pair (U', T') and (U'', T'') of bicliques in $\mathcal{C}$, neither δ-dominates the other. Finally, they introduce the notion of when a collection $\mathcal{C}$ of k ε-bicliques *swamps* a biclique (U', T'): either one of the k bicliques in $\mathcal{C}$ δ-dominates (U', T'), or (U', T') does not contain many more edges than any biclique in $\mathcal{C}$. Armed with these definitions, they pose the problem of computing a collection $\mathcal{C}$ of k ε-bicliques that are diverse and swamp every large biclique in the graph (a biclique is *large* if it contains at least some fraction ρ_G of the genes and some fraction ρ_S of the samples). Their algorithm runs in time linear in d, logarithmic in n, quasi-polynomial in k, ρ_G and ρ_S, and exponential in poly$(1/\varepsilon)$.

Tanay, Sharan, and Shamir [TSS02] construct a bipartite graph between genes and samples that represents a discretized version of the data. They assess edge weights in this graph based on a statistical model. They define a bicluster to be a bipartite clique (biclique) of large total edge weight. Under the assumption that each gene is connected to at most a constant number of samples, they simply enumerate all bipartite cliques in this graph. In practice, they supplement this approach with local searches to improve the weight of bipartite cliques and with hashing techniques to speed up the search. In a follow-up paper, they extend their formulation to integrate analysis of different types of genome-wide data [TSKS04]. In this work, the bipartite graph connects genes to properties. Properties include expression of a gene in a sample, regulation of a gene by a transcription factor, and response of a gene to a chemical treatment. In another follow-up study, Tanay *et al.* [TSKS05] extend these techniques to analyze gene expression data from a new study in the context of a large compendium of data from prior studies; they recast the new dataset in terms of biclusters computed from the other datasets and new biclusters discovered only upon the addition of the new data.

Motivated by the approach proposed by Tanay, Sharan, and Shamir, Tan [Tan08] considers the problem of finding the biclique of largest total weight in a weighted bipartite graph where edge weights are positive or negative integers. Under the assumption that the absolute value of the ratio of the smallest edge weight to the largest edge weight is in the range $\Omega(n^{\delta-1/2}) \cap O(n^{1/2-\delta})$,[2] where $\delta > 0$ is an arbitrarily-small constant, Tan shows that this problem is hard to approximate within a factor of ε for some $\varepsilon > 0$ unless RP = NP. Tan *et al.* [TCZZ07] prove that other formulations of biclustering [LO02, BDCKY02] are also hard to approximate.

The approaches discussed above are applicable when real-valued gene expression data is discretized. When such a discretization is not preferred, a geometric viewpoint may be more appropriate. From this perspective, the approaches discussed above compute *orthogonal* biclusters, i.e., each bicluster is a projection of a subset of the genes into an orthogonal subspace of $\mathbb{R}^d$ spanned by a subset of samples. See Figure 1.3 for examples of such biclusters. This image is based on the approach proposed by Procopiuc *et al.* [PJAM02]. In this model, a gene is an element of a bicluster if and only if the gene's expression levels in the samples in the bicluster span an interval of width at most w, where $w > 0$ is a parameter. They consider *dense* projective clusters, i.e., those that contain at least an α-fraction of the samples, $0 \le \alpha \le 1$. In addition, they introduce a condition that specifies the trade-off between the number of genes and number of samples in a bicluster; this condition depends on a parameter β. Under this formulation, they present a Monte Carlo algorithm to compute a bicluster with the largest number of genes, and with width at most $2w$ with probability at least 1/2, in $O(nd^{\log(2/\alpha)/\log(1/(2\beta))})$ time. Melkman and Shaham [MS04] describe a closely-related algorithm for the following model: for every pair of genes participating in a bicluster, the ratio of the expression levels of this pair of genes in each of the samples in the bicluster is a constant depending only on the two genes. They introduce the notion of sleeve-width to allow noise in this model.

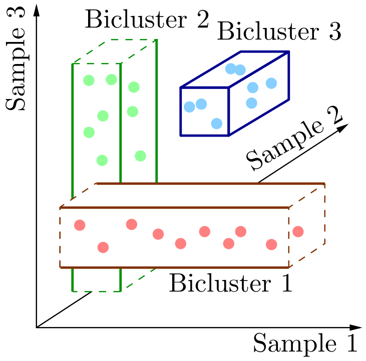

FIGURE 1.3: Example of orthogonal biclusters. Samples 2 and 3 belong to bicluster 1, samples 1 and 2 are elements of bicluster 2, and all three samples are in bicluster 3. A dashed face of a box indicates the dimension along which the box is unbounded.

Attention has also been paid to the problem of non-orthogonal projective clustering. The

[2] We have rewritten Tan's condition assuming $n \ge d$.

typical formulation of this problem seeks to approximate a set V of n points in $\mathbb{R}^d$ by a collection F of k shapes in $\mathbb{R}^d$. The shapes in F may be points, lines, j-dimensional subspaces ($j < d$), or non-linear shapes. By assigning each point in V to the closest subspace in F, we obtain a projective clustering of V. Algorithms attempt to minimize a value such as the sum of the distances from each point to the closest shape, the sum of the squares of these distances, or the largest of these, yielding the k-median, k-mean, or k-centre projective clustering problems respectively. When k or d are part of the input, Megiddo and Tamir [MT83] have shown that many versions of these problems are NP-Hard. Feldman *et al.* [FFSS07] summarize known results for approximating the quality of the best clustering. They point out that all such approximations must be super-polynomial in k, unless $P = NP$. Motivated by this observation, they consider (α, β) bi-criteria approximation algorithms that compute α j-dimensional flats whose quality is within a β factor of the best approximation by k such flats. Their algorithm achieves performance guarantees of $\alpha(k, j, n) = \log n(jk \log\log n)^{O(j)}$ and $\beta(j) = 2^{O(j)}$ in time $dn(jk)^{O(j)}$ with probability at least 1/2. Their algorithm applies simultaneously to the median, mean, and centre versions of the problem.

We note that orthogonal projective clustering algorithms are more likely to be useful in practice, since they are easier to interpret: each bicluster is simply a set of genes and a set of samples. Moreover, existing orthogonal biclustering algorithms allow different computed biclusters to have differing numbers of genes and samples, a property essential to capturing the diversity of different biological processes.

1.5 Structure of the Wiring Diagram

In Section 1.2, we defined a wiring diagram as the network composed of the known molecular interactions for an organism. Specifically, the wiring diagram is a graph where each node is a molecule and each edge is a directed or undirected interaction between two molecules. A pair of molecules may be connected by multiple edges; each edge is usually annotated with information on the type of the interaction, e.g., physical interaction, phosphorylation, or regulation. As mentioned earlier, wiring diagrams are now available for a number of organisms. Some types of networks (e.g., PPI networks) have been experimentally studied on a much larger scale and for many more organisms than others (e.g., transcriptional regulatory networks [HGLR04]). Limitations in experimental techniques lead to considerable noise in available networks; they contain erroneous interactions (false positives) and miss many interactions (false negatives). Assessing error rates at the level of experiment types and individual interactions is an area of active research [SSR$^+$06]. There are other types of uncertainty inherent in these data. For example, we may know that a set of proteins interact to form a protein complex, but we may not know precisely which pairs of proteins interact within the complex [BVH07].

Nevertheless, computational studies of wiring diagrams (especially, PPI network and transcriptional regulatory networks) have yielded numerous insights into their structure and evolution. Preliminary studies of the PPI networks and metabolic networks suggested that their degree distributions follow the power law [AJB00, JTA$^+$00]. More specifically, the fraction of nodes with degree $d \geq 1$ is proportional to $d^{-\gamma}$, with typical values of γ ranging between 2 and 3. More recent studies have cast doubts on these results, arguing that power law distributions may arise from experimental biases and artifacts caused by sampling [HDBe05, SWM05].

