# Learning Bayesian Network Parameters under Equivalence Constraints

Tiansheng Yao[1,*], Arthur Choi, Adnan Darwiche

*Computer Science Department*
*University of California, Los Angeles*
*Los Angeles, CA 90095 USA*

**Abstract**

We propose a principled approach for learning parameters in Bayesian networks from incomplete datasets, where the examples of a dataset are subject to equivalence constraints. These equivalence constraints arise from datasets where examples are tied together, in that we may not know the value of a particular variable, but whatever that value is, we know it must be the same across different examples. We formalize the problem by defining the notion of a constrained dataset and a corresponding constrained likelihood that we seek to optimize. We further propose a new learning algorithm that can effectively learn more accurate Bayesian networks using equivalence constraints, which we demonstrate empirically. Moreover, we highlight how our general approach can be brought to bear on more specialized learning tasks, such as those in semi-supervised clustering and topic modeling, where more domain-specific approaches were previously developed.

## 1. Introduction

In machine learning tasks, the examples of a dataset are generally assumed to be independent and identically distributed (i.i.d.). There are numerous situations, however, where this assumption does not hold, and there may be additional information available that ties together the examples of a dataset. We can then, in turn, exploit this background knowledge to learn more accurate models.

Consider, as a motivating example, the following scenarios that arise in medical diagnosis, where we would like to learn a model that could be used to diagnose diseases from symptoms. Typically, we would have data consisting of patient records, which we assume to be independent. However, we may obtain further information that ties some of these records together. For example, we may learn that two patients are identical twins, and hence may both be subject to increased risk of certain genetic diseases, i.e., they share the same genetic variants that may cause certain genetic disorders. We may

*Corresponding author
[1]University of California, Los Angeles, 4801 Boelter Hall, Los Angeles, CA 90095 USA. Tel. 1-310-794-4343, Fax. 1-310-794-5057.

also, for example, learn that two patients were both exposed to a third patient, who was diagnosed with a contagious disease. When learning a model from data, we would like to be able to take advantage of this type of additional information, when it is available.

We can view this type of additional information more generally as *equivalence constraints* that bear on an incomplete dataset, where we may not know the particular value of a variable, but whatever that value is, we know that it must be the same across different examples in our dataset. In this paper, we introduce a simple but principled way to deal with such additional information. In particular, we introduce and formalize the problem of *learning under equivalence constraints.* We first introduce the notion of a *constrained dataset*, which implies a corresponding *constrained log likelihood.* We then define the problem of learning the parameters of a Bayesian network from a constrained dataset, by maximizing the constrained log likelihood.

There are a variety of applications, across a variety of different domains, that can be viewed as learning from a constrained dataset. For example, in the information extraction task of named-entity recognition, we seek to label the elements of a text by the type of entity that they refer to (e.g., in an abstract for a talk, we would want to identify those elements that refer to the speaker). Hence, if we see a name that appears multiple times in the same text, we may presume that they all refer to an entity of the same type [1] (an equivalence constraint). As another example, in the task of (vision-based) activity recognition [2], our goal is to annotate each frame of a video by the activity that a human subject is involved in. In this case, a video could be partially annotated by a human labeler, specifying that different frames of a video that depict the same activity (again, an equivalence constraint).

Indeed, the notion of an equivalence constraint, for the purposes of learning, has appeared before in a variety of different domains (either implicitly or explicitly), where a variety of domain-specific approaches have been developed for disparate and specialized tasks. One notable domain, is that of semi-supervised clustering [3]. Here, the notion of a *must-link constraint* was proposed for $k$-means clustering, to constrain those examples that are known to belong to the same cluster;[2] see, e.g., [4, 5]. For example, when clustering different movies, a user may find that the clusters they learned assigned two different movies to two different clusters, when they should have been assigned to the same cluster (say, based on their personal preferences). In the topic modeling domain, a significantly different approach was proposed to accommodate must-link constraints (based on Dirichlet forest priors), to assert that different words should appear in the same topic (with high probability) [6].

In this paper, we show how the different tasks described above can be viewed uniformly as learning a Bayesian network from a dataset that is subject to equivalence constraints. We further propose a simple but principled way of learning a Bayesian network from such a dataset, which is competitive with, and sometimes outperforming, more specialized approaches that were developed in their own domains. Given the simplicity and generality of our approach, we further relieve the need to (a) derive new and tailored solutions for applications in new domains, or otherwise (b) adapt or generalize existing solutions from another domain (both non-trivial tasks).

[2] Similarly, *must-not-link constraints* were also considered, for examples that belong to different clusters.

Our paper is organized as follows. In Section 2, we review the task of learning Bayesian networks from incomplete datasets. In Section 3, we introduce the notion of a constrained dataset, and in Section 4 we introduce the corresponding notion of a constrained log likelihood. In Section 5, we consider the problem of evaluating the constrained log likelihood, and in Section 6, we discuss an iterative algorithm for optimizing it. In Section 7, we evaluate our approach for learning Bayesian networks from constrained datasets, further comparing it with more specialized approaches from two different domains: semi-supervised clustering and topic modeling. Finally, we review related work in Section 8, and conclude in Section 9.

## 2. Technical Preliminaries

We use upper case letters ($X$) to denote variables and lower case letters ($x$) to denote their values. Sets of variables are denoted by bold-face upper case letters ($\mathbf{X}$), and their instantiations by bold-face lower case letters ($\mathbf{x}$). Generally, we will use $X$ to denote a variable in a Bayesian network and $\mathbf{U}$ to denote its parents. A network parameter will further have the general form $\theta_{x|\mathbf{u}}$, representing the probability $Pr(X=x|\mathbf{U}=\mathbf{u})$. We will further use $\theta$ to denote the set of all network parameters.

Given a network structure $G$, our goal is to learn the parameters of the corresponding Bayesian network, from an incomplete dataset. We use $\mathcal{D}$ to denote a dataset, and $\mathbf{d}_i$ to denote an example. Typically, one seeks parameter estimates $\theta$ that maximize the log likelihood, defined as:

$$LL(\theta \mid \mathcal{D}) = \sum_{i=1}^{N} \log Pr_\theta(\mathbf{d}_i), \tag{1}$$

where $Pr_\theta$ is the distribution induced by network structure $G$ and parameters $\theta$. In the case of complete data, the maximum likelihood (ML) parameters are unique and easily obtainable. In the case of incomplete data, obtaining the ML parameter estimates is more difficult, and iterative algorithms, such as Expectation-Maximization (EM) [7, 8], are typically employed.

In this paper, we are interested in estimating the parameters of a Bayesian network from a similar perspective, but subject to certain equivalence constraints, which we introduce in the next section. Our approach is largely motivated by the use of *meta-networks*, which are more commonly used for Bayesian parameter estimation [9, 10]. In a meta-network, the parameters $\theta$ that we want to learn are represented explicitly as nodes in the network. Moreover, the dataset $\mathcal{D}$ is represented by replicating the original Bayesian network, now called a *base network*, as many times as there are examples $\mathbf{d}_i$ in the data. Each example $\mathbf{d}_i$ of the dataset $\mathcal{D}$ is then asserted as evidence in its corresponding base network. Such a meta-network explicitly encodes an i.i.d. assumption on the dataset $\mathcal{D}$, where data examples are conditionally independent given the parameters $\theta$ (which follows from d-separation).

**Example 1.** Consider a Bayesian network $A \rightarrow B$ with Boolean variables $A$ and $B$,

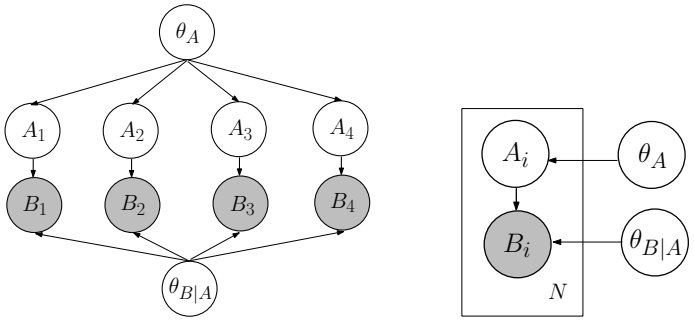


Figure 1: A meta-network for a Bayesian network $A \rightarrow B$, and the dataset $\mathfrak{D}$ given in Example 1 (left), and the corresponding plate representation (right).

and the following incomplete dataset $\mathfrak{D}$:

| example | $A$ | $B$ |
|---|---|---|
| 1 | ? | true |
| 2 | ? | false |
| 3 | ? | true |
| 4 | ? | true |

Here we have four examples, each row representing a different example $\mathbf{d}_i$. The symbol ? denotes a missing value. In this example, variable $B$ is fully observed (its value is never missing), whereas variable $A$ is fully unobserved (its value is always missing). The corresponding meta-network for this dataset is depicted in Figure 1, along with the corresponding plate representation, which is also commonly used [10]. In the meta-network, each example $\mathbf{d}_i$ has a corresponding base network $A_i \rightarrow B_i$ where instance $\mathbf{d}_i$ is asserted as evidence (observed nodes are shaded). Moreover, the network parameters $\theta_A$ and $\theta_{B|A}$ are represented explicitly as random variables. Here, the probability of variable $A$ depends on the parameters $\theta_A$, and the probability of variable $B$ depends on its parent $B$ and the parameters $\theta_{B|A}$.

