JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 19, Number 3, 2012
© Mary Ann Liebert, Inc.
Pp. 316–336
DOI: 10.1089/cmb.2011.0234

Review Article

Probabilistic Logic Methods and Some Applications to Biology and Medicine

NIKITA A. SAKHANENKO[1] and DAVID J. GALAS[1,2]

ABSTRACT

For the computational analysis of biological problems—analyzing data, inferring networks and complex models, and estimating model parameters—it is common to use a range of methods based on probabilistic logic constructions, sometimes collectively called machine learning methods. Probabilistic modeling methods such as Bayesian Networks (BN) fall into this class, as do Hierarchical Bayesian Networks (HBN), Probabilistic Boolean Networks (PBN), Hidden Markov Models (HMM), and Markov Logic Networks (MLN). In this review, we describe the most general of these (MLN), and show how the above-mentioned methods are related to MLN and one another by the imposition of constraints and restrictions. This approach allows us to illustrate a broad landscape of constructions and methods, and describe some of the attendant strengths, weaknesses, and constraints of many of these methods. We then provide some examples of their applications to problems in biology and medicine, with an emphasis on genetics. The key concepts needed to picture this landscape of methods are the ideas of probabilistic graphical models, the structures of the graphs, and the scope of the logical language repertoire used (from First-Order Logic [FOL] to Boolean logic.) These concepts are interlinked and together define the nature of each of the probabilistic logic methods. Finally, we discuss the initial applications of MLN to genetics, show the relationship to less general methods like BN, and then mention several examples where such methods could be effective in new applications to specific biological and medical problems.

Key words: Bayesian networks, first-order logic, hierarchical Bayesian networks, machine learning, Markov logic networks, probabilistic Boolean networks, probabilistic graphical models, propositional logic.

1. INTRODUCTION

LOGIC AND PROBABILISTIC GRAPHICAL MODELS ARE NATURAL TOOLS OF COMPUTING and have been studied and used in computer science for many years. Some of these methods have emerged in recent years as promising approaches to a range of problems in artificial intelligence. In biology and medicine, where many of the computational problems involve data analysis and the inference of underlying complex models, simple forms of these methods, such as Hidden Markov Models (HMM), and simple forms of Bayesian Networks

[1]The Institute for Systems Biology, Seattle, Washington.
[2]Luxembourg Centre for Systems Biomedicine, University of Luxembourg, Esch-sur-Alzette, Luxembourg.

(BN), have become more commonly used in recent years (Baldi and Brunak, 2001). There is now a wide range of these methods that have been usefully applied to biological and medical problems, but it is often difficult to see in the reports of successful applications in the literature how they are related to one another. In this review, we attempt to address this difficulty and elucidate some of the key relationships, not specifically for the experts in the field, but more for the computational biologists that are grappling with the realities and complexities of current problems in systems biology, genetics and their applications to modern medicine. This review is not a comprehensive survey of methods, nor of the specific techniques and implementations in the literature, but rather is focused on providing a map of some of the key concepts that we hope will allow computational biologists to see the relationships, contrasts, and overlaps among widely used methods. Even more importantly, in our view, we hope to point out where new applications of probabilistic logic methods may provide powerful new approaches to the sometimes dauntingly complex problems of understanding biological systems. Our plan for this review is to begin with the most general combination of logic and probability, and illustrate how simplifying restrictions and constraints lead to more familiar methods, in order to elucidate the key relationships, and to indicate where some power is lost and where efficiency is gained. There are several relevant logical systems and several different forms, and representations, of probability distributions. These are briefly summarized below.

First-Order Logic (FOL) is a very general formal logic, and represents a more powerful tool for expression of logical relationship than propositional logic (Ershov and Palyutin, 1986). Using propositional logic, we are not able to express assertions about classes of objects or events. FOL is considerably richer than propositional logic and allows just these expressions. FOL allows the propositional symbols to have arguments that range over elements of sets, which allows us to make assertions about the sets or classes. FOL can also deal with recursive statements, while propositional logic cannot.

Probability distributions can involve a wide range of simple and complex dependencies and are often represented in graphical form in statistics, where the independencies of the variables represented as nodes are encoded in the edges. In this review, we illustrate the key relationships by beginning with the most general and proceeding to more restrictive, and constrained representations. While there are some significant differences in the graphical representations used in various methods the logical components of the probabilistic logic are where most of the restrictions occur.

Markov Logic Networks (MLN) represent a new and general approach to modeling based on full FOL and Markov Random Fields (MRF) (Richardson and Domingos, 2006). In addition to high representational power of FOL, MRFs provide a very compact way of representing probability distributions and are very useful for modeling and reasoning in noisy, uncertain environments. For physicists, MRFs are probably most familiar as the mathematical representation of Ising models, which are simplified, statistical models of the interaction of arrays of magnetic spins (Kindermann and Snell, 1980). MLNs thus combine (and generalize) FOL and MRFs to gain most of the advantages of both the logic and probabilistic modeling worlds. Among the advantages of MLN are the ability to handle arbitrary classes of variables, recursive statements, and the ability to construct multiple templates for MRFs.

Using MLN can allow us to perform sophisticated probabilistic modeling while directly incorporating biological knowledge, particularly including partial knowledge, into the models, which is one of our major motivations for developing this general approach. We think that this capability is fundamentally important for the development of system level analysis and modeling of biological systems, including metabolic, regulatory, and genetic aspects of these systems.

It is useful here to presage, or summarize, the conclusions of the later sections here by giving a compact overview of the key relationships that put the MLN and other methods into some context. We use the examples elaborated later in this article to illustrate this overview. Since the most widely used probabilistic graphical models in computational biology are various versions of BN, it is specifically useful to note the key differences between these and MLN. The major differences are these three:

1. The probability distributions that can be represented in MLN are those of MRF, which are more general than those of BN.
2. Relationships and probabilities can be assigned to classes of elements and/or events in MLN, whereas in BN probabilities are assigned to individual elements or events.
3. The representation of the logical relationships in MLN is much more compact than in BN (for those that can be expressed in BN), particularly for more complex models.

These differences will be illustrated and discussed in later sections. A fourth difference is that MLNs are much more computationally intensive to implement, which can be an important practical limitation.

Now we turn to a specific example of model selection in genetic analysis. We have applied MLN by using a search method that allows us to solve various model selection problems with different biological knowledge constraints naturally embedded in them. In a previous article, we used MLN to select models for the influence of the genotype (specific sets of markers, indicating gene variants) on the phenotype (e.g., properties of the organism as expressed in measurements: size, color, shape, gene expression levels). We first used a single-marker model iteratively that focused on relationships between genetic markers and a given phenotype (Sakhanenko and Galas, 2010). Every marker associated with a gene selected in the model had a specific influence on the phenotype, even when considered alone. The model is asked to predict the phenotype given the genotype of a single organism, and the markers were iteratively selected to improve the prediction. Later, in this review, we discuss a natural extension of this application that uses a pair-wise gene model that explicitly represents relationships between a phenotype and pair-wise interacting genes. The models discussed here to illustrate the approach, while complex in implementation, are still rather simple genetic models, and the full power of MLNs will come with the natural extension to much more complex models. This is one of the major points we would like to emphasize—that much more powerful applications are to come, but probabilistic logic represents a ''language'' for expression and calculation with such complex models.

To illustrate the method in a simple form, a single-marker model encodes an influence of a set of genetic markers on a phenotype as an aggregation of influences of individual markers. When conditioned on the alleles of the markers (predicting the phenotype values given the genotype data), modeling using a single-marker model can be seen (as we specifically show in a later section) as performing a logistic regression of the marker alleles on the phenotype values. Note however that the single-marker model does not make the assumption that the data is identically, independently distributed (iid) as opposed to the case of a true logistic regression. At the first iteration of the search method (see Algorithm 1 of Section 4), when each model is based on only one marker, the corresponding logistic regression has two predictors (assuming a genetic marker can have two possible alleles). For the following iterations, as we add more markers to the predictive markers set, the corresponding logistic regression is expanded to include more predictors (four, six, and so on). It is important to note however that the MLN description of the models at different levels of iteration does not change; what changes is a set of possible values of the MLN variables that correspond to the predictors of the logistic regression. The method is then a systematic application of a *template* specified by MLN to different subsets of the data. This template is essentially a model. We can use this template then for adding biological domain knowledge into the probabilistic search—information about how the various parts of a biological system interact or influence one another. For example, by using *pair-wise* models, we expand the template to specifically allow for pair-wise gene interactions.

A pair-wise model encodes an aggregation of joint influences of pairs of markers together with their interactions on a phenotype. This model is an MLN template that explicitly takes into account possible gene interactions. As with the single-marker model, when predicting phenotype values from genotype values, the pair-wise model can also be seen as a logistic regression. The difference between the single-marker and the pair-wise models becomes apparent when we look closely at their corresponding forms of regression. A logistic regression derived from the pair-wise model contains all the terms of the regression of the single-marker model given the same set of markers, plus the terms that correspond to possible interactions in each pair of the markers. If the markers have no gene interactions, then a corresponding pair-wise model reduces to a single-marker model. On the other hand, if the markers do interact, then the pair-wise model can encompass that effect when predicting the phenotype, even if the markers individually have little influence on the phenotype and cannot therefore be detected as contributing in the single gene model. This is why the pair-wise models have been successful at capturing synthetic gene interactions that confer a phenotype only when both specific alleles of a pair of genes are present. However, if two genes have an effect on phenotype individually, but also have a significant interaction, the interaction may be detected by the single gene template as a non-additive effect of the two genes on prediction accuracy (because of the induced correlation or dependence). The pair-wise model, on the other hand can detect significant interaction effects even if the single gene effects are in the noise and not detectable. The pair-wise model, while more complex than the single gene model, is clearly just the tip of the iceberg of the kinds of complex models that likely underlie real complex genetic traits.