What "systems-level" insights into cellular function can wiring diagrams reveal? One of the guiding principles of systems biology is that molecules within the cell organize themselves into "modules" [HHLM99]. A *module* may be loosely defined as a group of interacting molecules that act coherently in the cell. Modules can share both nodes and edges, especially since many genes and proteins are multi-functional. Modules can be hierarchical in the sense that one module may contain another. In a sense, such modules constitute "building blocks" of wiring diagrams. Examples of modules are densely interacting proteins (perhaps forming complexes), protein sub-networks that may be evolutionarily conserved in many organisms, biochemical pathways that synthesize a particular compound, and sets of genes that have co-evolved and are found in multiple genomes.

1.5.1 Network decomposition

Graph clustering, or automatic decomposition of a network into modules or communities, has a rich history, with many problem formulations and techniques [CF06]. In the context of systems biology, a number of papers have studied the problem of decomposing the wiring diagram, mainly PPI networks, into modules [SI06]. These methods use various *ad hoc* heuristics, e.g., repeatedly removing the edge with largest betweenness centrality [DDS05], local searches around multiple seeds [BH03, Bad03], and approaches akin to simulated annealing [SM03]. We describe a few approaches that find a partition or cover of an undirected graph into multiple modules using principled ideas.

Given an undirected graph $G = (V, E)$, Hartuv and Shamir [HS00] define an induced subgraph H of G to be *highly connected* if the minimum number of edges that must be removed in order to disconnect H is at least half the number of nodes in H. They present an output-sensitive recursive algorithm to compute all highly-connected subgraphs of G. For each subgraph returned, the algorithm runs in time taken to compute the minimum cut in G.

Newman [New06] measures the quality of a partitioning of G into two subgraphs G_1 and G_2 using the notion of *modularity*, defined as

$$\frac{1}{4|E|}\sum_{(u,v)}\left(w_{uv} - \frac{d_u d_v}{|E|}\right) s_u s_v, \tag{1.3}$$

where the summation is over all pairs of nodes in V, $w_{uv} = 1$ if $(u, v) \in E$ and 0 otherwise, d_u is the degree of node $u \in V$, and $s_u = 1$ (respectively, -1) if $u \in G_1$ (respectively, G_2). He optimizes this quantity by computing the leading eigenvector of a symmetric matrix whose values are the elements within the summation. To find multiple modules, he recursively applies this algorithm, stopping when the largest eigenvalue for a subgraph is 0. Brandes *et al.* [BDGe06] prove that maximizing modularity is strongly NP-complete.

The $-d_u d_v/|E|$ term in (1.3) arises from the fact that if the edges in G are rewired randomly while maintaining the degree of the nodes, then the probability that u and v are connected is $\frac{d_u d_v}{|E|}$. Intuitively, subtracting this quantity accounts for any modularity that a random graph with the same degree sequence as G may have. This notion arises repeatedly in CSB: what is the probability that an observed network module may arise in "random" data? A typical approach to answering this question empirically is to sample multiple times from the distribution of random networks with the same degree sequence as G, run the network decomposition algorithm on each sample, and use the distribution of module sizes thus obtained to estimate the desired probability. The Markov Chain Monte Carlo method is useful in this situation, but current algorithms have large run-times, making them unsuitable for graphs with tens of thousands of nodes [GMZ03]. Given a degree sequence where the maximum degree $d_{\max} = O(|E|^{1/4-\tau})$, where τ is any positive constant, Bayati, Kim, and Saberi [BKS07] develop an algorithm that runs in $O(|E|d_{\max})$ time and generates any graph with the given degree sequence with probability within $1 \pm o(1)$ factor of uniform.

They also use an approach called sequential importance sampling to convert this algorithm into a fully polynomial randomized approximation scheme. Specifically, for any $\varepsilon, \delta > 0$, with probability at least $1-\delta$, the output of their algorithm is a graph with the given degree sequence; this graph is drawn from the set of all such graphs with uniform probability upto a multiplicative error of $1 \pm \varepsilon$. Their algorithm runs in time $O(|E|d_{\max}\varepsilon^{-2}\log(1/\delta))$.

1.5.2 Evolutionarily-conserved modules

It has been observed that many protein-protein interactions (PPIs) are evolutionarily conserved between different species [YLLe04], i.e., if proteins a and b in one organism interact and if a (respectively, b) is orthologous to protein a' (respectively, b') in another organism, then a' and b' interact. It is natural to ask whether larger sets of interactions may be conserved and how such sets could be automatically computed from PPI networks for two different organisms. In the CSB community, a number of approaches have been developed that use evolutionary constraints to compute such Conserved Protein Interaction Modules (CPIMs) [SI06]. See Figure 1.4 for an illustration.

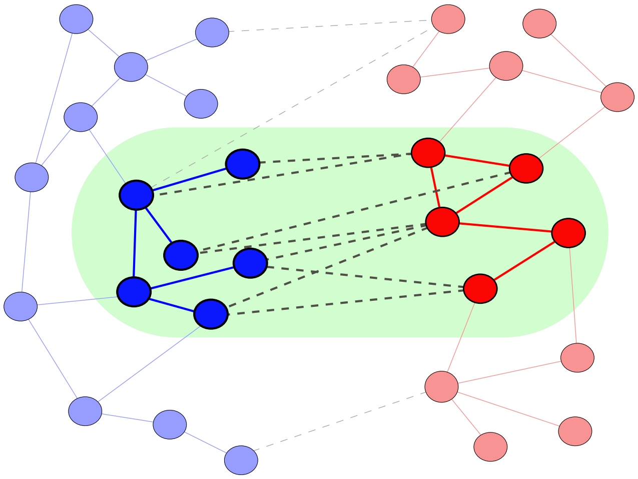

FIGURE 1.4: An illustration of a Conserved Protein Interaction Module (CPIM). Circles represent proteins. Solid lines connect interacting proteins. Dashed lines connect orthologous proteins. The figure contains two PPI networks, one on the left and the other on the right. The darker sub-networks and the pairs of orthologous proteins in those sub-networks (the nodes and edges within the shaded oval) constitute a CPIM.

A number of these approaches [KYLe04, SIKe04, KKTS06, SSI05] share many common features. They combine the PPI networks of two species into a single "alignment graph". A node in the alignment graph represents two orthologous proteins, one from each PPI network. An edge in the alignment graph represents an interaction that is conserved in both PPI networks. These methods add an edge to the alignment graph only if the proteins contributing to the nodes are connected through at most one intermediate protein in the respective PPI networks. The weight of an edge represents the likelihood that the corresponding interactions are conserved; this weight depends on the degree of orthology between the proteins and on assessed confidence estimates that the individual PPIs indeed take place in the cell. After constructing the alignment network, these authors find CPIMs by using various approaches to compute paths, complexes, and subgraphs of high weight in the alignment network and then expanding each such subgraph into the constituent PPIs. Sharan *et al.* [SSKe05] generalize this idea to more than two PPI networks. Liang *et al.* [LXTN06] propose a method where each node in the alignment graph is a pair of conserved PPIs. They develop criteria to connect two such nodes and reduce the problem of computing CPIMs to the problem of finding all maximal cliques in the alignment graph.

Narayanan and Karp have recently presented the Match-and-Split algorithm [NK07]. Like other methods, they define a pair of proteins to be similar if their sequence similarity is at least some threshold. They use combinatorial criteria to decide when the local neighborhoods of a pair of orthologs match. Under their model, they prove that a given pair of proteins can belong to at most one CPIM. This observation leads to a top-down partitioning algorithm that finds all maximal CPIMs in polynomial time.

