On Inclusion-Driven Learning of Bayesian Networks

Robert Castelo RCASTELO@IMIM.ES
Biomedical Informatics Group (GRIB)
Department of Experimental and Health Sciences
Pompeu Fabra University
Passeig Marítim 37-49
E-08003 Barcelona, Spain

Tomáš Kočka KOCKA@CS.AUC.DK
Decision Support Systems Group
Department of Computer Science
Aalborg University
Fredrik Bajers Vej 7E
DK-9220 Aalborg, Denmark
and
Laboratory for Intelligent Systems Prague
University of Economics Prague
Ekonomicka 957
148 01 Praha, Czech Republic

Editor: Craig Boutilier

Abstract

Two or more Bayesian network structures are Markov equivalent when the corresponding acyclic digraphs encode the same set of conditional independencies. Therefore, the search space of Bayesian network structures may be organized in equivalence classes, where each of them represents a different set of conditional independencies. The collection of sets of conditional independencies obeys a partial order, the so-called "inclusion order."

This paper discusses in depth the role that the inclusion order plays in learning the structure of Bayesian networks. In particular, this role involves the way a learning algorithm traverses the search space. We introduce a condition for traversal operators, the *inclusion boundary condition*, which, when it is satisfied, guarantees that the search strategy can avoid local maxima. This is proved under the assumptions that the data is sampled from a probability distribution which is *faithful* to an acyclic digraph, and the length of the sample is unbounded.

The previous discussion leads to the design of a new traversal operator and two new learning algorithms in the context of heuristic search and the Markov Chain Monte Carlo method. We carry out a set of experiments with synthetic and real-world data that show empirically the benefit of striving for the inclusion order when learning Bayesian networks from data.

Keywords: Bayesian networks, graphical Markov model inclusion, inclusion boundary, structure learning

1. Introduction

Graphical Markov models (GMMs) in general, and Bayesian networks in particular, allow us to describe structural properties of a family of probability distributions using graphs. These structural properties are *conditional independence restrictions*, CI restrictions hereafter. The following statement is an example of a CI restriction:

> "Cholesterol intake is conditionally independent of the occurrence of a heart attack given the cholesterol blood level."

Such statements can be formally written by identifying the random variables that involve the events described (e.g., cholesterol intake X_{in}, heart attack X_{ha}, cholesterol blood level X_{cb}), and using the following notation introduced by Dawid (1979):

$$X_{in} \perp\!\!\!\perp X_{ha} \,|\, X_{cb}[P]\,,$$

where P refers to the probability distribution that satisfies it. In Section 2 we will see that graphs can also *satisfy* a particular CI restriction and hence one writes $X \perp\!\!\!\perp Y \,|Z[G]$, where G is the graph that satisfies the CI restriction. Sometimes, it will be clear from the context which probability distribution P, or which graph G, satisfies a given CI restriction, and we will remove $[P]$, or $[G]$, from the notation.

One usually says that a GMM, or its associated graph, *encodes* a set of CI restrictions. As we shall see later in detail, the collection of sets of CI restrictions for a given class of GMMs and a given number of variables, obeys a partial order relation called the *graphical Markov model inclusion partial order*, or *inclusion order* for short.

Structure learning is the process of learning from data the graphical structure of the GMM. The graphical structure of Bayesian networks is an acyclic digraph, also known as DAG.[1] DAGs make Bayesian networks a class of GMMs that allow for an efficient computation of their marginal likelihood given a dataset. This enables us to compare many Bayesian networks in a very short time and hence, devise automatic procedures to learn their graphical structure from data. However, the number of Bayesian network structures grows more than exponentially in the number of variables considered and this makes the learning problem a difficult one. A learning algorithm for Bayesian network structures typically consists of three components:

1. A *scoring metric* that in the light of data ranks two or more alternative Bayesian network structures.

2. A *traversal operator* that by means of local transformations of the Bayesian network structure creates a set of *neighbors*, i.e., a *neighborhood*.

3. A *search strategy* that repeatedly applies the traversal operator to a particular subset of Bayesian network structures. The members of this particular subset are considered in relation to the ranking that the scoring metric provides, and some policy that the search strategy follows.

1. Sometimes also referred as ADG.

When the learning algorithm stops and selects a model, or a subset of models, that do not provide the maximum score among all the possible models, one says that the learning algorithm ended up in a local maximum, or in a subset of local maxima.

The seminal paper by Cooper and Herskovits (1992) started a long stream of research works on structure learning. However, the relevance of the inclusion order has been addressed only in few of them: the PhD dissertation by Meek (1997), the authors' paper (Kočka and Castelo, 2001) and the very recent contribution by Chickering (2002b).

This paper elaborates on the approach presented by Kočka and Castelo (2001) which consists of providing a traversal policy for the search space of DAGs that accounts for the inclusion order. We provide a theoretical justification for this approach, considering only the circumstance where all variables, and all values throughout the records, are observed. We extend the experiments presented by Kočka and Castelo (2001), showing empirically the benefit of using such inclusion-driven learning algorithms.

The rest of the paper is organized as follows. In the next section we introduce the basic concepts regarding DAGs, Bayesian networks and their equivalence relation. In Section 3 we describe in detail the inclusion order and discuss its relevance to the learning problem. In Section 4 we introduce a traversal operator for DAGs that accounts for the inclusion order. Using this new traversal operator we implement two new learning algorithms for heuristic search and the Markov Chain Monte Carlo (MCMC) method. The performance of these two new learning algorithms is assessed in Section 5 by a set of experiments on synthetic and real-world data. Finally, we summarize and discuss the contributions of this paper in Section 6.