MLNs are powerful tools for systems genetics since they allow us to incorporate explicit biological knowledge into genome-wide studies. For example, when using the pair-wise model, we can control the interaction terms of the corresponding logistic regression through specific constraints in the logical relationships based on our biological knowledge. We could also define specific constraints concerning the gene interactions in the model. Moreover, the logic component of MLNs makes it possible to represent large probabilistic models (a logistic regression with many interaction terms, for example) in a highly compact way. Using MLNs as model templates in genetic studies is a huge step beyond Genome-Wide Association Studies (GWAS) and moves us substantially toward true systems genetics.

It is important to compare MLNs to other established and related modeling methods in computational biology. Since MLNs are based on MRFs and FOL, it is natural that MLNs represent a generalization of both of these (Richardson and Domingos, 2006). In the most extreme case, we can reduce MLNs simply to FOL by setting all the weights in an MLN to an arbitrarily large number. Conversely, we can also reduce MLNs to MRFs by assigning an MLN formula for each clique of an MRF and then using the cliques potential as the weight of that formula. As with MRFs, we can also express any probability distribution represented by a BN using MLNs. Consequently, other kinds of BNs, such as Hierarchical BNs (HBN) and Dynamical BNs (DBN), can also be represented in the MLN framework. We note, for example, that an HMM is actually a particularly simple form of a DBN.

To briefly summarize the relationships between these methods, we present a simplified table of properties here (Table 1). These elements will be described and discussed in the body of the review, and though some of the entries in this table may be less than clear at this point, the table represents a rough map of the methods review and of our intent. We revisit the summary of relationships more explicitly in a later section (Fig. 3).

To further explore and analyze the relationship between MLNs and other methods, we will now consider in more detail the probabilistic, logic, and structural components of each method and indicate where the methods differ in how constrained their components are. The MLNs have the least constrained logic component, and so we will use them as the general, master method. Next, in Section 2, we set out the notation carefully, and we briefly introduce FOL, MRFs, and MLNs with a little more rigor. In Section 3, we show explicitly the relationship between MLNs and BNs, and then discuss how MLNs fit into a broader picture of probabilistic logic methods. To make it explicitly clear, we illustrate a specific MLN representation of a BN in an example in Section 3. In Section 4, we illustrate the use of MLN-based methods applied thus far to genetics. These examples show two types of models, single-marker and pair-wise models, which are analyzed in Sections 5 and 6, respectively. Finally, we conclude by describing some future, possible applications of MLNs to other biological problems in Section 7.

Table 1. Brief Summary of Some of the Key Elements of Probabilistic Logic Methods

Name	*Probabilistic component*	*Logic component*	*Graphical representation*
Markov Logic Networks	Markov Random Fields	First-order logic	General undirected graphs
Markov Random Fields	Conditional independence via graph separation	Propositional logic	General undirected graphs
Bayesian Networks	Conditional probability chain rule	Propositional logic with mutually exclusive constraints and acyclicity requirements	Directed acyclic graphs (DAGs)
Hierarchical Bayesian Networks	Conditional probability chain rule	Propositional logic with mutually exclusive constraints and structural requirements	Directed trees with links inside each layer
Dynamic Bayesian Networks	Conditional probability chain rule, Markov chain	Propositional logic with mutually exclusive constraints and structural sequential requirements	Sequence of snapshots (DAGs) connected in order
Probabilistic Boolean Networks	Markov chain	Boolean logic	Sequence of snapshots connected in order

2. PRELIMINARIES

We describe here all the necessary preliminary information, definitions, and notation for logic, MRFs, and networks.

2.1. Logic

Among various systems of logic, the most general one we will use here is FOL. On the other hand, propositional logic has the simplest semantics, but many concepts of propositional logic generalize to FOL (Ershov and Palyutin, 1986).

In propositional logic, there are atomic (simple) assertions, consisting of *propositional letters*, and compound assertions, composed from the atomic assertions and the logical connectives, *and* ($\wedge$), *or* ($\vee$), *not* ($\neg$), *implication* ($\Rightarrow$), and *equivalence* ($\Leftrightarrow$). An *interpretation* in propositional logic is a mapping that assigns a truth value (*True* or *False*) to every propositional letter. Once the atomic assertions in a proposition have received an interpretation, we can compute the truth value of the proposition. We can do this since all propositional formulas are *inductively* constructed from the atomic assertions, and the logical connectives are interpreted with the *truth table* given in Table 2.

In propositional logic, propositions that are equivalent irrespective of their interpretations form an equivalence class. A structure based on these equivalence classes is called *Boolean algebra*.

Using propositional logic, we are not able to express assertions about elements of structures. FOL is considerably richer than propositional logic and allows these expressions. FOL allows the propositional symbols to have arguments that range over elements of different structures, which allows us to make assertions about the sets of elements of structures.

In FOL, there are *atomic formulas*, consisting of *predicates* applied to logical *terms*. A term is an entity inductively constructed from variables and functions and has many levels of complexity. Note that the simplest term is a value of a variable, called a *constant*, that can be seen as a function that takes no arguments, or has the same value irrespective of its arguments. Note also that a simplest atomic formula is a predicate that takes no arguments, which is similar to a propositional letter in *propositional logic*. In FOL, there are formulas, composed inductively from the atomic formulas, the logical connectives, $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$ (as in propositional logic), and the quantifiers, *universal* ($\forall$, the usual mathematical symbol for "each and every") and *existential* ($\exists$, the usual mathematical symbol for "there exists"). In order to assign meaning to the symbols in FOL, we first must define a *domain* (a *universe*), which is the domain of the logical variables. An *instantiation* of a first-order formula is, then, a replacement of variables of the formula with logical terms. Note that the most common instantiation replaces the variables with the values (constants) from their domain. An *interpretation* in FOL is a mapping that assigns a truth value to every atomic formula for every instantiation of variables. An interpretation is then recursively defined on complex formulas. Note that, given an instantiation, formulas composed from atomic formulas and logical connectives, are interpreted just as in *propositional logic*. The interpretation of quantified formulas is as follows:

$$(\forall x\ A(x)) \text{ is true iff } A(d) \text{ is true for any } d \in \mathit{Domain}$$
$$(\exists x\ A(x)) \text{ is true iff } A(d) \text{ is true for some } d \in \mathit{Domain}$$

A *possible world* (a *Herbrand interpretation*) in FOL is an assignment of truth-values to *all possible* predicates whose variables are replaced with every possible combination of values. For example, one

TABLE 2. TRUTH TABLE DEFINING THE INTERPRETATION OF LOGICAL CONNECTIVES OF PROPOSITIONAL LOGIC

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
True	True	False	True	True	True	True
True	False	False	False	True	False	False
False	True	True	False	True	True	False
False	False	True	False	False	True	True

TABLE 3. POSSIBLE WORLD (IN FOL) FOR THE PROGRAM (13)

Phenotype(s_1, t_1)	True
Phenotype(s_1, t_2)	False
...	...
Phenotype(s_{N_4}, t_{N_3})	True
Allele(s_1, m_1, v_1)	False
...	...
Allele(s_{N_4}, m_{N_1}, v_{N_2})	False

Here predicates **Phenotype**(s_i, t_j) and **Allele**(s_i, m_k, v_l) encode two facts (two data points) that t_j is a phenotype value of strain s_i and that v_l is an allele of marker m_k for strain s_i.

possible world for the program (13) can be given in Table 3. Note that various model restrictions can be applied by reducing the set of all possible predicates and a set of possible values for the variables.

2.2. *Probabilistic graphical models*

A BN is a probabilistic graphical model that represents random variables and their probabilistic dependencies with a directed, acyclic graph (Pearl, 1988; Koller and Friedman, 2009). A BN is represented by a directed, acyclic graph where each vertex represents a random variable and each edge represents a conditional dependence between two vertices. Since the graph is directed, for every edge we distinguish a parent, a vertex from which the edge originates, and a child, a vertex to which the edge goes. Consequently, every vertex is assigned a conditional probability, in a table defining the probability of a variable given every possible set of values of the parents of the vertex (corresponding to all edges going in the vertex). We could say ''conditioned on'' those values. One powerful feature of BNs is their ability to graphically encode conditional dependence between random variables: a variable is conditionally independent from any of non-descendent variables, given the values of its parents. This feature allows BNs to specify probability distributions in a compact and efficient way, but only those distributions that fit these constraints.

Consider a set of random variables $\mathbf{V} = \{V_1, \ldots, V_N\}$ and consider a directed, acyclic graph $G = (\mathbf{V}, \mathbf{E})$ based on the set $\mathbf{V}$ (thus, every V_i will be referred to as either a *variable* or a *vertex*). Given a parameter set $\Theta = \{\Theta_1, \ldots, \Theta_N\}$, where each Θ_i is a conditional probability distribution of V_i given its parents, $\Theta_i = \Pr(V_i|parents(V_i))$, then $\langle \mathbf{V}, G, \Theta \rangle$ is a BN if a joint probability distribution on $\mathbf{V}$ can be factorized as

$$\Pr(\mathbf{V}) = \prod_{i=1}^{N} \Theta_i = \prod_{i=1}^{N} \Pr(V_i|parents(V_i)). \tag{1}$$

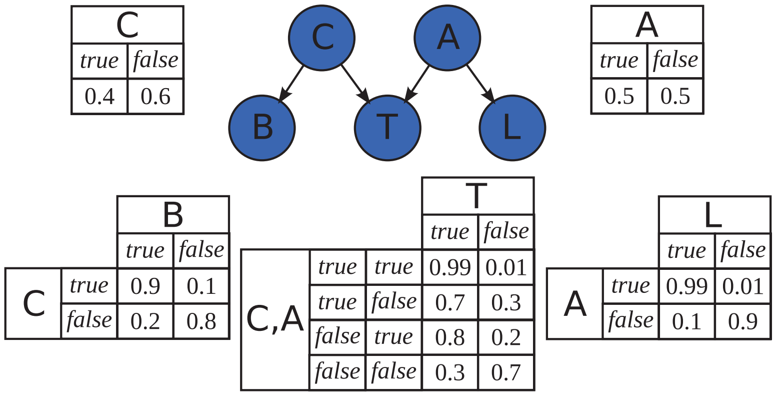
Note that in many applications of BNs researchers give causal meaning to the edges of a network. In general, however, the directionality of edges does not imply causality. Note the significance of the factorization is related to the chain rule for conditional probabilities. The interpretation of this rule is the source of these incorrect causality arguments.

