Constrained Community-based Gene Regulatory Network Inference

Ferdinando Fioretto Agostino Dovier Enrico Pontelli

Abstract

The problem of *Gene Regulatory Network inference* is a major concern of Systems Biology. In recent years, a novel methodology has gained momentum, called *Community Network* approach. Community networks integrate predictions from individual methods in a "meta predictor", in order to compose the advantages of different methods and soften individual limitations. This paper proposes a novel methodology to integrate prediction ensembles using *Constraint Programming*, a declarative modeling and problem solving paradigm. Constraint Programming naturally allows the modeling of dependencies among components of the problem as constraints, facilitating the integration and use of different forms of knowledge. The new paradigm, referred to as *Constrained Community Network*, uses constraints to capture properties of the regulatory networks (e.g., topological properties) and to guide the integration of knowledge derived from different families of network predictions. The paper experimentally shows the potential of this approach: the addition of biological constraints can offer significant improvements in prediction accuracy.

1 Introduction

In a cellular context, *genes* interact to orchestrate a variety of fundamental tasks, such us the response of cells to environmental stimuli (e.g., a drug), the cell proliferation, and its apoptosis (i.e., cells death). Research in the field of *Systems Biology* has highlighted the importance of investigating such interactions at the different levels in which they occur, in order to provide a comprehensive understanding of the behavior of a biological system. At the cellular level, proteins are considered to be amongst the most important components to carry out those functions which are necessary for cell regulation. In simple terms, the information needed to produce proteins is encoded within the genes. The process used to express a protein can be abstracted as a two-step process: first, the information of coding DNA (gene) is *transcribed* into a *messenger RNA* (mRNA) (Fig. 1(a)). Next, the mRNA is *translated* into a sequence of *amino acids*, that constitute a protein (Fig. 1(b)).

The products of each of these steps may be involved in the process of gene regulation. For instance proteins called *Transcription Factors (TFs)* can bind directly to meaningful regions of the DNA, leading to **(1)** enhanced mRNA production associated to a gene, and possibly its translation into proteins, or **(2)** inhibition of the process associated to transcription and, hence, gene expression. Some non-coding RNA fragments—i.e., RNA which is not translated into proteins—are also associated with regulation of gene expression. For instance *micro RNA (miRNA)* may bind to mRNA, promoting its degradation or preventing it from being translated into proteins [33].

A detailed description of the system involving each of these regulatory mechanisms would not appear a viable option for studying cells at a system level, due to its enormous complexity. Therefore this machinery is simplified and projected onto the transcriptomic level, where only genes are considered (DNA level in Fig. 1).

The set of regulatory interactions involving genes in a cell is referred to as *Gene Regulatory Network (GRN)*. GRNs capture both transcriptomic and proteomic regulatory events (Fig. 1 (c)), which are implicitly encoded in the gene regulation process and difficult to interpret in physical terms. In turn, uncovering the nature of gene regulatory interactions is referred to as *GRN inference* and it is of central importance in Systems Biology. Its use is crucial in understanding system regulations and to devise effective medical interventions, and it has been shown to be very promising in understanding some genetic diseases such as cancer [35].

GRNs can be reconstructed from manual literature curation [9], or using reverse engineering computational approaches [17]. These two flows have different limitations. The former lacks the possibility of integrating novel measurements, for instance, in possibly compromised signaling networks, and therefore it cannot detect cellular responses under specific biological stimuli. Causal signaling links can vary depending on lineage and (epi)genetic background, such that the same perturbation can lead to different signaling responses in different backgrounds. Thus, it is important to be able to feed a prediction method with experimental data that can be acquired for the specic biological context of interest. A biological context may, e.g., be defined by a combination

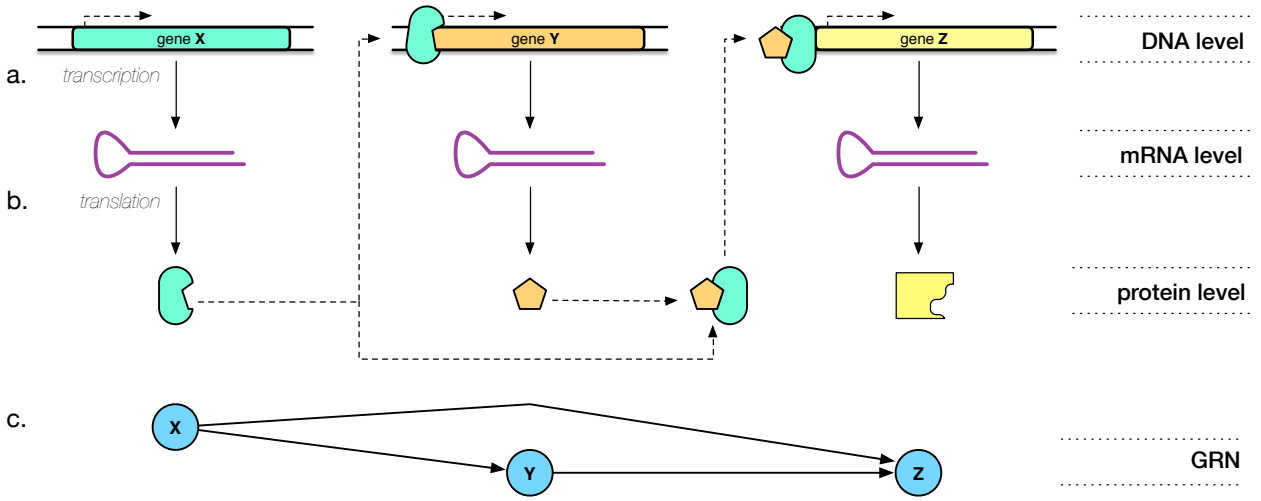

Figure 1: *Simplified representation of the regulatory mechanisms involved in a Gene Regulatory Network. Gene X* regulates *gene Y, by encoding a transcription factor which activates the transcription of gene Y. Genes X and Y* co-regulate *gene Z: the proteins produced by X and Y interact to form a complex, which activates gene Z. In the Gene Regulator Network inference problem, one aims to reconstruct the relations between genes (bottom of the figure).*

of cell line (a specific type of cell) and growth condition (an external stimuli which enhances the production of some cellular product). The latter methods inherently take account of the biological context of an experiment. On the other hand they may fail to observe well studied interactions, when the experimental data with which they are fed only includes particular cell conditions.

The development of new technologies in molecular biology, e.g., DNA microarray or high-throughput sequencing, has made available a wealth of genomic data, encouraging the development of novel computational methods for GRN inference. However, data sets are highly heterogeneous, containing information which is limited and difficult to analyze [53]. This impacts the performance of GRN inference methods, which tend to be biased towards specific types of data [31].

To alleviate these difficulties, several alternatives have been proposed, such as methods to integrate heterogeneous data into the inference model [47, 20] or to integrate a collection of predictions across different inference methods—as in the *Community Network (CN)* approach [36]. Methods based on integration of heterogeneous data are a promising research direction, but they face several challenges, which span from how to relate different types of data to issues of data normalization [50]. The CN method has the advantage of promoting the benefits of individual methods, while smoothing out their drawbacks. CN does not exclude the use of heterogeneous data in the initial prediction set, and has been shown to be robust across species and data sets [36]. The CN approach poses many challenges, e.g., **(i)** how to account for strengths/weaknesses of individual methods—e.g., the difficulty of Mutual Information methods to discriminate TFs; and **(ii)** how to use information not handled by the individual methods.

We propose a methodology based on *Constraint Programming (CP)* to combine community predictions and integrate biological knowledge—leading to a new paradigm for community networks, referred to as *Constrained Community Networks*. CP is a declarative problem solving paradigm, where logical rules are used to model problem properties and to guide the construction of solutions. CP offers a natural environment where heterogeneous information can be actively handled. The use of constraint expressions allows the incremental refinements of a model. This is particularly suitable to model biological knowledge integration, when such knowledge cannot be directly handled by individual prediction methods. CP provides an effective framework to model different types of network information that may become available during problem modeling or hypotheses testing, and use such information in the inference process.

We tested our method on a set of 360 benchmarks, including large networks proposed by the DREAM3 [41] and DREAM4 [27] challenges. We perform our experiments with three type of data obtained in two different experimental setups. We show significant improvements in prediction accuracy compared to a state of the art CN-based approach, up to 29.5%, when the integration of knowledge about target networks acquired in biological relevant settings is applied.

2 Background

2.1 Constraint Programming

Constraint Programming (CP) is a declarative programming paradigm commonly used to address combinatorial search problems. It focuses on capturing properties of the problem in terms of *variables* (representing the unknowns of the problem) and *constraints* over the variables (i.e., relations among the components of the problem), which are satisfied exclusively by solutions of the problem. Solutions to the problem are represented by assignments of values to the variables. CP models are fully declarative and elaboration tolerant, enabling the incremental integration of new knowledge and the use of sophisticated problem solving techniques (e.g., propagation and filtering methods, search heuristics).

A *Constraint Satisfaction Problem (CSP)* is formalized as a triple $\langle \mathbf{X}, \mathbf{D}, \mathbf{C} \rangle$. $\mathbf{X} = \langle x_1, \ldots, x_n \rangle$ is an n-tuple of variables—i.e., the unknowns of the problem. $\mathbf{D} = \langle D_1, \ldots, D_n \rangle$ is a corresponding n-tuple of domains; each D_i is a set of values, specifically the admissible values for the variable x_i. $\mathbf{C} = \langle C_1, \ldots, C_k \rangle$ is a k-tuple of constraints. Let us consider a subset $S_j \subseteq \mathbf{X}$ of the variables; a constraint C_j over S_j is a subset of the Cartesian product of the domains of the variables in S_j—i.e., $C_j \subseteq \Pi_{x_r \in S_j} D_r$. Intuitively, a constraint over the variables S_j restricts what are the joint assignments of values to the variables in S_j.

Given an n-tuple $A = \langle a_1, \ldots, a_n \rangle \in D_1 \times \cdots \times D_n$, we denote with $A|_{S_j}$ the projection of the tuple on the variables in S_j. For example, if $S_j = \{x_1, x_2\}$, then $A|_{S_j} = \langle a_1, a_2 \rangle$. The largest (resp. smallest) value that can be assigned to a variable x_i is denoted by $\max(D_i)$ (resp. $\min(D_i)$).

A *solution* of a CSP $\langle \mathbf{X}, \mathbf{D}, \mathbf{C} \rangle$ is an n-tuple $A = \langle a_1, \ldots, a_n \rangle$ where $a_i \in D_i$ (for $1 \leq i \leq n$) and $A|_{S_j} \in C_j$ (for $1 \leq j \leq k$)—i.e., the projection of A on the set of variables involved in C_j satisfies the relation C_j. Typical resolution algorithms for CSPs rely on efficient *search procedures*, to explore the space of possible solutions, and on *consistency methods*, where constraints are used to remove infeasible elements from the domains of not yet assigned variables. This search is made by exploring a data structure called *prop-labeling-tree* [5] composed by two kind of nodes: (**i**) nodes with as many children as the current size of the domain of a selected variable (non-deterministic choices) and (**ii**) nodes with a unique child obtained deterministically by a process of constraint propagation. Search strategies are developed for alternative visiting of the search tree (that is dynamically computed using backtracking). Incomplete methods are used for large problems, where the search is guided by random choices.

2.2 Gene Regulatory Networks and Inference Methods

A *Gene Regulatory Network (GRN)* can be described by a weighted directed graph $\mathcal{G} = (V, E)$, where V is the set of regulatory elements of the network and $E \subseteq V \times V \times [0, 1]$ is the set of regulatory interactions. The presence of an edge $\langle s, t, w \rangle \in E$ indicates that an interaction between the regulatory elements s and t is present with *confidence* value $w \in [0, 1] \subseteq \mathbb{R}$. The number $|V|$ is referred to as the *size* of the GRN. If the GRN has no uncertainty, then each edge in E will have weight equal to 1.

In the problem of *GRN inference*, we are given the set of vertices V (in this paper each $g_i \in V$ represents a gene) and a set of experiments, describing the behavior of the regulatory elements. The goal is to accurately detect the set of regulatory interactions E. The observations associated to the expression profiles of the gene g_i in a GRN are described via a random variable G_i whose values are typically normalized in $[0, 1]$.

We provide an overview of the network inference methods adopted in our investigation. We classify methods in five classes according to their main component.