1.5.3 Network motifs

Milo *et al.* [MSOIe02] pioneered the study of "network motifs" and the bottom-up assembly of complex networks from such motifs. Informally, given two graphs G and H (where H is connected), we say that H *occurs* in G if H is isomorphic to a subgraph of G. We say that H is a *network motif* of G if the number of times that H occurs in G is surprisingly large (we make this notion precise below). Network motifs may play a key role in processing information in regulatory networks [SOMMA02].

A typical approach to computing network motifs (i) identifies the number of subgraphs of G that are isomorphic to a candidate network motif H, (ii) determines the probability that H occurs at least this many times in random graphs with the same degree sequence as G. and (iii) declares H to be a network motif if this probability is smaller than a user-specified threshold. These methods usually enumerate isomorphic subgraphs explicitly, which can be computationally expensive. Wernicke [Wer06] defines the *concentration* of a k-node subgraph H to be the ratio of the number of occurrences of H in G to the total number of occurrences in G of all connected k-node subgraphs. He presents a randomized algorithm that computes an unbiased estimator of the concentration of every connected k-node subgraph that occurs in G. He also shows how to adapt theorems by Bender and Canfield [Ben74, BC78] to estimate the expected concentration of a given k-node subgraph in random graphs with the same degree sequence as G without explicitly generating such random graphs.

Wiring diagrams contain interactions of multiple types. Yeger-Lotem *et al.* [YLSKe04] and Zhang *et al.* [ZKWe05] consider the question of finding "multi-colored" network motifs. Researchers have also considered whether motifs might assemble into larger structures [GK07, ZKWe05] and how such relationships between consolidated subgraphs may reveal insights into the structure of the wiring diagram.

1.6 Condition-Specific Analysis of Wiring Diagrams

As described in the previous section, existing wiring diagrams are tremendous resources for systems biology, since they integrate information on multiple types of molecular interactions obtained from a variety of different experimental sources. However, such an experiment often does not yield information on when an interaction is activated within the cell. Therefore, the potential impact of wiring diagrams is diluted since they typically represent the *universe* of interactions that take place across diverse contexts in the cell. Another deficiency of existing wiring diagrams is that they are highly incomplete, in spite of decades of small-scale experimentation and recent advances in high-throughput screening. For instance, a recent estimate [STdSe08] suggests that the human PPI network may contain 650,000 edges, about an order of magnitude greater than the number obtained by combining multiple existing databases [DMS08]. Note that this estimate is solely for PPIs. Our knowledge of other types of interactions (e.g., between transcription factors and their target genes, between small molecules and metabolites, or between recently-discovered regulatory molecules such as microRNAs and their targets) is even more scarce than for PPIs. In this section, we focus on two classes of methods developed to address the two issues raised above.

1.6.1 Response networks

Many algorithms have been developed to integrate the wiring diagram with gene expression measurements for a single condition (e.g., the time-course of response of a cell to a stress or data from patients diagnosed with a particular disease) in order to compute the sub-network of interactions that is perturbed in that condition. These approaches take a wiring diagram $G = (V, E)$ and a gene expression dataset $V_S = \{g_S \mid g \in V\}$ as input, where S is the set of samples. Their goal is to compute the subgraph G_S of G such that the genes in G_S show the most similar expression patterns over all subgraphs of G.

A common experimental design is to divide the set S of samples into two classes, a set corresponding to a treatment and a set corresponding to a control. In this situation, gene expression measurements are better represented by estimates of differential expression for each gene. It is possible to use a hypothesis testing framework to assess how different the expression levels of a gene in the treatment samples are from its expression levels in the control samples, e.g., by using the t-test. For each gene g, this computation yields a p-value $0 \leq p_g \leq 1$ representing the statistical significance of the difference between the two sets of expression levels of the gene. Given such a data set, Ideker *et al.* [IOSS02] apply an assumption that p_g arises from a normal distribution, and compute a z-score $z_g = N^{-1}(p_g)$, where N^{-1} is the inverse of the normal distribution function. They define the *z-score* $z(G')$ of a subgraph G' of G to be the sum of the z-scores of the nodes in G' divided by the square root of the number of nodes in G'. Their goal is to compute the subgraph of G with the largest z-score. After showing that a version of this problem is NP-complete, they proceed to use simulated annealing to solve the problem.

Murali and Rivera [MR08] propose a method that is applicable when S contains enough samples to estimate the co-expression of any pair of genes. For every edge $e = (g, h)$ in E, they compute a weight w_e that is the absolute value of Pearson's correlation coefficient between g_S and h_S. They define the *density* of a subgraph G' of G as the total weight of the edges in G' divided by the number of nodes in G. Their goal is to compute the subgraph of G with largest density. This problem can be solved in polynomial time using parametric network flows [GGT89]. In practice, they use the greedy algorithm suggested by Charikar [Cha00], which computes a subgraph at least half as dense as the densest subgraph.

These two approaches have the drawback that they consider co-expression relationships only between pairs of genes that are adjacent in G. We have discussed earlier that G is incomplete for many organisms. In such situations, these approaches may ignore many co-expressed pairs of genes. Ulitsky and Shamir [US07] propose an innovative approach to mitigate this problem. They compute an undirected graph X_S where two genes are connected if they are highly co-expressed. In this graph, their goal is to find dense subgraphs under the constraint that each dense subgraph must induce a connected network in G. Thus, two genes may belong to a dense subgraph in X_S even if they are not directly connected in G. Ulitsky and Shamir develop a statistical model for this problem and propose a number of heuristics to compute multiple dense modules [US07].

In principle, these problems are related to the question of computing the largest clique in a graph, a problem well-known to be NP-complete [GJ79], and hard to approximate [Hås99]. Apart from the two papers mentioned above [Cha00, GGT89], theoretical studies of similar problems have usually dealt with unweighted graphs. Feige, Peleg, and Kortsarz [FKP01] compute the densest k-vertex subgraph of a given graph, namely, the subgraph with k vertices that contains the most edges among all k-vertex subgraphs. They develop an approximation algorithm for the problem, with approximation ratio $O(n^{\delta})$, for some $\delta < 1/3$. Khot proves that this problem does not admit a PTAS [Kho06].

Holzapfel *et al.* [HKMT06] pose the γ-CLUSTER problem, where given an undirected graph G and a natural number k, they ask if G has a subgraph on k vertices whose average degree is at least $\gamma(k)$; they allow $\gamma : \mathbb{N} \rightarrow \mathbb{Q}_+$ to be any function that can be computed in polynomial time and satisfies $\gamma(k) \leq k - 1$, for all $k \in \mathbb{N}$. For $\gamma(k) = k - 1$, this problem is the clique problem. In contrast, for $\gamma(k) = 2$, the problem can be solved in polynomial time. They show that the problem remains NP-complete if $\gamma = 2 + \Omega(1/k^{1-\varepsilon})$ for some $\varepsilon > 0$ and has a polynomial-time algorithm for $\gamma = 2 + O(1/k)$.

The spectral radius of an undirected graph G is the largest eigenvalue of the adjacency matrix of the graph. It is well-known that the spectral radius of a graph is at least as large as its average degree. Andersen and Cioaba [AC07] pose the (k, λ)-spectral radius problem: does G have a subgraph on at most k vertices whose spectral radius is at least λ? When such a subgraph exists, they present an approximation algorithm that runs in $O(n\Delta k^2)$ time, where Δ is the maximum degree of the graph, that outputs a subgraph with spectral radius at least $\lambda/4$ and with at most Δk^2 vertices.