2. Preliminaries

In this section we introduce first the basic notions about graphs. Afterwards, the concepts of Bayesian network and Markov equivalence are described. Finally, we give a formal description of the notion of neighborhood.

2.1 Graphical Terminology

The terminology and notation used for graphs within the context of GMMs have been borrowed mainly from Lauritzen (1996). A directed graph G is a pair (V, E) where V is the set of vertices and E is the set of directed edges or arcs. Since in this paper we deal mainly with directed graphs we will sometimes use the shorter form *edge*, or *arc*, to denote *directed edge*. Two vertices are *adjacent* when there is an edge joining them.

A sequence of vertices $a = v_0, v_1, \ldots, v_n = b$, $n > 0$, forms a *path* between vertices a and b if for every pair (v_i, v_{i+1}), there is either an arc $v_i \to v_{i+1}$ or $v_i \leftarrow v_{i+1}$. The path is *directed* if the arc is always $v_i \to v_{i+1}$. A directed cycle is a directed path, as the previous one, where $a = b$. An acyclic digraph, or DAG, is a directed graph where its set of arcs E does not induce directed cycles nor has loops (arcs where the two endpoints are the same vertex) nor has multiple edges (two or more edges joining the same pair of vertices). The *skeleton* of a DAG is its underlying undirected graph, i.e., the DAG itself without regard to the directions of the arcs.

Given a vertex v, the set of vertices reachable from v by directed paths is known as the set of *descendants* of v, and denoted by $de(v)$. Consequently, the set of *non-descendants*

of v is defined as $nd(v) = V \backslash \{de(v) \cup \{v\}\}$. For a given vertex v, the set of vertices that can reach v by directed paths form the *ancestor set* of v, denoted by $an(v)$. A subset of vertices $A \subseteq V$ is *ancestral* if and only if for every vertex $v \in A$, $an(v) \subseteq A$. For any subset of vertices $A \subseteq V$ there is always some larger subset that contains A and it is ancestral. The smallest of the ancestral sets containing A is called the *smallest ancestral set* of A and noted $An(A)$.

The set of vertices with arcs pointing to a vertex v is the *parent set* of v, denoted by $pa(v)$. When two vertices of $pa(v)$ are not adjacent, i.e., they are not connected by an arc in either direction, it is said that they form an *immorality* with respect to v, as for example in the graph $a \rightarrow b \leftarrow c$. A DAG that has no immoralities is said to be *moral*. A DAG, either moral or not, can be *moralized* by converting it into an undirected graph as follows. Every pair of non-adjacent parents that induce an immorality is married by joining them with an undirected edge, and the directions on the rest of the edges are dropped. The moralized version of a DAG G is denoted by G^m. A *subgraph* $G_S = (S, E_S)$ of a graph $G = (V, E)$ *induced* by a subset of vertices $S \subseteq V$ has its edge set defined as $E_S = E \cap (S \times S)$.

2.2 Bayesian networks

The standard definition of Bayesian network is a pair (G, θ), where G is a DAG and θ is a particular set of parameters. The set of parameters θ specifies the conditional probability distributions associated to the random variables represented in G, and provides a succinct description of the joint probability distribution of these variables. The DAG G is also known as the *structure* of the Bayesian network.

In the context of structure learning, the Bayesian network structure is often identified as the Bayesian network itself because learning the parameters can be done once the structure has been learned. However, another implicit reason, which is important for this paper, is that the Bayesian network structure conveys a model on its own, a conditional independence model. More concretely, as Whittaker (1990, pg. 207), we use the term *model* to specify an arbitrary family of probability distributions that satisfies a set of CI restrictions in the following way. A probability distribution P is Markov over a graph G if every CI restriction encoded in G is satisfied by P. A graphical Markov model (GMM), denoted by $\mathbf{M}(G)$, is the family of probability distributions that are Markov over G (Whittaker, 1990, pg. 13). One also says that G *determines* the GMM $\mathbf{M}(G)$.

A class $\mathcal{M}$ of GMMs is the set of GMMs determined by the same type of graph. Bayesian networks form the class of GMMs where the graph G, i.e., the Bayesian network structure, that determines the model, is a DAG. A DAG encodes a set of CI restrictions through a particular graphical criterion, the directed global Markov property (DGMP).

Definition 2.1 *Directed global Markov property (DGMP)*
Let $G = (V, E)$ be a DAG. A probability distribution P is said to satisfy the directed global Markov property *(DGMP) if, for any triplet (A, B, S) of disjoint subsets of V, where A, B are non-empty, such that S separates A from B in the moralized version of the subgraph induced by the vertices in $An(A \cup B \cup S)$, i.e., in $G^m_{An(A \cup B \cup S)}$, P satisfies*

$$A \perp\!\!\!\perp B \,|\, S[P]\,.$$

The DGMP means that two non-empty subsets of vertices A, B are conditionally independent given a third subset S, if S separates A and B in the moralized subgraph induced by the smallest ancestral set of $A \cup B \cup S$. The DGMP is the sharpest possible graphical criterion that permits reading CI restrictions from a given DAG (Pearl and Verma, 1987, Lauritzen et al., 1990).

An alternative way of reading conditional independencies in a DAG is using the *d-separation* criterion of Pearl and Verma (1987), which we review now. A vertex v_i in a path $v_0, v_1, \ldots, v_n$, $n > 1$, is a *collider* if v_{i-1} and v_{i+1} are parent vertices of v_i A vertex in the path which is not a collider is a *non-collider*. Given two vertices $u, v \in V$ and a subset $S \subseteq V$ where $u, v \notin S$, one says that a path between u and v is *active* with respect to S if

1. every non-collider in the path is not in S, and

2. every collider in the path is in S or has a descendant in S.