2.2.1 Correlation

Correlation-based network inference methods rely on the notion of statistical dependence, a condition in which random variables do not satisfy a requirement of probabilistic independence. The correlation between gene expression levels is expressed by a number in $[-1, 1] \subseteq \mathbb{R}$ to indicate the presence of a regulatory interaction. A positive (negative) value indicates an activating (inhibitory) interaction. We consider three standard correlation coefficient: *Pearson*, *Spearman*, and *Kendall*. Pearson coefficient relates the standard deviation of the expression profiles of two genes with their covariance and it is limited to capture linear dependencies. The other two relate the ranked expression levels of genes and can capture how well two variables can be described via a monotonic function. The correlation coefficients considered are symmetric: i.e., for two random variables X and Y it holds that $corr(X, Y) = corr(Y, X)$. Therefore, additional information is required to assign directionality to

the inferred interactions. As our evaluation does not discriminate inhibiting from activating interactions, we focus on the absolute value of the correlation coefficients.[1] Correlation measures are widely adopted in practice to study the relationships among gene expressions, e.g., in [19] they have been used to reconstruct the GRN associated to the central nervous system development in rats.

2.2.2 Mutual Information

One of the limitations of correlation-based methods is their inability to identify non-linear relationships among variables. *Mutual Information (MI)* methods overcome this limitation, by measuring the common information in two random variables X and Y. Let us assume that X (Y) range on a finite set $\mathcal{D}_X$ ($\mathcal{D}_Y$). In a GRN inference context these values emerges from a discretization of the expression levels of the associated genes emerging from experimental measurements, possibly normalized in $[0, 1]$. This approach quantifies to which extent knowing one of these variables reduces the uncertainty about the other. For instance, if X and Y are independent, no additional information about Y is produced by knowing X and vice versa; thus, their mutual information is 0. We denote with $I(X;Y)$ the mutual information of variables X and Y. MI-based methods cannot infer the direction of an interaction.

The *Context Likelihood of Relatedness* (CLR) [23] assigns a score z_{ij} to each interaction between genes g_i, g_j: $z_{ij} = \sqrt{z_i^2 + z_j^2}$, with $z_i = \max_{j \neq i} \left(0, \frac{I(G_i;G_j) - \mu_i}{\sigma_i}\right)$ where μ_i and σ_i are respectively the mean and the standard deviation of the empirical distribution of the MI values $I(G_i;G_k)$ of G_i for all the variables G_k, $k \neq i$. This represents the background distribution of the MI for gene g_i and it plays a central role in the CLR algorithm by aiming at reducing the prediction of false interactions—based on false correlations—and indirect interactions. CLR has been successfully applied to decipher the *E. coli* transcriptional regulatory network [23].

The *Algorithm for Reconstruction of Accurate Cellular Networks* (ARACNE) [37] aims at filtering out indirect interactions by applying the Data Processing Inequality (DPI). The DPI states that if gene g_i interacts with gene g_j through a gene g_k then $I(G_i;G_j) \leq \min(I(G_i;G_k), I(G_j;G_k))$. After computing the MI of the pair of genes involved, ARACNE filters out all the interactions for which their MI does not exceed a given threshold. Then it prunes the weakest interactions within each triplet of genes if it violates the DPI test . This approach has been validated using microarray dataset from reconstructing the GRN associated to human B cells [37].

The *Conservative Causal Core* (C3NET) [4] algorithm consists of two steps. First, it detects the non-significant connections among gene pairs g_i, g_j. This is realized by testing the statistical significance of their MI $I(G_i;G_j)$, by assessing whether the null hypothesis $H_0 : I(G_i;G_j) = 0$ cannot be rejected for a given significance level. For each gene g_i, it selects the most significant link (g_i, g_j) for which the null hypothesis cannot be rejected based on their mutual information estimate.

BC3NET is an extension of C3NET in which an ensemble of datasets is generated via bootstrapping and each of the bootstrapped dataset is fed to the C3NET network inference procedure. The inferred networks are hence aggregated employing a binomial test [18]. A C3NET algorithm has been used to identify tumor specific gene interactions in prostate cancer datasets [3].

2.2.3 Other Statistical Tests

The *Generalized Logical Network* (GLN) models interactions as many-to-one relationships between a set of TFs and a target gene [49]. An interaction is ranked by its p-value in the χ^2 test; in the case of ties, interactions with lower degrees of freedom are ranked higher. The significance of the χ^2 statistics accounts for both linear and non-linear interactions. GLN has been adopted to identify genes from major neuronal pathways in the alcohol response mechanism from the brains of alcohol-treated mice [49].

2.2.4 Feature Selection

In the context of supervised learning, feature selection is the process of selecting a subset of relevant features to be used in the model construction. This process can be viewed as an optimization problem, where the measure to be optimized is a score of the different subsets of features. Since the general problem of selecting the best

[1] The refinement of interaction types (inhibiting vs. excitatory) will be a future step in our research.

subset of features is computationally intractable, several techniques based on (incomplete) local search methods are commonly adopted [30].

MRNET infers interactions between genes by using mutual information between expression profiles and a feature selection procedure called *Maximum Relevance Minimum Redundancy (MRMR)* [39]. For each variable G_j, treated as a target gene, MRNET aims at selecting a set of regulators $\mathbf{S}_j$ of g_j having high MI with G_j (maximum relevance) and low MI between them (minimum redundancy). The selection of the set $\mathbf{S}_j$ is made via a forward selection procedure, which starts by including the variable with the highest MI with the target G_j. The other variables being selected will be the ones having high MI with G_j and low MI with the variables already in $\mathbf{S}_j$. A specific network can then be inferred by only keeping edges whose score lies above a given threshold (similarly to what done in CLR).

GEne Network Inference with Ensemble of trees (GENIE3) is similar to MRNET, in that: **(a)** it considers each gene individually, treating it as the target gene regulated by the other genes, and **(b)** it employs a feature selection procedure to identify the best set of regulator genes. GENIE3 uses a decision tree learning approach, where leaves nodes of the decision tree describe class labels, while each internal node represents a test on an attribute, and each branch represents the outcome of test. A path from the root to a leaf node represents a classification rule. A decision tree can be learned by *splitting* the set of items into subsets based on an attribute value test, so to create different branches, and repeating this process recursively on each derived subset.

The feature selection step of GENIE3 is performed via Random Forests [14]. At each feature selection step, GENIE3 generates an ensemble of $1,000$ trees, built using a bootstrap sample composed of $p-1$ randomly selected attributes, where p is the number of potential regulators. In each tree, each node n selected for a split, is augmented with a score that accounts for the total reduction of the variance of the output variable due to the split: $I(n) = |\mathbf{S}|\sigma(\mathbf{S}) - |\mathbf{S}_t|\sigma(\mathbf{S}_t) - |\mathbf{S}_f|$, where $\mathbf{S}$ denotes the set of samples reaching node n, $\mathbf{S}_t$ and $\mathbf{S}_f$ denote the subsets of $\mathbf{S}$ for which the test was respectively true or false, and $\sigma(\cdot)$ is the variance of the output variable in a given set. For each target gene, the importance of a gene as its regulator is computed by summing the tree nodes where such gene is used as a variable to split the tree, and averaging the results across the ensemble of trees. The results of each subproblem are aggregated to get the final ranked list of regulatory interactions. GENIE3 was the best performer in the DREAM 4 challenge [27].

Tigress, similarly to MRNET and GENIE3, employs a feature selection strategy to estimate a score $s_j(i)$ of each candidate regulator g_i for a target gene g_j. This is determined as the solution of a regression problem, aimed at predicting the expression level G_j from the expression level of its candidate regulators $G_i \in \mathbf{S}_j \subseteq \mathbf{G}$: $G_j = f_j(\mathbf{S}_j) + \epsilon$, where $\mathbf{G}$ is the set of all the G_i's associated to the genes of the GRN, f_j is a regression function and ϵ is a term modeling some noise. The algorithm does not aim to model the regression function f_j, but rather to find a small set of regulators $\mathbf{S}_j$ which are sufficient to provide a good model for G_j. The score $s_j(i)$ is associated to each candidate regulator g_i, and it assesses the likelihood of G_i to be involved in the regression model f_j; this is computed via a *Least Angle Regression (LARS)* [21] with *stability selection* [38]. Tigress was evaluated to be the best linear regression-based method in the DREAM5 gene network inference challenge [36] and one among the top overall performers.

2.2.5 Meta approaches

The *Inferelator pipeline* is a meta approach based on re-sampling combining *median-corrected z-scores (MCZ)*, to rank edges based on a z-score derived from TF-deletion data, *time-lagged CLR (tlCLR)*, for the analysis of time-series data, and a linear *ordinary differential equation (ODE)* model constrained by Lasso [27]. The kernel of the Inferelator is based on a ODE model which governs the time evolution of a gene product accounting for both RNA production and degradation rates for each gene (see [11] for details).

The Inferelator was used to predict a large portion of the regulatory network of the *archaeon Halobacterium NRC-1* under specific perturbations [11].

2.2.6 Community Inference as Committees

Combining different models for solving classification problems has been an active topic of research in machine learning [10, 43]. The use of multiple trained models in combination often results in improved performance and enhanced robustness with respect to merely using a single model in isolation [43]. A widely adopted aggregation strategy is that of *committees*, a meta predictor where multiple models are combined by averaging the results of each individual predictor.

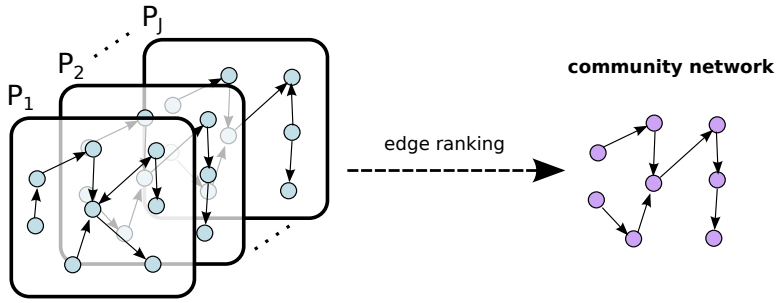

Figure 2: *Community Network* generation via committee models integration.

Consider a simple regression problem where we want to predict the value of a continuous function $t(x)$. The committee prediction for an input x is given by averaging the predictions of the M committee members: $\hat{t}(x) = \frac{1}{M}\sum_{i=1}^{M} f_i(x)$, where $f_i(x)$ is the prediction of the i-th method in the committee at input x. If the prediction errors made by the individual predictors are all uncorrelated and have 0 mean, the average error of a model could be reduced by a factor of M simply by averaging M members. Even though typically errors are highly correlated and the performance gain could be small, the work of Perrone [40] shows that even when committee members are correlated and biased, the squared prediction error of the committee (obtained through an averaging process) is no worse than the mean squared prediction error of the individual committee members, i.e., $(\hat{t}(x) - t(x))^2 \leq \frac{1}{M}\sum_{i=1}^{M}(f_i(x) - t(x))^2$. Informally, this means that an averaging process can only improve the results, provided that the committee members make better than random predictions.

In a binary classification problem, $t(x)$ could be interpreted as the probability of belonging to one class (e.g., the presence of a regulatory relation), and $1 - t(x)$ as the probability of belonging to the other class (e.g., the absence of a regulatory relation); $f_i(x)$ represents the prediction of the i-th method for $t(x)$.

In the context of GRN inference, committees are referred to as *Community Networks (CNs)* and are used to integrate multiple inference methods to obtain a common consensus prediction, as illustrated in Fig. 2, where the P_i's represent the GRN predictions obtained by different methods. CNs have been shown to achieve better average confidence across different datasets and produce more robust results with respect to the individual methods being composed [36]. A simple scheme for combining predictions in a community network has been proposed by Marbach et al. where each interaction is re-scored by averaging the ranks it obtained within each of all the employed predictions: we will refer to this method as CN_{rank}.

The inference methods adopted in this study is listed in Tab. 1.

Pearson	Spearman	Kendall	MRNET	Aracne	CLR
C3Net	BC3Net	Tigress	Genie3	Inferelator	GLN

Table 1: *The Community Network prediction methods adopted.*

3 Constrained Community GRN inference

Constraint Technologies and *Constraint Programming (CP)* have been recently successfully applied in the field of System Biology [52]. For example, Answer Set Programming has been adopted to address problems in network inconsistencies detection [26] and in metabolic network analysis [48]. CP has been used to investigate discrete network models, under the Thomas' GRN Model [51], where GRNs are modeled using multi-valued variables and transition rules [16]. In particular, CP is used to represent GRNs' possible dynamics and to test, for a given structure of a GRN, the consistency of a set of hypotheses—allowing the relaxation of the constraints imposed on the network behavior when inconsistencies arise [15]. The *Biocham* platform [22] makes use of temporal and other classes of constraints to support modeling and simulation of regulatory networks. Concurrent constraint programming has also been used to support modeling of biological systems [13], where interacting molecules are viewed through the lenses of communicating processes.