Consider here an example from a textbook (Luger, 2008). Suppose there is one event that can cause orange barrels to appear on the road, $B = true$: a road construction, $C = true$. There is also one event that can cause flashing lights to appear on the road, $L = true$: an accident, $A = true$. Suppose then that there are two events that can cause bad traffic, $T = true$: either an accident or a road construction. This example can be modeled by a BN shown in Figure 1, whose parameters are given in the tables below. Note that all the variables are binary in this case.

The probabilistic independencies encoded by the BN in Figure 1 allows us to express a joint probability distribution in a compact way with a small number of parameters. Here, the rule is the probabilities of nodes not directly connected by edges are independent and can be multiplied:

$$\Pr(C, A, B, T, L) = \Pr(C) \times \Pr(A) \times \Pr(B|C) \times \Pr(T|C, A) \times \Pr(L|A).$$

MRFs are another kind of probabilistic graphical model, which is similar to BNs in how dependencies are represented (Kindermann and Snell, 1980; Pearl, 1988; Koller and Friedman, 2009). As opposed to BNs though, MRFs are defined on *undirected* graphs. Removing edge directionality eliminates the asymmetry between a parent vertex and a child vertex, which allows MRFs to represent cyclic dependencies that



C	
true	*false*
0.4	0.6

A	
true	*false*
0.5	0.5

C	B *true*	B *false*
true	0.9	0.1
false	0.2	0.8

C	A	T *true*	T *false*
true	*true*	0.99	0.01
true	*false*	0.7	0.3
false	*true*	0.8	0.2
false	*false*	0.3	0.7

A	L *true*	L *false*
true	0.99	0.01
false	0.1	0.9

FIG. 1. Structure of a road traffic Bayesian network. Here *B* stands for orange barrels on the street, *C*—for a road construction, *T*—for traffic, *A*—for an accident, and *L*—for flashing lights.

cannot be represented by BNs. Note however that there are dependencies, such as induced dependencies, that can be represented by BNs, but not by MRFs. Thus, MRFs offer an alternative graphical semantics for probability distributions where graph separation of two vertices by a separating set implies conditional independence of the two vertices given the separating set, and we are able to make different conditional independence statements in MRFs than in BNs.

More formally, given three disjoint sets of vertices, A, B, C, in an undirected graph, the set B *separates* A from C in the graph if *every path* from A to C contains at least one vertex from B. We now use the definition of graph separation to define an MRF. Consider a set of random variables $\mathbf{V}=\{V_1, \ldots, V_N\}$ and consider an undirected graph $G=(\mathbf{V}, \mathbf{E})$ based on the set $\mathbf{V}$. An MRF defined on $\mathbf{V}$ is a probability distribution such that there exists a G with a condition that, for disjoint sets of vertices A, B, C, if A and C are separated by B, then vertices from A are conditionally independent from vertices from C given B. The conditional independence property of MRFs implies that if we have two vertices V_i and V_j that are not directly connected, then they are conditionally independent given all other vertices, i.e.,

$$\begin{aligned}&\Pr(V_i=v_i, V_j=v_j|\mathbf{V}\setminus\{V_i, V_j\}=\mathbf{v}\setminus\{v_i, v_j\})=\\&=\Pr(V_i=v_i|\mathbf{V}\setminus\{V_i, V_j\}=\mathbf{v}\setminus\{v_i, v_j\})\Pr(V_j=v_j|\mathbf{V}\setminus\{V_i, V_j\}=\mathbf{v}\setminus\{v_i, v_j\}).\end{aligned}$$

Here, $V_i=v_i$ stands for an event that a variable V_i takes on a value v_i, $\mathbf{V}=\mathbf{v}$ is shorthand for $V_1=v_1, \ldots, V_N=v_N$, and $\mathbf{V}\setminus\{V_i, V_j\}$ is a set $\mathbf{V}$ without vertices V_i and V_j. Note also that two directly connected vertices are not conditionally independent given all other vertices. This can be used to factorize an MRF.

A clique c of a graph G is a subgraph such that all vertices of c are fully connected. A Gibbs distribution defined on G is

$$\Pr(\mathbf{V}=\mathbf{v})=\frac{1}{Z}\prod_{c\in Cl}\exp(-\phi_c(\mathbf{v}_c)). \tag{2}$$

The so-called partition function $Z=\sum_{\mathbf{v}}\prod_{c\in Cl}\exp(-\phi_c(\mathbf{v}_c))$ normalizes the probability to ensure that $\sum_{\mathbf{v}}\Pr(\mathbf{V}=\mathbf{v})=1$. Here Cl is the set of all cliques of G, $\mathbf{v}_c$ is a restriction of a configuration of $\mathbf{v}$ to a clique c, and ϕ_c is a real-valued potential function assigned to a clique c. Given a Gibbs distribution on G,

$$\begin{aligned}&\Pr(V_1=v_1|\mathbf{V}\setminus\{V_1\}=\mathbf{v}\setminus\{v_1\})=\frac{\Pr(\mathbf{V}=\mathbf{v})}{\Pr(\mathbf{V}\setminus\{V_1\}=\mathbf{v}\setminus\{v_1\})}\\&=\frac{\prod\limits_{c\in Cl}\exp(-\phi_c(v_1, v_2, \ldots, v_N))}{\sum\limits_x\prod\limits_{c\in Cl}\exp(-\phi_c(x, v_2, \ldots, v_N))}=\frac{\prod\limits_{\{c\in Cl|V_1\in c\}}\exp(-\phi_c(v_1, v_2, \ldots, v_N))}{\sum\limits_x\prod\limits_{\{c\in Cl|V_1\in c\}}\exp(-\phi_c(x, v_2, \ldots, v_N))}.\end{aligned}$$

Note that although for convenience we write $\phi_c(V_1, \ldots, V_N)$, this potential function depends only on the vertices from the clique c, hence the right side of the derivation above depends only on the variables from the cliques containing V_1. Thus, the variable V_1 is conditionally dependent on only the variables form the same cliques it belongs to, which means that any Gibbs distribution is an MRF. On the other hand, the

Hammersley-Clifford theorem proves the converse, that every positive MRF corresponds to some Gibbs distribution.

Without loss of generality, we can represent an MRF conveniently as a log-linear model:

$$\Pr(\mathbf{V}=\mathbf{v}) = \frac{1}{Z}\exp\Big(\sum_i w_i f_i(\mathbf{v})\Big), \tag{3}$$

where the f_i are real-valued functions defining features of the MRF and w_i are the real-valued weights of the MRF, that are the parameters of the model (Pietra et al., 1997). Features, that can be as simple as indicator functions representing the presence of some attributes, and can overlap in arbitrary ways providing representational flexibility.

2.3. *Markov logic networks*

Because of their flexibility and the potential for representing complex relationships we have proposed using probabilistic logic methods for analyzing genetic data. From a set of related logic-based probabilistic methods, we chose the most general of these, MLNs, and have used the method to identify genetic loci that predict quantitative phenotypes (Sakhanenko and Galas, 2010).

MLNs merge MRFs with first-order logic (see Sections 2.1 and 2.2). Any strictly formal completely logic system (not including probabilities) is not suitable for applications where the data contain any uncertainty or noise. This is because a set of first-order formulas specifying a logical model is seen as a set of uncompromising requirements so that the model is either true or false by comparison with the data. In other words, models in FOL can only have probability values 1 or 0, and since noise and uncertainty exist in all real data, no model is satisfied exactly and all must therefore be false. Since the data we wish to analyze are actually rich in information while not being exactly satisfied, this is not a useful point of view. MLNs relax this constraint by allowing a model with unsatisfied formulas with a lesser probability than 1. The model with the smallest measure of unsatisfied formulas is the most probable, and will therefore represent the most successful extraction of knowledge of reality from the data.

An MLN is a set of first-order formulas, F_i, with assigned weights w_i. An MLN, together with the set of possible values of its variables, is then converted to an MRF as follows. For every predicate of the MLN whose variables are instantiated with every possible combination of values, we create one random variable in the MRF. The value of the random variable is either 1 or 0 corresponding to whether the instantiated predicate is true or false. Furthermore, for every possible instantiation of every formula, F_i, we construct one feature of the log-linear MRF (see Section 2.2) whose value is either 1 or 0, depending on the truth value of the instantiated formula. The weight of the MRF feature is the weight w_i assigned to the formula F_i in the MLN. Taking the original definition (3) and replacing all the features corresponding to false formulas with 0, we obtain the probability distribution represented by an instantiated MLN

$$\Pr(\gamma) = \frac{1}{Z}\exp\Big(\sum_i w_i n_i(\gamma)\Big), \tag{4}$$

where $n_i(\gamma)$ is a number of times the formula F_i is instantiated to a true proposition in the state γ (which corresponds to our data). Note that this probability distribution depends on the set of possible values of MLN variables. Therefore, MLNs can be seen as templates specifying classes of MRFs, similar to FOL specifying propositional formulas (see Section 2.1).

3. MARKOV LOGIC NETWORKS VERSUS BAYESIAN NETWORKS

3.1. *Representing Baysian networks with conjunctive normal forms*

Darwiche (2002) showed that a BN can be encoded in a Conjunctive Normal Form (CNF). A CNF C is a conjunction of logical clauses D_i: $C = D_1 \wedge \ldots \wedge D_N$, where a logical clause D_i is a disjunction of either propositional variables or their negations (although a variable and its negation cannot be in the same clause): thus, $V_{i_1} \vee \neg V_{i_2} \vee \ldots \vee V_{i_K}$.