1.6.2 Reverse-engineering gene networks

The algorithms described in the previous section assume that a wiring diagram is available. However, as mentioned at the beginning of Section 1.6, existing wiring diagrams are incomplete. To surmount this difficulty, methods have been developed to reverse engineer interactions between genes from gene expression data. The primary assumption underlying these techniques is that if two genes are highly co-expressed, i.e., if their expression levels under one or more conditions have high correlations, then the genes may have a functional interaction. Based on this hypothesis, numerous methods have been developed to infer interactions between pairs of genes [BBAIB07, MS07]. Approaches investigated for gene network construction include gene relevance networks [BK99, DWFS98], Gaussian graphical models [dlFBHM04, SS05], mutual information based networks [BMSe05, BK00, ZAA08], and Bayesian networks [FLN00, Y$^+$02].

In spite of the excitement surrounding these approaches, there is considerable debate about a number of issues. How should the co-expression between two genes be measured and what are the relative advantages of each measure? For example, Pearson's correlation coefficient can be computed in time linear in the number of samples and estimated with confidence even for relatively few samples, but it can only capture linear dependencies. More complex methods typically come with greater computational cost and/or the need for a large number of samples. Does the co-expression of two genes imply stable binding between the proteins that the genes code for, or a cause-and-effect relationship between the genes? These issues have been discussed in a number of papers in the last few years [MS07, BBAIB07, ZSA08, SBA07].

An important problem that arises in gene network construction is to find a sufficient number of samples relative to the network size to be inferred. One could limit the network size if the goal is to infer a subnetwork focused around a biological process that involves a subset of genes, but this requires knowing the genes in advance. In many cases, one would want to infer a gene network to precisely identify such subnetworks and discover unknown genes that may be part of such networks. There are tens of thousands of genes in any complex organism, and in many cases it is impossible to find a reliable way to limit analysis to only a subset of them. The number of samples can be significantly increased by tapping into public repositories of gene expression profiles, resulting from microarray experiments carried out by many laboratories worldwide. Even so, the number of available samples falls short of what is ideally required by the underlying computational methods, with even the number of genes in an organism significantly outnumbering the number of available samples at present. In addition, the use of such large number of samples raises computational and statistical challenges. Gene expression data is inherently noisy and significantly influenced by many experiment-specific attributes. As a result, it is not meaningful to directly compare expression levels across multiple samples. Finally, little is known about general regulatory mechanisms (e.g., post-transcriptional effects) and thus no satisfactory models of genetic regulation are available.

To provide more insight into the construction of gene networks, we present mutual information based methods in greater detail. Mutual information (MI) can capture nonlinear dependencies, the underlying algorithms have polynomial complexity, and recent work demonstrates that these methods generate networks with good quality. Let n denote the number of genes and m denote the number of samples. In MI based methods, the expression level of gene g_i is taken to be random variable X_i, for which we have m recorded observations. The MI between a pair of genes g_i and g_j, denoted $\mathcal{I}(X_i; X_j)$, is given by

$$\mathcal{I}(X_i; X_j) = \mathcal{H}(X_i) + \mathcal{H}(X_j) - \mathcal{H}(X_i, X_j),$$

where the entropy $\mathcal{H}(X)$ of a continuous variable X is given by:

$$\mathcal{H}(X) = -\int p_X(\xi) \log p_X(\xi) d\xi,$$

and p_X is a probability density function for X. In this case p_{X_i}, p_{X_j}, and the joint probability density function p_{X_i,X_j} are unknown and have to be estimated based on available gene expression samples.

To reverse engineer gene networks using this approach, a method for computing the MI between a pair of genes and a criterion for assessing when the MI value is significant are needed. Several different techniques to estimate MI have been proposed [K$^+$07], differing in precision and complexity. Simple histogram methods [BK00] are very fast but inaccurate, especially when the number of observations is small. Gaussian kernel estimators utilized by Margolin *et al.* [MNBe06] provide good precision but take $O(m^2)$ run-time. Daub *et al.* [DSSK04] propose a linear time method that is competitive with the Gaussian kernel estimator. Their method is based on binning, which in its simplest form estimates the probability density of a random variable by dividing samples into fixed number of bins and counting samples per bin. Such a method is imprecise and sensitive to the selection of boundaries of bins [MRL95]. Daub *et al.* overcome this by using B-splines as a smoothing criterion: each observation belongs to k bins simultaneously with weights given by B-spline functions up to order k.

A standard way to assess the significance of MI value between g_i and g_j is to randomly permute the expression values of one of the genes, say g_i, and computing the MI again based on the permuted expression values of g_i and unaltered expression values of g_j. A large number of such permutation tests are conducted, and $\mathcal{I}(X_i; X_j)$ is deemed significant if it is greater than the MI value of at least a fraction $1 - \epsilon$ of the permutations tested. Such

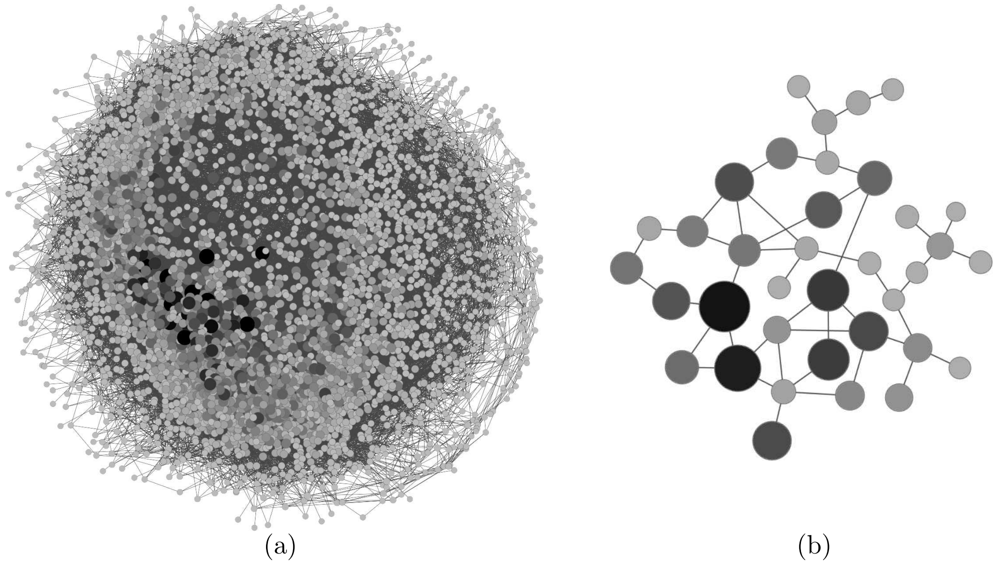

FIGURE 1.5: (a) Yeast regulatory network on 4000 genes inferred by TINGe software. The size of a node is proportional to its degree in the graph. (b) A closer look at the connectivity between some of the genes involved in response to oxidative stress.

testing is expensive, particularly when repeated for the $\binom{n}{2}$ gene pairs. Zola *et al.* [ZAA08] propose a method to make a permutation test applicable to all gene pairs, thereby reducing the complexity of permutation testing by a factor of $\Theta(n^2)$.

A particularly vexing issue in network inference is the difficulty of distinguishing indirect interactions from direct interactions. Consider three genes g_i, g_j, and g_k, where g_i directly interacts with g_j and g_j directly interacts with g_k, but g_i and g_k have no direct interaction. If all three genes are up-regulated in a condition, all three pairs of expression profiles will be correlated. Disentangling direct interactions from indirect ones is difficult, and becomes more complex with larger sets of genes that interact in more intricate ways. Mutual information methods address this using Data Processing Inequality (DPI) [CT91], which states that in case of the example mentioned above, $\mathcal{I}(X_i, X_k) \leq \mathcal{I}(X_i, X_j)$ and $\mathcal{I}(X_i, X_k) \leq \mathcal{I}(X_j, X_k)$. After computing the MI between all pairs of genes, the DPI is run in reverse by identifying such triplets of gene pairs and removing the one with the smallest MI value in each triplet [MNBe06, ZAA08]. Margolin *et al.* [MNBe06] prove that this algorithm correctly recovers the interaction network if the mutual information values can be estimated without errors, the network contains only pairwise interactions, and the network is a tree. They show that their algorithm can reconstruct networks with loops under certain other assumptions. ARACNe, the software implementation of their algorithm, runs in $O(n^3 + n^2m^2)$ time. Zola *et al.* [ZAA08] developed a parallel method for MI based inference that combines Daub *et al.*'s $O(m)$ time B-spline MI estimator with a new method for reducing permutation testing complexity by $\Theta(n^2)$. Their software implementation TINGe scales to whole genome networks and much larger number of samples than previous approaches.