The CN approach adopted in this work is built by combining multiple GRN inference procedures and creating an *inference ensemble*. The methods used to create the ensemble have been selected based on their performance, popularity and availability.

The methods selection process used to build an ensemble starts from a set of 12 methods from the classes

described in Sect. 2.2. While building the final ensemble, we impose the constraint that exactly one representative methods from each class should be an ensemble component; the only exception is when two methods in a class are distinguished by a secondary component. This choice provides robustness and diversity, while avoiding redundancies that could potentially bias the inference ensemble.

A preliminary version of the research described in this paper has been presented in [24]. The work presented in this paper provides a more detailed model description and formalization of the CN-based GRN inference problem, introducing a new set of constraints, more general and effective of those originally discussed in [24]. The present work also removes several restrictive assumptions used in our previous work, by limiting, analyzing and automatically tuning the constraints' parameters. In addition, while our previous work restricted the constraint solver to integer finite domains (and therefore creating the potential for discretization errors), the constraint solver introduced in this paper is capable of handling real values. The present manuscript introduces a comprehensive assessment of the CN-based GRN inference schema. It includes an extensive evaluation of a broad set of individual prediction methods and their combination in committees, and analyzes the proposed method on a wide set of large GRNs and three different datasets.

3.1 Problem Formulation

Given a set of n genes, we describe a GRN inference problem as a CSP $\langle \mathbf{X}, \mathbf{D}, \mathbf{C} \rangle$, where:

- $\mathbf{X}$ is a set of $n^2 - n$ variables, each of them referred as $X_{i \to j}$, with $i, j \in \{1, \ldots, n\}$, $i \neq j$. These variables describe regulatory relations (excluding self regulations);
- $\mathbf{D}$ is the set of domains for the variables in $\mathbf{X}$. Each $D_{i \to j}$ is a finite set of elements in $[0, 1] \subset \mathbb{R}$, describing the possible confidence values associated with the regulatory relation modeled by $X_{i \to j}$. Values close to 0 indicate high confidence about the *absence* of a regulatory relation (with 0 denoting the highest confidence), whereas values close to 1 indicate high confidence about the presence of a regulatory relation (with 1 denoting the highest confidence);
- $\mathbf{C}$ is a k-tuple of constraints $\langle C_1, \ldots, C_k \rangle$. Each C_j is a constraint over a set of variables $S_j \subseteq \mathbf{X}$. Constraints expressing restrictions of peculiar network topologies will be discussed in Sect. 3.3.

A variable $X_{i \to j}$ is said to be *assigned* when its associated domain $D_{i \to j}$ is a singleton. We adopt the notation $d(X_{i \to j})$ to indicate the value of an assigned variable $X_{i \to j}$. A solution to the above CSP defines a GRN prediction $G = (V, E)$, with $V = \{1, \ldots, n\}$ and $E = \{\langle i, j, d(X_{i \to j}) \rangle \mid i \in V, j \in V, d(X_{i \to j}) > 0\}$.

3.2 Ensemble Analysis and Initial Domains Construction

The proposed CSP solution leverages the collection of GRN predictions within the prediction ensemble by: **(1)** tightening the size of the solution search space[2] and **(2)** taking into account the discrepancies among the community predictions. These objectives are achieved by mapping the edge confidence levels of each prediction to the corresponding CSP variable domain. The greater the agreement in the inference ensemble, the smaller is the set of values in the domain of the variable representing the relation being considered. Thus, the size of each domain captures the degree of uncertainty expressed by an edge prediction within the inference ensemble.

Let us consider a set of J predictions $\mathcal{P}$ of a GRN $G = (V, E)$. For $j \in \{1, \ldots, J\}$, let us denote with $P_j = (V, E_j)$ the j-th prediction, where E_j are the edges that have been identified by P_j. For a labeled edge $(s, t, w) \in E_j$ we identify w as the confidence assigned for the presence of that edge in the GRN, and $\omega_j^{\#}(s, t)$ as its rank—its position in the descending ordered list of confidence values of E_j—normalized in $[0, 1]$, where 0 is associated to the last position, and 1 to the first one. Furthermore, let θ_d ($0 \leq \theta_d \leq 1$) be a given threshold, referred to as disagreement threshold.

The procedure described in Algorithm 1 populates the domains in $\mathbf{D}$ with at most three values. For each edge (s, t), we calculate (line 4) the average confidence value `w_rank`, according to the *Borda* count election method, as presented by Marbach et al. [36]. This method averages the ranked edge confidence values ($\omega_j^{\#}(s, t)$) assigned by each prediction P_j. Line 4 also determines the discrepancy value `w_d` within $\mathcal{P}$. The discrepancy value captures the ensemble prediction disagreement for a given edge, averaging the pairwise differences of the edge ranks associated to each prediction of the ensemble. If the discrepancy value exceeds the discrepancy threshold θ_d and the average confidence value is not strongly informative—that is, it lies between values $L \in [0, 1]$ and $U \in [0, 1]$ (line 6)—then we force the domain $D_{s \to t}$ to account for the prediction disagreement by

[2] For domains having all size b the search space of a GRN inference problem of size n is $b^{n^2 - n}$.

ALGORITHM 1: *Initialization of the Domains of the Variables.*

/* Require normalized $P_j \in \mathcal{P}, \theta_d, G = (V, E)$ */

1 $J \leftarrow |\mathcal{P}|$
2 **for** $(s,t) \in E$ **do**
3 $\quad D_{s \to t} \leftarrow \emptyset$
4 $\quad (\texttt{w\_rank}, \texttt{w\_d}) \leftarrow \left(\frac{1}{J} \sum_{j=1}^{J} \omega_j^{\#}(s,t), \quad \frac{1}{\binom{J}{2}} \sum_{j=1}^{J} \sum_{i=j+1}^{J} \left| \omega_j^{\#}(s,t) - \omega_i^{\#}(s,t) \right| \right)$
5 $\quad D_{s \to t} \leftarrow \{\texttt{w\_rank}\}$
6 $\quad$ **if** $\texttt{w\_d} \geq \theta_d \ \wedge \ L < \texttt{w\_rank} < U$ **then**
7 $\quad\quad D_{s \to t} \leftarrow D_{s \to t} \cup \left\{ \max\left(0, \texttt{w\_rank} - \frac{\texttt{w\_d}}{2}\right), \min\left(1, \texttt{w\_rank} + \frac{\texttt{w\_d}}{2}\right) \right\}$
8 $\quad$ **end**
9 **end**

adding a variation of w_d/2 to the average confidence value. Line 5 ensures the presence of the value w_rank in $D_{s \to t}$. All the parameters of the algorithm—i.e., θ_d, L and U—are automatically tuned, and they depend entirely on the prediction ensemble (see Sect. 4.3.1 for details).

3.3 Constraints

3.3.1 Sparsity constraints

It is widely accepted that the GRN machinery is controlled by a relatively small number of genes. Several state-of-the-art methods for predicting GRNs encourage *sparsity* in the inferred networks [36]. Nevertheless, when combining predictions in a community based approach, no guarantees on the sparsity of the resulting prediction can be provided. To address this issue, we introduce a `sparsity` *constraint* which is built from two more general constraints: `atleast_k_ge` and `atmost_k_ge`. They both enforce a relation among a set of variables, to ensure that at least (resp. at most) k of the variables have values exceeding a given threshold:

$$\texttt{atleast\_k\_ge}(k, \mathbf{S}, \theta) \equiv \quad \left| \{ X_i \in \mathbf{S} \mid d(X_i) > \theta \} \right| \geq k \tag{1}$$

(1) enforces a lower bound ($k \in \mathbb{N}$) on the number of variables in $\mathbf{S} \subseteq \mathbf{X}$ whose confidence value is greater than θ (with $0 \leq \theta \leq 1$). The constraint `atmost_k_ge` is defined in the same way where $\geq k$ is replaced by $\leq k$.

These constraints are used to filter the domains of the variables involved, through a *propagation* process. The propagation of the `atmost_k_ge` constraint is exploited during solution search to enforce its semantics and performed by the following rewriting rule:[3]

$$\texttt{atmost\_k\_ge}(k, \mathbf{S}, \theta): \quad \frac{\mathbf{T} = \{ X_i \in \mathbf{S} \mid \min(D_i) > \theta \}, \ |\mathbf{T}| = k}{\bigwedge_{X_i \in \mathbf{S} \setminus \mathbf{T}} D_i = D_i \cap [0, \theta]} \tag{2}$$

Intuitively, if there are already k variables in $\mathbf{S}$ whose possible values are greater than θ, then all other variables should have θ as an upper bound to their admissible value.

For the `atleast_k_ge` constraint, early failures can be detected during the solution search, by checking the upper bound on the number of variables not yet instantiated which satisfy property (1). The associated propagation rule is:

$$\texttt{atleast\_k\_ge}(k, \mathbf{S}, \theta): \quad \frac{\mathbf{T} = \{ X_i \in \mathbf{S} \mid \max(D_i) \leq \theta \}, \ |\mathbf{S} \setminus \mathbf{T}| = k}{\bigwedge_{X_i \in \mathbf{S} \setminus \mathbf{T}} D_i = D_i \cap (\theta, 1]} \tag{3}$$

The `sparsity` constraint ranges over the variables in $\mathbf{X}$. It enforces lower and upper bounds on the number of edges whose confidence value is outside a given threshold. Formally, given $k_l, k_m, \theta_l, \theta_m$ (these parameters are discussed in Sect. 4.3.1) :

$$\texttt{sparsity}(\mathbf{X}, k_l, \theta_l, k_m, \theta_m) \equiv \quad \texttt{atleast\_k\_ge}(k_l, \mathbf{X}, \theta_l) \ \cap \ \texttt{atmost\_k\_ge}(k_m, \mathbf{X}, \theta_m) \tag{4}$$

[3] $\frac{condition}{consequence}$ indicates that the domain transformation *consequence* is applied whenever *condition* is satisfied

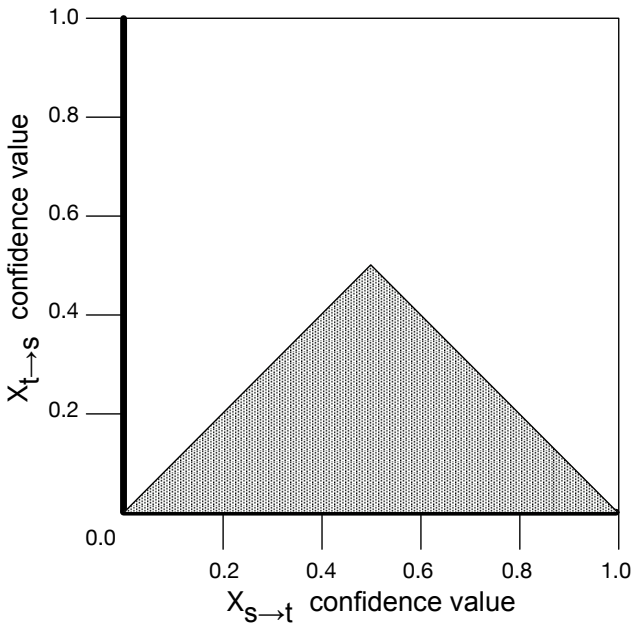

Figure 3: Values for $X_{t\to s}$

3.3.2 Edge orientation constraint

Given two variables $X_{s\to t}$ and $X_{t\to s}$ the *edge orientation constraint* (`orient`) exploits the confidence value assigned to $X_{s\to t}$ to impose an upper bound on the values that can be assigned to $X_{t\to s}$. This constraint imposes an orientation for an edge between two given nodes and it is described as follows:

$$\texttt{orient}(X_{s\to t}, X_{t\to s}) \equiv \quad X_{t\to s} \leq \min(X_{s\to t}, 1 - X_{s\to t}). \tag{5}$$

This constraint bounds the variable $X_{t\to s}$ to the confidence value of $X_{s\to t}$ and, if the existence of the edge (s,t) is predicted with a high confidence (> 0.5), by a factor which is inversely proportional to $X_{s\to t}$. Figure 3 depicts the upper bound for the confidence values of the variable $X_{t\to s}$ (the solid line) at the varying of the values of the variable $X_{s\to t}$ (x-axis).