Following Darwiche's notation, we first establish the alphabet of propositional logic corresponding to the objects of a BN. We use two types of propositional symbols here, λ_v and $\theta_{v|\mathbf{u}}$. A propositional symbol λ_v for each value v of a random variable V is interpreted as being true iff $V = v$. Note that V can have more than two values. A propositional symbol $\theta_{v|\mathbf{u}}$, for each value v of V and for each combination of values $\mathbf{u}$ of a set $\mathbf{U}$ of parent vertices of V, is interpreted as being true iff there is an entry in a conditional probability table whose value is $\Pr(V = v|\mathbf{U} = \mathbf{u})$. These elements can be used to define precisely the logical structure inherent in BNs.

Consider then a BN $\langle \mathbf{V}, G, \Theta \rangle$ and construct a CNF encoding this BN. First, for each network variable V, whose possible values are $v_1, \ldots, v_K$, we must include the following clauses (disjunctions): $\lambda_{v_1} \vee \ldots \vee \lambda_{v_K}$ and $\neg\lambda_{v_i} \vee \neg\lambda_{v_j}, i \neq j$. These clauses ensure that we use exactly one value for each random variable when evaluating our BN. In addition, for each entry of every conditional probability table of the BN, we must include in the CNF encoding the following clauses: $\lambda_v \wedge \lambda_{u_1} \wedge \ldots \wedge \lambda_{u_M} \Leftrightarrow \theta_{v|u_1,\ldots,u_M}$. These clauses ensure that, while evaluating the BN, the evidence $v, u_1, \ldots, u_M$ is necessary and sufficient for the BN to include a conditional probability table that contains $\Pr(v|u_1, \ldots, u_M)$.

Let us now construct a CNF encoding the road traffic BN from Figure 1. Our CNF is a conjunction of the disjunctions (clauses) shown in Table 4. The CNF models (possible truth assignments to the propositional variables) are in one-to-one correspondence with the instantiations of the network variables. Following Darwiche (2002), we can now apply weighted model counting to the CNF by assigning weights to all the letters and their negations: the weight of λ_v, $\neg\lambda_v$, and $\neg\theta_{v|\mathbf{u}}$ is 1, and the weight of $\theta_{v|\mathbf{u}}$ is a value $\Pr(V = v|\mathbf{U} = \mathbf{u})$ from a corresponding CPT of the BN. Consequently, the probability of an event e can be computed by weighted model counting of the CNF in conjunction with e.

For example, one of the 32 ($2^{|\mathbf{V}|} = 2^5$) possible CNF models correspond to the following instantiation of the random variables of the BN:

$$C \leftarrow c_2, A \leftarrow a_1, B \leftarrow b_1, T \leftarrow t_1, L \leftarrow l_2.$$

In this model, the following propositional letters are true:

$$\lambda_{c_2}, \lambda_{a_1}, \lambda_{b_1}, \lambda_{t_1}, \lambda_{l_2}, \theta_{c_2}, \theta_{a_1}, \theta_{b_1|c_2}, \theta_{t_1|c_2,a_1}, \theta_{l_2|a_1}.$$

Consequently, the weight of this model is a product of weights of these propositional letters, $0.6 \times 0.5 \times 0.2 \times 0.8 \times 0.01 \approx 0.0005$.

Thus, it is possible to represent and elucidate the logical structure of BNs using the CNF formulation. We can use this in turn to make the connection directly to MLNs.

3.2. Representing Bayesian networks with Markov logic networks

The relationship between these two kinds of networks can best be elucidated by explicitly formulating one in terms of the other. Let us then encode a BN using MLNs by a construction similar to the CNF

Table 4. Set of Propositional Clauses Representing Each Element of Every Conditional Probability Table (CPT) of the BN Given in Figure 1

BN component	*CNF clauses*	
Variable C	$\lambda_{c_1} \vee \lambda_{c_2}$	$\neg\lambda_{c_1} \vee \neg\lambda_{c_2}$
Variable A	$\lambda_{a_1} \vee \lambda_{a_2}$	$\neg\lambda_{a_1} \vee \neg\lambda_{a_2}$
Variable B	$\lambda_{b_1} \vee \lambda_{b_2}$	$\neg\lambda_{b_1} \vee \neg\lambda_{b_2}$
Variable T	$\lambda_{t_1} \vee \lambda_{t_2}$	$\neg\lambda_{t_1} \vee \neg\lambda_{t_2}$
Variable L	$\lambda_{l_1} \vee \lambda_{l_2}$	$\neg\lambda_{l_1} \vee \neg\lambda_{l_2}$
CPT for C	$\lambda_{c_1} \Leftrightarrow \theta_{c_1}$	$\lambda_{c_2} \Leftrightarrow \theta_{c_2}$
CPT for A	$\lambda_{a_1} \Leftrightarrow \theta_{a_1}$	$\lambda_{a_2} \Leftrightarrow \theta_{a_2}$
CPT for B	$\lambda_{b_1} \wedge \lambda_{c_1} \Leftrightarrow \theta_{b_1|c_1}$	$\lambda_{b_1} \wedge \lambda_{c_2} \Leftrightarrow \theta_{b_1|c_2}$
	$\lambda_{b_2} \wedge \lambda_{c_1} \Leftrightarrow \theta_{b_2|c_1}$	$\lambda_{b_2} \wedge \lambda_{c_2} \Leftrightarrow \theta_{b_2|c_2}$
CPT for T	$\lambda_{t_1} \wedge \lambda_{c_1} \wedge \lambda_{a_1} \Leftrightarrow \theta_{t_1|c_1,a_1}$	$\lambda_{t_1} \wedge \lambda_{c_1} \wedge \lambda_{a_2} \Leftrightarrow \theta_{t_1|c_1,a_2}$
	$\lambda_{t_1} \wedge \lambda_{c_2} \wedge \lambda_{a_1} \Leftrightarrow \theta_{t_1|c_2,a_1}$	$\lambda_{t_1} \wedge \lambda_{c_2} \wedge \lambda_{a_2} \Leftrightarrow \theta_{t_1|c_2,a_2}$
	$\lambda_{t_2} \wedge \lambda_{c_1} \wedge \lambda_{a_1} \Leftrightarrow \theta_{t_2|c_1,a_1}$	$\lambda_{t_2} \wedge \lambda_{c_1} \wedge \lambda_{a_2} \Leftrightarrow \theta_{t_2|c_1,a_2}$
	$\lambda_{t_2} \wedge \lambda_{c_2} \wedge \lambda_{a_1} \Leftrightarrow \theta_{t_2|c_2,a_1}$	$\lambda_{t_2} \wedge \lambda_{c_2} \wedge \lambda_{a_2} \Leftrightarrow \theta_{t_2|c_2,a_2}$
CPT for L	$\lambda_{l_1} \wedge \lambda_{a_1} \Leftrightarrow \theta_{l_1|a_1}$	$\lambda_{l_1} \wedge \lambda_{a_2} \Leftrightarrow \theta_{l_1|a_2}$
	$\lambda_{l_2} \wedge \lambda_{a_1} \Leftrightarrow \theta_{l_2|a_1}$	$\lambda_{l_2} \wedge \lambda_{a_2} \Leftrightarrow \theta_{l_2|a_2}$

encoding in the previous section. Consider a set of random variables $\mathbf{V} = \{V_1, \ldots, V_N\}$ of a BN, where $V_i \in \{v_i^1, \ldots, v_i^K\}$. For each value v_i^j of every variable V_i we define a predicate $IV_i(x)$ such that $IV_i(v_i^j) = True \Leftrightarrow V_i = v_i^j$. In BNs, each random variable can take on *one and only one* value, therefore we have to add the following formulas to the corresponding encoding MLN to specify this restriction:

$$\exists v_i^k \; IV_i(v_i^k). \tag{5}$$

$$(v_i^k \neq v_i^l) \wedge IV_i(v_i^k) \Rightarrow \neg IV(v_i^l). \tag{6}$$

Note that formulas (5,6) are ‘‘pure’’ FOL formulas (they are either true or false, meaning that their probabilistic weight is large without bound.)

Consider a set of parameters $\Theta = \{\Theta_1, \ldots, \Theta_N\}$ of the BN we are trying to encode, where each Θ_i is a conditional probability table assigned to a variable V_i. Assuming variable V_i has K parents in the BN, $V_{i_1}, \ldots, V_{i_K}$, then

$$\Theta_i \equiv \{\Pr(V_i = v_i | V_{i_1} = v_{i_1}, \ldots, V_{i_K} = v_{i_K}), \forall v_i, v_{i_1}, \ldots, v_{i_K}\}. \tag{7}$$

For every BN parameter Θ_i, the corresponding encoding MLN contains the following set of formulas

$$\begin{array}{ll} \ldots & \\ w_i & IV_i(v_i) \wedge IV_{i_1}(v_{i_1}) \wedge \ldots \wedge IV_{i_K}(v_{i_K}) \\ \ldots & \end{array} \tag{8}$$

where the weight $w_i = \ln(\Pr(V_i = v_i | V_{i_1} = v_{i_1}, \ldots, V_{i_K} = v_{i_K}))$. Note that the set (8) contains $N_i \times N_{i_1} \times \ldots \times N_{i_K}$ formulas, one formula per each element of the conditional probability table Θ_i. Here N_i and N_{i_j} are the number of values variables V_i and V_{i_j} can take. If $\Pr(V_i = v_i | V_{i_1} = v_{i_1}, \ldots, V_{i_K} = v_{i_K}) = 0$, then we use a ‘‘hard’’ encoding formula (whose weight is large without bound):

$$\neg IV_i(v_i) \wedge \neg IV_{i_1}(v_{i_1}) \wedge \ldots \wedge \neg IV_{i_K}(v_{i_K}). \tag{9}$$

As indicated in equation (4), an MLN represents the following distribution:

$$\Pr(V_1 = v_1, \ldots, V_N = v_N) = \frac{1}{Z} \exp\left(\sum_{\{F_i\}} w_i f_i(v_1, \ldots, v_N)\right), \tag{10}$$

where the sum is taken over all formulas of the MLN. For each formula F_i, w_i is its weight and f_i is its characteristic function:

$$f_i(v_1, \ldots, v_N) = \begin{cases} 1, & F_i(V_1 \leftarrow v_1, \ldots, V_N \leftarrow v_N) = True, \\ 0, & \text{otherwise.} \end{cases}$$

Because of the way we constructed this MLN to encode a BN, equation (10) is a product of elements of the BN conditional probability tables corresponding to the BN instantiation $v_1, \ldots, v_N$. Note also that the partition function Z defined as $Z = \sum_{v_1, \ldots, v_N} \exp(\sum_{\{F_i\}} w_i f_i(v_1, \ldots, v_N))$ is 1 in this case, since values of random variables are mutually exclusive and all the formulas exhaust all the possibilities, thus we are essentially adding up the joint probabilities for all possible instantiations of the set of random variables.