When a path between two vertices u, v *is not* active with respect to S, one says that the path is *blocked* by S. Given these notions of active and blocked path, the d-separation criterion is defined as follows.

Definition 2.2 *d-separation*
Let $G = (V, E)$ be a DAG. For any triplet (A, B, S) of disjoint subsets of V, where A, B are non-empty, A and B are d-separated by S if every path between the vertices in A and B is blocked by S.

Lauritzen et al. (1990) prove that the *d-separation* criterion encodes in a DAG exactly the same CI restrictions as the DGMP.

A DAG Markov model, or DAG model, denoted by $\mathbf{D}(G)$, is a GMM whose set of probability distributions satisfy the DGMP relative to the DAG G (Andersson et al., 1995, Definition 3.2). Other classes of GMMs may be analogously specified and we refer the interested reader to the books of Whittaker (1990) and Lauritzen (1996) for definitions of other Markov properties that lead to some of the different classes of GMMs.

A Bayesian network structure determines, therefore, a DAG model. In this paper we use the term Bayesian network to mean a DAG model, and therefore we also denote a Bayesian network by $\mathbf{D}(G)$. We will often use the shorter term DAG to denote the Bayesian network structure. Sometimes we will introduce concepts in the wider scope of GMMs and, for clarity, we will use $\mathbf{M}(G)$ when we deal with GMMs in general, and $\mathbf{D}(G)$ when we deal with Bayesian networks in particular.

Note that each GMM, and therefore each Bayesian network, is determined by a graph G that represents it. Different graphs of different types (or of the same type as we shall see in the next subsection) can encode the same set of CI restrictions and thus represent the same model. Given a Bayesian network $\mathbf{D}(G)$ determined and represented by a DAG G, we consider the neighborhood of G as a set of DAGs. Such a neighborhood represents a set of neighboring models of the DAG G, because each DAG member of the neighborhood represents a DAG model. From a different perspective, one might define neighborhood in terms of models, but we are interested in using neighborhoods that, while being different, they may be created from different graphs representing the same model, and therefore, a neighborhood definition in terms of graphs is required.

Let $G = (V, E)$ be a DAG whose vertex set V indexes a set of n categorical random variables $\mathbf{X}_V = \{X_i | i \in V\}$, where each variable X_i takes values from a domain $\mathcal{X}_i$. The random variables in $\mathbf{X}_V$ allow for sampling a collection of categorical observations that form a dataset D following a multinomial distribution defined on a product space $\mathcal{X}_V = \times(\mathcal{X}_i | i \in V)$. Elements of $\mathcal{X}_V$ are denoted by $\mathbf{x}_V$, while elements of each $\mathcal{X}_i$ are denoted by x_i. Lauritzen et al. (1990) show that by using the conditional probability distributions θ_i of every X_i given $\mathbf{X}_{pa(i)} = \mathbf{x}_{pa(i)}$, the joint distribution $p(\mathbf{x}_V | G, \theta)$, where $\theta = \{\theta_i | i \in V\}$, admits the unique *recursive factorization*:

$$p(\mathbf{x}_V | G, \theta) = \prod_{i \in V} p(x_i | \mathbf{x}_{pa(i)}, \theta_i) \,. \tag{1}$$

The fact that the vertices $\{i\} \cup pa(i)$ form a complete subgraph in the moralized version of G, i.e., in G^m, implies this factorization and the more interesting fact that $p(\mathbf{x}_V | G, \theta)$ obeys the DGMP relative to G^m (Lauritzen et al., 1990).

The recursive factorization in (1) allows us to obtain a closed formula for the marginal likelihood of the data D given a Bayesian network $M \equiv \mathbf{D}(G)$, $p(D|M)$, under a certain set of assumptions about D (Buntine, 1991, Cooper and Herskovits, 1992, Heckerman et al., 1995). The logarithm of the marginal likelihood and the prior of the model, $\log[p(D|M)p(M)]$, is often used as a scoring metric for Bayesian networks. Throughout this paper we have used the BDeu scoring metric, which corresponds to the BDe metric from Heckerman et al. (1995) with uniform hyper-Dirichlet priors (Buntine, 1991). We have also considered a uniform prior distribution $p(M)$ over the space of Bayesian networks.

2.3 Markov Equivalence

The search space of Bayesian network structures is defined as the space of DAGs, which we call "DAG-space." An equivalence class of Bayesian network structures comprises all DAGs that encode the same set of CI restrictions. Each equivalence class has a canonical representation in the form of an acyclic partially directed graph where the edges may be directed and undirected and satisfy some characterizing conditions (Spirtes et al., 1993, Chickering, 1995, Andersson et al., 1997a). This representation has been introduced independently by several authors under different names: *pattern* (Spirtes et al., 1993), *completed PDAG* (Chickering, 1995) and *essential graph* (Andersson et al., 1997a). We adopt here the term essential graph (EG). An EG equivalent to a DAG has the same skeleton and each edge in the EG is directed iff this edge has the same orientation in all equivalent DAGs, otherwise it is undirected.

Consider a Bayesian network $\mathbf{D}(G)$ whose structure G belongs to an equivalence class with two or more members. Then G has at least one edge that can be reversed in such a way that the resulting DAG remains in the same equivalence class. Chickering (1995) characterized such edges in the following way.