The propagation of the `orient` constraint is exploited during the solution search to enforce property (5) and implemented by the rule:

$$\texttt{orient}(X_{s\to t}, X_{t\to s}): \quad \frac{v = d(X_{s\to t}),\ l = \min(v, 1-v)}{D_{t\to s} = D_{t\to s} \cap [0, l)} \tag{6}$$

3.3.3 Redundant edge constraints

Several state-of-the-art inference methods rely on mutual information or correlation techniques. The community approach adopted in this work employs methods that use both correlation and mutual information as principal components for the inference process. One of the disadvantages of such methods is the difficulty in speculating on the directionality of a given prediction. We define a constraint that can aid in detecting the edge directionality based on the collective decisions of the CN predictions, among the non MI- or correlation-based methods.

Let us consider a collection of predictions $\mathcal{P} = \{P_1, \ldots, P_J\}$ for a GRN $G = (V, E)$, and a non-empty set of predictions $\mathcal{H} \subseteq \mathcal{P}$ derived from MI-based or correlation-based methods. An edge (t, s) is said to be *redundant* if:

$$\forall\, P_i \in \mathcal{P} \setminus \mathcal{H}\,. \quad \omega_i(s,t) > \omega_i(t,s) + \beta_i \tag{7}$$

where $\omega_i(s,t) : V \times V \to [0,1]$ expresses the confidence value of the edge (s,t) in the prediction P_i, and $\beta_i \in \mathbb{R}$ is a real value associated to each prediction method in $\mathcal{H}$. Similarly, we define a variable $X_{s\to t}$ to be redundant if the corresponding edge (s,t) is redundant. We use the proposition `red-e`(t,s) to denote a redundant edge (t,s), and given a redundant edge (t,s) we call the edge (s,t) the *required* edge. The `redundant_edge` constraint

enforces a relation between two variables $X_{s\to t}$ and $X_{t\to s}$ by imposing an edge orientation constraint on the redundant variable and the required variable. Let X_R be the set of all the required and redundant variables.[4] For a pair of variables $X_{s\to t}, X_{t\to s} \in X_R$ we express a redundant edge constraint as:

$$\texttt{redundant}(X_{s\to t}, X_{t\to s}) \equiv \texttt{red-e}(t,s) \to \texttt{orient}(X_{s\to t}, X_{t\to s}).$$

which naturally translates to the propagation rule:

$$\texttt{redundant}(X_{s\to t}, X_{t\to s}): \quad \frac{v = d(X_{s\to t}),\ l = \min(v, 1-v),\ \texttt{red-e}(t,s)}{D_{t\to s} = D_{t\to s} \cap [0, l)} \tag{8}$$

3.3.4 Transcription Factor constraint

GRN-specific information, e.g., sequence DNA-binding TFs or functional activity of a set of genes, is often available from public sources (e.g., DBD [32] or Gene Ontology [29]). Moreover, several studies show that similar mRNA expression profiles are likely to be regulated via the same mechanisms [1]. On the other hand, not every method may be designed to handle such information, or this information may be only partially available, and hence not suitably usable by prediction methods. We propose constraints to incorporate such information in the CN model.

The property that a TF regulates the production of other genes is described by a condition on the out-degree of the involved gene—for those edges with an adequate confidence value. The `t-factor` constraint over a gene s requires the condition: $\texttt{atleast\_k\_ge}(k, \mathbf{X}_s, \theta)$, with $\mathbf{X}_s = \{X_{s\to u} \in \mathbf{X} \mid u \in V\}$, and k represents the co-expression degree, i.e., the number of genes targeted by the TF. In addition, we impose a directionality constraint between each variable $X_{s\to u}$ and $X_{u\to s}$ whenever u has not been identified as a TF:

$$\texttt{t-factor}(s,k,\theta) \equiv \quad \texttt{atleast\_k\_ge}(k, \mathbf{X}_s, \theta) \ \cap \bigcap_{u \in X_s \setminus \mathbf{TF}} \texttt{orient}(X_{s\to u}, X_{u\to s}) \tag{9}$$

where $\mathbf{TF}$ is the set of all the putative transcription factors for the predicted GRN.

3.3.5 Co-regulator constraint

Multiple TFs can cooperate to regulate the transcription of specific genes; these are referred to as *co-regulators*. When this information is available, it can be expressed by a `co-reg` constraint. This constraint involves two TFs, s' and s''. The constraint enforces a relation on a set of variables $\mathbf{X_S}$, to guarantee the existence of at least k elements that are co-regulated by both s' and s'', for which an interaction is predicted with confidence values greater than θ $(0 < \theta \leq 1)$.

Given two distinct TFs s', s'' and a threshold θ, the set of all elements co-regulated by both s' and s'' is defined as follows:

$$\mathbf{CR}_{\langle s', s'', \theta\rangle} = \{t \mid t \in V, t \neq s', t \neq s'', d(X_{s'\to t}) > \theta, d(X_{s''\to t}) > \theta\}.$$

As for the `t-factor` constraint, a directionality constraint is imposed between the transcription regulators s' and s'' and each of their targets $u \in \mathbf{X_S} \setminus \mathbf{TF}$. Given a co-regulation degree k, a real value $\theta \in (0, 1]$ and a set of variables $\mathbf{X_S} = \{X_{s'\to t}, X_{s''\to t}\}$, for some $s', s'', t \in V$ different among each other and such that whenever both $X_{s'\to t}$ and $X_{s''\to t}$ are in $\mathbf{X_S}$, then $X_{t\to u} \notin \mathbf{X_S}$, for all $u \in V$, the co-regulator constraint is expressed by:

$$\begin{aligned}
&\texttt{co-reg}(s', s'', k, \mathbf{X_S}, \theta) \equiv \\
&\quad |\{\langle X_{s'\to t}, X_{s''\to t}\rangle \mid X_{s'\to t} \in \mathbf{X_S}, X_{s''\to t} \in \mathbf{X_S}, t \in \mathbf{CR}_{\langle s', s'', \theta\rangle}\}| > k \ \cap && (10)\\
&\quad \bigcap_{u \in X_s \setminus \mathbf{TF}} \texttt{orient}(X_{s'\to u}, X_{u\to s'}) \cap \bigcap_{u \in X_s \setminus \mathbf{TF}} \texttt{orient}(X_{s''\to u}, X_{u\to s''}) && (11)
\end{aligned}$$

We call the (10) $\texttt{coregulation}(s', s'', k, \mathbf{X_S}, \theta)$ and its associated propagation rule is described by the following:

$$\frac{T = |\{(s', s'', t) \mid \max(D_{s'\to t}) \leq \theta \wedge \max(D_{s''\to t}) \leq \theta\}|,\ |\mathbf{S} \setminus \mathbf{T}| = k}{\bigwedge_{X_i \in \mathbf{S}\setminus\mathbf{T}} D_i = D_i \cap (\theta, 1]} \tag{12}$$

[4] $X_{s\to t}$ is required/redundant if the corresponding edge (s,t) is required/redundant.

Expressing biological hypotheses and network properties as constraints may assist the phase of experimental design for GRN inference. The solver verifies the existence of a set of solutions consistent with the hypotheses, and its size can be related to confidence strength of the answer w.r.t. the collective prediction decisions. Consider the case where a gene is inaccurately identified as a transcription factor, or if none of its targets is identified among its putative target genes X_s. In such a case, the CSP will return no valid models.[5] In order to revise the model and generate valid solutions, a relaxation of the `tf` constraint is required. This can be achieved by either removing such constraint from the model or by changing the putative target set X_s. In the case where X_s includes the entire set of genes in the network, and the model is still unsatisfiable, then, it can be claimed that, w.r.t. the knowledge leveraged by the CN prediction ensemble, such transcription factor has no direct effects on the genes analyzed, and the associated `tf` constraint can be removed. When performing inference on gene expression data, analysis of gene sub-networks may hide known global properties. It is crucial to test the biological value of a model prior generating the GRN prediction.

3.4 Solution Search

The proposed modeling of GRN prediction allows a significant degree of flexibility in exploring the solution space. We implement an incomplete search strategy, that explores the prop-labeling tree (i.e., the search space of assignments to the variables—see Sect. 2.1) making use of *Monte Carlo (MC)* methods, via a left-most (fixed) variable selection strategy or prioritizing the variables $X_{s\to t}$, with $s \in \mathbf{TF}$, when the **TF** set is known. We visit the prop-labeling-tree executing a random choice (i.e., a random value selection) when non-deterministic choices occur. After every non-deterministic choice is done, the propagation rules described earlier are applied to possibly reduce the non-determinism of unlabeled variables. The search stops after a given number of trials or a when a given number solutions have been found.

3.5 GRN Consensus

A challenge in GRN inference is the absence of a widely accepted objective function to drive the solution search. We decided to generate an ensemble of m solutions and propose three criteria to compute the final GRN prediction. Given a set of m solutions $\mathbf{S} = \{S_1, \ldots, S_m\}$, where each $S_i = \langle a_1^i, \ldots, a_{n^2-n}^i \rangle$, let $\mathbf{S}|_{X_k} = \bigcup_{i=1}^m \{a_k^i\}$ be the set of values assigned to the variable X_k[6] in the different solutions, and $\texttt{freq}(S, a, k)$ be the function counting the occurrences of the value a among the assignments to X_k in the solution set S. The consensus value a_k^* associated with the variable x_k is computed by:

- *Mode:* $a_k^* = \arg\max_{a \in \mathbf{S}|_{X_k}} (\texttt{freq}(S, a, k))$. This estimator rewards the edge confidence value appearing with the highest frequency in the solution set. The intuition is that edge-specific confidence values appearing in many solutions may be important for the satisfaction of the constraints.
- *Average:* $a_k^* = \frac{1}{m} \sum_{i=1}^m a_k^i$. It computes the average edge consensus among all solution in order to capture recurring predictive trends.
- Hamming distance: $a_k^* = a_k^h$, where: $h = \min_i \sum_j HD(S_i, S_j)$, $HD(S_i, S_j) = \sum_k |l(X_k^i) - l(X_k^j)|$, and $l(X_k^i)$ is the position of the value a_k^i in the domain D_k, whose elements are listed in increasing order. This measure is a *global* measure, that acts collectively on the prediction values of all edges, returning the solution which minimizes the Hamming distance among all edge prediction values.

3.6 A Case Study

We provide an example to illustrate our approach. We extract a subnetwork of ten nodes from the E. coli regulatory network (Fig. 4 (a)) and simulate its dynamics using GeneNetWeaver (GNW) [44]—a standard software for GRN inference evaluation. The target network has two transcription factors (*leuO* and *bglJ*) which are in turn co-regulators for genes *bglG, bglF, bglB*, and it has 11 interactions.

Phase 1: CN Predictions. The prediction methods employed to construct the final community network are: **(i)** BC3NET, **(ii)** CLR, **(iii)** GENIE3, **(vi)** GLN, **(v)** Inferelator, **(vi)** Pearson Correlation, and **(vii)** Tigress. The prediction ensemble is obtained by feeding a multi-factorial expression dataset composed of 10 measurements to each of the methods aforementioned. The data is generated via GNW. In addition, we generate a Community

[5] Accordingly to the interaction patterns detected by the prediction methods employed in the committee.
[6] For readability we write here X_k rather than $X_{i\to j}$.

Network prediction, CN_{rank} (Fig. 4 (b)), by averaging the ranks obtained within each individual prediction, as done in [36], and use it as baseline to build the domain variables (see Algorithm 1) and for evaluation.

Phase 2: Modeling the CSP. The execution of Algorithm 1 for the prediction disagreements analysis reduces the initial domain sizes to 1 for 21 cases, and to 3 for the others. We automatically tuned the parameters of the algorithm, as described in Sect. 4.3.1.

A `sparsity` constraint is imposed at the global level:

$$\texttt{atleast\_k\_ge}(k_l, \mathbf{X}, \theta_l) \cap \texttt{atmost\_k\_ge}(k_m, \mathbf{X}, \theta_m) \qquad (\texttt{sp})$$

where $\mathbf{X}$ are the variables describing all possible interactions of the network.

As the inference ensemble adopted employs methods that may suffer from the *edge redundancy* problem, we impose a `redundant` constraint for all the edge pairs $(s,t),(t,s)$ that satisfy the redundant property (see (7)) as:

$$\texttt{red-e}(t,s) \rightarrow \texttt{orient}(X_{s\rightarrow t}, X_{t\rightarrow s}). \qquad (\texttt{re})$$

This constraint is able to reduce the value uncertainty for 12 additional variables—only one element in their domains can possibly satisfy the conditions above for any value choice of the required edge variable.