In Bayesian parameter estimation, one typically estimates the network parameters by considering the posterior distribution obtained from conditioning the meta-network on the given dataset $\mathfrak{D}$. For our purposes, we want to condition instead on the parameter variables, asserting a given parameterization $\theta$ (as in maximum likelihood estimation). In this case, one induces a meta-distribution $\mathcal{P}(. \mid \theta)$ over the variables of the base networks. If our dataset $\mathfrak{D}$ is specified over variables $\mathbf{X}$, then let $\mathbf{X}_i$ denote the variables of the base-network corresponding to example $i$ in the meta-network. Moreover, let $\mathbf{X}_{1:N} = \cup_{i=1}^{N} \mathbf{X}_i$ denote the set of all base-network variables in the meta-network, and let $\mathbf{x}_{1:N}$ denote a corresponding instantiation. Our meta-distribution is then:

$$\mathcal{P}(\mathbf{x}_{1:N} \mid \theta) = \prod_{i=1}^{N} \mathcal{P}(\mathbf{x}_i \mid \theta) = \prod_{i=1}^{N} Pr_\theta(\mathbf{x}_i)$$

where, again, $Pr_\theta(\mathbf{X}) = \mathcal{P}(\mathbf{X}_i \mid \theta)$ is the distribution induced by a network with

structure $G$ and parameters $\theta$. The likelihood of a set of parameters $\theta$ is then:

$$\mathcal{P}(\mathbf{\mathcal{D}} \mid \theta) = \sum_{\mathbf{x}_{1:N} \sim \mathbf{\mathcal{D}}} \mathcal{P}(\mathbf{x}_{1:N} \mid \theta) = \prod_{i=1}^{N} Pr_{\theta}(\mathbf{d}_i) \tag{2}$$

where $\mathbf{x}_{1:N} \sim \mathbf{\mathcal{D}}$ denotes compatibility between a complete instantiation $\mathbf{x}_{1:N}$ and the (incomplete) dataset $\mathbf{\mathcal{D}}$, i.e., each $\mathbf{x}_{1:N}$ is a valid completion of dataset $\mathbf{\mathcal{D}}$. The corresponding log likelihood, of the meta-network, is thus equivalent to the log likelihood of Equation 1:

$$\log \mathcal{P}(\mathbf{\mathcal{D}} \mid \theta) = LL(\theta \mid \mathbf{\mathcal{D}}).$$

Again, when estimating the parameters of a Bayesian network from data, we typically seek those parameters that maximize the log likelihood. In this paper, we take advantage of this meta-network perspective on this parameter estimation task, as it facilitates the learning of Bayesian networks from constrained datasets, which we discuss next.

## 3. Constrained Datasets

As a motivating example, consider the following problem that we may encounter in the domain of medical diagnosis.

**Example 2.** Consider an incomplete dataset $\mathbf{\mathcal{D}}$ composed of four medical records:

| record | $V$ | $D$ | $H$ | $T$ |
|---|---|---|---|---|
| 1 | true | ? | false | true |
| 2 | ? | ? | true | true |
| 3 | false | ? | ? | true |
| 4 | ? | ? | false | ? |

where each row represents a medical record $\mathbf{d}_i$ over four binary features:

- "has genetic variant" ($V$), which can be true or false,
- "has diabetes" ($D$), which can be true or false,
- "has increased hunger" ($H$), which can be true or false,
- "has increased thirst" ($T$), which can be true or false.

Here, we consider a genetic variant, whose presence increases the risk of diabetes in a patient, and two symptoms of diabetes, increased hunger and thirst. Suppose that we obtain information that record 2 and record 4 correspond to two patients, who are identical twins, and hence share the same genetic variants. Suppose, however, that we do not know whether the genetic variant is present or absent in the twins: we know that either they both possess the variant or both do not possess the variant. Even if both twins possess the variant ($V$), they may not both develop diabetes ($D$), nor may they exhibit the same symptoms ($H$ and $T$).

Our goal is to take advantage of the type of background information available in Example 2 (in this case, that two records correspond to identical twins), in order to learn more accurate models. In particular, we view this type of background information as an *equivalence constraint*, which constraints the values of a variable to be the same across different examples in a dataset, although not necessarily to a specific value.

More formally, consider the following definition of an equivalence constraint.

**Definition 1** (Equivalence Constraint)**.** An *equivalence constraint* on a variable $X$, in a dataset $\mathcal{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_N\}$, is an index set $C_X \subseteq \{1, \ldots, N\}$, which constrains the corresponding instances of $X$ to have the same value, i.e., we have that $X_i \equiv X_j$ in examples $\mathbf{d}_i$ and $\mathbf{d}_j$, for all pairs of indices $i, j \in C_X$.

We further define a trivial equivalence constraint $C_X$ to be one that contains a single index $i$, i.e., variable $X_i$ must be equivalent to itself, which is vacuous. We denote a *set* of equivalence constraints on a variable $X$ by $\boldsymbol{\mathcal{C}}_X$. Typically, we consider sets $\boldsymbol{\mathcal{C}}_X$ of equivalence constraints that partition the examples $\{1, \ldots, N\}$. Such a partition may also include trivial constraints $C_X$ over a single index. We further assume, without loss of generality, that the equivalence constraints $C_X$ of a set $\boldsymbol{\mathcal{C}}_X$ are pairwise disjoint (otherwise, we could merge them into a single equivalence constraint).

**Example 3.** Consider again the medical dataset of Example 2. In this example, we had an equivalence constraint $C_V = \{2, 4\}$ that asserts that the state of a genetic variant in examples 2 and 4 must be the same (either both true, and the variant is present, or both false, and the variant is absent). We can also partition the examples into a set of equivalence constraints $\boldsymbol{\mathcal{C}}_G = \{\{2, 4\}, \{1\}, \{3\}\}$, which includes two equivalence constraints which are trivial: $\{1\}$ and $\{3\}$. This dataset also respects the equivalence constraint $C_T = \{1, 2, 3\}$, as all three examples observe the same value true on the variable $T$.

In general, we may constrain multiple variables $X$ in a dataset. We thus introduce the notion of a *constrained dataset*.

**Definition 2** (Constrained Dataset)**.** A *constrained dataset* over variables $\mathbf{X}$, is composed of two components: (1) a traditional dataset $\mathcal{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_N\}$ over variables $\mathbf{X}$, where each example $\mathbf{d}_i$ is a partial instantiation of the variables; and (2) a set of equivalence constraints $\boldsymbol{\mathcal{C}} = \{\boldsymbol{\mathcal{C}}_X \mid X \in \mathbf{X}\}$ over dataset $\mathcal{D}$, where each $\boldsymbol{\mathcal{C}}_X \in \boldsymbol{\mathcal{C}}$ is a set of equivalence constraints on variable $X \in \mathbf{X}$.

Finally, we will in general assume that the values of variables involved in equivalence constraints have hidden values in the data. Constraints on fixed values are not useful here, when they are already equivalent (otherwise, it is not meaningful to assert an equivalence constraint on two variables that are fixed to two different values).

## 4. Learning with Constrained Datasets

Now that we have introduced the notion of a constrained dataset, we can consider the problem of learning from one. Given a traditional dataset $\mathcal{D}$, we would typically want to seek parameter estimates $\theta$ that maximize the likelihood of Equation 2. When

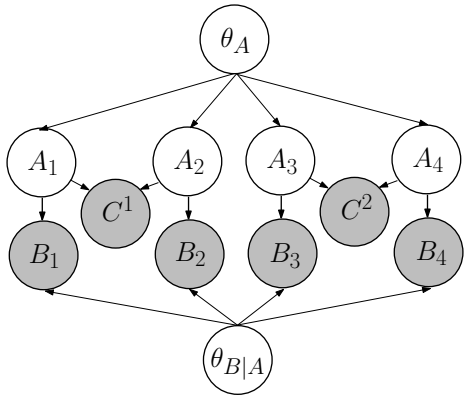


Figure 2: A meta-network for a Bayesian network $A \to B$, and the constrained dataset given in Example 4.

we subject the dataset $\mathcal{D}$ to equivalence constraints $\boldsymbol{\mathcal{C}}$, we propose instead to maximize the likelihood, but conditional on the equivalence constraints $\boldsymbol{\mathcal{C}}$. That is, we seek parameters $\theta$ that maximize the *constrained likelihood*:

$$\mathcal{P}(\mathcal{D} \mid \boldsymbol{\mathcal{C}}, \theta) = \sum_{\mathbf{x}_{1:N} \sim \mathcal{D}} \mathcal{P}(\mathbf{x}_{1:N} \mid \boldsymbol{\mathcal{C}}, \theta)$$

Next, we show how to represent and evaluate this conditional distribution.

### *4.1. Encoding Equivalence Constraints*

We can encode each individual equivalence constraint $C_X \in \boldsymbol{\mathcal{C}}_X$ for each set $\boldsymbol{\mathcal{C}}_X \in \boldsymbol{\mathcal{C}}$, locally in the meta-network. In particular, we introduce an observed variable $C_X$, fixed to the value $c_x$, where $C_X$ has parents $\mathbf{X}_C = \{X_i \mid i \in C_X\}$. The CPT of $C_X$ is then:

$$\mathcal{P}(C_X = c_x \mid \mathbf{X}_C = \mathbf{x}_C) = \begin{cases} 1 & \text{if } \mathbf{x}_C \text{ sets all } X_i \in \mathbf{X}_C \text{ to the same value;} \\ 0 & \text{otherwise.} \end{cases}$$

**Example 4.** Consider again the simple Bayesian network $A \to B$, and the traditional dataset $\mathcal{D}$, of Example 1. Suppose that we obtain a constrained dataset from $\mathcal{D}$ by asserting the constraint that variable $A$ is equivalent in examples 1 and 2, and variable $A$ is equivalent in examples 3 and 4, i.e. $A_1 \equiv A_2$ and $A_3 \equiv A_4$. That is, we have the set of equivalence constraints $\boldsymbol{\mathcal{C}} = \{\boldsymbol{\mathcal{C}}_A\}$ where $\boldsymbol{\mathcal{C}}_A = \{C^1, C^2\} = \{\{1, 2\}, \{3, 4\}\}$. In the corresponding meta-network, depicted in Figure 2, we introduce additional variables $C^1$ and $C^2$ for each equivalence constraint. Variable $C^1$ has parents $A_1$ and $A_2$, and variable $C^2$ has parents $A_3$ and $A_4$. By conditioning on the instantiation $\boldsymbol{\mathcal{C}} = (C^1 = c_1, C^2 = c_2)$, we enforce the above equivalence constraints in the meta-distribution $\mathcal{P}(\mathbf{X}_{1:4} \mid \boldsymbol{\mathcal{C}}, \theta)$.

Note that a given equivalence constraint $C_X$ is independent of all other equivalence constraints (and further, the network parameters $\theta$), given the variables $\mathbf{X}_C$, which follows from d-separation.

### *4.2. The Constrained Log Likelihood*

As a meta-network induces a log likelihood, a constrained meta-network induces a *constrained log likelihood (CLL)*:

$$CLL(\theta \mid \mathcal{D}, \mathcal{C}) = \log \mathcal{P}(\mathcal{D} \mid \mathcal{C}, \theta). \tag{3}$$

To learn the parameters of a Bayesian network, subject to equivalence constraints, we seek to obtain those estimates $\theta$ maximizing Equation 3. Consider first the following theorem, that decomposes the constrained log likelihood, into two components.