Definition 2.3 *Covered Edge (Chickering, 1995)*
Let $G = (V, E)$ be a DAG. An edge $a \to b \in E$ is covered *in G if $pa(b) = \{a\} \cup pa(a)$.*

Using this characterization we can describe more formally the notion of Markov equivalence among Bayesian network structures.

Lemma 2.1 *Let G and G' be two Bayesian network structures. The following three conditions are equivalent:*

1. $\mathbf{D}(G) = \mathbf{D}(G')$*, i.e., G and G' are Markov equivalent.*
2. *G and G' have the same skeleton and contain the same immoralities.*
3. *There exists a sequence $L_1, \ldots, L_n$ of DAGs such that $L_1 = G$ and $L_n = G'$ and L_{i+1} is obtained from L_i by reversing a covered arc in L_i for $i = 1, \ldots, n-1$.*

The equivalence (1) $\Leftrightarrow$ (2) was proven by Verma and Pearl (1990) and Andersson et al. (1997a), and in the more general framework of chain graphs by Andersson et al. (1997b, Theorem 3.1).[2] The equivalence (1) $\Leftrightarrow$ (3) was proven by Chickering (1995), Heckerman et al. (1995), Andersson et al. (1997a).

A scoring metric, for instance, BDe (Heckerman et al., 1995), is *score equivalent* if it gives equal scores to Markov equivalent Bayesian network structures.

The relationship of Markov equivalence organizes DAG-space into equivalence classes which form what we call "EG-space." Each member of EG-space has a graphical representation as an EG. From the asymptotic number of DAGs given by Robinson (1973) it follows that DAG-space grows more than exponentially in the number of vertices. One may think that EG-space departs substantially from such rate but recent empirical investigation gives a quite different perspective.

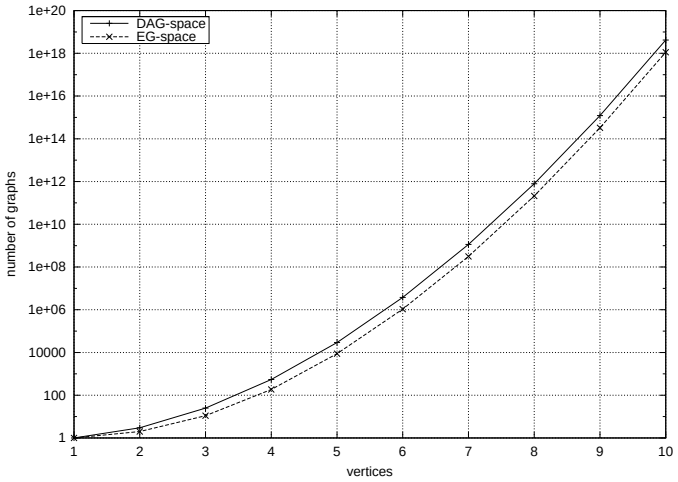


Figure 1: Cardinalities of DAG-space and EG-space.

Observation 2.1 *(Gillispie and Perlman, 2001)*
The average ratio of DAGs per equivalence class seems to converge to an asymptotic value smaller than 3.7. This was observed up to 10 vertices.

2. Frydenberg (1990) also proved it but under the additional condition of the fifth graphoid axiom CI5 (see Pearl, 1988).

In Figure 1 we see the cardinalities of DAG-space and EG-space plotted, up to 10 vertices. From a non-causal perspective one is interested in learning equivalence classes of Bayesian networks from data, and for that purpose one uses a score equivalent scoring metric. In such situation it makes sense to use EG-space instead of DAG-space. This argument has been further supported by several authors (Heckerman et al., 1995, Madigan et al., 1996) who cite the following advantages:

1. The cardinality of EG-space is smaller than in DAG-space.

2. The scoring metric is no longer constrained to give equal scores to Markov equivalent Bayesian networks. The use of a score equivalent scoring metric imposes strong constraints on the prior distribution of both, the graphs and the parameters (Andersson et al., 1997a, Section 7.2).

3. Quantities computed from Bayesian model averaging outputs cannot be biased by the size of the equivalence classes of Bayesian networks.

According to Observation 2.1, the first advantage does not alleviate substantially the learning problem. The second advantage is not effective as long as we use a DAG-based scoring metric, and hence the need to first develop scoring metrics specifically for EGs - cf. (Castelo and Perlman, 2002). The third advantage affects only MCMC learning algorithms and it is currently an open question the extent to which the use of DAG-space biases the correct output of the learning process.

Learning algorithms using EG-space were formerly developed by Spirtes and Meek (1995), Chickering (1996) and Madigan et al. (1996). While they improve the quality of the learned models, with respect to DAG-space approaches, their computational overhead is a serious burden. In the algorithms of Spirtes and Meek (1995), Chickering (1996) this overhead was mainly produced by the need to switch between EG-space and DAG-space to score the models, and in the algorithm of Madigan et al. (1996) it was mainly produced by the need to consider transformations on two adjacencies at a time to fulfill irreducibility of a Markov chain.

Recently, Chickering (2002a) has provided an algorithm that works in EG-space and computes the scores efficiently but it does not account for the inclusion order. More recently, Chickering (2002b) has developed a learning algorithm that combines the efficient scoring scheme for EG-space from Chickering (2002a) with a search policy that respects the inclusion order.

2.4 Concepts of Neighborhood

The transformations of a single adjacency of a DAG performed by learning algorithms for Bayesian network structures are usually addition, removal and reversal of the arc such that acyclicity in the directed graph is preserved. Using these transformations we may formalize the following concepts of neighborhood:

- **NR** (No Reversals) All DAGs with one arc less, and one arc more that does not introduce a directed cycle.