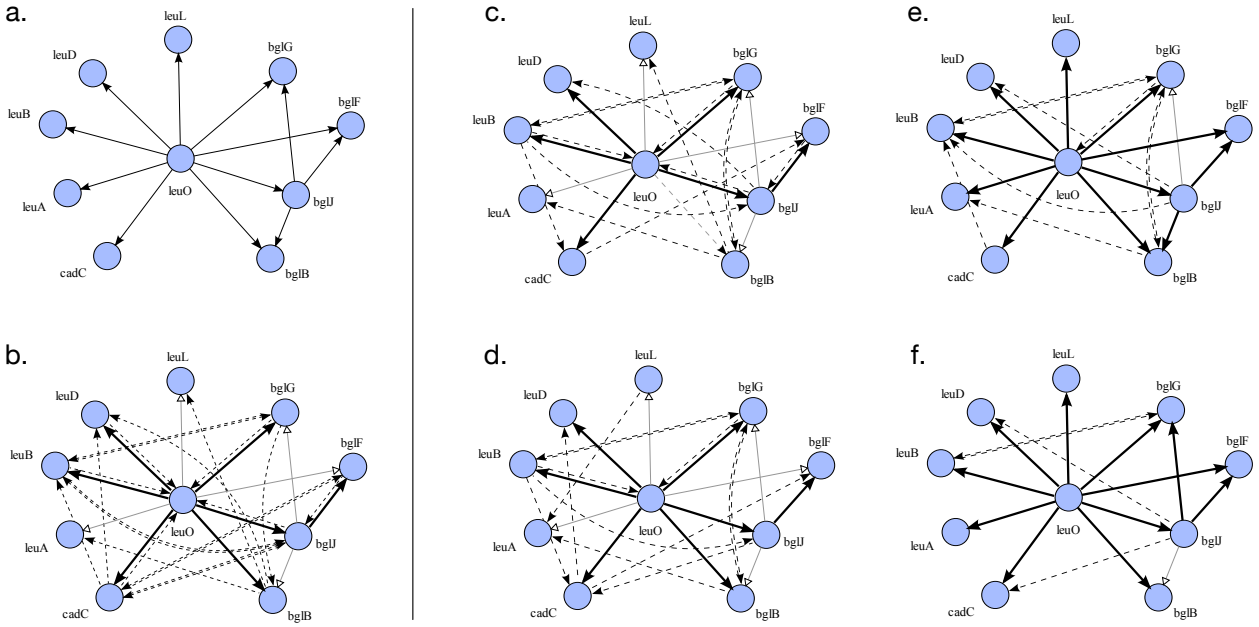

Figure 4: *An extract of* 10 *node E.coli GRN (a) (from [25]) and its the* CN_{rank} *consensus prediction (b). The CCN predictions after the integration of the* `sparsity` *constraint (c), the* `redundant` *constraint (d), the* `t-factor` *constraints (e) and* `co-reg` *(f).*

Phase 3: Generating the Consensus. We perform 1,000 Monte Carlo trials producing a set of solutions which we refer to as *Constrained Community Networks (CCNs).*

To illustrate the effects of constraints integration on the CCNs, we consider the best prediction returned by each CSP exhibiting a different combinations of the imposed constraints. We plot it as a graph containing all and only the edges of highest confidence necessary to make such graph weakly connected. These resulting predictions are illustrated in Fig. 4 (c–d), together with the CN_{rank} (b). In each network, the thick edges denote the true positive predictions, the dotted edges denote the false positive predictions, and the gray edges with white arrows denote the false negatives. The results are also summarized in Table 2, where we report the AUROC and AUPR scores [7] for the best prediction (CCN_{best}) generated and for each CCN generated by the evaluation criteria presented in in Sect. 3.4: Mode (CCN_{mode}), Average (CCN_{avg}) and Hamming distance (CCN_{hd}).[7]

[7] AUROC and AUPR are popular measures from the machine learning literature—see Sect. 4.2.

Phase 4: Employing network specific information. Let us now model some specific information about the target network. The target network includes two TFs: *leuO* and *bglJ*, which can be modeled via two `t-factor` constraints as:

$$\texttt{atleast\_k\_ge}(k, \mathbf{X}_{leuO}, \theta) \ \cap \bigcap_{u \in X_{leuO} \setminus \{leuO, bglJ\}} \texttt{orient}(X_{leuO \to u}, X_{u \to leuO}),$$

$$\texttt{atleast\_k\_ge}(k, \mathbf{X}_{bglJ}, \theta) \ \cap \bigcap_{u \in X_{bglJ} \setminus \{leuO, bglJ\}} \texttt{orient}(X_{bglJ \to u}, X_{u \to bglJ}), \qquad (\texttt{tf})$$

with $\mathbf{X}_i = \{X_{i \to j} \in \mathbf{X} \mid j \in V\}$ for $i = leuO, bglK$, and the parameters k and θ set as described in Sect. 4.3.1. Fig. 4 and Table 2 show the improvements using this final formalization. Finally, speculation about the activity of genes *leuO* and *bglJ* as co-regulators can be captured via a `co-reg` constraint expressed by:

$$\begin{aligned} \texttt{coregulation}(leuO, bglJ, k, \mathbf{X_S}, \theta) \cap & \bigcap_{u \in X_s \setminus \{leuO, bglJ\}} \texttt{orient}(X_{leuO \to u}, X_{u \to leuO}) \cap \\ & \bigcap_{u \in X_s \setminus \{leuO, bglJ\}} \texttt{orient}(X_{bglJ \to u}, X_{u \to bglJ}), \qquad (\texttt{cr}) \end{aligned}$$

where $\mathbf{X}_S = \{X_{s \to t} \mid s \in \{leuO, bglJ\}, t \in V\}$ is the set of all candidate regulations having *leuO* or *bglJ* as TF, and k and θ set as described in Sect. 4.3.1. The application of this additional constraint produces further improvements (Fig. 4 and Table 2).

CCNs	CN_{rank}		CCN_{best}		CCN_{avg}		CCN_{mode}		CCN_{hd}	
	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR	AUROC	AUPR
`sp`	0.716	0.302	0.831	0.4	0.789	0.328	0.671	0.283	0.745	0.291
+`re`	0.716	0.302	0.855	0.396	0.814	0.341	0.73	0.304	0.773	0.326
+`tf`	0.716	0.302	0.942	0.551	0.848	0.415	0.777	0.366	0.824	0.421
+`cr`	0.716	0.302	0.95	0.659	0.902	0.498	0.777	0.366	0.781	0.299

Table 2: *The prediction accuracy of the* CN_{rank} *and CCNs on the* 10*-node "E.coli" GRN.*

4 Performance evaluation

We systematically assessed the ability of the Constrained Community Network schema to accurately reconstruct GRNs both in an ab-initio scenario (only datasets information available) and in presence of prior information in biologically relevant settings. We started by conducting experiments to select a subset of methods to be used in the committee schema to construct a Community Network. We assess the power of each constraint individually and how the constraint interaction affects the performance predictions. Finally, we evaluate the performance of the CCNs predictions against the CN schema. Performance is validated against the set of gold standard GRNs.

4.1 Benchmark networks & datasets

The benchmark networks adopted to asses the performance of our method were produced extracting sub-networks from the *Escherichia coli* [2, 25] and *Saccharomyces cerevisiae* [28, 42] regulatory networks. The datasets used to simulate the dynamics of such networks were generated in GeneNetWeaver [44], a tool commonly adopted for the generation of synthetic GRN benchmarks, and used to generate the synthetic datasets for the DREAM3 [41] and DREAM4 [27] competitions.

We adopt two type of steady-state expression data[8]:
(1) *Multifactorial*: these are measurements obtained by (slightly) perturbing all genes simultaneously. Multifactorial data might correspond, for example, to expression profiles obtained from different patients or biological

[8]Experiments where the mRNA expressions are observed once when at a steady state.

replicates. Such type of data is simpler and less expensive to obtain than other types of data, such as knockout/knockdown or time series data, and is thus more common in practice; however, it is also less informative for the prediction of edge directionality [8] and therefore makes the regulatory network inference task more challenging.
(2) *Knockout*: these are steady-state levels of a single-gene knockout (deletion). The datasets are built by performing an independent knockout for a subset of genes. A knockout experiment is simulated by setting the gene's transcription rate to zero.

The complete benchmark network set adopted in this work is composed of 20 large GRNs, each described by 3 datasets: one containing multi-factorial data, one containing knockout data, and another one containing both data types. We will refer to these three datasets as *mf*, *ko*, and *all*, respectively. The benchmark set is composed of:
(1) *FDP Networks:* 5 networks of sizes 100 extracted from the *E. coli* and other 5 networks of the same size from the *S. cervise* regulatory networks, denoted respectively by $\texttt{E}_i$ and $\texttt{S}_i$ with $i = 1, \ldots, 5$. The associated datasets are generated via GNW, by setting the value of *Seed* to random and the *Neighbor selection* to 20%. The model was generated by producing 100 microarray datasets, enabling both the ODE and the *Stochastic Differential Equation* options, with the SDE value equal to 0.05. The noise level was set to simulate the microarray standard noise with default parameters.
(2) *D3:* 5 networks from the DREAM3 competition [41], denoted by $\texttt{D3}_1, \ldots, \texttt{D3}_5$ and consisting of 100 genes, built by extracting modules from the *E. coli* and from a yeast genetic interaction network [42]. Data normalization was made by the competition organizers.
(3) *D4:* 5 GRNs from the DREAM4 competition [27], denoted by $\texttt{D4}_1, \ldots, \texttt{D4}_5$ and consisting of 100 genes. The network topologies have been extracted from the transcriptional regulatory networks of *E. coli* and *S. cerevisiae*. The data corresponds to noisy measurements of mRNA levels based on SDEs (Langevin equations) and has been normalized to values in $[0, 1]$.

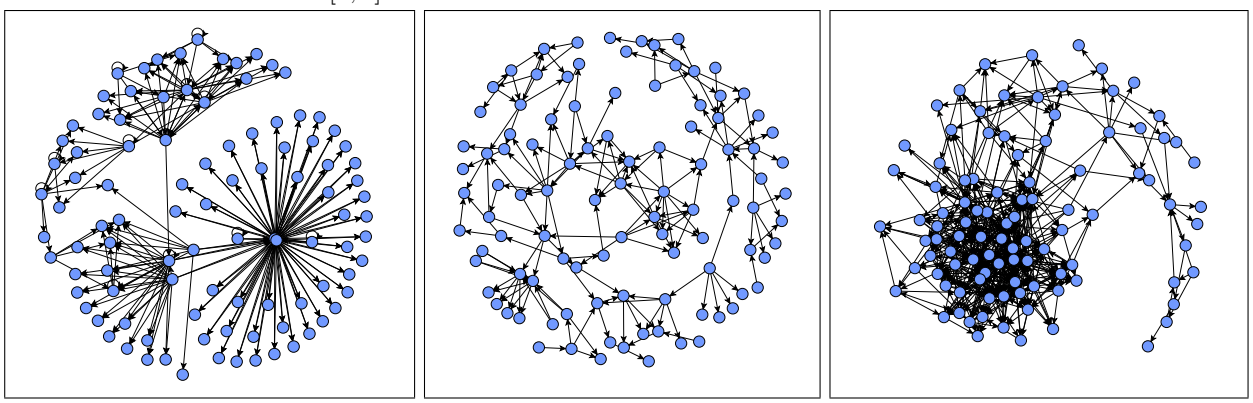

Figure 5: *Gene regulatory networks with IDs* $\texttt{E}_2$ *(left),* $\texttt{D3}_3$ *(center) and* $\texttt{D3}_5$ *(right).*

The GRNs adopted in this study are representative of a diverse range of heterogeneous network topologies, varying properties such as sparsity, number of hubs and local connectivity—as illustrated in Fig. 5. A summary of the network topologies used in the performance assessment is given in Table 3, where we report the network ID, the transcription regulatory network from where the network has been extracted, the total number of gene-gene interactions (edges) and the number of bidirectional regulatory interactions (in parenthesis), the number of TFs, and the number of network hubs—defined as the number of nodes whose out-degree exceeds the average TFs out-degree for the same GRN.

We adopt the datasets associated to the *FDP* GRNs to asses the individual methods prediction performance and to train the construction of the community network prediction under the *Borda* rank election schema (CN_{rank}). This results in 4,320 individual predictions and 360 CN predictions. We use the complete collection of datasets to assess the performances of the CCNs against the CN predictions.