**Theorem 4.1.** *Given a Bayesian network with structure $G$, parameters $\theta$, and a constrained dataset $(\mathcal{D}, \mathcal{C})$, the constrained log likelihood is:*

$$CLL(\theta \mid \mathcal{D}, \mathcal{C}) = LL(\theta \mid \mathcal{D}) + PMI(\mathcal{D}, \mathcal{C} \mid \theta).$$

*where $PMI(\mathcal{D}, \mathcal{C} \mid \theta) = \log \frac{\mathcal{P}(\mathcal{D}, \mathcal{C} \mid \theta)}{\mathcal{P}(\mathcal{D} \mid \theta)\mathcal{P}(\mathcal{C} \mid \theta)}$ is the pointwise mutual information between the dataset $\mathcal{D}$ and the equivalence constraints $\mathcal{C}$.*

*Proof.* Starting from Equation 3, we have:

$$\begin{aligned} CLL(\theta \mid \mathcal{D}, \mathcal{C}) &= \log Pr(\mathcal{D} \mid \mathcal{C}, \theta) \\ &= \log Pr(\mathcal{D} \mid \theta) + \log \frac{\mathcal{P}(\mathcal{D}, \mathcal{C} \mid \theta)}{\mathcal{P}(\mathcal{D} \mid \theta)\mathcal{P}(\mathcal{C} \mid \theta)} \\ &= LL(\theta \mid \mathcal{D}) + PMI(\mathcal{D}, \mathcal{C} \mid \theta) \end{aligned}$$

as desired. □

Maximizing the constrained log likelihood is thus balancing between maximizing the traditional log likelihood and the pointwise mutual information [11] between the data and the constraints (i.e., maximizing the likelihood that the data and constraints appear together, as opposed to appearing independently). Moreover, when there are no equivalence constraints (i.e., $\mathcal{C} = \emptyset$), the constrained log likelihood reduces to the traditional log likelihood $LL(\theta \mid \mathcal{D})$, i.e., the pointwise mutual information $PMI(\mathcal{D}, \mathcal{C} \mid \theta)$ is equal to zero.

### *4.3. Computing The Constrained Log Likelihood*

To evaluate the traditional log likelihood $LL(\theta \mid \mathcal{D}) = \mathcal{P}(\mathcal{D} \mid \theta)$, as in Equations 1 & 2, it suffices to compute the factors $Pr_\theta(\mathbf{d}_i)$ (in the meta-network, the examples $\mathbf{d}_i$ are independent given the parameters $\theta$). Hence, to compute the log likelihood, we only require inference in the base network (and not in the meta-network), which we assume is tractable, using a jointree algorithm (for example). In general, computing the *constrained* log likelihood is intractable, as it does not factorize like the traditional log likelihood. In particular, the terms $\mathcal{P}(\mathcal{D}, \mathcal{C} \mid \theta)$ and $\mathcal{P}(\mathcal{C} \mid \theta)$ do not necessarily

factorize, as the equivalence constraints create dependencies across different examples $\mathbf{d}_i$ in the meta-network.

In the following section, we consider exact and approximate inference in the constrained meta-network. In order to evaluate the constrained log likelihood, and further to optimize it, we will need to compute (or approximate) the relevant quantities, which are in general intractable to compute. Primarily, we will be concerned with approximate inference, although we will consider some special cases where exact inference in the constrained meta-network is feasible (which are applicable to certain tasks in semi-supervised clustering and topic modeling).

## 5. Inference in the Constrained Meta-Network

For inference in the constrained meta-network, several alternatives are available. This choice further impacts the subsequent algorithm that we propose for estimating the parameters of a Bayesian network from a constrained dataset.

As we just discussed, exact inference is in general intractable in the constrained meta-network, so we must appeal to approximate inference algorithms. Popular approaches include stochastic sampling, (loopy) belief propagation and variational inference. Further, all are commonly used in lieu of exact inference, in EM and in other algorithms, for the purposes of parameter learning [12, 13, 14, 15]. Gibbs sampling and importance sampling, however, are known to be inefficient in the presence of deterministic constraints (such as the equivalence constraints used in our constrained meta-network), requiring exponentially many samples, or slow convergence to the stationary distribution [16, 17, 18].[3] Variational approximations and variational EM offer a number of attractive properties, such as lower bounds on the log likelihood [22]. On the other hand, mean-field approximations also suffer from other problems, such as many local optima, and may lead to coarser approximations, compared to (for example) belief propagation [23], although belief propagation does not provide any bounds, and is not guaranteed to converge [24, 25].

For inference in the constrained meta-network, we shall in fact appeal to a class of belief propagation approximations, which are based on the Relax-Compensate-Recover (RCR) framework for approximate inference in probabilistic graphical models; for an overview, see [26]. This choice is particularly suitable for our purposes as RCR is expressly based on relaxing equivalence constraints in a probabilistic graphical model, in order to obtain a tractable approximation (and it is precisely the equivalence constraints that we introduce in a constrained meta-network, that makes inference intractable).

More specifically, RCR is an approach to approximate inference that is based on performing exact inference in a model that has been simplified enough to make infer-

[3] Another approach to inference, that we remark further on, is *blocked* Gibbs sampling [19]. Using single-site updating, Gibbs sampling may get stuck in parts of the search space, when variables are highly correlated. When variables are further subject to equivalence constraints, some parts of the search space may even become unreachable (i.e., the Markov chain is not ergodic); see, e.g., [20, 21]. Although the particular method for approximate inference is not the focus of our later empirical evaluations, we remark that a blocked approach to Gibbs sampling may become viable, if we update the variables of an equivalence constraint as a block.

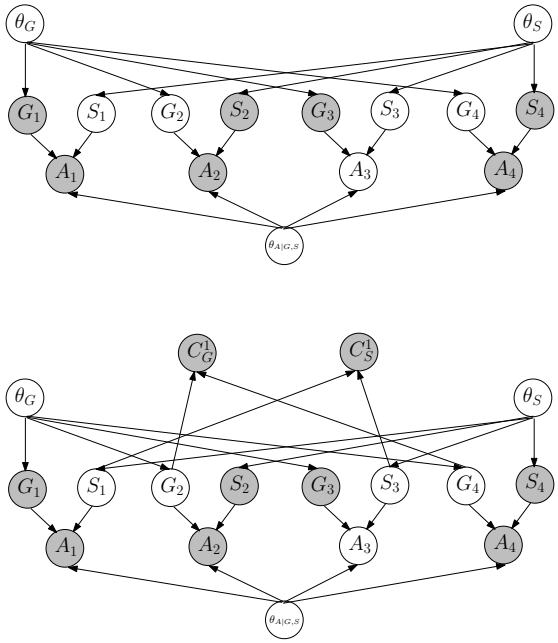


Figure 3: A meta-network for a Bayesian network $G \rightarrow A \leftarrow S$, for a traditional dataset (top) and for a constrained dataset (bottom).

ence tractable. Here, we apply the Relax and Compensation steps of RCR, without Recovery, which yields an approximation that corresponds to an iterative joingraph propagation (IJGP) approximation [27, 28]. In the extreme case, where a fully-disconnected approximation is assumed, RCR corresponds to the influential iterative belief propagation algorithm (and also the Bethe free energy approximation) [29, 30].

For inference in the constrained meta-network, we shall only relax the equivalence constraints $\boldsymbol{C}$, while performing exact inference in each base-network (which, again, we assume is tractable using, for example, a jointree algorithm). This corresponds to an iterative joingraph propagation algorithm, with a corresponding free energy approximation of the likelihood [27, 28]. Later, we shall also consider some interesting cases where this RCR approximation will be exact.

### *5.1. An Approximation: Relax and Compensate*

By asserting the equivalence constraints $\boldsymbol{C}$ in our meta-network, we introduce complexity to its topology, which can make evaluating the constrained likelihood an intractable inference problem. Consider the following example.

**Example 5.** Consider a Bayesian network structure $G \rightarrow A \leftarrow S$, and the following

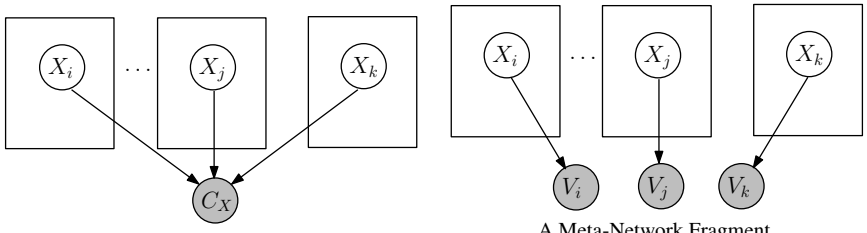


Figure 4: (Left) A meta-network fragment which depicts a variable $X$ that is subject to an equivalence constraint. (Right) The corresponding fragment after relaxing the equivalence constraint $C_X$, and then compensating for the relaxation using soft evidence (using the technique of virtual evidence).

dataset:

| record | $G$ | $A$ | $S$ |
|---|---|---|---|
| 1 | male | true | ? |
| 2 | ? | true | false |
| 3 | female | ? | ? |
| 4 | ? | false | true |

Figure 3 depicts two meta-networks for this example, one without equivalence constraints, and one with equivalence constraints. In the meta-network without equivalence constraints (top), one can verify by inspection that each example $\mathbf{d}_i$ is independent of the other examples (by d-separation), when the values of the parameters $\theta$ are clamped. Hence, to compute the log likelihood, it suffices to compute the probability of each $\mathbf{d}_i$, for a given parameterization $\theta$, independently. However, when we assert equivalence constraints in a meta-network (bottom), this independence (and d-separation) no longer holds. In our example, the base networks of examples 1 and 3 are tied due to the constraint $C_S^1$. Similarly, the base networks of examples 2 and 4 are tied due to the constraint $C_G^1$. In general, as we introduce more constraints, we increase the connectivity among the base networks, which correspondingly makes inference (and evaluating the constrained log likelihood) more challenging. In the extreme case, inference would be exponential in the number of unobserved values in the dataset (which, at worst, would entail summing over all completions of the dataset).