An illustration of a gene network inferred using the MI approach is shown in Figure 1.5. The network in Figure 1.5(a) shows interactions among 4,000 Yeast genes, with node size reflecting the degree of the node. A closer look at the connectivity among some of the genes involved in response to oxidative stress is shown in Figure 1.5(b). Given the complexity of networks, visualization and navigation of these networks is of considerable importance to biologists. Shannon *et al.* developed a widely used software environment, termed Cytoscape, for this purpose [SMe03], which allows third party plugins for further enhancing the functionality of the environment as necessary. Another problem is that of comparing across multiple network inference methods, all the more important due to the diversity of approaches being applied. While a few biologically validated networks exist, it is useful to have a wide array of benchmark data sets with different numbers of genes, samples, quality of samples etc. It is common practice to validate network inference algorithms using synthetically generated networks by programs such as SynTReN [dBLNe06] and COPASI [HSe06]. SynTReN takes a gene network as input (such as a biologically validated network of Yeast), and creates a synthetic benchmark network of a specified size with similar or desired topological properties, and the desired number of samples with user-specified noise. COPASI is capable of generating time series data – such data is particularly valuable in inferring directionality of interactions. For example one may infer that gene g_i regulates gene g_j, if a time delay is observed between the rise in expression value of g_j when compared to the rise in expression value of g_i. Time series data can be especially valuable for gene network construction, but requires access to carefully timed experiments. Thus, undirected network inference using large public data repositories continues to be of value to gather sufficient number of samples to build robust gene networks.

1.7 Outlook and Resources for Further Study

Although the field is young, research in computational systems biology is taking place at an intense level commensurate with the importance and potential applications of this field. This is hardly surprising given the potential for broad impact in such critical areas as health and human disease, and management of agricultural and animal resources. In this interdisciplinary field, apart from computer science and molecular biology, many other disciplines contribute to knowledge discovery including chemical engineering, physics, control theory, and statistics. In keeping with the scope and expected audience of this handbook, our coverage of topics is heavily influenced by areas with substantial contributions from computer science, and the role of algorithms and theory even within that. Not covered in detail here are many topics related to contributions with techniques from other areas of computer science such as text mining for annotation and network extraction, heterogeneous data integration, models of network evolution, and data and graph mining. A number of techniques normally considered outside the realm of algorithms and theory have also been utilized in making substantial contributions to systems biology. Examples include metabolic flux analysis, physics driven models, and modeling of biological processes as complex control systems, although algorithmic questions naturally arise when these models eventually result in the need for computational solutions.

The field of systems biology is expected to grow quite rapidly, aided by both increasing availability of experimental data and continued discovery and refinement of computational models and techniques. On the experimental side, high throughput experimental techniques are continually being improved and increasing amounts of such data are being generated, e.g., comprehensive gene expression profile measurements for various organisms. The predominant culture in the community of open data sharing through web portals is a significant driver of innovation, drawing in scientists from many fields with no interest in conducting the experiments *per se*, or in developing such expertise. New experimental techniques to measure various aspects of cellular activity are expected to come on line, along with refined measurement capability for existing instrumentation. To match these experimental advances on the computational end, we need better graph theoretic models of biological processes, approaches to reason about networks, and techniques for modeling experiments and their results. In the future, we envision the routine use of computational models as a mechanism for suggesting useful experiments for biologists and the incorporation of the results of these experiments into more refined models of the cell.

For the reader interested in further forays into the field or keeping abreast of this rapidly growing field, a number of resources are available, some of which we mention here. The open access *PLoS Computational Biology* journal publishes a series on “Getting started in . . . ”, which is a valuable resource for understanding many topics relevant to systems biology. The *Nature* Insights series provides an editorial and a compendium of commentaries on specific focus topics. *Nature* also launched a series of “Connections essays” to explore how large number of interacting components result in systems level behavior. Finally, *Nature Cell Biology* and *Nature Reviews Molecular Cell Biology* have partnered together to publish several review articles in various subfields of systems biology. We encourage readers of the chapter to tap into these and other resources for further study of this fascinating and emerging area of scientific discovery. Finally, computational biology is a vast research area and to permit a reasonable exposition within this chapter, we limited ourselves to the emerging important area of systems biology. Readers interested in a comprehensive introduction to the field of computational biology are referred to the handbook edited by one of the authors [Alu06].

Acknowledgements

Work on this book chapter is supported in part by the National Science Foundation under CCF-0811804.

1.8 Glossary

Molecules

Gene a DNA sequence (a substring of a chromosome) that is involved in encoding one or more functional products such as an RNA or a protein. The sequence includes coding regions that code for the functional product(s), and non-coding regions such as introns.

Protein a chain of amino acids synthesized in a specific order from the transcription and translation of a gene via the genetic code.

Transcription factor a protein that binds to a specific DNA sequence (cis-element) in the promoter region of a gene to control its transcription.

Cis-element a short DNA sequence found in the upstream non-coding region of the gene that controls the transcriptional activity of a gene via transcription factors or other DNA-binding elements.

Promotor a regulatory sequence of DNA located in the upstream 5’ non-coding region of the gene that controls transcription of the gene.

Enzyme a protein that catalyzes a biochemical reaction.

Kinase a protein that adds a phosphate group to a protein, the substrate.

Phosphatase a protein that removes a phosphate group from a protein.

Homolog a gene whose nucleotide sequence exhibits similarity to another gene or a set of genes.

Ortholog a gene in one organism is *orthologous* to a gene in another organism if both genes have evolved from the same gene in a common ancestral organism. Orthology is usually established by comparing genetic sequences.

Interactions

Protein-protein interactions An association between a set of protein molecules to form long-term protein complexes.

Transcriptional regulatory interaction an interaction between a transcription factor and promotor of a gene to regulate gene expression.

Genetic interaction interaction between a set of genes where the action of one gene is modified by another gene or a set of genes.

Biochemical reaction a process where an enzyme interacts with one or more molecules (substrate) to produce a product.

Phosphorylation addition of a phosphate group to organic molecules by enzymes called kinases.

Processes

Gene expression the conversion of a gene (DNA sequence) into a functional gene product such as RNA or a protein.

Protein translation the process of converting messenger RNA, the product of gene expression, into a chain of amino acids using the genetic code.

Post-translational modification changes made to a protein after translation such as addition of a chemical group, or formation of a protein complex, or making structural changes to the protein.

Signal transduction a process by which signals are transmitted by proteins and other molecules from the outside of a cell to its interior or within a cell.

Gene Functions

Gene Ontology a standard nomenclature that is used to describe genes and gene product attributes across organisms.

Cellular component a component of the cell.

Molecular function an activity, such as catalytic or binding activity, that occurs at the molecular level.

Biological process a series of events accomplished by one or more ordered assemblies of molecular functions.

Protein localization positioning of a protein in an appropriate cellular area (e.g., an organelle, an interior membrane, etc.) where its activity is needed.

Phenotype an observable characteristic or trait of an organism.

Organisms

Eukaryote an organism whose cells contain nuclei. Genomic DNA is contained within the nucleus of each cell.

Prokaryote an organism whose cells do not have nuclei.