- **AR** (All Reversals) The NR neighborhood plus all DAGs with one arc reversed that does not introduce a directed cycle.
- **CR** (Covered Reversals) The NR neighborhood plus all DAGs with one covered arc reversed.[3]
- **NCR** (Non-Covered Reversal) The NR neighborhood plus all DAGs with one non-covered arc reversed that does not introduce a directed cycle.

For any given DAG G the previously described neighborhoods will be denoted by $\mathcal{N}_{\mathrm{NR}}(G)$, $\mathcal{N}_{\mathrm{AR}}(G)$, $\mathcal{N}_{\mathrm{CR}}(G)$ and $\mathcal{N}_{\mathrm{NCR}}(G)$, respectively. The NR neighborhood is used in MCMC search by the MC^3 algorithm of Madigan and York (1995). The NR neighborhood may lead easily to local maxima in heuristic search, or to an extremely small probability of reaching the most probable model given the data in MCMC search. This problem may be alleviated by using an AR neighborhood, which is quite common in many other learning algorithms. The CR and NCR neighborhoods are variations of the AR neighborhood that are not intended to enhance the AR neighborhood but are used here, as we shall see later, for comparison purposes.

3. Graphical Markov Model Inclusion

By *graphical Markov model inclusion* we denote a particular partial order among GMMs. A partial order is reflexive, asymmetric and transitive, and some pairs of elements may not be related, otherwise it would be a *total order*. The intuition behind the inclusion order is that one GMM $\mathbf{M}(G)$ precedes another GMM $\mathbf{M}(G')$ if and only if all the CI restrictions encoded in G are also encoded in G'.

A complete DAG G_c that encodes no CI restriction at all, determines a Bayesian network $\mathbf{D}(G_c)$ that consists of all possible discrete probability distributions over the corresponding set of random variables, due to the fact that any of such distributions is always Markov over the complete DAG G_c.

On the opposite side, we find the empty DAG $G_\emptyset$, with no edges, under which all random variables are marginally independent. It encodes all possible CI restrictions among these random variables under the closure of the semi-graphoid axioms (Pearl, 1988). The DAG $G_\emptyset$ determines a Bayesian network $\mathbf{D}(G_\emptyset)$ that consists of "only" those discrete probability distributions under which all the random variables are marginally independent. Clearly, the set of probability distributions that are Markov over G_c includes those that are Markov over $G_\emptyset$, therefore

$$\mathbf{D}(G_\emptyset) \subseteq \mathbf{D}(G_c)\,. \tag{2}$$

However, the CI restrictions encoded by G_c (none) are included into those encoded by $G_\emptyset$ (all), and this latter notion is the one that determines the inclusion order. The notation used in (2) might be somewhat counterintuitive with the idea that $\mathbf{D}(G_c)$ precedes $\mathbf{D}(G_\emptyset)$ under the inclusion order. Therefore, we will explicitly express the set of the CI restrictions encoded by a graph G that determines the GMM $\mathbf{M}(G)$ as:

3. The reversal of a covered arc cannot introduce a directed cycle (Kočka et al., 2001).

$$\mathbf{M}^I(G) = \{(A, B, S) \,:\, A, B \neq \emptyset \wedge A \perp\!\!\!\perp B | S[G]\}\,.$$

Here we are using $\mathbf{M}^I(G)$, instead of $\mathbf{D}^I(G)$, to emphasize that this notation applies to any given class of GMMs. Now, in order to denote the inclusion relationship between the fully restricted Bayesian network $\mathbf{D}(G_\emptyset)$ and the unrestricted Bayesian network $\mathbf{D}(G_c)$ we simply write it as:

$$\mathbf{D}^I(G_c) \subseteq \mathbf{D}^I(G_\emptyset)\,.$$

The graphical characterization of the inclusion order for two arbitrary Bayesian networks has been an open question for a long time. The first attempt to provide necessary and sufficient conditions to characterize the inclusion order among Bayesian networks was done by Verma and Pearl (1988), and Kočka (2001) shows that these conditions are necessary but not sufficient. Later, Meek (1997) conjectured the following operational criterion to decide the inclusion order among Bayesian networks.

Conjecture 3.1 *Meek's conjecture, Meek (1997)*
Let $\mathbf{D}(G)$ and $\mathbf{D}(G')$ be two Bayesian networks determined by two DAGs G and G'. The conditional independence model induced by $\mathbf{D}(G)$ is included in the one induced by $\mathbf{D}(G')$, i.e., $\mathbf{D}^I(G) \subseteq \mathbf{D}^I(G')$, if and only if there exists a sequence of DAGs $L_1, \ldots, L_n$ such that $G = L_1$, $G' = L_n$ and the DAG L_{i+1} is obtained from L_i by applying either the operation of covered arc reversal or the operation of arc removal for $i = 1, \ldots, n$.

Despite its simple and intuitive appearance, it took a few years until a proof was found by Kočka et al. (2001) for the particular case in which G and G' differ in, at most, one adjacency. Recently, Chickering (2002b) has provided a constructive proof (by means of an algorithm) of the following theorem that verifies and sharpens Meek's conjecture.