Computing the constrained log likelihood is challenging because of the equivalence constraints $\boldsymbol{C}$ that we assert in our meta-network, which may make the topology of the meta-network too complex for exact inference algorithms. Hence, we shall temporarily *relax* these equivalence constraints, which will make the constrained log likelihood as easy to compute as the traditional log likelihood again. However, just relaxing these constraints may result in a coarse approximation. Hence, we *compensate* for these relaxations, which shall restore a *weaker* notion of equivalence, but in a way that does not increase the complexity of performing inference in the meta-network. The RCR framework specifies a particular way to perform this "compensation" step, which we analyze in more depth here, in the context of the CLL. In particular, the "weaker notion of equivalence" is based on a special case where we assert an equivalence constraint on a set of *independent* variables. We consider this special case, in the following example.

**Example 6.** Let $X_1, \ldots, X_k$ be a set of $k$ variables in a Bayesian network, over the same domain of values. In particular, let $x_{i,s}$ and $x_{i,t}$ denote the $s$-th and the $t$-th state of variable $X_i$. We shall refer to these states by $x_{.,s}$ and $x_{.,t}$, when the index of the variable is not relevant. Suppose now that variables $X_1, \ldots, X_k$ are marginally independent, i.e., $Pr(X_1, \ldots, X_k) = \prod_i Pr(X_i)$. To characterize this distribution, it suffices for us to consider the odds:

$$O(x_{i,s}, x_{i,t}) = \frac{Pr(x_{i,s})}{Pr(x_{i,t})}$$

for each variable $i$, and for each pair of states $s$ and $t$.[4] Suppose that we assert an equivalence constraint $C_X$ over the variables $X_1, \ldots, X_k$. The resulting marginals, and hence the odds, for each variable $X_i$ must also be equivalent. We shall refer to these *consensus* odds by $O(x_{.,s}, x_{.,t} \mid C_X)$. These consensus odds can be characterized by the original odds $O(x_{i,s}, x_{i,t})$, prior to conditioning on the equivalence constraint:[5]

$$O(x_{.,s}, x_{.,t} \mid C_X) = \prod_i O(x_{i,s}, x_{i,t}).$$

In other words, when we assert an equivalence constraint on a set of *independent* variables, the resulting consensus odds is found by simply accumulating the odds of the variables being constrained.

Consider an equivalence constraint $C_X$ over the variables $X_1, \ldots, X_k$; see Figure 4 (left). By relaxing, or deleting, each edge $X_i \rightarrow C_X$ in the meta-network (as in RCR), we ignore the dependencies that exist between different examples $\mathbf{d}_i$, due to the equivalence constraint $C_X$. We can compensate for this relaxation by asserting *soft evidence* on each variable $X_i$, which can be used to restore a weaker notion of equivalence; see Figure 4 (right). In lieu of the equivalence constraint, we use the soft evidence to enforce that the variables $X_1, \ldots, X_k$ have at least equivalent *marginals*. In particular, we will enforce that these marginals correspond to the ones that they would have, as if we asserted an equivalence constraint on independent variables (as in our example above). If the variables $X_1, \ldots, X_k$ are indeed independent in the meta-network, then these marginals would be correct.

In the special case where there is a single equivalence constraint $C_X$ in the meta-network, then the variables $X_1, \ldots, X_k$ would indeed be independent (after relaxing the equivalence constraint). Hence, the compensation would yield exact marginals for variables $X_1, \ldots, X_k$.[6] However, in general, these examples may interact through

[4] For $k$ variables and $u$ states, there are only $k \cdot (u-1)$ independent parameters.

[5] Since instantiations $x_{1,s}, \ldots, x_{k,s}$ and $x_{1,t}, \ldots, x_{k,t}$ satisfy the equivalence constraint $C_X$ (i.e., they are all set to the $s$-th or $t$-th value):

$$O(x_{.,s}, x_{.,t} \mid C_X) = \frac{Pr(x_{1,s}, \ldots, x_{k,s} \mid C_X)}{Pr(x_{1,t}, \ldots, x_{k,t} \mid C_X)} = \frac{Pr(x_{1,s}, \ldots, x_{k,s})}{Pr(x_{1,t}, \ldots, x_{k,t})} = \prod_i \frac{Pr(x_{i,s})}{Pr(x_{i,t})}$$

which is $\prod_i O(x_{i,s}, x_{i,t})$.

[6] In fact, any query $\alpha$ over the variables in example $i$ would be exact [29].

other equivalence constraints on example $i$. Hence, when we compensate for the relaxation of one equivalence constraint, we may violate the "weaker notion of equivalence" that we restored for a previous equivalence constraint (i.e., the equivalent marginals). In general, however, we can iterate over each equivalence constraint that we relax, and compensate for them until convergence. At convergence, each equivalence constraint that we relaxed will indeed satisfy their respective "weaker notion of equivalence." This iterative procedure basically corresponds to the iterative algorithm proposed for RCR, for probabilistic graphical models in general [26].

Consider again Figure 4 (right), where we have asserted soft evidence on each variable $X_i$, where soft evidence, more specifically, is an observation that increases or decreases the belief in an event, but not to the point of certainty [24, 31]. To assert soft evidence on a variable $X_i$, which was subject to a constraint $C_X$ that we relaxed, we need to specify a vector over the values of variables $X$. This vector, which we denote by $\lambda_{C_X}(X_i)$, specifies the strength of our soft evidence. We can implement soft evidence by the method of virtual evidence [24, 31], which introduces a variable $V_i$ as a child of the variable $X_i$, which is clamped to the value $v_i$, and whose CPT is set according to $Pr(v_i \mid x_i) = \lambda_{C_X}(x_i)$. We further note that a given example $i$ may be involved in multiple equivalence constraints. Let $\mathbf{V}_i$ denote the virtual evidence variables introduced to example $i$ by relaxing its equivalence constraints, and let $\mathbf{v}_i$ denote the corresponding instantiation.

We thus want to enforce that each variable $X_i$, that was constrained by an equivalence constraint $C_X$, to have the consensus odds:

$$O_{\theta,\lambda}(x_{i,s}, x_{i,t} \mid \mathbf{d}_i, \mathbf{v}_i) = \frac{Pr_{\theta,\lambda}(x_{i,s} \mid \mathbf{d}_i, \mathbf{v}_i)}{Pr_{\theta,\lambda}(x_{i,t} \mid \mathbf{d}_i, \mathbf{v}_i)} = \prod_{j \in C_X} O_{\theta,\lambda}(x_{j,s}, x_{j,t} \mid \mathbf{d}_j, \mathbf{v}_j \setminus V_j).$$

Here, $Pr_{\theta,\lambda}$ denotes the distribution of the base network parameterized by the CPT parameters $\theta$, but also by the soft evidence parameters $\lambda$ that we introduced after relaxation. Similarly, $O_{\theta,\lambda}$ denotes the corresponding odds.

Consider again Figure 4 (right). The odds $O_{\theta,\lambda}(x_{i,s}, x_{i,t} \mid \mathbf{d}_i, \mathbf{v}_i)$ corresponds to the odds of $X_i$, given example $\mathbf{d}_i$ and all soft observations $\mathbf{v}_i$ introduced for example $i$. In contrast, the odds $O_{\theta,\lambda}(x_{i,s}, x_{i,t} \mid \mathbf{d}_i, \mathbf{v}_i \setminus V_i)$ corresponds to the same odds of $X_i$, except that we retract the soft observations $V_i$ for the constraint $C_X$. Hence, to obtain the desired consensus odds, we need to set the soft evidence vector $\lambda_{C_x}(X_i)$ to obtain the corresponding odds change (i.e., Bayes factor):

$$F(x_{i,s}, x_{i,t}) = \frac{O_{\theta,\lambda}(x_{i,s}, x_{i,t} \mid \mathbf{d}_i, \mathbf{v}_i)}{O_{\theta,\lambda}(x_{i,s}, x_{i,t} \mid \mathbf{d}_i, \mathbf{v}_i \setminus V_i)}$$

(the required strengths of soft evidence satisfy $F(x_{i,s}, x_{i,t}) = \frac{\lambda_{C_X}(x_{i,s})}{\lambda_{C_X}(x_{i,t})}$; see, e.g., [31]). As we described before, updating the soft evidence for one equivalence constraint may disturb the consensus odds that were obtained for a previous equivalence constraint. Hence, one typically performs these updates in an iterative fashion, until all updates converge.

Once we have relaxed all equivalence constraints, and compensated for them, the resulting meta-network induces a distribution that approximates the original one. In

particular, it corresponds to an iterative joingraph propagation (IJGP) approximation [29, 27, 28]. The resulting meta-distribution now factorizes according to its examples, and thus inference in the meta-network requires only inference in the respective base networks, which we assume is tractable using, e.g., a jointree algorithm. In some (restricted) cases, however, the resulting computations may still be exact, allowing us to evaluate the CLL exactly (as well as certain marginals).

### *5.2. On Computing the CLL Exactly*

In some relevant cases, the RCR approximation of the meta-network that we just described, can still be used to compute the CLL exactly. Consider again the CLL:

$$CLL(\theta \mid \mathcal{D}, \mathcal{C}) = \log \mathcal{P}(\mathcal{D}, \mathcal{C} \mid \theta) - \log \mathcal{P}(\mathcal{C} \mid \theta).$$

We can approximate the terms $\mathcal{P}(\mathcal{D}, \mathcal{C} \mid \theta)$ and $\mathcal{P}(\mathcal{C} \mid \theta)$ in the RCR approximation of the meta-network, which corresponds to (corrected) partition functions in the RCR framework [30]. However, these approximations are known to be exact in some known cases [30]. The following proposition characterizes a class of constrained datasets where the above approximation of the CLL will also be exact.

**Proposition 5.1.** *Say we are given a Bayesian network with structure $G$, parameters $\theta$, and a constrained dataset $(\mathcal{D}, \mathcal{C})$, where each example $\mathbf{d}_i$ of the dataset $\mathcal{D}$ is constrained by at most one equivalence constraint in $\mathcal{C}$. For such a constrained dataset, the RCR approximation of the constrained log likelihood is exact. Moreover, the RCR approximations for marginals over families $X\mathbf{U}$ are also exact.*

This special case is interesting because it captures a variety of learning tasks in different domains, which can be reduced to the problem of learning the parameters of a Bayesian network from a constrained dataset.

**Example 7.** In semi-supervised clustering, we can learn naive Bayes models and Gaussian mixture models with "must-link" constraints, where it suffices to assert at most one constraint on each example. In particular, for each maximal set of examples that are known to belong to the same cluster, we assert a single equivalence constraint. In this case, we can seek to optimize the constrained log likelihood, which we can now evaluate exactly.