Let us now illustrate with an example how a BN can be encoded by an MLN. Consider a toy BN in Figure 2.

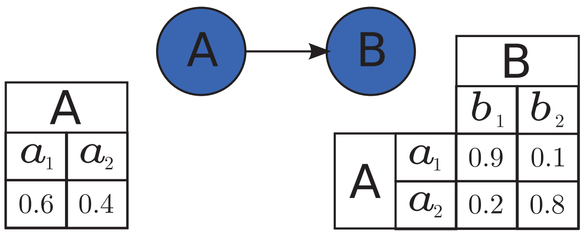


FIG. 2. Structure of a toy Bayesian network. Here *A* and *B* stand for two events.

Let us define predicates $IA(A)$ and $IB(B)$:

$$IA(a_i) = True \Leftrightarrow A = a_i \tag{11}$$

$$IB(b_i) = True \Leftrightarrow B = b_i \tag{12}$$

The MLN encoding the toy BN in figure 2 consist of the following formulas

$$\begin{array}{ll}
 & \exists x \quad IA(x). \\
 & (x \neq y) \wedge IA(x) \Rightarrow \neg IA(y). \\
 & \exists x \quad IB(x). \\
 & (x \neq y) \wedge IB(x) \Rightarrow \neg IB(y). \\
\ln(0.6) & IA(a_1) \\
\ln(0.4) & IA(a_2) \\
\ln(0.9) & IA(a_1) \wedge IB(b_1) \\
\ln(0.1) & IA(a_1) \wedge IB(b_2) \\
\ln(0.2) & IA(a_2) \wedge IB(b_1) \\
\ln(0.8) & IA(a_2) \wedge IB(b_2)
\end{array}$$

The top four first-order formulas of this MLN ensure that the arguments of *IA* and *IB* are mutually exclusive and exhaustive, whereas the remaining formulas represent every element of the corresponding probability tables of the BN. Note that the representational complexity of this MLN and the original BN is the same, since we are essentially mapping every element of the BN to a formula in MLN. Note that the description of the MLN is a bit lengthy, since we want to express exactly the same probability distribution represented by the BN. As a result, most of the formulas of this MLN are propositional (since they were manually instantiated) with explicitly specified weights. The representational power of MLN becomes clearer in this example if we decide to learn the weights from data, in which case the propositional part of the MLN can be replaced with a more concise

$$\begin{array}{l}
IA(+x) \\
IA(+x) \wedge IB(+y)
\end{array}$$

Moreover, this structure does not change, if we change the domains of the logical variables x and y. This demonstrates both that BNs can be represented as MLNs and that part of this representation consists of restrictions not necessary if we do not impose the BN constraints.

3.3. *MLN in the landscape of probabilistic logic methods*

Markov logic networks are based on a combination of MRFs with FOL. An MLN can be seen as a template (Richardson and Domingos, 2006) for constructing MRFs according to specific logical patterns. MLNs are more general than MRFs, since we can represent any MRF by an MLN (in the worst case we can simply list all the cliques of the MRF using statements in propositional logic). On the other hand, an MLN can be seen as a relaxation of otherwise strict logical rules specified in FOL, allowing us to define the likelihood of logical models (Richardson and Domingos, 2006). MLNs can thus be seen also as a generalization of FOL: when the weights are equal and infinitely large, MLNs become FOL. Richardson and Domingos showed that, when all weights are equal and tend to infinity, an MLN represents a uniform distribution over all possible worlds satisfying the set of first-order formulas (Richardson and Domingos, 2006). Moreover, using the MLN with infinitely large weights, we can check whether or not a formula can be logically inferred from the set of MLN formulas by computing the probability that the formula is true and checking whether it is true or not.

Figure 3 schematically depicts the relationship between MLNs and other well-known probabilistic methods. Each method is classified here according to three major components: a probabilistic component, a logic component, and a graphical representation. Note that these components are not completely separate from each other, for example the graphical representation is intrinsically connected with the probabilistic dependency specified in the probabilistic component. On the other hand, using these three components allows us to illustrate the overall relationships between MLNs and other probabilistic methods. In particular, we can see how methods derive from others by imposing more constraints.

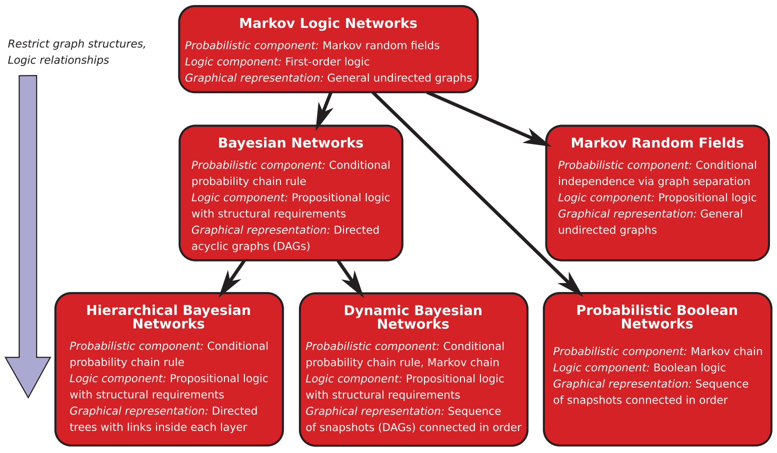

Restrict graph structures, Logic relationships
Markov Logic Networks
Probabilistic component: Markov random fields
Logic component: First-order logic
Graphical representation: General undirected graphs
Bayesian Networks
Probabilistic component: Conditional probability chain rule
Logic component: Propositional logic with structural requirements
Graphical representation: Directed acyclic graphs (DAGs)
Markov Random Fields
Probabilistic component: Conditional independence via graph separation
Logic component: Propositional logic
Graphical representation: General undirected graphs
Hierarchical Bayesian Networks
Probabilistic component: Conditional probability chain rule
Logic component: Propositional logic with structural requirements
Graphical representation: Directed trees with links inside each layer
Dynamic Bayesian Networks
Probabilistic component: Conditional probability chain rule, Markov chain
Logic component: Propositional logic with structural requirements
Graphical representation: Sequence of snapshots (DAGs) connected in order
Probabilistic Boolean Networks
Probabilistic component: Markov chain
Logic component: Boolean logic
Graphical representation: Sequence of snapshots connected in order

FIG. 3. The relationships between MLN and different probabilistic representations.

In Figure 3, we position MRFs and BNs on the same level, right under MLNs, since both are the most expressive methods among those based on propositional logic. MLNs generalize MRFs and BNs by using FOL. Note that BNs typically imply some (temporal) ordering: a true value of a variable may ‘‘cause’’ another variable to be true. Although MRFs imply no such ordering, MLNs use FOL to make such an implication (see previous section). Two well-known methods, HBNs and DBNs, are the specialized versions of BNs; therefore, we place them right under BNs in Figure 3. HBNs and DBNs are essentially BNs with additional structural constraints: an HBN is a directed hierarchical tree where some ‘‘sibling’’ vertices may be linked, and a DBN is a sequence of BNs, representing a systems snapshot, such as a systems state at a moment of time, interlinked in one direction, representing a series of events, such as a progression of time. MLNs, which can express BNs, are also able to represent HBNs and DBNs, whose structural constraints can be easily expressed in FOL. Another well-known class of probabilistic models, probabilistic boolean networks (PBN), deals with systems dynamics, similarly to DBNs. Moreover, a very close relationship between PBNs and DBNs was shown in Lahdesmaki et al. (2006). Thus, we place PBNs on the same level with DBNs and directly under MLNs, since Boolean logic, which PBNs are based on, is directly generalized by FOL.

4. THE APPLICATION OF MLNS TO GENETICS

We have used MLNs to study the interactions of genetic loci in predicting phenotypes (Sakhanenko and Galas, 2010) and to capture the influence of these interconnected loci on phenotypes. We use a regression-type MLN: given N predictor variables X_i (that could represent alleles of genetic markers), predict a value of an outcome variable Y (that could represent a phenotype, for example). Algorithm 1 summarizes the method that handles a model selection problem of finding a model that captures best the probability distribution over the training data:

Algorithm 1: MLN-based predictor selection

```
foreach predictor variable $X_i$ do
    repeat
        shuffle data;
        train and test $MLN(X_i, Known, Y)$;
        obtain a cross-validation score $e(X_i)$;
    until the average score $e(X_i)$ does not change;
if $e(\hat{X}_i)$ is a max outlier then
    $Known = Known \bigcup \{\hat{X}_i\}$;
    go to line 1;
```

The inner *repeat-until* loop trains and evaluates the same model on a reordered, or shuffled, data set: since MLN modeling is path-dependent, this loop reduces the effect of path-dependency on a prediction score. The outer *foreach* loop traverses the set of all predictor variables (genetic markers) and computes how well each marker, together with a small set of known markers, predicts an outcome variable (a phenotype). Once all the markers in the given set are traversed and assigned a score, the most significantly predictive marker is selected and added to the set of known markers, prompting another iteration of the search procedure. The search stops when there are no markers left with significant predictive power. This heuristic search keeps the structure of MLN and the outcome variable Y the same, but considers different subsets of predictor variables X_i in order to find a model that best ''explains'' the variation of Y (Sakhanenko and Galas, 2010). In the next sections, we present and compare two types of MLNs using Algorithm 1, single-marker models and pair-wise models. We also explain what it means for two markers to predict a phenotype *together* when using each of these models.