Theorem 3.1 *Chickering (2002b)*
Let $\mathbf{D}(G)$ and $\mathbf{D}(G')$ be two Bayesian networks determined by any pair of DAGs G and G' such that $\mathbf{D}^I(G) \subseteq \mathbf{D}^I(G')$. Let r be the number of edges in G that have opposite orientation in G', and let m be the number of edges in G that do not exist in either orientation in G'. There exists a sequence of at most $r + 2m$ distinct edge reversals and additions in G' with the following properties:

1. *Each edge reversed is a covered edge.*
2. *After each reversal and addition G' is a DAG and $\mathbf{D}^I(G) \subseteq \mathbf{D}^I(G')$.*
3. *After all reversals and additions $G = G'$.*

Note that Theorem 3.1 and Lemma 2.1 are results that allow one to make strong claims about complex interaction models by pure graphical criteria. As we shall see later, Theorem 3.1 leads to other results that have a more direct impact in learning Bayesian networks from data.

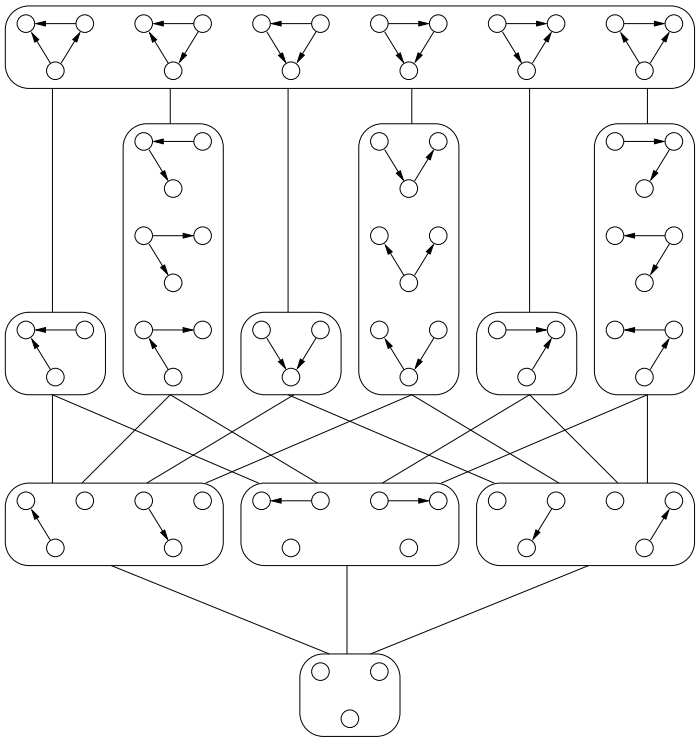


Figure 2: Hasse diagram of the space of Markov equivalence classes of Bayesian network structures over three variables.

Consider the collection of GMMs $\mathcal{K}_{\mathcal{M},n} = (\mathbf{M}(G_1), \ldots, \mathbf{M}(G_p))$ where all $G_i, 1 \leq i \leq p$, have n vertices and all $\mathbf{M}(G_i), 1 \leq i \leq p$, belong to the same class $\mathcal{M}$ of GMMs. The collection of sets of CI restrictions $(\mathbf{M}^I(G_1), \ldots, \mathbf{M}^I(G_p))$, obtained from $\mathcal{K}_{\mathcal{M},n}$, forms a *poset*. A *poset* $(S, \leq)$ is a set S equipped with a partial order relation $\leq$, and it can be represented by means of a *Hasse diagram*. A *Hasse diagram* of a poset S is a representation of an undirected graph in the plane such that for any pair of elements $x, y \in S$, x is below y in the plane iff $x < y$, and x is adjacent to y iff $x < y$ and $x \leq z < y \Rightarrow z = x$. In Figure 2 we see this representation for the Markov equivalence classes of Bayesian networks over three variables.

From the perspective of the search space that the Hasse diagram in Figure 2 provides, the concept of *inclusion boundary* follows. This concept applies to every type of GMM. As we shall see throughout the paper, this concept is the key to understanding the relevance of the inclusion order in the learning task.

Definition 3.1 *Inclusion boundary (Kočka, 2001)*
Let $\mathbf{M}(H), \mathbf{M}(L)$ *be two GMMs determined by the graphs* H *and* L. *Let* $\mathbf{M}^I(H) \prec \mathbf{M}^I(L)$ *denote that* $\mathbf{M}^I(H) \subset \mathbf{M}^I(L)$ *and for no graph* K, $\mathbf{M}^I(H) \subset \mathbf{M}^I(K) \subset \mathbf{M}^I(L)$. *The* inclusion boundary *of the GMM* $\mathbf{M}(G)$, *denoted by* $\mathcal{IB}(G)$, *is*

$$\mathcal{IB}(G) = \{\mathbf{M}(H) \,|\, \mathbf{M}^I(H) \prec \mathbf{M}^I(G)\} \cup \{\mathbf{M}(L) \,|\, \mathbf{M}^I(G) \prec \mathbf{M}^I(L)\}\,.$$

Intuitively, the inclusion boundary of a given GMM $\mathbf{M}(G)$ consists of those GMMs $\mathbf{M}(G_i)$ that induce a set of CI restrictions $\mathbf{M}^I(G_i)$ which *immediately* follow or precede $\mathbf{M}^I(G)$ under the inclusion order. In the Hasse diagram of Figure 2, the inclusion boundary for a given node is formed by those nodes adjacent to it.