4.2 Performance assessment

To measure prediction accuracy we evaluate each prediction as a binary classification task—where interactions are predicted as either being present or absent. The ranked list of interactions is compared against a gold

ID	GRN	Edges	TFs	Hubs	ID	GRN	Edges	TFs	Hubs
E_1	E. coli (*)	148 (4)	17	8	S_1	S. cerevisiae (**)	174 (0)	17	9
E_2	E. coli (*)	151 (6)	19	7	S_2	S. cerevisiae (**)	205 (0)	18	9
E_3	E. coli (*)	218 (6)	18	7	S_3	S. cerevisiae (**)	207 (0)	18	6
E_4	E.coli (*)	159 (4)	17	7	S_4	S. cerevisiae (**)	168 (0)	18	9
E_5	E. coli (*)	171 (10)	20	5	S_5	S. cerevisiae (**)	202 (0)	20	7
$D3_1$	E. coli (***)	125 (0)	26	11	$D4_1$	Synthetic (*,**)	176 (14)	40	8
$D3_2$	E. coli (***)	119 (0)	19	7	$D4_2$	Synthetic (*,**)	249 (14)	35	7
$D3_3$	Yeast (****)	166 (0)	59	26	$D4_3$	Synthetic (*,**)	195 (6)	44	17
$D3_4$	Yeast (****)	389 (0)	70	25	$D4_4$	Synthetic (*,**)	211 (8)	40	14
$D3_5$	Yeast (****)	551 (0)	80	30	$D4_5$	Synthetic (*,**)	193 (4)	34	15

Table 3: *Properties of the benchmark network topologies. (*) [25], (**) [6], (***) [46], (****) [42].*

standard via two measures largely adopted in machine learning: the *area under the precision vs. recall curve* (AUPR) and the *area under the receiver operating characteristic curve* (AUROC)—true positive rate vs. false positive rate [7]. To compute the AUROC and the AUPR curves, we express the measures of *true positive rate* (TPR), *false positive rate* (FPR), *precision* and *recall* as functions of a cutoff threshold (k), which denotes the number of ranked edges to be considered. Let us denote with P the number of interactions in the gold standard, with N the number of negatives (absent interactions), and with $T = P + N$ the total number of putative edges.

The true/false positive rates and precision and recall are defined as follows:

$$TPR(k) = \frac{TP(k)}{P} \qquad FPR(k) = \frac{FP(k)}{N} \qquad precision(k) = \frac{TP(k)}{TP(k)+FP(k)} \qquad recall(k) = \frac{TP(k)}{P}$$

where $TP(k)$ ($FP(k)$) is the number of true correct (incorrect) predictions among the first k elements in the interaction list. An AUROC value of 0.5 (1.0) corresponds to a random (perfect) prediction. AUPR values close to 0 indicate the predominance of erroneous predictions, while a value of 1.0 denotes a prediction with no errors.

4.3 Settings

We performed the various experiments using the R language and a generic CSP solver, capable of handling real valued variables and of exploring the search space using a prop-labeling tree with random value selection (Sect. 3.4). We use R to generate the GRN predictions for each method presented in Sect. 2.2 and to assess their individual performance and the performance as committees in ranked-based community networks. The parameters associated to each prediction method have been set to the default values, and the predictions generated are fed to the constraint solver. The constraint solver generates the constrained community networks. Our CSP solver explores the queue of constraints using techniques based on the notion of event (a change in the domain of a variable) [45] and is implemented in C++.

For each experiment, we perform 10,000 Monte Carlo trials and generate the CCNs using all of the solutions found. We generate four CCN consensus solutions, one for each estimator described in Sect. 3.4 (CCN_{mode}, CCN_{avg}, CCN_{hd}) and CCN_{best}, as the best prediction with respect to the AUROC score. We notice that the CCN produced via the mode and avg estimators may outperform the CCN_{best}, as they generate a new solution, starting from those found during the search phase, which may not be part of such ensemble. All the experiments have been performed on an *Intel Core i7 3770* machine, 3.4GHz with 16 GB of RAM, equipped with the SuSE Linux operative system.

4.3.1 Automatic parameter tuning

Let us now discuss the algorithms and constraints parameters tuning adopted by default by the system. In what follow, we will assume that the edge predictions in CN_{rank} are sorted from the most likely one to the least likely one, and we will refer to $\text{CN}_{\text{rank}}[i]$ as the confidence value assigned to the highest i^{th} edge prediction in CN_{rank}. We will denote by n and $e = n^2 - n$ the size of a GRN and the number of putative regulators, respectively.

Domain construction: The domains analysis and reduction phase, described in Algorithm 1 is used to assess the community prediction disagreements on a given edge. Recall that the bounds L and U are used to discriminate whether the confidence for the presence/absence of an interaction is strong within the community network

prediction. Their values are selected to be, respectively, the highest confidence value in the last $\mathrm{CN}_{\mathrm{rank}}$ decile and the lowest confidence value in the first $\mathrm{CN}_{\mathrm{rank}}$ decile. The disagreement threshold θ_d is set to the average of the discrepancy values `w_d` across all the edges of the network. A summary of the parameters values is reported below:

$$L = \mathrm{CN}_{\mathrm{rank}}[0.9\,e] \qquad U = \mathrm{CN}_{\mathrm{rank}}[n] \qquad \theta_d = \frac{1}{|E|} \sum_{(s,t)\in E} \Big(\frac{1}{\binom{J}{2}} \sum_{j=1}^{J} \sum_{i=j+1}^{J} |\omega_j^{\#}(s,t) - \omega_i^{\#}(s,t)| \Big)$$

We also tried to populate domains in a non symmetric way, e.g., by looking at how many methods lie inside which quantile and by assigning a value representative of each quantile. This method led to worst results w.r.t. the one adopted in our experiments.

Sparsity constraint: To guide the parameter selection for the `sparsity` constraint, we set the values k_l and k_m (see Eq. (4)) to be, respectively, n and $n\,\log(n)$, since the number of edges in a sparse network is considered to be $O(n\,\log(n))$.

To identify the thresholds θ_l and θ_m, we select for each gene the minimum (resp. maximum) weight $\bar{w}$, such that the number of outgoing edges in the $\mathrm{CN}_{\mathrm{rank}}$ predicted with confidence greater than $\bar{w}$ is greater than (resp. less than or equal to) 1 (resp. $\log(n)$) and average these values. By doing so, we try to impose a restrictive condition for the satisfaction of the `atleast_k_ge` and `atmost_k_ge` constraints. The `sparsity` constraint values are summarized next:

$$\begin{array}{l} k_l = n \qquad \theta_l = \frac{1}{n}\sum_{s\in V} \theta_l^s \qquad k_m = n\,\log(n) \qquad \theta_m = \frac{1}{n}\sum_{s\in V} \theta_m^s \\ \theta_l^s = \min_{\bar{w}} |\{(s,t,w) \in \mathrm{CN}_{\mathrm{rank}} \mid t \in V, w \geq \bar{w}\}| > 1 \\ \theta_m^s = \max_{\bar{w}} |\{(s,t,w) \in \mathrm{CN}_{\mathrm{rank}} \mid t \in V, w \geq \bar{w}\}| \leq \log(n) \end{array}$$

Redundant edge constraint: The parameters β_i introduced in Eq. (7) for the redundant edge definition are set to the mean of all the differences of the confidence values of the pairs of edges (s,t) and (t,s) in each CN ensemble prediction $P_i \in \mathcal{P} \setminus \mathcal{H}$ (see Eq. (7)):

$$\beta_i = \frac{1}{|E|} \sum_{(s,t)\in E} (w_{s\to t}^i - w_{t\to s}^i),$$

where $w_{s\to t}^i$ is the confidence value associated to the edge (s,t) reported by P_i.

We also tried to learn the values of the β_i of Eq. 7 using a set of 10 training networks. In our study we compare the confidence values assigned to the true edges $w_{s\to t}$—present in the gold standard—against the confidence assigned to the opposite edges $w_{t\to s}$—true negatives—for each individual prediction P_i in $\mathcal{P} \setminus \mathcal{H}$ and estimate the $\beta_i \geq 0$ solving the following linear program:

$$\begin{array}{rl} \textbf{maximize} & |\{(w_{s\to t}^i - w_{t\to s}^i) > \beta_i \mid (s,t) \in E_{GS} \wedge (t,s) \notin E_{GS}\}| \\ \text{subjected to :} & |\{(w_{s\to t}^i - w_{t\to s}^i) > \beta_i \mid (s,t) \notin E_{GS} \wedge (t,s) \in E_{GS}\}| = 0, \end{array}$$

where E_{GS} is the set of edges in the gold standard. Applying the determined β_i to Eq. (7) in our experiments resulted in almost all cases in detecting no redundant edges.

Transcription factor constraint: The `atleast_k_ge` constraint parameters k and θ employed to express a `t-factor` constraint are automatically set so that $k = \log(n)$ and $\theta = \mathrm{CN}_{\mathrm{rank}}[k]$. To guarantee the constraint satisfaction, we add the value θ in the domains of those variables X_i involved in the `tf` constraint, having $\max(D_i) < \theta$.

Co-regulator constraint: The values for the `co-reg` constraint parameters k and θ (see Eq. (7)) are the same as those used in the `tf` constraint. Also in this case, we expand the domain of the variables involved as described in the previous paragraph.

To guarantee the satisfaction of the constraints $\texttt{orient}(X_{s\to t}, X_{t\to s})$, we introduce a value 0 in the domains of the variables $X_{t\to s}$ if $\min(D_{t\to s}) \geq \max(D_{s\to t})$.

4.4 Analysis of individual methods and Community Network construction

To construct the CN schema, we evaluate each individual prediction over the complete collection of datasets associated to the *FDP* networks. For each dataset (*mf*, *ko* and *all*) we assess the performance of the prediction

methods from Sect. 2.2, by averaging the AUROCs and AUPRs values associated to each prediction of the *FDP* benchmark networks. In Table 4, we report, for each dataset, the list of the methods sorted according to the quality of their performance. The final community network schema results in a selection of 7 of the 12 GRN prediction methods. We consider both the methods *GENIE3* and *Tigress* because of their different approaches in performing the feature selection step: *GENIE3* relies on Random Forests while *Tigress* uses a regression step. We also discriminate the method BC3NET from the other MI-based models, as it employs an additional bagging step. For each data type (*mf*, *ko* and *all*), the final methods ensemble selected to form the community network includes: *Tigress, GENIE3, Pearson Correlation, GLN, BC3Net, Inferelator* and *CLR*—these appear underlined in Table 4. All other methods fall in a category already represented in the ensemble. A report of the performance of the individual methods is provided in the Appendix, in Tables I–IV.

	***mf* data**			***ko* data**			***all* data**		
#	**met**	**AUROC**	**AUPR**	**met**	**AUROC**	**AUPR**	**met**	**AUROC**	**AUPR**
1	tigr	0.710	0.141	tigr	0.797	0.109	tigr	0.789	0.164
2	geni	0.725	0.120	geni	0.770	0.070	geni	0.802	0.143
3	clr	0.690	0.107	pear	0.770	0.069	pear	0.754	0.089
4	pear	0.694	0.097	gln	0.693	0.057	infe	0.707	0.103
5	mrne	0.688	0.102	bc3n	0.627	0.063	bc3n	0.653	0.112
6	kend	0.684	0.097	infe	0.672	0.045	gln	0.667	0.072
7	bc3n	0.636	0.112	clr	0.557	0.024	clr	0.659	0.077
8	spea	0.683	0.094	spea	0.553	0.024	spea	0.657	0.067
9	infe	0.639	0.091	kend	0.553	0.024	kend	0.657	0.067
10	gln	0.653	0.082	mrne	0.553	0.024	mrne	0.655	0.067
11	arac	0.597	0.086	arac	0.532	0.026	arac	0.574	0.058
12	c3ne	0.589	0.085	c3ne	0.520	0.027	c3ne	0.564	0.056

Table 4: *Ranked method lists for the multi-factorial dataset (left) the knockout dataset (center) and the combined dataset (right).*

4.5 Analysis of individual constraint in the CCN

In this section, we analyze the construction of the CCNs by measuring the impact of the application of individual constraints on the quality of the resulting solution ensemble. The tests are performed over the datasets associated to the *FDP* networks and, for each experiment, we report the median predictions found with respect to the AUROC score. Fig. 6 illustrates the impact of each constraint on the AUROC score for each network of the *FDP* multi-factorial dataset. The plot reports the median of a set of 10,000 solutions associated to the CCNs generated exploiting only the `redundant` constraint (`re`), the `sparsity` constraint (`sp`), the `t-factor` constraint (`tf`), and the `co-reg` constraint (`cr`), together with the trace of all the solutions generated via an unconstrained problem with same settings (grey stripe). The results for the datasets based on *ko* and *all* data follow the same trend.