5. SINGLE-MARKER MODELS

We have used MLNs as an underlying model representation in the method that detects genetic loci determining quantitative phenotypes (Sakhanenko and Galas, 2010). This method is an iterative procedure that scans the set of all possible genetic markers during every iteration and selects a marker that, in combination with other known predictors, has a significant predictive power of the phenotype. The selected marker is then added to the set of known predictors and the method continues to the next iteration.

The first iteration of the method is similar to GWAS in the following way. The method scans all the markers and finds those markers whose individual predictive power is high. However, our method is different from the traditional statistical tools of GWAS: during the next iterations, our method searches for subsets of markers whose compound effect on the phenotype is high. Consequently, our method is a model selection procedure, where the relationships between markers (gene interactions) and phenotypes are hypothesized and encoded by a model, and the method then evaluates all possible such models and identifies the most probable one from the available data. Furthermore, the models are represented using MLNs allowing us to bring the power of both logic and probabilistic reasoning into genetic analyses, which will allow direct extension to arbitrarily complex models.

5.1. The connection with logistic regression

Following the MLN syntax used in Alchemy (Kok et al., 2007), a single-marker model is expressed by the following program:

$$\begin{aligned}&\mathbf{Phenotype}(\mathit{Strain},\ +T)\\&\mathbf{Allele}(\mathit{Strain},\ +\mathit{Marker},\ +V)\Rightarrow\mathbf{Phenotype}(\mathit{Strain},\ +T).\end{aligned}\tag{13}$$

Here, **Phenotype** and **Allele** are logical predicates, and *Strain*, *Marker*, *V*, and *T* are variables. Assuming $\mathit{Marker} = m_1$, $V = v_1$, $T = t_1$, and $\mathit{Strain} = s_1$, predicates $\mathbf{Phenotype}(s_1, t_1)$ and $\mathbf{Allele}(s_1, m_1, v_1)$ encode two facts (two data points) that t_1 *is a phenotype value of strain* s_1 and that v_1 *is an allele of marker* m_1 *for strain* s_1. Furthermore, a logical formula, $\mathbf{Allele}(\mathit{Strain}, m_1, v_1) \Rightarrow \mathbf{Phenotype}(\mathit{Strain}, t_1)$, encodes the following statement:

$$\begin{aligned}&\textbf{for all strains, an allele value } v_1 \textbf{ of a marker } m_1\\&\textbf{implies a phenotype value } t_1.\end{aligned}\tag{14}$$

If $\mathit{Marker} \in \{m_1, \ldots, m_{N_1}\}$, $V \in \{v_1, \ldots, v_{N_2}\}$, $T \in \{t_1, \ldots, t_{N_3}\}$, and $\mathit{Strain} \in \{s_1, \ldots, s_{N_4}\}$, then program (13) is actually a set of N_3 instances of a first-order formula (15) and $N_1 \times N_2 \times N_3$ instances of a first-order formula (16) with *separate* weights w_i and w_{jkl} correspondingly:

$$\ldots$$
$$w_i : \mathbf{Phenotype}(\mathit{Strain}, t_i)\tag{15}$$
$$\ldots$$
$$w_{jkl} : \mathbf{Allele}(\mathit{Strain}, m_j, v_k)\Rightarrow\mathbf{Phenotype}(\mathit{Strain}, t_l)\tag{16}$$
$$\ldots$$

Note that all variables of (15,16) are replaced with various combinations of values, except for the variable *Strain*, which represents the specific, representative organism. Consequently, each formula represents a truth statement for *any* value of the variable *Strain*. A weight assigned to a formula indicates the probability of the truth of the encoded statement: the higher the weight, the greater the difference in log probability between the *world* that satisfies the statement and the one that does not. Note that while a weighted formula (16) models a probability distribution over logic statements (14), a weighted formula (15) models a binomial probability distribution of the phenotype values across all possible strains.

The model represented by the formulas (15) and (16) is equivalent to

$$\begin{aligned} &\ldots \\ w_i &: \mathbf{Phenotype}(\mathit{Strain}, t_i) \qquad (17) \\ &\ldots \\ w_{jkl} &: \mathbf{Allele}(\mathit{Strain}, m_j, v_k) \wedge \mathbf{Phenotype}(\mathit{Strain}, t_l) \qquad (18) \\ &\ldots \end{aligned}$$

if conditioned on **Allele**(*Strain*, m_i, v_j) for all i and j. The equivalency between models (15, 16) and (17, 18) will be discussed later in this section.

Let us introduce characteristic functions a_{ij} and p_k:

$$a_{ij} = \begin{cases} 1, & \text{allele of marker } m_i \text{ is } v_j, \\ 0, & \text{otherwise} \end{cases} \qquad p_k = \begin{cases} 1, & \text{phenotype is } t_k. \\ 0, & \text{otherwise} \end{cases}$$

Note that a_{ij} = 1 and p_k = 1 for a strain s iff the corresponding predicates **Allele**(s, m_i, v_j) and **Phenotype** (s, t_k) are true. The MLN represented by the formulas (17, 18) encodes the joint probability distribution (see equation (4)):

$$\Pr(p_k, a_{11}, \ldots, a_{N_1N_2}) = \frac{1}{Z} exp\left(w_k p_k + \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} w_{ijk} a_{ij} p_k \right). \qquad (19)$$

This equation yields a logistic regression formula where every characteristic function a_{ij} of each allele is a binary predictor variable of a phenotype function p_k (an outcome variable of the logistic regression):

$$\log\left(\frac{\Pr(p_k = 1 | a_{11}, \ldots, a_{N_1N_2})}{\Pr(p_k = 0 | a_{11}, \ldots, a_{N_1N_2})} \right) = w_k + \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} w_{ijk} a_{ij}. \qquad (20)$$

Let us now come back to the equivalency of models (15, 16) and (17, 18). For simplicity we will compare the following two models:

$$\begin{aligned} w_1 &: \ \mathbf{B}(x) \\ w_2 &: \ \mathbf{A}(x) \wedge \mathbf{B}(x) \qquad (21) \end{aligned}$$

$$\begin{aligned} w_1' &: \ \mathbf{B}(x) \\ w_2' &: \ \mathbf{A}(x) \Rightarrow \mathbf{B}(x) \qquad (22) \end{aligned}$$

As in our earlier discussion, we introduce two characteristic functions

$$a = \begin{cases} 1, & \mathbf{A} \text{ is true,} \\ 0, & \text{otherwise,} \end{cases} \qquad b = \begin{cases} 1, & \mathbf{B} \text{ is true,} \\ 0, & \text{otherwise.} \end{cases}$$

We also introduce two binary feature functions representing a truth-value of a compound formula

$$f_1 = \begin{cases} 1, & \mathbf{A} \wedge \mathbf{B} \text{ is true,} \\ 0, & \text{otherwise,} \end{cases} \qquad f_2 = \begin{cases} 1, & \mathbf{A} \Rightarrow \mathbf{B} \text{ is true,} \\ 0, & \text{otherwise.} \end{cases}$$

Table 5. Truth Table for Logical Conjunction and Implication

A	***B***	$A \wedge B$	$A \Rightarrow B$
True	True	True	True
True	False	False	False
False	True	False	True
False	False	False	True

Table 5 provides the truth table for logical concatenation and implication. This table can be rewritten in terms of the characteristic functions (Table 6) revealing the dependency of f_1 and f_2 on a and b, which can be written as:

$$f_1 = ab \text{ and } f_2 = 1 - a(1 - b).$$

The expression of f_2 in terms of a and b is also evident if we recall an equivalent representation of the logical implication through the concatenation and negation: $(\mathbf{A} \Rightarrow \mathbf{B}) \equiv \neg(\mathbf{A} \wedge \neg\mathbf{B})$, sometimes termed a ‘‘contrapositive.’’

In terms similar to (19), we can now express the joint probability distribution encoded by model (21) as

$$\Pr(b, a) = \frac{1}{Z} exp(w_1 b + w_2 f_1). \tag{23}$$

Rewriting this as a logistic regression model, we get:

$$\frac{\Pr(b=1|a)}{\Pr(b=0|a)} = \frac{exp(w_1 + w_2 f_1|_{b=1})}{exp(w_2 f_1|_{b=0})} = \\ = \frac{exp(w_1 + w_2 a)}{exp(0)} = exp(w_1 + w_2 a). \tag{24}$$

Model (22) encodes the joint probability distribution expressed as:

$$\Pr(b, a) = \frac{1}{Z} exp(w'_1 b + w'_2 f_2). \tag{25}$$

and can therefore be represented as a logistic regression formula:

$$\frac{\Pr(b=1|a)}{\Pr(b=0|a)} = \frac{exp(w'_1 + w'_2 f_2|_{b=1})}{exp(w'_2 f_2|_{b=0})} = \\ = \frac{exp(w'_1 + w'_2)}{exp(w'_2(1-a))} = exp(w'_1 + w'_2 a). \tag{26}$$

We can see then that, conditioned on a, the logistic regression models (24) and (25) are equivalent.