Model organism an organism that is extensively studied with the expectation that knowledge gained here provides valuable insights into other related organisms.

References

References

[ABBB00] M. Ashburner, C.A. Ball, J.A. Blake, and D. Botstein *et al.* Gene Ontology: tool for the unification of biology. the Gene Ontology Consortium. *Nature Genetics*, 25(1):25–29, 2000.

[AC07] R. Andersen and S.M. Cioaba. Spectral Densest Subgraph and Independence Number of a Graph. *Journal of Universal Computer Science*, 13(11):1501–1513, 2007.

[AJB00] R. Albert, H. Jeong, and A.L. Barabasi. Error and attack tolerance of complex networks. *Nature*, 406(6794):378–382, 2000.

[Alu06] S. Aluru, editor. *Handbook of computational molecular biology.* Chapman & Hall/CRC Computer and Information Science Series, Boca Raton, FL, 2006.

[AMS07] C. Ambühl, M. Mastrolilli, and O. Svensson. Inapproximability results for sparsest cut, optimal linear arrangement, and precedence constrained scheduling. In *Proc. 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 329–337, 2007.

[Bad03] J. S. Bader. Greedily building protein networks with confidence. *Bioinformatics*, 19(15):1869–74, 2003.

[BBAIB07] M. Bansal, V. Belcastro, A. Ambesi-Impiombato, and D. Bernardo. How to infer gene networks from expression profiles. *Molecular Systems Biology*, 3:78, 2007.

[BC78] E.A. Bender and E.R. Canfield. The asymptotic number of labeled graphs with given degree sequences. *Journal of Combinatorial Theory Series A*, 24(3):296–307, 1978.

[BCS06] G.D. Bader, M.P. Cary, and C. Sander. Pathguide: a pathway resource list. *Nucleic Acids Research*, 34(Database issue):D504–D506, 2006.

[BDCKY02] A. Ben-Dor, B. Chor, R.M. Karp, and Z. Yakhini. Discovering local structure in gene expression data: the order-preserving submatrix problem. In *Poc. 6th International Conference on Computational Biology (RECOMB)*, pages 49–57, 2002.

[BDGe06] U. Brandes, D. Delling, M. Gaertler, and R. Goerke *et al.* Maximizing Modularity is hard. *Arxiv preprint physics/0608255*, 2006.

[Ben74] E.A. Bender. The asymptotic number of non-negative integer matrices with given row and column sums. *Discrete Mathematics*, 10:217–223, 1974.

[BH03] G. D. Bader and C. W. Hogue. An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinformatics*, 4(1), January 2003.

[BJDGe03] Z. Bar-Joseph, E.D. Demaine, D.K. Gifford, and N. Srebro *et al.* K-ary clustering with optimal leaf ordering for gene expression data. *Bioinformatics*, 19(9):1070–1078, 2003.

[BK99] A.J. Butte and I.S. Kohane. Unsupervised knowledge discovery in medical databases using relevance networks. *Proc. American Medical Informatics Association Symposium*, pages 711–715, 1999.

[BK00] A.J. Butte and I.S. Kohane. Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. In *Proc. Pacific Symposium on Biocomputing*, pages 418–429, 2000.

[BKS07] M. Bayati, J.H. Kim, and A. Saberi. A sequential algorithm for generating random graphs. In *Approximation, Randomization, and Combinatorial*

Optimization. Algorithms and Techniques, 10th International Workshop, APPROX 2007, and 11th International Workshop, RANDOM 2007, Proceedings, volume 4627 of *Lecture Notes in Computer Science*, pages 326–340. Springer, 2007.

[BMSe05] K. Basso, A.A. Margolin, G. Stolovitzky, and U. Klein *et al.* Reverse engineering of regulatory networks in human B cells. *Nature Genetics*, 37(4):382–390, 2005.

[BOR04] A. Borodin, R. Ostrovsky, and Y. Rabani. Subquadratic Approximation Algorithms for Clustering Problems in High Dimensional Spaces. *Machine Learning*, 56(1):153–167, 2004.

[BST06] Z. Barutcuoglu, R.E. Schapire, and O.G. Troyanskaya. Hierarchical multilabel prediction of gene function. *Bioinformatics*, 22(7):830–6, 2006.

[BVH07] A. Bernard, D.S. Vaughn, and A.J. Hartemink. Reconstructing the Topology of Protein Complexes. In *Proc. 11th International Conference on Research in Computational Molecular Biology (RECOMB)*, pages 32–46, 2007.

[CF06] D. Chakrabarti and C. Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Computing Surveys*, 38(1):Article 2, 2006.

[Cha00] M. Charikar. Greedy approximation algorithms for finding dense components in a graph. In *Proc. 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 84–95, 2000.

[CSZ06] O. Chapelle, B. Schölkopf, and A. Zien, editors. *Semi-Supervised Learning.* MIT Press, Cambridge, MA, 2006.

[CT91] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley-Interscience, New York, NY, 1991.

[CX04] Y. Chen and D. Xu. Global protein function annotation through mining genome-scale data in yeast Saccharomyces cerevisiae. *Nucleic Acids Res*, 32(21):6414–24, 2004.

[dBLNe06] T. Van den Bulcke, K. Van Leemput, B. Naudts, and P. Van Remortel *et al.* SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms. *BMC Bioinformatics*, 7:43, 2006.

[DDS05] R. Dunn, F. Dudbridge, and C. M. Sanderson. The use of edge-betweenness clustering to investigate biological function in protein interaction networks. *BMC Bioinformatics*, 6(1), March 2005.

[DFKe04] P. Drineas, A. Frieze, R. Kannan, and S. Vempala *et al.* Clustering Large Graphs via the Singular Value Decomposition. *Machine Learning*, 56(1):9–33, 2004.

[DL05] S. Dasgupta and P.M. Long. Performance guarantees for hierarchical clustering. *Journal of Computer and System Sciences*, 70(4):555–569, 2005.

[dlFBHM04] A. de la Fuente, N. Bing, I. Hoeschele, and P. Mendes. Discovery of meaningful associations in genomic data using partial correlation coefficients. *Bioinformatics*, 20(18):3565–74, 2004.

[DMS08] M.D. Dyer, T.M. Murali, and B.W. Sobral. The landscape of human proteins interacting with viruses and other pathogens. *PLoS Pathogens*, 4(2):e32, 2008.

[DSSK04] C.O. Daub, R. Steuer, J. Selbig, and S. Kloska. Estimating mutual information using B-spline functions – an improved similarity measure for analysing gene expression data. *BMC Bioinformatics*, 5:118, 2004.

[DWFS98] P. D'haeseleer, X. Wen, S. Fuhrman, and R. Somogyi. *Information Processing in Cells and Tissues*, chapter Mining the gene expression matrix: inferring gene relationships from large scale gene expression data, pages 203–

212. 1998.

[EKO03] A.J. Enright, V. Kunin, and C.A. Ouzounis. Protein families and tribes in genome sequence space. *Nucleic Acids Research*, 31(15):4632–4638, 2003.

[ESBB98] M.B. Eisen, P.T. Spellman, P.O. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceeding of the National Academy of Sciences USA*, 95:14863–14868, 1998.

[FFSS07] D. Feldman, A. Fiat, M. Sharir, and D. Segev. Bi-criteria linear-time approximations for generalized k-mean/median/center. In *Proc. 23rd annual Symposium on Computational Geometry*, pages 19–26, 2007.

[FKP01] U. Feige, G. Kortsarz, and D. Peleg. The dense k-subgraph problem. *Algorithmica*, 29:410–421, 2001.

[FLN00] N. Friedman, M. Linial, and I. Nachman. Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, 7:601–620, 2000.