We observe that the median values of the solutions generated using the `sparsity` constraint are close to the best solutions generated in the unconstrained problem; this confirms the effectiveness of the constraint. The biological information encoded by the `t-factor` and `co-reg` constraints enhances the quality of the solutions beyond the capabilities of the unconstrained search. As for the `redundant` constraint, the median solutions returned are slightly better that the median solutions found by the unconstrained problem. To validate the effectiveness of such constraint, we examine the number of redundant edges correctly identified via the `red-e` property (`tf`→`tg` and `tg`→`tg`)—where `tf` denotes a TF gene and `tg` a gene targeted by some TF, but not itself a TF—and the number of edges wrongly predicted as redundant (`tf`→`tf` and `tg`→`tf`). In Table 7, we report the extended results for the multi-factorial dataset (top) and we summarize the results for all the 3 datasets in the bottom part of the table. Observe that the constraint is more effective in the *mf* dataset, where fewer errors occur.

Let us analyze the impact of combining two constraints. We adopt the same settings as in the previous experiments. The results are reported in Fig. 8. The plots illustrate the median of a set of 10,000 solutions associated to the CCNs generated via a combination of the `sparsity` constraint (top-left), the `redundant` constraint (top-right), the `t-factor` constraint (bottom-left), and the `co-reg` constraint (bottom-right) with all the others. As for Fig. 6, we mark with a grey stripe the scores for the solutions generated by the unconstrained CCNs. The knockout and the combined datasets follow the same trend as those in Fig. 8. Observe that the

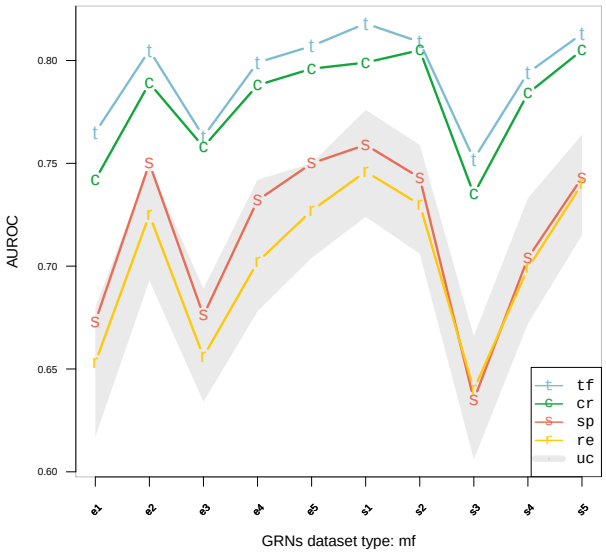

Figure 6: *The impact of each constraint on the AUROC score for the multi-factorial dataset in the FDP networks.*

interaction among constraints is important to improve the quality of the solutions. An evaluation of the impact of each constraint combination on the FDP networks is reported in the Appendix, Fig. I(a)–(m).

4.6 Constrained Community Networks vs Community Networks

To assess the ability of the CCN approach to accurately reconstruct GRNs, we focus on two subproblems: **(1)** We examine the predicted CCNs using only general network topological information, such as the `sparsity` and the `redundant` constraints, to leverage community-method features and networks properties; **(2)** We integrate network specific biological knowledge available. Due to the observations made in the previous section, we opt not to involve the `redundant` constraint for the datasets containing knockouts data. We categorize the benchmarks by datasets (*mf*, *ko* and *all*) and networks (*FDP*, *D3* and *D4*), and average their respective AUROC and AUPR scores. Tables 5, 6 and 7 report the average AUROC and AUPR improvements (as a percentage) for the networks inferred from the *mf*, *ko* and *all* datasets, respectively, of the CCN_{best}, CCN_{avg}, CCN_{hd} and CCN_{mode} with respect to CN_{rank}. We first focus on the results of the CCNs obtained when only the `sparsity` (`sp`) and the `redundant` (`re`) constraints are active, given the restrictions aforementioned (first row of each Table).

For the *mf* datasets (Table 5), the CCNs outperform the community network schema in every benchmark network. For the datasets that include knockout data (*ko* and *all* in Tables 6 and 7, respectively), the CCNs increase the prediction accuracy with respect to the CNs for the the *FDP* and the *D4* networks; the only exception is the `mode` estimator, which produces a slight performance degradation in the *ko* dataset. The AUROC measures for the *D3* networks result in a performance degradation ranging from 0.52% to 0.78%. On the other hand, the precision vs. recall score produces an enhancement of the prediction accuracy up to 2.8% for the `best` estimator.

As showed in [24], the prediction accuracy for the CCNs is consistent and often better than that of a CN schema in the ab-initio scenario.

It is interesting to observe that the CCNs consistently outperform the CNs prediction in the multi-factorial datasets. This result is appealing, as such type of data has been shown to be less informative than knockout or time-series data, while it is substantially cheaper to produce and more abundant.

The next experiment extends the set of constraints adopted to model the GRNs to include specific knowledge about individual networks. We enable the `t-factor` (`tf`) and the `co-reg` constraints (`cr`) over the set of genes which are known to be TFs or co-regulators in the target networks. The results are reported in Tables 5, 6 and 7 in second and third row for the addition of the `tf` and `cr` constraints, respectively. The integration of such additional knowledge results in significant improvements of the GRN predictions, both in terms of AUROC (up

Net	Data	tf→tg	tf→tf	tg→tf	tg→tg
E_1	*mf*	4	0	0	6
E_2	*mf*	4	0	0	16
E_3	*mf*	0	1	0	6
E_4	*mf*	5	1	1	10
E_5	*mf*	3	0	0	11
S_1	*mf*	4	0	1	13
S_2	*mf*	6	0	1	11
S_3	*mf*	6	0	0	20
S_4	*mf*	7	0	1	17
S_5	*mf*	4	0	0	13
		Average			
FDP	*mf*	4.30	0.2	0.4	12.30
FDP	*ko*	2.5	0.1	0.90	16.60
FDP	*all*	7.1	0.2	1.20	17.4

Figure 7: *Summary of the type of edges detected by the* `redundant` *constraint.*

	FDP				*DREAM3*				*DREAM4*			
AUROC	`best`	`avg`	`hd`	`mode`	`best`	`avg`	`hd`	`mode`	`best`	`avg`	`hd`	`mode`
`sp (+re)`	1.163	1.083	1.033	0.963	1.108	1.028	1.028	1.108	0.859	0.839	0.839	0.839
`+tf`	17.543	18.883	13.623	5.483	18.948	18.108	15.308	9.648	24.939	24.799	21.539	12.919
`+cr`	18.333	21.223	16.203	5.483	20.888	22.148	17.768	9.648	27.459	29.479	24.559	12.919
AUPR												
`sp (+re)`	1.173	0.434	0.273	0.189	0.296	0.046	0.040	0.050	0.053	0.029	0.025	0.027
`+tf`	9.488	8.568	6.291	4.658	2.716	1.542	1.570	0.732	3.307	1.699	2.089	0.627
`+cr`	5.658	12.238	1.878	4.658	4.936	2.448	2.866	0.732	5.715	2.585	3.969	0.627

Table 5: *AUROC and AUPR % improvements for the CCN with* `best`*, average (*`avg`*), hamming distance (*`hd`*) and* `mode` *estimators w.r.t. the* CN_{rank} *in multi-factorial data.*

to 29.5%) and AUPR scores (up to 15.2%). This result supports our hypothesis that the addition of biological knowledge can better guide the predictions, even when the same inference ensemble is used.

Let us also observe that the best improvements in terms of AUROC and AUPR scores can be found in the CCN with the average estimator, and this is true for all the network considered and every dataset, including the ones with knockout data.

5 Conclusions

In this paper, we introduced the Constrained Community Networks (CCNs) paradigm to solve the gene regulatory network inference problem. CCNs use constraint programming techniques to guide the integration of predictions in a community network.

The use of constraints to model topological and biologically relevant prior information of a regulatory network provides several advantages in the creation of community networks. Our approach does not impose any hypothesis on the datasets adopted nor on the type of inference methods. Furthermore, constraints can naturally handle heterogeneous knowledge, facilitating the balancing of the strengths and weaknesses of the individual inference methods composing the inference ensemble.

We introduced a class of constraints able to **(1)** guarantee GRNs' specific properties and **(2)** take into account the community prediction collective agreements on each edge, and the limitations of each specific method. Experiments performed over a set of more than 300 benchmarks, including large networks proposed in the DREAM challenges, show that our approach can consistently outperform the consensus networks constructed by averaging individual edges ranks, as proposed in [36].

We have shown how the integration of knowledge about target networks acquired in biological relevant settings can provide significant improvements in terms of GRN prediction quality when compared to a state

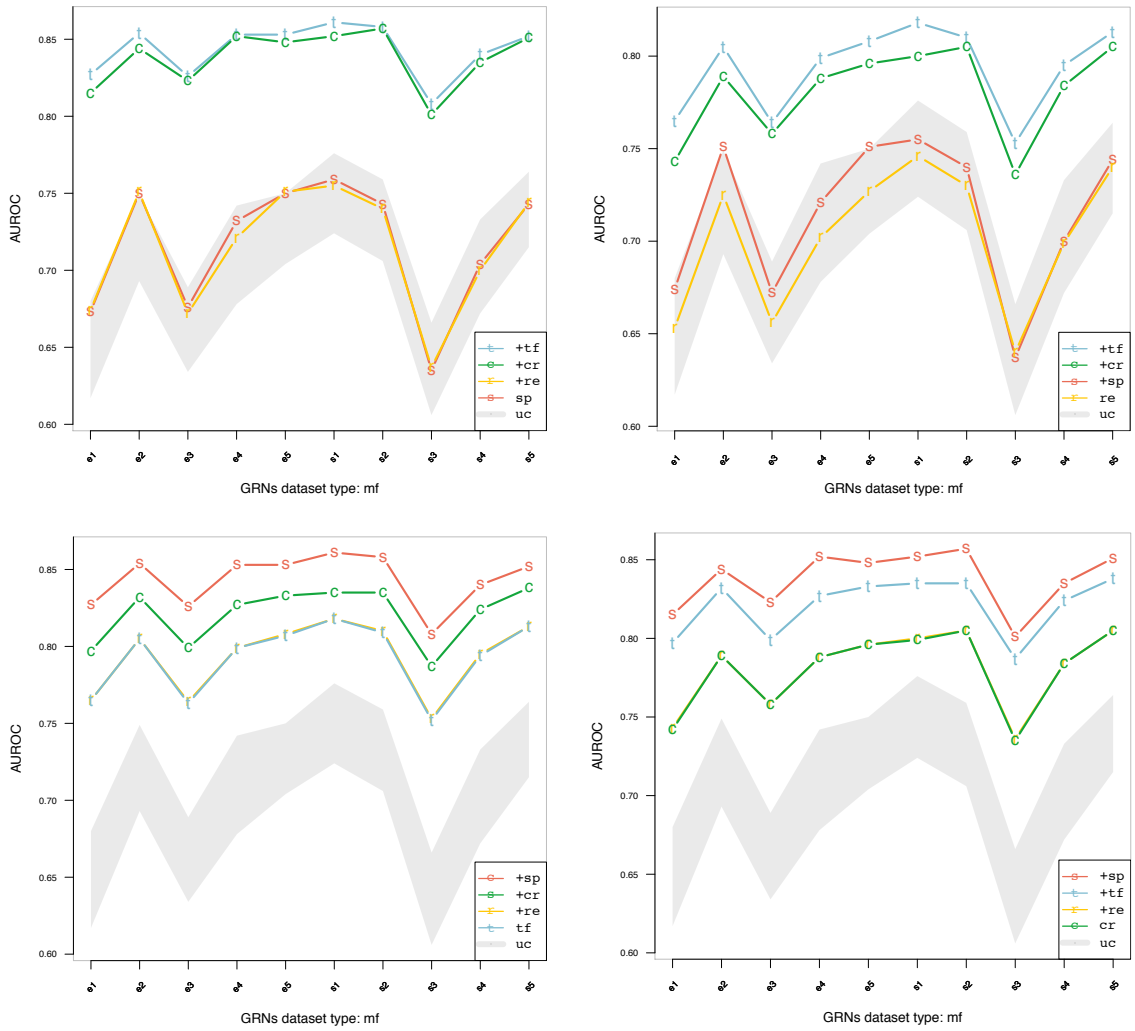

Figure 8: *The impact on the AUROC scores for the multi-factorial dataset in the FDP networks when combining 2 constraints fixing the* `sparsity` *(top-left), the* `redundant` *(top-right), the* `t-factor` *(bottom-left) or the* `co-reg` *(bottom-right) constraints.*

of the art CN approach (up to 29.5% and 15.2% for AUROC and AUPR measures respectively). This was possible as our model encourages the modular integration of biological knowledge, in form of logical rules.