We remark that Proposition 5.1 follows from the RCR framework. In particular, if we relax (delete) an edge $Y \rightarrow X$ that splits a network into two independent sub-networks, then we can compensate for the relaxation exactly, and recover the true probability of evidence [30]. This is a generalization of the correctness of the Bethe free energy approximation in polytree Bayesian networks [32], since deleting any edge $Y \rightarrow X$ in a polytree splits the network into two.

## 6. Optimizing the CLL

**Algorithm 1** Optimize $CLL'$

**input:**

$G$: A Bayesian network structure
$\mathcal{D}$: An incomplete dataset $\mathbf{d}_1, \ldots, \mathbf{d}_N$
$\boldsymbol{\mathcal{C}}$: A set of equivalence constraints
$\theta$: An initial parameterization of structure $G$

1: **while** not converged **do**

2: **Update** soft evidence parameters $\lambda_1, \lambda_2$ and compute the marginals for each instantiation $x\mathbf{u}$ of each family $X\mathbf{U}$:

$$\begin{array}{ll} Pr_{\theta,\lambda_1}(x\mathbf{u} \mid \mathbf{d}_i, \mathbf{v}_i) & Pr_{\theta,\lambda_1}(\mathbf{u} \mid \mathbf{d}_i, \mathbf{v}_i) \\ Pr_{\theta,\lambda_2}(x\mathbf{u} \mid \mathbf{v}_i) & Pr_{\theta,\lambda_2}(\mathbf{u} \mid \mathbf{v}_i) \end{array}$$

3: **Update** network parameters $\theta$, for each instantiation $x\mathbf{u}$ of each family $X\mathbf{U}$:

$$\theta_{x|\mathbf{u}} = \frac{\sum_i Pr_{\theta,\lambda_1}(x\mathbf{u} \mid \mathbf{d}_i, \mathbf{v}_i) + \theta_{x|\mathbf{u}} \sum_i Pr_{\theta,\lambda_2}(\mathbf{u} \mid \mathbf{v}_i)}{\sum_i Pr_{\theta,\lambda_2}(x\mathbf{u} \mid \mathbf{v}_i) + \theta_{x|\mathbf{u}} \sum_i Pr_{\theta,\lambda_1}(\mathbf{u} \mid \mathbf{d}_i, \mathbf{v}_i)} \theta_{x|\mathbf{u}}$$

4: **return** parameterization $\theta$

Our task is now to learn the parameters $\theta$ of a Bayesian network, from a dataset $\mathcal{D}$ that is subject to the constraints $\boldsymbol{\mathcal{C}}$. We propose to seek those parameter estimates $\theta$ that *maximize* the constrained log likelihood, as given in Equation 3

$$CLL(\theta \mid \mathcal{D}, \boldsymbol{\mathcal{C}}) = \log \mathcal{P}(\mathcal{D} \mid \boldsymbol{\mathcal{C}}, \theta) = \log \mathcal{P}(\mathcal{D}, \boldsymbol{\mathcal{C}} \mid \theta) - \log \mathcal{P}(\boldsymbol{\mathcal{C}} \mid \theta).$$

However, due to the constraints $\boldsymbol{\mathcal{C}}$, it may be intractable to even evaluate the CLL, for a given candidate set of estimates $\theta$. Hence, we will propose to optimize instead an approximation of the CLL found by relaxing the equivalence constraints $\boldsymbol{\mathcal{C}}$, and then compensating for the relaxations, as dictated by the RCR framework (which we further described in the previous section). This approach to parameter estimation is akin to approaches that use a tractable approximation of the log likelihood, such as one obtained by loopy belief propagation; see, e.g., [14, 15].

First, when we relax all equivalence constraints $\boldsymbol{\mathcal{C}}$, we compensate by asserting soft evidence $\mathbf{v}_i$ on each example $i$. Moreover, the meta-distribution factorizes according to the examples $i$. This leads to the following approximation of the constrained log likelihood, that also factorizes according to the examples $i$:

$$CLL'(\theta, \lambda_1, \lambda_2 \mid \mathcal{D}, \boldsymbol{\mathcal{C}}) = \underbrace{\sum_i \log Pr_{\theta,\lambda_1}(\mathbf{d}_i, \mathbf{v}_i)}_{\approx \log \mathcal{P}(\mathcal{D}, \boldsymbol{\mathcal{C}} \mid \theta)} - \underbrace{\sum_i \log Pr_{\theta,\lambda_2}(\mathbf{v}_i)}_{\approx \log \mathcal{P}(\boldsymbol{\mathcal{C}} \mid \theta)} \quad (4)$$

Here, $Pr_{\theta,\lambda}$ denotes the distribution of the base network, which is now determined by two sets of parameters: (1) the parameter estimates $\theta$ of the Bayesian network that we seek to learn, and (2) the parameters $\lambda$ of the soft observations $\mathbf{v}_i$ that are used to compensate for the relaxed equivalence constraints $\boldsymbol{\mathcal{C}}$. Note that we use two sets of

compensations $\lambda_1$ and $\lambda_2$, to approximate $\mathcal{P}(\mathcal{D}, \mathcal{C} \mid \theta)$ (with the dataset observed in the meta-network) and $\mathcal{P}(\mathcal{C} \mid \theta)$ (with the dataset unobserved in the meta-network).

We are now interested in optimizing our approximation $CLL'$ of the constrained log likelihood, which is done with respect to the network parameters $\theta$, but also with respect to the soft evidence parameters $\lambda_1, \lambda_2$. We propose a simple fixed-point iterative algorithm, for doing so, which is summarized in Algorithm 1.

Our fixed-point algorithm alternates between updating the soft evidence parameters $\lambda_1, \lambda_2$, and updating the parameter estimates $\theta$. We first fix the parameter estimates $\theta$ and then update the soft evidence parameters $\lambda_1, \lambda_2$. This corresponds to performing relax-and-compensate in the meta-network, as described in the previous section. We next fix the soft evidence parameters $\lambda_1, \lambda_2$, and update the parameter estimates $\theta$, which we shall discuss further next. These two steps are repeated, until all parameters converge to a fixed-point.

**Proposition 6.1.** *Fixed-points of Algorithm 1 are stationary points of the approximation to the constrained log likelihood of Equation 4.*

Note that optimal parameter estimates, with respect to the (approximate) CLL, must be stationary points of the (approximate) CLL, but not necessarily vice versa.

Consider the first partial derivative of the CLL, with respect to a parameter $\theta_{x|\mathbf{u}}$, for the instantiation $x\mathbf{u}$ of family $X\mathbf{U}$ (keeping the soft evidence parameters $\lambda_1, \lambda_2$ fixed):

$$\frac{\partial CLL'}{\partial \theta_{x|\mathbf{u}}} = \sum_i \frac{Pr_{\theta,\lambda_1}(x\mathbf{u} \mid \mathbf{d}_i, \mathbf{v}_i) - Pr_{\theta,\lambda_2}(x\mathbf{u} \mid \mathbf{v}_i)}{\theta_{x|\mathbf{u}}} \tag{5}$$

We are interested in the stationary points of the CLL, but subject to sum-to-one constraints on the parameters $\theta_{x|\mathbf{u}}$. Hence, we construct the corresponding Lagrangian, and set the gradient to zero. We can then obtain the following stationary conditions which we can use in an iterative fixed-point algorithm:

$$\theta_{x|\mathbf{u}} = \frac{\sum_i Pr_{\theta,\lambda_1}(x\mathbf{u} \mid \mathbf{d}_i, \mathbf{v}_i) + \theta_{x|\mathbf{u}} \sum_i Pr_{\theta,\lambda_2}(\mathbf{u} \mid \mathbf{v}_i)}{\sum_i Pr_{\theta,\lambda_2}(x\mathbf{u} \mid \mathbf{v}_i) + \theta_{x|\mathbf{u}} \sum_i Pr_{\theta,\lambda_1}(\mathbf{u} \mid \mathbf{d}_i, \mathbf{v}_i)} \theta_{x|\mathbf{u}}. \tag{6}$$

We remark further that the above stationary condition for the constrained log likelihood reduces to a stationary condition for the traditional log likelihood, when no equivalence constraints are used:

$$\theta_{x|\mathbf{u}} = \frac{\sum_i Pr_\theta(x\mathbf{u} \mid \mathbf{d}_i)}{\sum_i Pr_\theta(\mathbf{u} \mid \mathbf{d}_i)}. \tag{7}$$

The above stationary conditions further correspond to the EM algorithm for traditional datasets, when used as an update equation in an iterative fixed-point algorithm.

In general, however, Algorithm 1 seeks the stationary points of an approximation of the CLL (Equation 4), which is based on an RCR approximation. RCR is in turn a generalization of loopy belief propagation [26], and hence inherits some of its drawbacks (i.e., it is not guarantee to converge in general, and does not provide bounds). Our learning algorithm, is thus more akin to approaches to parameter estimation that use loopy belief propagation in lieu of exact inference, for example, to approximate the expectations in EM, when such computations are otherwise intractable [14, 15].

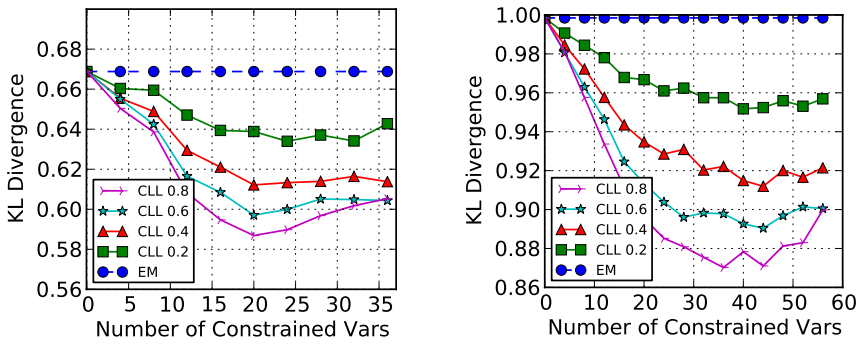


Figure 5: From left-to-right, we generally see improved parameter estimates ($y$-axis), as we increase the number of constrained variables ($x$-axis). We also vary the proportion of the missing values that are subject to constraints (0.2,0.4,0.6,0.8), for each variable.

Other alternatives, in terms of inference and learning paradigms, could similarly be employed to optimize the CLL. However, as we shall see in the next section, our proposed algorithm performed well, for the purposes of empirically evaluating the CLL, as an approach to learning from constrained datasets.