We will say that a sequence of GMMs $\mathbf{M}(G_1), \ldots, \mathbf{M}(G_n)$ forms an *inclusion path*, or are *in inclusion*, if either $\mathbf{M}^I(G_1) \supset \ldots \supset \mathbf{M}^I(G_n)$ or $\mathbf{M}^I(G_1) \subset \ldots \subset \mathbf{M}^I(G_n)$. We will say that an inclusion path between $\mathbf{M}(G_1)$ and $\mathbf{M}(G_n)$ is *maximal* if each GMM $\mathbf{M}^I(G_i)$ in this path has its neighbors in its inclusion boundary i.e., $\mathbf{M}^I(G_{i+1}), \mathbf{M}^I(G_{i-1}) \in \mathcal{IB}(G_i)$. For each pair of GMMs in inclusion $\mathbf{M}^I(G) \supset \mathbf{M}^I(H)$ there is at least one maximal inclusion path with endpoints at $\mathbf{M}^I(G)$ and $\mathbf{M}^I(H)$. Its existence follows from the definition of inclusion boundary.

The following definition of *inclusion boundary condition* establishes a necessary condition that a traversal operator must satisfy in order to avoid local maxima during the learning process. Later we will show that in the case of Bayesian networks, and under some additional assumptions, this condition is sufficient to avoid local maxima.

Definition 3.2 *Inclusion boundary condition (Kočka, 2001)*
A traversal operator satisfies the inclusion boundary condition *if for every GMM determined by a graph G, the traversal operator can create a neighborhood $\mathcal{N}(G)$ such that*

$$\{\mathbf{M}(G_i) \,|\, G_i \in \mathcal{N}(G)\} \supseteq \mathcal{IB}(G)\,.$$

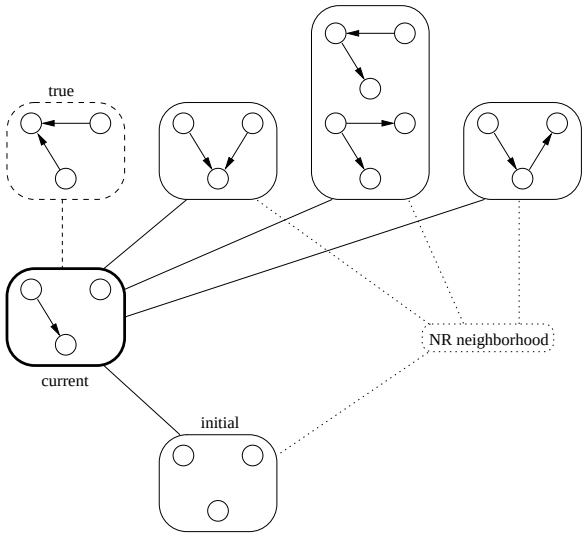


Figure 3: Example of getting stuck in a local maximum because the concept of neighborhood employed (NR) does not contain the inclusion boundary.

In Figure 3 we find a situation in the context of Bayesian networks, in which the model highlighted with a thick box is our current model represented by a particular DAG. The

model highlighted with a dashed box is the one from which the data is sampled (the *true* model).

All other DAGs are the NR neighborhood of the DAG that represents the current model and they are grouped in a box when they represent the same model (i.e., when the DAGs belong to the same equivalence class). All models around the current one form its inclusion boundary. As we may appreciate, because the models represented by the current neighborhood do not cover the entire the inclusion boundary, it is not possible to reach the *true* model from the current one in a single step. In this situation, it may become very difficult for the learning algorithm to reach the *true* model. Note that starting from the fully restricted model represented by the empty graph, it may easily happen that we select the current model, provided that a score equivalent metric would score equally in either direction the addition of a single isolated edge.

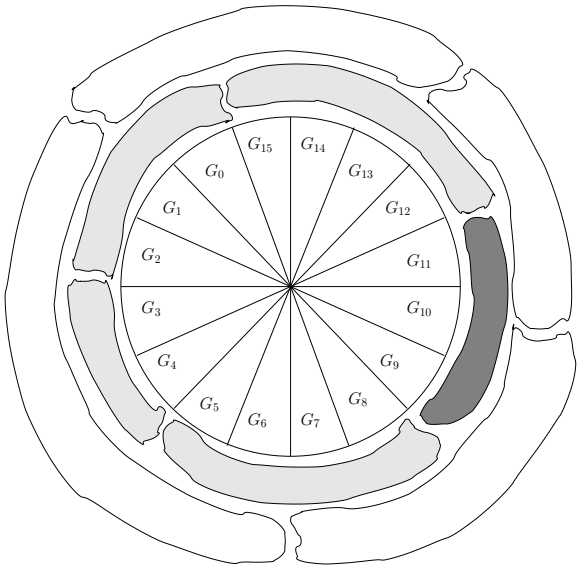


Figure 4: No member $G_0, \ldots, G_{15}$ of an equivalence class can reach the entire inclusion boundary (shaded classes) by a transformation of a single adjacency.

From the previous example, it follows that the NR neighborhood does not retain the inclusion boundary condition. Later on, in Theorem 3.2 we will see that the neighborhood AR does not retain it either, which also holds for the CR and NCR neighborhoods as they are subsets of AR.

In Figure 4 we may find this problem more clearly illustrated. In the center we have a circle that represents an equivalence class of DAGs with 16 members determined by the graphs G_0 to G_{15}. Surrounding this circle we have other equivalence classes, where those that are shaded represent its inclusion boundary. Furthermore, an equivalence class can reach those depicted next to it by a transformation in a single adjacency of one of the DAGs within the class.

Consider the equivalence class in the inclusion boundary that is more darkly shaded. As it is clear from the figure, only G_9, G_{10} and G_{11} can reach this equivalence class by a

transformation in a single adjacency. Therefore, if our learning algorithm does not consider any of the G_9, G_{10} or G_{11}, it will not be able to reach that class within the inclusion boundary in a single move.