5.2. Single-markers models in Algorithm 1

Working with haploid yeast genetics, every marker can have only 2 allele values, A and B, so $N_2 = 2$. When applying a single-marker model (13) to one marker only (such as when we perform the first iteration of the marker search in Algorithm 1), $N_1 = 1$ and the simplified model (20) becomes

Table 6. Relationship Between Characteristic Functions a, b, Representing Two Propositions, and f_1, f_2, Representing a Conjunction and Implication Based on These Propositions

a	b	f_1	f_2
1	1	1	1
1	0	0	0
0	1	0	1
0	0	0	1

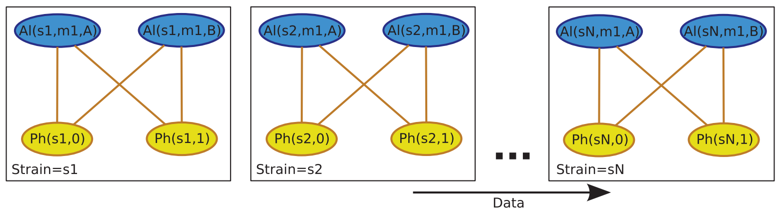


FIG. 4. Structure of a logical model underlying a single-marker MLN based on one marker. A single-marker MLN, defined by (13) and applied to a single marker m_1, encodes a MRF. For the illustration we assumed there are s_N strains, and the phenotype can have two values. **Al**() and **Ph**() stand for **Allele**() and **Phenotype**().

$$\log\left(\frac{\Pr(p_k = 1|a_{11}, a_{12})}{\Pr(p_k = 0|a_{11}, a_{12})}\right) = w_k + w_{11k}a_{11} + w_{12k}a_{12}, \tag{27}$$

where

$$a_{11} = \begin{cases} 1, & \text{allele of } m_1 \text{ is } A, \\ 0, & \text{otherwise} \end{cases} \qquad a_{12} = \begin{cases} 1, & \text{allele of } m_1 \text{ is } B. \\ 0, & \text{otherwise} \end{cases}$$

Figure 4 shows a structure of a MRF imposed by the logical formulas of the single-marker MLN (13) applied to one marker. Each node of the structure graph corresponds to a predicate (either **Allele** or **Phenotype**) whose variables are substituted with every possible combination of values. There is an edge between two nodes of the graph if the corresponding predicates appear in the same formula. In our example, edges show all possible dependencies of **Phenotype** on **Allele**: probabilistic dependency of p_1 and p_2 on a_{11} and a_{12}, and thus every edge is assigned a probabilistic weight. Note that the graphical structure is disjoint, since there are no edges between nodes corresponding to different strains. Note that edges, such as **Allele**(s_1, m_1, A) – **Phenotype**(s_1, 0), **Allele**(s_2, m_1, A) – **Phenotype**(s_2, 0), etc, are instances of the same statistical template, **Allele**($\cdot$, m_1, A) – **Phenotype**($\cdot$, 0), and thus are assigned the same weight, learned from the entire network.

At the *second* iteration of Algorithm 1 using a single-marker model, each model is applied to two markers, one of which is a known marker $\tilde{m}_1$ selected after the first iteration. Therefore, $N_1 = 2$ and the model (20) becomes

$$\log\left(\frac{\Pr(p_k = 1|a_{11}, a_{12}, a_{21}, a_{22})}{\Pr(p_k = 0|a_{11}, a_{12}, a_{21}, a_{22})}\right) = w_k + w_{11k}a_{11} + w_{12k}a_{12} + w_{21k}a_{21} + w_{22k}a_{22}, \tag{28}$$

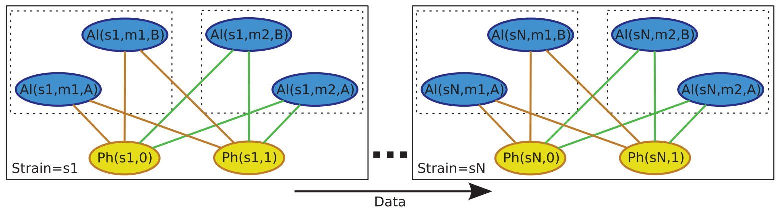


FIG. 5. Structure of a logical model underlying a single-marker MLN based on two markers. A single-marker MLN, defined by (13) and applied to a pair of markers m_1 and m_2, encodes a MRF. We assume there is a data set with s_N strains, and for each strain the phenotype can have one of two possible values. In brown we show the subnetwork, which is the entire network in figure 4, illustrating the increase of complexity of the model when switching to the second iteration of Algorithm 1.

where

$$a_{11} = \begin{cases} 1, & \text{allele of } \tilde{m}_1 \text{ is } A, \\ 0, & \text{otherwise} \end{cases} \qquad a_{12} = \begin{cases} 1, & \text{allele of } \tilde{m}_1 \text{ is } B, \\ 0, & \text{otherwise} \end{cases}$$

$$a_{21} = \begin{cases} 1, & \text{allele of } m_2 \text{ is } A, \\ 0, & \text{otherwise} \end{cases} \qquad a_{22} = \begin{cases} 1, & \text{allele of } m_2 \text{ is } B. \\ 0, & \text{otherwise} \end{cases}$$

Figure 5 shows a graphical representation of a model generated for two markers at the second iteration. Notice that this model is similar to the model generated at the first iteration (Fig. 4). The only difference is in the number of predictor variables.

6. PAIR-WISE MODEL

Like a single-marker model (13), a pair-wise model is expressed by the following program including more than one marker and variant:

$$\begin{aligned} &\mathbf{Phenotype}(Strain, +T) \\ &\mathbf{Allele}(Strain, +M_1, +V_1) \wedge \mathbf{Allele}(Strain, +M_2, +V_2) \Rightarrow \\ &\quad \Rightarrow \mathbf{Phenotype}(Strain, +T). \end{aligned} \tag{29}$$

Assuming $M_1 = m_1$, $V_1 = v_1$, $M_2 = m_2$, $V_2 = v_2$, and $T = t_1$, this program encodes a statement: *for all strains, the allele values v_1 and v_2 of markers m_1 and m_2 together (as a pair) imply a phenotype value t_1.*

As in the case of a single-marker model, the pair-wise MLN (29) encodes a joint probability distribution:

$$\Pr(p_m, a_{11}, \ldots, a_{N_1N_2}) = \frac{1}{Z} exp\left(w_m p_m + \sum_{i=1}^{N_1}\sum_{k=1}^{N_2}\sum_{j=1}^{N_1}\sum_{l=1}^{N_2} w_{ijm}^{kl} a_{ik} a_{jl} p_m \right). \tag{30}$$

Since a genetic marker cannot have two different allele values at the same time, which means that $a_{im}a_{in} = 0$ if $m \neq n$, expression (30) can be rewritten as:

$$\begin{aligned} &\Pr(p_m, a_{11}, \ldots, a_{N_1N_2}, b_{11}^{11}, \ldots, b_{N_2N_2}^{N_1N_1}) = \\ &= \frac{1}{Z} exp\left(w_m p_m + \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} w_{ijm} a_{ij} p_m + \right. \\ &\left. + \sum_{\substack{(i,j)\in N_1\times N_1 \\ i\neq j}} \sum_{(k,l)\in N_2\times N_2} w_{ijm}^{kl} b_{ij}^{kl} p_m \right), \end{aligned} \tag{31}$$

where a_{ij} are the same characteristic functions as in the single-marker model case, and

$$b_{ij}^{kl} = \begin{cases} 1, & \text{allele of marker } m_i \text{ is } v_k \text{ and allele of marker } m_j \text{ is } v_l. \\ 0, & \text{otherwise} \end{cases}$$

Note that $b_{ij}^{kl} = a_{ik}a_{jl}$.

The pair-wise MLN can be seen as a logistic regression model where all characteristic functions a_{ij} and b_{ij}^{kl} are the predictor variables of an outcome variable p_m:

$$\log\left(\frac{\Pr(p_m = 1 | a_{11}, \ldots, a_{N_1N_2}, b_{11}^{11}, \ldots, b_{N_2N_2}^{N_1N_1})}{\Pr(p_m = 0 | a_{11}, \ldots, a_{N_1N_2}, b_{11}^{11}, \ldots, b_{N_2N_2}^{N_1N_1})} \right) = w_m + \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} w_{ijm} a_{ij} + \sum_{\substack{(i,j)\in N_1\times N_1 \\ i\neq j}} \sum_{(k,l)\in N_2\times N_2} w_{ijm}^{kl} b_{ij}^{kl}. \tag{32}$$

Note that this can also be seen as *logistic regression with interaction terms* of every pair of predictor variables a_{ij}.

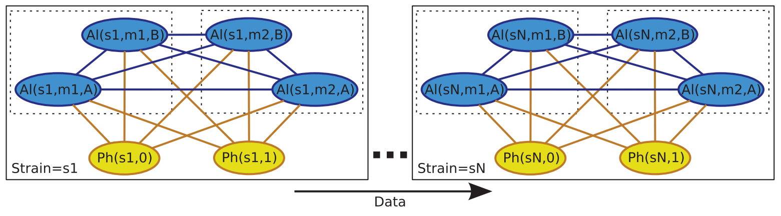


FIG. 6. Structure of a logical model underlying a pair-wise MLN based on two markers. A pair-wise MLN, defined by (29) and applied to a pair of markers m_1 and m_2, encodes a MRF. As before, we assume a data set with s_N strains and two phenotypes. In brown we show the sub-network, which is the entire network in figure 5, illustrating the increase of complexity when switching from a single-marker model to a pair-wise model. All the new edges, not present in figure 5, correspond to the pair-wise interactions and are colored in blue.

When applying a pair-wise model (29) to two markers, $N_1 = 2$ and the simplified model (32) becomes

$$\begin{aligned}
&\log\left(\frac{\Pr(p_m = 1|a_{11}, a_{12}, a_{21}, a_{22})}{\Pr(p_m = 0|a_{11}, a_{12}, a_{21}, a_{22})}\right)\\
&= w_m + w_{11m}a_{11} + w_{12m}a_{12} + w_{21m}a_{21} + w_{22m}a_{22} +\\
&+ w^{11}_{12m}b^{11}_{12} + w^{12}_{12m}b^{12}_{12} + w^{21}_{12m}b^{21}_{12} + w^{22}_{12m}b^{22}_{12} +\\
&+ w^{11}_{21m}b^{11}_{21} + w^{12}_{21m}b^{12}_{21} + w^{21}_{21m}b^{21}_{21} + w^{22}_{21m}b^{22}_{21}.
\end{aligned} \tag{33}$$

Recall that p_m is a characteristic function equal to 1 when the phenotype has a value t_m. Note that this long expression of logistic regression with many predictor variables is compactly represented by a pair-wise model (29) emphasizing the representational power of MLN.