## 7. Experiments

In our first set of experiments, we study the CLL as an objective function for learning Bayesian networks, showing how it can learn more accurate models as more side information, in the form of equivalence constraints, is provided. Subsequently, we consider two different learning tasks in two different domains, that we can reduce to the problem of learning a Bayesian network from a constrained dataset: (1) semi-supervised clustering with naive Bayes models, and (2) topic modeling with domain-specific knowledge.

### *7.1. Synthetic Data*

We consider first the constrained log likelihood, as an objective function for learning Bayesian networks. In particular, we evaluate our ability to learn more accurate models, as more side information is given. We consider two classical networks, `alarm` and `win95pts`, with 37 variables and 69 variables, respectively. We simulated complete datasets from each network, then obtained an incomplete dataset for each by hiding values at random, with some probability. We further simulated equivalence constraints, for each incomplete dataset, by randomly constraining pairs of missing values (which were known to have the same value in the original complete dataset). Our baseline, in this set of experiments, is the standard EM algorithm for traditional (unconstrained) datasets. EM does not incorporate side information in the form of equivalence constraints, and our main goal first is to illustrate the benefits of doing so.

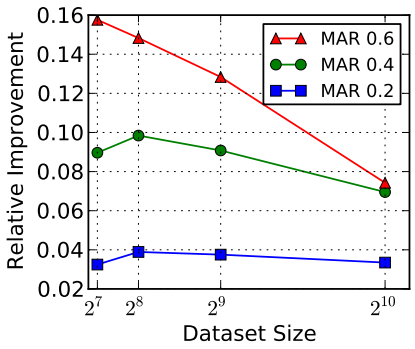


Figure 6: From right-to-left, we generally see equivalence constraints are more effective ($y$-axis), as we *decrease* the size of the dataset ($x$-axis). We also vary the proportion of the missing values (0.2,0.4,0.6).

Since we are simulating datasets, we measure the quality of an algorithm's estimates using the KL-divergence between (1) the distribution $Pr_{\theta^\star}$ induced by the parameters $\theta^\star$ that generated the dataset originally, and (2) the distribution $Pr_\theta$ induced by the parameters $\theta$ estimated by the corresponding learning algorithm:[7]

$$\begin{aligned} KL(Pr_{\theta^\star}, Pr_\theta) &= \sum_w Pr_{\theta^\star}(w) \log \frac{Pr_{\theta^\star}(w)}{Pr_\theta(w)} \\ &= \sum_{X\mathbf{U}} \sum_{\mathbf{u}} Pr_{\theta^\star}(\mathbf{u}) KL(\theta^\star_{X|\mathbf{u}}, \theta_{X|\mathbf{u}}) \end{aligned}$$

where we iterate over all families $X\mathbf{U}$ of our Bayesian network, and all parent instantiations $\mathbf{u}$, i.e., we iterate over all CPT columns of the network. Note that the KL-divergence is non-negative, and equal to zero iff the two distributions are equivalent. We also assume pseudo-counts of one, i.e., Laplace smoothing, for both the EM and CLL optimization algorithms. Both algorithms were also initialized with the same random seeds. Further, each algorithm was given 5 initial seeds, and the one that obtained their best respective likelihood was chosen for evaluation.

Figure 5 highlights our first set of experiments where we observe the increase in quality of our parameter estimates (on the $y$-axis), as we assert equivalence constraints on more variables (on the $x$-axis). Here, each plot point represents an average of 30 simulated datasets of size $2^9$, where values were hidden at random with probability $0.4$. On the $x$-axis, from left-to-right, we progressively selected a larger number of variables to subject to constraints. For each constrained variable, we randomly sampled a proportion of the hidden values under constraints; the proportions that we constrained were also varied, using separate curves.

[7] Since our training data is simulated from Bayesian networks that we have access to, our evaluation is based on the KL-divergence, rather than the log likelihood of simulated testing data, which would be equivalent in expectation; see, e.g., [10, Section 16.2.1].

At one extreme, when no variables are constrained, our CLL optimization algorithm performs the same as EM, as expected. As we move right on the plot, where we select more variables to constrain, we see that our CLL optimization algorithm does, generally speaking, produce more accurate parameter estimates, as compared to EM (which does not exploit equivalence constraints). Based on the individual curves, from top-to-bottom, where we increase the proportion of missing values that are subject to constraints, we see the quality of the estimates generally improve further. We also observe that, as we approach an overly large number of constraints, the quality of our estimates appear to degrade. This is also expected, as the optimization algorithm that we proposed assumes an approximation of the CLL, based on relaxing (and then compensating for) all equivalence constraints, which we expect to be coarse when many equivalence constraints must be relaxed.

Figure 6 highlights our second set of experiments, where we measure now the relative improvement of our CLL optimization algorithm compared to EM, on the $y$-axis. Here, each point is an average over $120$ simulated datasets of a fixed dataset size where values were hidden with probabilities 0.2, 0.4, and 0.6. Further, $20$ variables were constrained and half the missing values were covered by equivalence constraints. As we move *right-to-left* on the $x$ axis, where we *decrease* the size of the dataset, we find that our CLL optimization algorithm is increasingly effective compared to EM, as less data is available, at least up to a point (at dataset size $N = 128$, which is a relatively small amount of data, the effectiveness starts to diminish again). This highlights how important additional side-information becomes as less data is available. We also observe the increased effectiveness as more values are missing in the data (going from the bottom curve to the top one).

### *7.2. Application: Semi-Supervised Clustering*

In our first illustrative application, we are interested in clustering tasks where we have *must-link constraints* that are used to constrain examples that are known to belong in the same cluster; see, e.g., [4]. We consider semi-supervised clustering here with naive Bayes models in particular, where the constrained variable is the class variable. Unlike our previous set of experiments, we are able in this case to evaluate the constrained log likelihood exactly. We compare our method with the EM algorithm proposed by [33] (originally proposed for Gaussian mixture models, but adapted here for naive Bayes models), which we refer to by em-sbhw. Note that em-sbhw was specifically proposed for the task of semi-supervised clustering, under the presence of must-link constraints. We also compare with the traditional EM algorithm, as a baseline, which again does not take advantage of any equivalence constraints.

We use datasets from the UCI Machine Learning repository, and in some cases, used Weka to discretize data with continuous values, and to fill-in missing values by mean/mode, as is commonly done in such evaluations. We start with complete datasets where the true clusterings are known, and then generate equivalence constraints based on these known labels. We then hide these labels to yield an incomplete dataset. We follow the approach of [33], where we first partition the dataset into $K$ partitions ($K = 5$) and then randomly select a fixed number $m$ of examples from each. In each of these $m$ selected examples, we asserted an equivalence constraint across those examples that had the same labels. Note that this yields equivalence constraints of varying sizes.

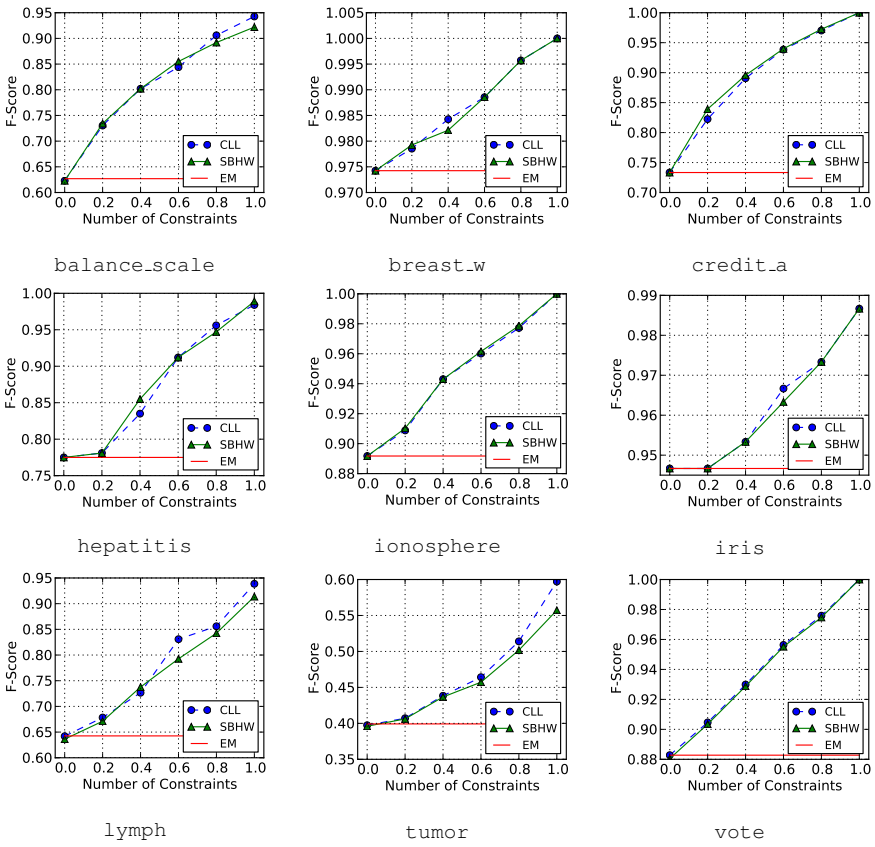


Figure 7: In GMMs, we observe the increase in clustering performance ($y$-axis), via $F$-measure, as the amount of side-information is increased ($x$-axis).

Effectively, $K \cdot m$ controls the amount of side-information available. When $K \cdot m$ equals the number of examples $N$, then every example in the dataset is subject to an equivalence constraint. Given a constrained dataset, we evaluate the results of each algorithm based on their performance in clustering, using the $F$-measure, which is computed using $f = \frac{2PR}{P+R}$, where $P$ and $R$ are the precision and recall, respectively.

Here, we steadily increase the amount of side-information available, from no side-information to perfect side-information (in the latter case, all examples that were assigned to the same cluster in the original complete dataset, were constrained to be in the same cluster in the incomplete dataset that we evaluated). In Figure 7, we observe the increase in $F$-measure as we increase the number of equivalence constraints made available. With no equivalence constraints given, all algorithms evaluated were equivalent to vanilla EM. As we increase the number of constraints, we see that both cem and em-sbhw exhibit smoothly increasing performance, and in some cases obtaining perfect clusterings when all known equivalences were provided. In datasets `lymph` and `tumor`, we see that our CLL optimization algorithm is superior. In dataset

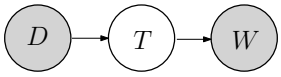


Figure 8: A Bayesian network representation of the LDA model, as employed in [34]; for simplicity, we have omitted the nodes representing LDA's parameters (which are now CPTs of this Bayesian network), which are typically explicated in plate representations. Here, variable $T$ represents the topic assignment, and variable $W$ represents the word observed. Further, the document index has been explicated using the variable $D$.