[FMS07] D. Feldman, M. Monemizadeh, and C. Sohler. A PTAS for k-means clustering based on weak coresets. In *Proc. 23rd Annual Symposium on Computational Geometry*, pages 11–18, 2007.

[GGT89] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM Journal on Computing*, 18(1):30–55, 1989.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, New York, NY, 1979.

[GJF07] A. Godzik, M. Jambon, and I. Friedberg. Computational protein function prediction: are we making progress? *Cellular and Molecular Life Sciences*, 64(19-20):2505–2511, 2007.

[GK07] J. Grochow and M. Kellis. Network motif discovery using subgraph enumeration and symmetry-breaking. In *Proc. 11th Annual International Conference on Research in Computational Molecular Biology (RECOMB), Springer-Verlag Lecture Notes in Computer Science*, volume 4453, pages 92–106, 2007.

[GMZ03] C. Gkantsidis, M. Mihail, and E. Zegura. The Markov chain simulation method for generating connected power law random graphs. In *In Proc. 5th Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 16–25, 2003.

[Gon85] T. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

[GT88] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. *Journal of the Association for Computing Machinery*, 35:921–940, 1988.

[Hås99] J. Håstad. Clique is hard to approximate within 1- ε. *Acta Mathematica*, 182(1):105–142, 1999.

[HDBe05] J-D.J. Han, D. Dupuy, N. Bertin, and M.E. Cusick *et al.* Effect of sampling on topology predictions of protein-protein interaction networks. *Nature Biotechnology*, 23(7):839–844, 2005.

[HGLR04] C.T. Harbison, D.B. Gordon, T.I. Lee, and N.J. Rinaldi *et al.* Transcriptional regulatory code of a eukaryotic genome. *Nature*, 431(7004):99–104, 2004.

[HHLM99] L.H. Hartwell, J.J. Hopfield, S. Leibler, and A.W. Murray. From molecular to modular cell biology. *Nature*, 402(6761 Suppl):C47–52, 1999.

[HKMT06] K. Holzapfel, S. Kosub, M.G. Maaß, and H. Täubig. The complexity of detecting fixed-density clusters. *Discrete Applied Mathematics*, 154(11):1547–1562, 2006.

[HPS05] S. Har-Peled and B. Sadri. How Fast Is the k-Means Method? *Algorithmica*, 41(3):185–202, 2005.

[HS00] E. Hartuv and R. Shamir. A clustering algorithm based on graph connectivity. *Information Processing Letters*, 76(4-6):175–181, 2000.

[HSe06] S. Hoops, S. Sahle, and R. Gauges *et al.* COPASI–a COmplex PAthway SImulator. *Bioinformatics*, 22:3067–3074, 2006.

[IL03] T. Ideker and D. Lauffenburger. Building with a scaffold: emerging strategies for high- to low-level cellular modeling. *Trends in Biotechnology*, 21(6):255–62, 2003.

[IOSS02] T. Ideker, O. Ozier, B. Schwikowski, and A.F. Siegel. Discovering regulatory and signalling circuits in molecular interaction networks. *Bioinformatics*, 18 Suppl 1:S233–S240, 2002.

[JTA$^+$00] H Jeong, B Tombor, R Albert, Z N Oltvai, and AL Barabasi. The large-scale organization of metabolic networks. *Nature*, 407(6804):651–4, 2000.

[K$^+$07] S. Khan et al. Relative performance of mutual information estimation methods for quantifying the dependence among short and noisy data. *Physical review. E*, 76(2 Pt 2):026209, 2007.

[Kar04] P.D. Karp. Call for an enzyme genomics initiative. *Genome Biology*, 5(8):401–401.2, 2004.

[KBDS93] S. Kasif, S. Banerjee, A. L. Delcher, and G. Sullivan. Some results on the complexity of symmetric connectionist networks. *Annals of Mathematics and Artificial Intelligence*, 9:327–344, 1993.

[Kho06] S. Khot. Ruling out PTAS for graph min-bisection, dense k-subgraph, and bipartite clique. *SIAM Journal on Computing*, 36(4):1025–1071, 2006.

[KKTS06] M. Koyuturk, Y. Kim, U. Topkara, and S. Subramaniam. Pairwise alignment of protein interaction networks. *Journal of Computational Biology*, 13(2):182–99, 2006.

[KL02] D. Krznaric and C. Levcopoulos. Optimal algorithms for complete linkage clustering in d dimensions. *Theoretical Computer Science*, 286(1):139–149, 2002.

[KMLe04] U. Karaoz, T. M. Murali, S. Letovsky, and Y. Zheng *et al.* Whole genome annotation using evidence integration in functional linkage networks. *Proceedings of the National Academy of Sciences USA*, pages 2888–2893, 2004.

[KYLe04] B.P. Kelley, B. Yuan, F. Lewitter, and R. Sharan *et al.* PathBLAST: a tool for alignment of protein interaction networks. *Nucleic Acids Research*, 32(Web Server issue), 2004.

[LLC$^+$08] Insuk Lee, Ben Lehner, Catriona Crombie, Wendy Wong, Andrew G. Fraser, and Edward M. Marcotte. A single gene network accurately predicts phenotypic effects of gene perturbation in caenorhabditis elegans. *Nature Genetics*, 40(2):181–188, January 2008.

[Llo82] S. P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28:129–137, 1982. Special issue on quantization.

[LMTK08] K. Liolios, K. Mavromatis, N. Tavernarakis, and N.C. Kyrpides. The Genomes On Line Database (GOLD) in 2007: status of genomic and metagenomic projects and their associated metadata. *Nucleic Acids Research*, 36(Database issue):D475–D479, 2008.

[LO02] L. Lazzeroni and A. Owen. Plaid models for gene expression data. *Statistica Sinica*, 12:61–86, 2002.

[LSY06] S. Lonardi, W. Szpankowski, and Q. Yang. Finding biclusters by random projections. *Theoretical Computer Science*, 368(3):217–230, 2006.

[LXTN06] Z. Liang, M. Xu, M. Teng, and L. Niu. Comparison of protein interaction networks reveals species conservation and divergence. *BMC Bioinformatics*, 7:457, 2006.

[MNBe06] A.A. Margolin, T. Nemenman, K. Basso, and C. Wiggins *et al.* ARACNE: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. *BMC Bioinformatics*, 7 Suppl 1, 2006.

[MO04] S.C. Madeira and A.L. Oliveira. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.

[MR08] T.M. Murali and C.G. Rivera. Network legos: Building blocks of cellular wiring diagrams. *Journal of Computational Biology*, 15(7):829–844, 2008.

[MRL95] Y. Moon, B. Rajagopalan, and U. Lall. Estimation of mutual information using kernel density estimators. *Physical review. E*, 52(3):2318–2321, 1995.

[MRS04] N. Mishra, D. Ron, and R. Swaminathan. A new conceptual clustering framework. *Machine Learning Journal*, 56(1–3):115–151, 2004.

[MS04] A.A. Melkman and E. Shaham. Sleeved coclustering. In *Proc. 10th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 635–640, 2004.

[MS07] F. Markowetz and R. Spang. Inferring cellular networks - a review. *BMC Bioinformatics*, 8(Suppl 6), 2007.

[MSOIe02] R. Milo, S. Shen-Orr, S. Itzkovitz, and N. Kashtan *et al.* Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[MT83] N. Megiddo and A. Tamir. Finding least-distances lines. *SIAM Journal on Algebraic Discrete Methods*, 2:207–211, 1983.

[MT07] F. Markowetz and O.G. Troyanskaya. Computational identification of cellular networks and pathways. *Molecular Biosystems*, 3(7):478–482, 2007.

[MWK06] T. M. Murali, C-J. Wu, and S. Kasif. The art of gene function prediction. *Nature Biotechnology*, 12:1474–1475, 2006.