The second line of equation (33) is similar to equation (28) of the single-marker model. That is why all the informative markers identified by algorithm 1 using a single-marker model can be also identified using a pair-wise model. The third and fourth lines of equation (33) are the interaction terms between alleles of different markers. Consider two markers that have a specific combination of alleles that are predictive of a phenotype, but that have no effect of the phenotype on their own (for example if the markers have a synthetic interaction). Equation (28) would not work on these markers, since each weight representing a predictive power of an individual marker will be zero. Similarly, the second line of equation (33) would disappear as well (the corresponding weights would also be zero). However, some of the weights of the interaction terms of equation (33) would not be zero allowing us to detect the synergy between two markers by using the pair-wise model.

Figure 6 illustrates a graphical representation of a pair-wise model generated for two markers. Notice that this model is somewhat similar to the single-marker model generated at the second iteration (Fig. 5). However, the major difference is that in a pair-wise model predictor variables are interconnected, forming cliques with phenotype variables (see blue edges in Fig. 6). Moreover, the probabilistic dependency is modeled between two predictor variables (alleles) and an outcome variable (a phenotype value); thus, the weights are assigned to every triangle (clique) connecting two marker alleles and a phenotype value.

Table 7 summarizes the three models described above: a single-marker MLN based on one and two markers and a pair-wise MLN based on two markers, and their corresponding representation as logistic regressions. The complexity of the regression formula for the pair-wise case points to the significant advantage of the first order logic expression, even when as here they can be effectively expressed as logistic regressions. The compactness of the expression of the model is striking. Many more complex models cannot, of course, be expressed at all as logistic regressions, even extremely large ones. The power of the compactness of expression of models in MLNs is illustrated by the rapid expansion of the equivalent logistic regressions shown above, relative to the modest expansion of the model complexity. Since we are

Table 7. Summary of Single-Marker MLNs Based on One and Two Markers as Well as a Pair-Wise MLN Based on Two Markers

	First-order logic	*Logistic regression*
1	**Allele**(S, $+M$, $+V$)$\Rightarrow$ **Phenotype**(S, $+T$), $M = m_1$	$\log\left(\frac{\Pr(p_k=1\|a_{11},a_{12})}{\Pr(p_k=0\|a_{11},a_{12})}\right) = w_k + w_{11k}a_{11} + w_{12k}a_{12}$
2	**Allele**(S, $+M$, $+V$)$\Rightarrow$ **Phenotype**(S, $+T$), $M \in \{m_1, m_2\}$	$\log\left(\frac{\Pr(p_k=1\|a_{11},a_{12},a_{21},a_{22})}{\Pr(p_k=0\|a_{11},a_{12},a_{21},a_{22})}\right) =$ $w_k + w_{11k}a_{11} + w_{12k}a_{12} + w_{21k}a_{21} + w_{22k}a_{22}$
3	**Allele**(S, $+M_1$, $+V_1$)$\wedge$ **Allele**(S, $+M_2$, $+V_2$)$\Rightarrow$ **Phenotype**(S, $+T$), $M \in \{m_1, m_2\}$	$\log\left(\frac{\Pr(p_m=1\|a_{11},a_{12},a_{21},a_{22})}{\Pr(p_m=0\|a_{11},a_{12},a_{21},a_{22})}\right) =$ $w_m + w_{11m}a_{11} + w_{12m}a_{12} + w_{21m}a_{21} + w_{22m}a_{22} +$ $w^{11}_{12m}b^{11}_{12} + w^{12}_{12m}b^{12}_{12} + w^{21}_{12m}b^{21}_{12} + w^{22}_{12m}b^{22}_{12} +$ $w^{11}_{21m}b^{11}_{21} + w^{12}_{21m}b^{12}_{21} + w^{21}_{21m}b^{21}_{21} + w^{22}_{21m}b^{22}_{21}$

The left column shows the formulas expressed in FOL defining the structure of the models, and the right column shows their corresponding representation as logistic regressions.

only scratching the surface of the underlying complexity of models that will be useful in future, the lesson here is evident.

7. OTHER APPLICATIONS IN BIOLOGY AND MEDICINE

In its most abstract form, genetic analysis is directed to the detection of causative patterns in heritable genomes that predict well the phenotypes of interest. In a similar vein, the detection of patterns in data that predict experimental outcomes is at the heart of almost any modern biological or medical problem. Framed this way we can quickly conclude that MLNs and other probabilistic logic methods are well suited to the solution of these kinds of problems. Problems in this class include predicting sub-networks of the functioning complex networks that are central to cellular function (we do not presume to infer the entire networks of any system quite yet). This includes inferring networks from mRNA expression data like array data and RNA Seq data. An important characteristic of biological problems of this kind is that we sometimes know something about the system, or have specific hypotheses about the system, that need to be incorporated into the models during inference. MLNs are perfectly well suited to this kind of problem.

Classes of data like mRNA expression data—which include microRNA data, alternative splicing data (exon usage and alternative UTR use), protein expression level data, metabolite level data, transcription factor binding site occupation levels, histone modification and methylation patterns, and a variety of other kinds of molecular and non-molecular data—present the same kinds of problems. The challenge is to use these different data sets together to extract more information about the reality of the complex models that predict function than can be inferred from the individual sets by themselves. Since we also know something about how one kind of data is related to the others (e.g., proteins are made from mRNA at a rate determined by their sequences, translation factors and the levels of modulators like miRNAs) we need to represent this knowledge as constraints on the models when using the data together. This concept is a central aspect of true data integration and, as the knowledge of biology and medicine grows, is a major potential application for the methods we describe here.

8. CONCLUSION

Probabilistic logic methods, particularly those that use graphical model representations as a central part of the structure, are powerful tools in the analysis of data. They are particularly potent in dealing with biological data, as the field is currently in a state where the detailed data generation volume is enormous and the knowledge of the underlying complex systems is substantial, but very partial and often uncertain, with non-negligible quantitative error levels. Taking all of this into account—huge diverse data sets, partial knowledge of various types—is all but impossible without the systematic structures provided by methods and models that combine the rigors of representing and tracking multiple logical relationships with the scoring of likelihoods reflecting reality in a balanced and integrated fashion. Neither strictly logical nor completely statistical and probabilistic methods can properly manage the complexity that is presented by the current state and dynamics of modern biological and medical research. Together they hold a great deal of promise.

We have summarized a wide range of both simple and complex mathematical and computational advances by focusing on a very general method, the MLN method, and attempting to put it into the context of some methods more familiar to most experimental and computational biologists. In the computational analysis of biological problems, analyzing data, inferring networks and complex models, and estimating model parameters, it has been common to use a range of methods based on various probabilistic logic models, sometimes collectively called ''machine learning'' methods. Inference methods based on the widely used BNs fall into this class, as do those based on HBNs, PBNs, and MLNs. We have tried to illustrate this landscape of methods, particularly contrasting MLNs with BNs, and describe some of the strengths and limitations. The pivotal concepts for the sketching of this landscape and comparisons of methods has been probabilistic graphical models, the structures of the graphs, and the scope of the logical language repertoire (from FOL to Boolean logic).

While these methods are powerful, the computational intensity is substantial for the most general, and therefore most potent of them. This certainly includes the application of MLN, for which the pair-wise genetic model has proven to be extremely computationally intensive. This limitation is, of course, temporary for several reasons, but two of them are evident. First, the available computing power to be brought to these problems is increasing rapidly and the costs are dropping. In addition, the engineering of software, and specialized hardware, for efficiency and speed in executing these algorithms, particularly using parallelization methods and hardware embodiment of algorithms, is substantial and growing. It is clear that the complexity of our understanding of biological problems, and the application of probabilistic logic methods are both at the very beginnings of their growth curves, and the future is very rich with possibilities.

ACKNOWLEDGMENTS

We are grateful to a number of colleagues for comments and suggestions on the manuscript: Aimee Dudley, Alex Skupin, and Hong Li. This work has been supported by the National Science Foundation FIBR Program and the Luxembourg-ISB Program.

DISCLOSURE STATEMENT

No competing financial interests exist.

REFERENCES

Baldi, P., and Brunak, S. 2001. *Bioinformatics: The Machine Learning Approach.* MIT Press, Cambridge, MA.

Darwiche, A. 2002. A logical approach to factoring belief networks. *Proc. KR* 409–420.

Ershov, B., Yu. L., and Palyutin, E.A. 1986. *Mathematical Logic.* Imported Publications, New York.

Kindermann, R., and Snell, J.L. 1980. *Markov Random Fields and Their Applications.* American Mathematical Society, New York.

Kok, S., Sumner, M., Richardson, M., et al. 2007. The alchemy system for statistical relational AI [Technical report]. University of Washington, Seattle.

Koller, D., and Friedman, N. 2009. *Probabilistic Graphical Models: Principles and Techniques.* MIT Press, Cambridge, MA.

Lahdesmaki, H., Hautaniemi, S., Shmulevich, I., et al. 2006. Relationships between probabilistic Boolean networks and dynamic Bayesian networks as models of gene regulatory networks. *J. Signal Process.* 86, 814–834.

Luger, G.F. 2008. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving.* Addison-Wesley, New York.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, New York.

Pietra, S.D., Pietra, V.D., and Lafferty, J. 1997. Inducing features of random fields. *IEEE Trans. Pattern Anal. Mach. Intell.* 19, 380–393.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Mach. Learn.* 62, 107–136.

Sakhanenko, N.A., and Galas, D.J. 2010. Markov logic networks in the analysis of genetic data. *J. Comput. Biol.* 17, 1491–1508.

Address correspondence to:
Dr. David J. Galas
The Institute for Systems Biology
401 Terry Avenue North
Seattle, WA 98109

E-mail: dgalas@systemsbiology.org