`credit`, we see that em-sbhw is mildly favorable. In most of the datasets, however, we see that both algorithms perform similarly. Hence, this suggests that our general approach, based on optimizing the CLL, is comparable (and sometimes outperforming) a domain-specific algorithm developed for the relatively specialized task of semi-supervised clustering.

### *7.3. Application: Topic Modeling*

We consider another application in topic modeling [35], where we want to assert some domain knowledge, in the form of equivalence constraints, in order to learn improved topics. In particular, one could use equivalence constraints to interactively refine the topics learned by a topic model. For example, a practitioner may inspect the topics assigned to the individual words of a document, and may observe that some words are associated with topics that are not reasonable, based on their background knowledge. In particular, they may find words that are associated with different topics, that they judge should be in the same topic. In this case, the practitioner can assert an equivalence constraint between the topics of these words, to encourage the topic model to associate the same topic with them. In this way, a practitioner can "debug", or otherwise have some refined mechanism to control, the topics that are learned by a topic model.

We next present some experiments, illustrating another example of how equivalence constraints can be used to refine the topics of a topic model. In particular, we propose to introduce equivalence constraints, in order to encourage the formation of new topics, whose words are otherwise scattered across disparate topics. The dataset that we consider here consists of 539 abstracts from the Journal of Artificial Intelligence Research (JAIR). The corpus covers a broad range of different topics related to AI, for example, *agent-based systems*, *heuristic search* and *logical reasoning* (as analyzed by the annual reports, produced by the editors of JAIR).

We first learn from this corpus a standard LDA model, over 10 topics (where the number of topics here is based roughly on the number of topics manually identified in the annual reports of JAIR).[8] A Bayesian network representation of the LDA model is provided in Figure 8. The top words of each topic are illustrated in Table 2. For example, topic 7 can clearly be interpreted as a *planning* topic.

Suppose now that a practitioner decides that they want other topics, based on background knowledge, to emerge from the model. Take for example, a possible "complex-

[8] We used the `lda-c` package, which is based on variational EM. The `lda-c` package is available at `http://www.cs.columbia.edu/~blei/lda-c/index.html`.

Table 1: Topics learned from JAIR abstracts. Each row is a constrained keyword, with a • indicates that a word is among the most 50 probable words within a topic.

| | LDA | | | | | | | | | | LDA-DF | | | | | | | | | | LDA-CLL | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| topic | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| bayesian | | | | | | • | | | • | | • | | | | | | | | | | | | | | | | • | | | |
| network | | • | | | | | | | • | | • | | | | | | | | | | | | | | | | • | | | |
| inference | | | | | | | | | • | | • | | | | | | | | | | | | | | | | • | | | |
| complexity | | | | | | • | • | | | | • | | | | | | | | | | • | | | | | | | | | |
| polynomial | | | | | | • | | | | | • | | | | | | | | | | • | | | | | | | | | |
| np | | | | | | • | | | | | • | | | | | | | | | | • | | | | | | | | | |

ity" topic (which is not purely an AI topic, per se, but a topic frequently discussed in AI papers). Words, such as "complexity", or "NP", or "polynomial", when appearing together in the same abstract, may plausibly come from the same topic, about "complexity." Hence, we propose to impose an equivalence constraint among the topics of these words, if they appear in the same abstract. Such an equivalence constraint would encourage the topic model to include these set of words in the same topic, and perhaps encourage the formation of a new topic, around these words.

Consider Table 3, where we have learned an LDA model while optimizing the CLL, based on the above constraints. We see that in topic 0, a new topic has formed, which contain the above words "complexity", "NP", and "polynomial" (in boxes). Moreover, we see that the other words in the topic are also strongly reminiscent of a "complexity" topic. In addition to the above constraints, we also constrained the topics of the words "Bayesian" and "inference" and "network", when they appeared together in the same abstract. We see also in topic 6, another topic has appeared around these words (in boxes), reminiscent of a "Bayesian network modeling and inference" topic.

We also compared with a more specialized method, which incorporates analogous (must-link) constraints via a Dirichlet forest prior (LDA-DF) [6], learned using the same constraints described above. First, we consider another visualization of the topics, as in [6], in Table 1. Here, we want to visualize whether our constrained words appear in the same topic, as desired, or whether they are dispersed across different topics. More specifically, we sorted the words of each learned topic by probability, and then selected the 50 most probable words for each topic. We then observed whether or not our selected keywords appeared as one of the most probable words, for each topic (if so, we denote it by a dot in Table 1).

In vanilla LDA, we found that our selected keywords were indeed dispersed across multiple topics. For example, the word "complexity" appears strongly in multiple topics (topics 5 and 6). By asserting the background knowledge in the form of constraints (CLL), we see that we are indeed able to learn a more specialized "complexity" topic, whose words are strongly associated to it (and none other). We observe similarly for LDA-DF. In Table 4, we visualize the topics learned by LDA-DF, more directly. We can see, in topic 0, a topic where all of our "complexity" keywords appear strongly. However, in contrast to topic 0 learned in Table 3 for LDA-CLL, we see that this topic

is much less reminiscent of "complexity" (it appears to be a topic much broader than "complexity"), and did not encourage as focused a topic, as suggested by the constrained keywords, to the extent that LDA-CLL did.

## 8. Related Work

As we discussed in our experiments, there are learning tasks in a variety of domains, where specialized methods were developed, that can be formulated as learning a Bayesian network under a constrained dataset. For example, the work of [33] proposed to learn Gaussian mixture models (GMMs) under equivalence constraints to improve clustering. Moreover, [6] considered the use of equivalence constraints in topic models, albeit less directly, to constrain words that should have similar importance in different topics. Here, a Dirichlet forest prior was used to accommodate such constraints, with a corresponding Gibbs sampling method for learning the parameters of the topic model.

The works of [33, 36] are more closely related to our proposal. Rather than connect two meta-network variables $X_i$ and $X_j$ to a common child, which constrained them to take the same value, these proposals effectively assumed a direct edge $X_i \rightarrow X_j$, where the CPT of $X_j$ was set so that variable $X_j$ assumes the value of variable $X_i$, i.e., $Pr(x_j \mid z_i) = 1$ iff $x_i = x_j$. This induces a meta-network with a different type of structure and likelihood, where equivalent variables were merged into a single node. However, the type of equivalence constraint implied by "merging" equivalent variables $X_i \rightarrow X_j$, does not generalize obviously when variable $X$ is not a root node.

## 9. Conclusion

In this paper, we proposed a general framework for learning Bayesian network parameters under equivalence constraints. These constraints assert that the values of unobserved variables in certain examples of a dataset must be the same, even if that particular value is not known. We proposed a notion of a constrained dataset, and a corresponding constrained log likelihood. We proposed a fixed-point iterative algorithm for optimizing the constrained log likelihood, and showed empirically that it can be effective at learning more accurate models given more background knowledge in the form of equivalence constraints. We further highlighted how our framework naturally models tasks in a variety of domains, where often domain-specific and (sometimes) less principled approaches have been previously proposed.

## Acknowledgments

We thank the anonymous reviewers for valuable comments that improved the paper. This work has been partially supported by ONR grant #N00014-12-1-0423 and NSF grant #IIS-1118122.

Table 2: Topics Learned by LDA

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| agents | algorithm | decision | search | data | learning | reasoning | planning | instances | action |
| agent | system | theory | algorithm | information | model | logic | plan | problem | agent |
| problems | constraint | agents | algorithms | system | based | knowledge | domains | inference | model |
| problem | based | domain | problem | approach | models | set | state | sat | control |
| optimal | information | game | problems | model | approach | complexity | domain | performance | agents |
| monitoring | consistency | paper | space | paper | paper | properties | plans | clauses | show |
| efficient | semantic | models | optimal | query | framework | class | planners | random | systems |
| information | paper | results | function | based | selection | show | problems | show | programs |
| strategy | level | algorithms | heuristic | resolution | data | description | planner | paper | learning |
| show | temporal | graphical | time | examples | representation | paper | actions | problems | logic |
| multi | show | set | solution | set | machine | propositional | competition | structure | rules |
| cost | problem | learning | solutions | task | methods | complete | techniques | results | preferences |
| present | present | making | state | time | application | belief | results | algorithm | actions |
| distributed | techniques | agent | local | sources | algorithms | representation | based | network | behavior |
| algorithm | approach | framework | show | results | technique | logics | performance | evidence | present |
| time | method | previous | policy | show | tasks | based | graph | class | paper |
| resource | performance | functions | methods | methods | reinforcement | constraints | probabilistic | networks | order |
| mechanisms | knowledge | games | pomdps | language | role | general | ipc | hard | semantics |
| environment | dialogue | networks | markov | problem | general | computational | heuristic | satisfiability | models |
| approach | user | equilibrium | large | real | work | programs | control | provide | results |
| social | binary | approach | paper | web | results | notion | present | solvers | work |
| case | satisfaction | classification | solving | method | set | languages | deterministic | algorithms | preference |
| auctions | measure | multi | number | approaches | empirical | semantics | describe | cost | execution |
| communication | constraints | work | size | evaluation | proposed | revision | order | sampling | limitations |
| systems | systems | attributes | processes | experiments | problem | temporal | time | max | programming |
| solutions | algorithms | utility | states | cases | task | operator | international | formulas | natural |
| mechanism | results | concept | decision | applications | current | change | language | phase | method |
| task | text | knowledge | find | perform | error | update | variables | complexity | number |
| techniques | tree | accuracy | policies | algorithm | analysis | problem | model | number | language |
| solving | sentence | based | method | natural | sets | study | causal | probability | single |