As part of our future work, we plan to investigate new optimization measures that take into account local and global network properties, e.g., the number of specific network motifs in a target GRN region or the scale free degree in a given a portion of the graph. This can be achieved by including *soft constraints* in our model. We plan to use this information to address method-specific biases towards different connectivity patterns. On the CP side, we will extend existing constraints, for instance by studying the most likely set where a `t-factor` constraint could be targeted, and model new constrains and propagators to capture different type of biological knowledge, such us information about cell line and conditions at the time of the experiment, or the information encoded in *functional modules*—groups of TFs which regulate a particular biological process. Moreover we plan to employ path-based constraints, e.g., acting on the cascade effects resulting in a reward or penalty of an edge confidence value.

Acknowledgment

We thank the anonymous reviewers for their comments. The research has been supported by NSF grants HRD-1345232, CNS-1042341, and DGE-0947465.

	FDP				*DREAM3*				*DREAM4*			
AUROC	best	avg	hd	mode	best	avg	hd	mode	best	avg	hd	mode
sp (+re)	0.529	0.409	0.369	0.319	-0.517	-0.577	-0.777	-0.677	0.261	0.201	0.161	0.021
+tf	11.159	13.459	8.669	3.719	1.843	6.763	-0.577	3.123	6.801	11.021	4.181	5.401
+cr	11.889	15.179	9.819	3.719	2.223	8.023	0.783	3.123	7.201	12.721	5.341	5.401
AUPR												
sp (+re)	0.981	0.129	0.187	-0.077	2.000	0.120	-0.516	-0.740	1.004	0.140	0.374	-0.202
+tf	4.990	5.715	2.858	1.208	2.940	5.140	-0.220	2.520	3.130	5.790	0.414	2.970
+cr	7.248	9.038	3.822	1.208	-1.180	8.320	-4.016	2.520	1.300	8.530	-1.416	2.970

Table 6: *AUROC and AUPR % improvements for the CCN with* `best`*, average (*`avg`*), hamming distance (*`hd`*) and* `mode` *estimators w.r.t. the* CN_{rank} *in the knockout data.*

	FDP				*DREAM3*				*DREAM4*			
AUROC	best	avg	hd	mode	best	avg	hd	mode	best	avg	hd	mode
sp (+re)	1.266	1.136	1.106	1.026	-0.599	-0.639	-0.779	-0.779	0.707	0.527	0.507	0.467
+tf	13.106	14.936	10.206	4.526	2.061	6.361	-0.479	2.941	22.947	21.987	19.607	11.967
+cr	13.796	16.636	11.796	4.526	2.361	7.801	0.761	2.941	24.087	24.907	20.347	11.967
AUPR												
sp (+re)	1.282	0.400	0.464	0.161	2.178	0.118	0.338	-0.802	0.251	0.075	0.033	0.047
+tf	8.394	8.454	4.827	4.008	3.338	4.598	-0.902	2.038	3.433	2.303	2.051	1.057
+cr	5.064	12.514	1.794	4.008	-0.614	7.458	-2.880	2.038	5.961	3.303	3.131	1.057

Table 7: *AUROC and AUPR % improvements for the CCN with* `best`*, average (*`avg`*), hamming distance (*`hd`*) and* `mode` *estimators w.r.t. the* CN_{rank} *in the combined (all) data.*

References

[1] D. Allocco, I. Kohane, and A. Butte. 2004. Quantifying the relationship between co-expression, co-regulation and gene function. *BMC Bioinformatics* 5, 1 (2004), 18+.

[2] U. Alon. 2007. Network motifs: theory and experimental approaches. *Nat Rev Genet* 8, 6 (2007), 450–461.

[3] G. Altay, M. Asim, F. Markowetz, and D. E. Neal. 2011. Differential C3NET reveals disease networks of direct physical interactions. *BMC Bioinformatics* 12 (2011), 296.

[4] G. Altay and F. E. Streib. 2010. Inferring the conservative causal core of gene regulatory networks. *BMC Systems Biology* 4, 1 (2010), 132+.

[5] K. Apt. 2009. *Principles of Constraint Programming.* Cambridge University Press.

[6] S. Balaji et al. 2006. Comprehensive Analysis of Combinatorial Regulation using the Transcriptional Regulatory Network of Yeast. *Journal of Molecular Biology* 360, 1 (2006), 213 – 227.

[7] P. Baldi et al. 2000. Assessing the accuracy of prediction algorithms for classification: an overview. *Bioinformatics* 16, 5 (2000), 412–424.

[8] M. Bansal et al. 2007. How to infer gene networks from expression profiles. *Mol Sys Biol* 3 (2007), 78.

[9] A. Bauer-Mehren, L. I. Furlong, and F. Sanz. 2009. Pathway databases and tools for their exploitation: benefits, current limitations and challenges. *Molecular systems biology* 5, 1 (2009).

[10] C. M. Bishop and N. M. Nasrabadi. 2006. *Pattern recognition and machine learning.* Vol. 1. Springer.

[11] R. Bonneau et al. 2006. The Inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets de novo. *Genome Biology* 7, 5 (2006), R36.

[12] J. C. Borda. 1781. Memoire sur les elections au scrutin. (1781).

[13] L. Bortolussi and A. Policriti. 2008. Modeling Biological Systems in Stochastic Concurrent Constraint Programming. *Constraints* 13, 1-2 (2008).

[14] L. Breiman et al. 1984. *Classification and Regression Trees.* Chapman & Hall, New York, NY.

[15] F. Corblin, E. Fanchon, and L. Trilling. 2010. Applications of a formal approach to decipher discrete genetic networks. *BMC Bioinformatics* 11 (2010), 385.

[16] F. Corblin et al. 2009. A declarative constraint-based method for analyzing discrete genetic regulatory networks. *Biosystems* 98, 2 (2009), 91–104.

[17] H. De Jong. 2002. Modeling and simulation of genetic regulatory systems: a literature review. *Journal of computational biology* 9, 1 (2002), 67–103.

[18] R. de Matos Simoes and F. Emmert-Streib. 2012. Bagging statistical network inference from large-scale gene expression data. *PloS one* 7, 3 (2012), e33624+.

[19] P. D'Haeseleer, X. Wen, S. Fuhrman, and R. Somogyi. 1999. Linear modeling of mRNA expression levels during CNS development and injury. In *Pacific Symposium on Biocomputing*. 41–52.

[20] F. Eduati et al. 2012. Integrating literature-constrained and data-driven inference of signalling networks. *Bioinformatics* 28, 18 (2012), 2311–2317.

[21] B. Efron et al. 2004. Least angle regression. *Ann. Statist* 32, 2 (2004), 407–499.

[22] F. Fages et al. 2010. Computational Systems Biology in BIOCHAM. *ERCIM News* 2010, 82, 36.

[23] J. J. Faith et al. 2007. Large-Scale Mapping and Validation of Escherichia coli Transcriptional Regulation from a Compendium of Expression Profiles. *PLoS Biol* 5, 1 (01 2007), e8.

[24] F. Fioretto and E. Pontelli. 2013. Constraint Programming in Community-based Gene Regulatory Network Inference. In *CMSB 2013 (LNBI)*, Vol. 8130. Springer-Verlag, 135–149.

[25] S. Gama-Castro et al. 2008. RegulonDB: gene regulation model of Escherichia coli K-12 beyond transcription, active annotated promoters and Textpresso navigation. *Nucleic Acids Research* 36, D120–D124.

[26] M. Gebser et al. 2008. Detecting Inconsistencies in Large Biological Networks with Answer Set Programming. In *Logic Programming*, LNCS, Vol. 5366. Springer Berlin Heidelberg, 130–144.

[27] A. Greenfield et al. 2010. DREAM4: Combining Genetic and Dynamic Information to Identify Biological Networks and Dynamical Models. *PLoS ONE* 5, 10 (10 2010), e13397.

[28] N. Guelzim et al. 2002. Topological and causal structure of the yeast transcriptional regulatory network. *Nature Genetics* 31, 1 (22 April 2002), 60–63.

[29] M. A. Harris et al. 2004. The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Res* 32, Database issue (2004), D258–61.

[30] T. Hastie, R. Tibshirani, and J. Friedman. 2009. *The elements of statistical learning: data mining, inference and prediction* (2 ed.). Springer, New York, NY.

[31] S. Kim, S. Imoto, and S. Miyano. 2003. Dynamic Bayesian Network and Nonparametric Regression for Nonlinear Modeling of Gene Networks from Time Series Gene Expression Data. *Biosystems*. 104–113.

[32] S. K. Kummerfeld and S. A. Teichmann. 2006. DBD: a transcription factor prediction database. *Nucl. Acids Res.* 34, suppl_1 (2006), D74–81.

[33] L. P Lim et al. 2005. Microarray analysis shows that some microRNAs downregulate large numbers of target mRNAs. *Nature* 433, 7027 (2005), 769–773.

[34] H-W. Ma et al. 2004. An extended transcriptional regulatory network of Escherichia coli and analysis of its hierarchical structure and network motifs. *Nucleic acids research* 32, 22 (2004), 6643–6649.

[35] P. Madhamshettiwar et al. 2012. Gene regulatory network inference: evaluation and application to ovarian cancer allows the prioritization of drug targets. *Genome Medicine* 4, 5 (1 May 2012), 41+.

[36] D. Marbach et al. 2012. Wisdom of crowds for robust gene network inference. *Nat Meth* 9, 8 (2012), 796–804.

[37] A. A. Margolin et al. 2006. ARACNE: An Algorithm for the Reconstruction of Gene Regulatory Networks in a Mammalian Cellular Context. *BMC Bioinformatics* 7, S-1 (2006).

[38] N. Meinshausen and P. Bühlmann. 2010. Stability selection. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72, 4 (1 Sept. 2010), 417–473.

[39] P. E. Meyer et al. 2007. Information-theoretic inference of large transcriptional regulatory networks. *EURASIP Journal on Bioinformatics & Systems Biology* (2007).

[40] M. P. Perrone. 1993. *Improving regression estimation: Averaging methods for variance reduction with extensions to general convex measure optimization.* Ph.D. Dissertation.

[41] R. J. Prill et al. 2010. Towards a Rigorous Assessment of Systems Biology Models: The DREAM3 Challenges. *PLoS ONE* 5, 2 (02 2010), e9202.

[42] T. Reguly et al. 2006. Comprehensive curation and analysis of global interaction networks in Saccharomyces cerevisiae. *Journal of biology* 5, 4 (08 June 2006), 11+.

[43] M. Renda and U. Straccia. 2003. Web metasearch: rank vs. score based rank aggregation methods. In *Proceedings of the 2003 ACM Symposium on Applied Computing.* ACM, 841–846.

[44] T. Schaffter, D. Marbach, and D. Floreano. 2011. GeneNetWeaver: In silico benchmark generation and performance profiling of network inference methods. *Bioinformatics* 27, 16 (22 June 2011), 2263–2270.

[45] C. Schulte and P. J. Stuckey. 2008. Efficient Constraint Propagation Engines. *ACM TOPLAS* 31, 1.

[46] S. S. Shen-Orr et al. 2002. Network motifs in the transcriptional regulation network of Escherichia coli. *Nature Genetics* 31 (2002), 1061–4036. Issue 1.

[47] A. Sîrbu, H. J. Ruskin, and M. Crane. 2012. Integrating heterogeneous gene expression data for gene regulatory network modelling. *Theory in Biosciences* 131, 2 (2012), 95–102.

[48] T. Soh and K. Inoue. 2010. Identifying Necessary Reactions in Metabolic Pathways by Minimal Model Generation. In *ECAI 2010.* IOS Press, Amsterdam, The Netherlands, The Netherlands, 277–282.

[49] M. Song et al. 2009. Reconstructing generalized logical networks of transcriptional regulation in mouse brain from temporal gene expression data. *EURASIP J. on Bioinformatics and Systems Biology* (2009), 5.

[50] N. Sun and H. Zhao. 2009. Reconstructing transcriptional regulatory networks through genomics data. *Statistical Methods in Medical Research* 18, 6 (2009), 595–617.

[51] R. Thomas. 1973. Boolean formalization of genetic control circuits. *J. Theoretical Biol.* 42, 3, 563–585.

[52] S. Videla et al. 2012. Revisiting the Training of Logic Models of Protein Signaling Networks with ASP. In *CMSB.* 342–361.

[53] X. Zhou, X. Wang, and E. Dougherty. 2006. *Genomic Networks: Statistical Inference from Microarray Data.* John Wiley & Sons.