[NBH07] W.S. Noble and A. Ben-Hur. *Bioinformatics - From Genomes to Therapies*, volume 3, chapter Integrating information for protein function prediction, pages 1297–1314. Wiley, 2007.

[New06] M.E.J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[NJAe05] E. Nabieva, K. Jim, A. Agarwal, and B. Chazelle *et al.* Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. *Bioinformatics*, 21 Suppl 1:i302–i310, 2005.

[NK07] M. Narayanan and R.M. Karp. Comparing protein interaction networks via a graph match-and-split algorithm. *Journal of Computational Biology*, 14(7):892–907, 2007.

[ORSS06] R. Ostrovsky, Y. Rabani, L.J. Schulman, and C. Swamy. The effectiveness of Lloyd-type methods for the k-means problem. In *Proc. 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 165–176, 2006.

[PCTMe08] L. Pena-Castillo, M. Tasan, C.L. Myers, and H. Lee *et al.* A critical assessment of *Mus musculus* gene function prediction using integrated genomic evidence. *Genome Biology*, 9 Suppl 1:S2, 2008.

[Pee03] R. Peeters. The maximum edge biclique problem is np-complete. *Discrete Applied Mathematics*, 131(3):651–654, 2003.

[PJAM02] C.M. Procopiuc, M.T. Jones, P.K. Agarwal, and T. M. Murali. A Monte-Carlo algorithm for fast projective clustering. In *Proc. International Conference on Management of Data*, pages 418–427, 2002.

[RKKe04] R.J. Roberts, P. Karp, S. Kasif, and S. Linn *et al.* An experimental approach to genome annotation. Report of a workshop organised by the American Academy of Microbiology, 2004.

[Rob04] R.J. Roberts. Identifying protein function–a call for community action. *PLoS Biol*, 2(3):E42, 2004.

[SBA07] N. Soranzo, G. Bianconi, and C. Altafini. Comparing association network algorithms for reverse engineering of large-scale gene regulatory networks: synthetic versus real data. *Bioinformatics*, 23(13):1640–1647, 2007.

[SI06] R. Sharan and T. Ideker. Modeling cellular machinery through biological network comparison. *Nat Biotechnol*, 24(4):427–33, 2006.

[SIKe04] R. Sharan, T. Ideker, B.P. Kelley, and R. Shamir *et al.* Identification of protein complexes by comparative analysis of yeast and bacterial protein interaction data. In *Proc. 8th annual International Conference on Computational Molecular Biology*, pages 282–289, 2004.

[SKA05] S. Seal, S. Komarina, and S. Aluru. An optimal hierarchical clustering algorithm for gene expression data. *Information Processing Letters*, 93(3):143–147, 2005.

[SM03] V. Spirin and L.A. Mirny. Protein complexes and functional modules in molecular networks. *Proceedings of the National Academy of Sciences USA*, 100(21):12123–12128, 2003.

[SMe03] P. Shannon, A. Markiel, and O. Ozier *et al.* Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Research*, 13(11):2498–2504, 2003.

[SOMMA02] S.S. Shen-Orr, R. Milo, S. Mangan, and U. Alon. Network motifs in the transcriptional regulation network of *Escherichia coli*. *Nature Genetics*, 31(1):64–8, 2002.

[SS05] J. Schafer and K. Strimmer. An empirical Bayes approach to inferring large-scale gene association networks. *Bioinformatics*, 21(6):754–764, 2005.

[SSI05] S. Suthram, T. Sittler, and T. Ideker. The *Plasmodium* protein network diverges from those of other eukaryotes. *Nature*, 438(7064):108–112, 2005.

[SSKe05] R. Sharan, S. Suthram, R.M. Kelley, and T. Kuhn *et al.* From the cover: Conserved patterns of protein interaction in multiple species. *Proceedings of the National Academy of Sciences USA*, 102(6):1974–1979, 2005.

[SSR$^+$06] S. Suthram, T. Shlomi, E. Ruppin, R. Sharan, and T. Ideker. A direct comparison of protein interaction confidence assignment schemes. *BMC Bioinformatics*, 7:360, 2006.

[STdSe08] M.P. Stumpf, T. Thorne, E. de Silva, and R. Stewart *et al.* Estimating the size of the human interactome. *Proceedings of the National Academy of Sciences USA*, 105(19):6959–6964, 2008.

[SUS07] R. Sharan, I. Ulitsky, and R. Shamir. Network-based prediction of protein function. *Molecular Systems Biology*, 3:88, 2007.

[SWM05] M.P.H. Stumpf, C. Wiuf, and R.M. May. Subnets of scale-free networks are not scale-free: sampling properties of networks. *Proceeding of the National Academy of Sciences USA*, 102(12):4221–4224, 2005.

[Tan08] J. Tan. Inapproximability of maximum weighted edge biclique and its applications. In *Proc. 5th Annual Conference on Theory and Applications of Models of Computation, Springer-Verlag Lecture Notes in Computer Science*, volume 4978, pages 282–293, 2008.

[TCZZ07] J. Tan, K.S. Chua, L. Zhang, and S. Zhu. Algorithmic and complexity issues of three clustering methods in microarray data analysis. *Algorithmica*, 48(2):203–219, 2007.

[TSKS04] A. Tanay, R. Sharan, M. Kupiec, and R. Shamir. Revealing modularity and organization in the yeast molecular network by integrated analysis of highly heterogeneous genomewide data. *Proceedings of the National Academy of Sciences USA*, 101(9):2981–2986, 2004.

[TSKS05] A. Tanay, I. Steinfeld, M. Kupiec, and R. Shamir. Integrative analysis of genome-wide experiments in the context of a large high-throughput data compendium. *Molecular Systems Biology*, 1(1):msb4100005–E1–msb4100005–E10, 2005.

[TSS02] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18 Suppl 1:S136–S144, 2002.

[TSS06] A. Tanay, R. Sharan, and R. Shamir. *Handbook of Computational Molecular Biology*, chapter Biclustering Algorithms: A Survey, pages 26–1. Chapman & Hall/CRC Press Computer and Information Science Series, 2006.

[US07] I. Ulitsky and R. Shamir. Identification of functional modules using network topology and high-throughput data. *BMC Syst Biol*, 1:8, 2007.

[Wer06] S. Wernicke. Efficient detection of network motifs. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(4):347–359, 2006.

[Y^+02] H. Yu et al. Using Bayesian network inference algorithms to recover molecular genetic regulatory networks. In *Proc. of International Conference on Systems Biology*, 2002.

[YLLe04] H. Yu, N.M. Luscombe, H.X. Lu, and X. Zhu *et al.* Annotation transfer between genomes: protein-protein interologs and protein-dna regulogs. *Genome Research*, 14(6):1107–1118, June 2004.

[YLSKe04] E. Yeger-Lotem, S. Sattath, N. Kashtan, and S. Itzkovitz *et al.* Network motifs in integrated cellular networks of transcription-regulation and protein-protein interaction. *Proceedings of the National Academy of Sciences USA*, 101(16):5934–5939, 2004.

[ZAA08] J. Zola, M. Aluru, and S. Aluru. Parallel information theory based construction of gene regulatory networks. In *Proc. 15th International Conference on High Performance Computing (HiPC), Springer-Verlag Lecture Notes in Computer Science*, volume 5374, pages 336–349, 2008.

[ZGL03] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proc. 20th International Conference on Machine Learning*, 2003.

[ZKWe05] L.V. Zhang, O.D. King, S.L. Wong, and D.S. Goldberg *et al.* Motifs, themes and thematic maps of an integrated Saccharomyces cerevisiae interaction network. *Journal of Biology*, 4(2):6, 2005.

[ZSA08] M. Zampieri, N. Soranzo, and C. Altafini. Discerning static and causal interactions in genome-wide reverse engineering problems. *Bioinformatics*, 2008.