Table 3: Topics Learned by LDA-CLL

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| complexity | agents | planning | sat | information | domain | model | programs | learning | system |
| problems | algorithm | search | theory | data | knowledge | inference | logic | decision | problem |
| np | problems | domains | random | system | planning | network | search | policy | paper |
| algorithms | search | heuristic | paper | paper | search | models | algorithms | algorithm | instances |
| problem | agent | based | monitoring | reasoning | show | based | constraints | paper | based |
| show | problem | algorithm | show | set | problem | networks | problem | markov | learning |
| polynomial | game | problems | clauses | based | agent | theory | algorithm | algorithms | control |
| optimal | algorithms | problem | problem | rules | plan | bayesian | set | based | show |
| decision | paper | plan | algorithm | semantic | agents | algorithm | constraint | problem | problems |
| class | solutions | state | results | approach | control | method | semantics | data | results |
| complete | show | techniques | description | preferences | plans | paper | bounds | model | systems |
| time | time | actions | formula | attributes | approach | show | show | reinforcement | model |
| provide | performance | planner | max | show | based | data | consistency | state | search |
| models | solution | plans | auctions | sources | order | results | programming | problems | dialogue |
| hard | results | planners | based | order | learning | learning | based | space | cost |
| general | number | competition | instances | web | paper | semantic | results | function | data |
| paper | systems | present | set | resolution | team | structure | paper | framework | recursive |
| algorithm | behavior | domain | formulas | time | system | representation | agent | results | number |
| reasoning | optimal | heuristics | previous | text | planners | terms | problems | pomdps | backbone |
| propositional | structure | graph | approach | knowledge | performance | proposed | models | policies | update |
| computational | games | action | model | queries | algorithm | set | properties | optimal | representation |
| classes | real | paper | data | preference | features | probabilistic | space | training | class |
| agents | method | space | state | logics | execution | knowledge | present | performance | algorithm |
| tractable | approach | results | logics | query | approaches | information | languages | robot | set |
| logic | general | performance | knowledge | model | domains | problem | binary | agent | methods |
| case | limitations | time | properties | word | reasoning | domain | answer | processes | function |
| finding | social | ipc | horn | qualitative | results | algorithms | knowledge | approach | knowledge |
| graphical | revision | system | number | temporal | present | belief | number | pomdp | clause |
| model | methods | show | context | peer | local | approach | called | examples | strategy |
| temporal | belief | representation | action | number | language | conditional | atoms | present | copeland |

Table 4: Topics Learned by LDA-DF

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| algorithms | algorithm | planning | search | problems | approach | agents | model | system | learning |
| show | paper | time | problem | paper | decision | domain | agent | agent | data |
| number | inference | control | space | representation | problem | theory | analysis | based | knowledge |
| results | method | actions | information | reasoning | systems | temporal | show | sat | logic |
| present | structure | optimal | cost | based | efficient | strategy | information | method | based |
| class | solutions | domains | solution | plan | action | framework | techniques | resolution | models |
| complexity | local | case | models | properties | provide | performance | features | cases | language |
| constraints | network | state | state | complete | terms | domains | sets | human | programs |
| set | describe | size | belief | performance | based | consistency | world | result | semantic |
| general | networks | policy | solving | results | programming | approaches | context | structure | present |
| variables | system | heuristic | probabilistic | semantics | problems | show | complex | optimal | set |
| hard | rules | plans | results | planners | random | methods | set | real | called |
| np | bayesian | technique | experimental | computational | work | specific | selection | systems | tree |
| constraint | standard | prove | presents | description | markov | classification | task | formula | languages |
| experiments | linear | process | perform | classes | conditional | approach | graph | implemented | machine |
| bounds | execution | planner | states | preferences | functions | efficiency | implementation | goal | natural |
| polynomial | optimization | function | order | finding | processes | order | operator | update | empirical |
| obtained | partial | values | art | techniques | making | operators | propose | processing | form |
| clauses | distribution | competition | reinforcement | propositional | presented | individual | respect | query | framework |
| shown | queries | independent | current | behavior | deterministic | common | types | demonstrate | trees |
| simple | order | monitoring | pomdps | multiple | tasks | design | rule | part | training |
| instances | version | dynamic | article | game | significantly | investigate | work | including | learn |
| study | exact | discuss | task | concept | general | strategies | user | communication | examples |
| provide | improve | graphical | environment | demonstrate | point | compare | sentence | previous | related |
| heuristic | resource | heuristics | required | theoretical | classical | existing | global | definition | concepts |
| real | compared | policies | instances | logics | structures | variety | relative | extensions | text |
| large | improved | key | shows | developed | probability | ipc | relations | methods | base |
| satisfiability | methods | assumptions | robot | make | computing | level | ordering | word | formalisms |
| applications | quality | bound | sources | important | procedure | social | change | potential | accurate |
| empirical | initial | phase | efficiently | large | previous | preference | studies | max | accuracy |

## References


[1] C. A. Sutton, A. McCallum, An introduction to conditional random fields, Foundations and Trends in Machine Learning 4 (2012) 267–373.

[2] R. Szeliski, Computer Vision: Algorithms and Applications, 1st ed., Springer-Verlag New York, Inc., New York, NY, USA, 2010.

[3] I. Davidson, Clustering with constraints, in: L. Liu, M. T. Özsu (Eds.), Encyclopedia of Database Systems, Springer US, 2009, pp. 393–396.

[4] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl, Constrained $k$-means clustering with background knowledge, in: Proceedings of the 18th Annual International Conference on Machine Learning (ICML), 2001, pp. 577–584.

[5] Z. Lu, T. K. Leen, Semi-supervised learning with penalized probabilistic clustering, in: Proceedings of the 17th Annual Conference on Neural Information Processing Systems (NIPS), 2004.

[6] D. Andrzejewski, X. Zhu, M. Craven, Incorporating domain knowledge into topic modeling via dirichlet forest priors, in: Proceedings of the 26th Annual International Conference on Machine Learning (ICML), 2009, p. 4.

[7] A. Dempster, N. Laird, D. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society B 39 (1977) 1–38.

[8] S. Lauritzen, The EM algorithm for graphical association models with missing data, Computational Statistics and Data Analysis 19 (1995) 191–201.

[9] A. Darwiche, Modeling and Reasoning with Bayesian Networks, Cambridge University Press, 2009.

[10] D. Koller, N. Friedman, Probabilistic Graphical Models: Principles and Techniques, MIT Press, 2009.

[11] K. W. Church, P. Hanks, Word association norms, mutual information, and lexicography, Computational Linguistics 16 (1990) 22–29.

[12] Z. Ghahramani, M. I. Jordan, Factorial hidden markov models, Machine Learning 29 (1997) 245–273.

[13] B. S. Caffo, W. Jank, G. L. Jones, Ascent-based monte carlo expectation-maximization, Journal of the Royal Statistical Society. Series B (Statistical Methodology) 67 (2005) pp. 235–251.

[14] M. J. Wainwright, Estimating the "wrong" graphical model: Benefits in the computation-limited setting, Journal of Machine Learning Research 7 (2006) 1829–1859.

[15] U. Heinemann, A. Globerson, What cannot be learned with bethe approximations, in: Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence (UAI), 2011, pp. 319–326.

[16] H. Poon, P. Domingos, Sound and efficient inference with probabilistic and deterministic dependencies, in: Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI), 2006, pp. 458–463.

[17] V. Gogate, R. Dechter, SampleSearch: Importance sampling in presence of determinism, Artificial Intelligence 175 (2011) 694–729.

[18] D. Venugopal, V. Gogate, GiSS: Combining Gibbs sampling and SampleSearch for inference in mixed probabilistic and deterministic graphical models, in: Proceedings of the Twenty-Seventh Conference on Artificial Intelligence (AAAI), 2013.

[19] C. S. Jensen, U. Kjærulff, A. Kong, Blocking Gibbs sampling in very large probabilistic expert systems, International Journal of Human Computer Studies 42 (1995) 647–666.

[20] K. P. Murphy, Machine Learning: A Probabilistic Perspective, MIT Press, 2012.

[21] V. Gogate, Sampling Algorithms for Probabilistic Graphical Models with Determinism, Ph.D. thesis, University of California, Irvine, 2009.

[22] M. J. Wainwright, M. I. Jordan, Graphical models, exponential families, and variational inference, Foundations and Trends in Machine Learning 1 (2008) 1–305.

[23] Y. Weiss, Comparing the mean field method and belief propagation for approximate inference in MRFs, in: D. Saad, M. Opper (Eds.), Advanced Mean Field Methods, MIT Press, 2001, pp. 229–239.

[24] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, 1988.

[25] G. Elidan, I. McGraw, D. Koller, Residual belief propagation: Informed scheduling for asynchronous message passing, in: Proceedings of the 22nd Conference in Uncertainty in Artificial Intelligence (UAI), 2006, pp. 165–173.

[26] A. Choi, A. Darwiche, Relax, compensate and then recover, in: T. Onada, D. Bekki, E. McCready (Eds.), New Frontiers in Artificial Intelligence, volume 6797 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, 2011, pp. 167–180.

[27] S. M. Aji, R. J. McEliece, The generalized distributive law and free energy minimization, in: Proceedings of the 39th Allerton Conference on Communication, Control and Computing, 2001, pp. 672–681.

[28] R. Dechter, K. Kask, R. Mateescu, Iterative join-graph propagation, in: Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence (UAI), 2002, pp. 128–136.

[29] A. Choi, A. Darwiche, An edge deletion semantics for belief propagation and its practical impact on approximation quality, in: Proceedings of the 21st National Conference on Artificial Intelligence (AAAI), 2006, pp. 1107–1114.

[30] A. Choi, A. Darwiche, Approximating the partition function by deleting and then correcting for model edges, in: Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence (UAI), 2008, pp. 79–87.

[31] H. Chan, A. Darwiche, On the revision of probabilistic beliefs using uncertain evidence, Artificial Intelligence 163 (2005) 67–90.

[32] J. S. Yedidia, W. T. Freeman, Y. Weiss, Constructing free energy approximations and generalized belief propagation algorithms, IEEE Transactions on Information Theory 51 (2005) 2282–2312.

[33] N. Shental, A. Bar-hillel, T. Hertz, D. Weinshall, Computing Gaussian mixture models with EM using equivalence constraints, in: Proceedings of the 16th Annual Conference on Neural Information Processing Systems (NIPS), 2003.

[34] A. Asuncion, M. Welling, P. Smyth, Y.-W. Teh, On smoothing and inference for topic models, in: UAI, 2009, pp. 27–34.

[35] D. M. Blei, Probabilistic topic models, Communications of the ACM 55 (2012) 77–84.

[36] D. J. Hu, L. K. Saul, A probabilistic topic model for music analysis, in: NIPS Workshop on Applications for Topic Models: Text and Beyond, 2009.