We are going to define a neighborhood in DAG-space that will be the same for any two Markov equivalent DAGs and for which the set of represented neighboring models coincides with the inclusion boundary. In the next section, we will provide an efficient implementation of this neighborhood.

Let $\mathbf{D}(G)$ be any given Bayesian network represented by a DAG G that belongs to some equivalence class $\mathcal{C} = \{G' | \mathbf{D}(G') = \mathbf{D}(G)\}$. Consider the following two neighborhoods for G:

- **ENR** (Equivalence class No Reversals) For every member of the equivalence class, $G' \in \mathcal{C}$, consider all DAGs with one arc less and one arc more from G'.
- **ENCR** (Equivalence class Non-Covered Reversals) For every member of the equivalence class, $G' \in \mathcal{C}$, consider all DAGs with one arc less, one arc more and one non-covered arc reversed from G'.

Now, we will investigate the relationships between the ENR and ENCR neighborhoods, and the other neighborhoods previously defined. We begin by reviewing the directed pairwise Markov property (Lauritzen et al., 1990):

Definition 3.3 *Directed pairwise Markov property (DPMP)*
Let $G = (V, E)$ be a DAG. A probability distribution P is said to satisfy the directed pairwise Markov property *(DPMP) with respect to a DAG G if, for any pair $u, v \in V$ of non-adjacent vertices such that $v \in nd(u)$, P satisfies*

$$u \perp\!\!\!\perp v \,|\, nd(u) \backslash \{v\} [P] \,.$$

Lauritzen et al. (1990) proved that all CI restrictions encoded by the DPMP are encoded by the DGMP and the d-separation criterion as well. The following two lemmas provide insight into the relationship between the inclusion order and the graphical structure of Bayesian networks.

Lemma 3.1 *Let $\mathbf{D}(G)$ and $\mathbf{D}(G')$ be two Bayesian networks represented by two DAGs G and G'. If G and G' have the same number of edges then either the two DAGs are equivalent, i.e., $\mathbf{D}(G) = \mathbf{D}(G')$, or the two Bayesian networks are not in inclusion, i.e., $\mathbf{D}^I(G) \not\subset \mathbf{D}^I(G')$ and $\mathbf{D}^I(G) \not\supset \mathbf{D}^I(G')$.*

Proof We will distinguish three cases. First when the two DAGs G and G' have different skeletons, second when G and G' have the same skeletons but different immoralities, and third when G and G' have the same skeletons and the same immoralities.

In the first case, the two skeletons of G and G' are different but have the same number of edges. This implies that there are at least two distinct vertices u and v that are adjacent in G and are non-adjacent in G'. Either $v \in nd(u)$, or $u \in nd(v)$, holds in G' as otherwise there

would be a cycle. By the DPMP there exists a CI restriction between u and v in G'. This CI restriction cannot hold in G according to the d-separation criterion because u and v are adjacent in G. Thus $\mathbf{D}^I(G) \not\supset \mathbf{D}^I(G')$. The same argument applies for $\mathbf{D}^I(G) \not\subset \mathbf{D}^I(G')$.

In the second case, G and G' have the same skeletons but different immoralities. Let $u \rightarrow w \leftarrow v$ be an immorality formed by three distinct vertices u, v and w which, without loss of generality, is in G but not in G'. This implies that the subgraph induced in G' by u, v and w will be either $u \leftarrow w \leftarrow v$ or $u \rightarrow w \rightarrow v$ or $u \leftarrow w \rightarrow v$ where u and v are non-adjacent as in G.

Again, either $v \in nd(u)$ or $u \in nd(v)$ holds in G'. By means of the DPMP there is a CI restriction $u \perp\!\!\!\perp v|C$ in G' for some C where $w \in C$. If $w \notin C$, $u \perp\!\!\!\perp v|C$ would not hold in G' according to the d-separation criterion. The same CI restriction $u \perp\!\!\!\perp v|C$, where $w \in C$, cannot hold in G because $w \in C$ creates an active path between u and v, so they are not d-separated. This leads to $\mathbf{D}^I(G) \not\supset \mathbf{D}^I(G')$. Analogously, for some subset C' where $w \notin C'$, the CI restriction $u \perp\!\!\!\perp v|C$ holds in G while it does not hold in G', therefore $\mathbf{D}^I(G) \not\subset \mathbf{D}^I(G')$. The third case follows from Lemma 2.1, which leads to $\mathbf{D}(G) = \mathbf{D}(G')$. ■

Lemma 3.2 *Let $\mathbf{D}(G)$ and $\mathbf{D}(G')$ be two Bayesian networks. If $\mathbf{D}^I(G) \subset \mathbf{D}^I(G')$ then G' has fewer edges than G.*

Proof We prove this lemma by contradiction. Assume $\mathbf{D}^I(G) \subset \mathbf{D}^I(G')$ and the two possibilities: (a) the two DAGs G and G' have the same number of edges or (b) G' has at least one more edge than G.

In (a) the contradiction follows from Lemma 3.1 which forces $\mathbf{D}^I(G)$ and $\mathbf{D}^I(G')$ to be either equivalent or not in inclusion.

In (b) the contradiction follows from the fact that there is a pair of vertices u, v which are non-adjacent in G and adjacent in G'. By the DPMP there is a CI restriction $u \perp\!\!\!\perp v|nd(u)\backslash v$ in G which does not hold in G', i.e., $\mathbf{D}^I(G) \not\subset \mathbf{D}^I(G')$. ■

The following result discusses some of the relationships among the different concepts of neighborhoods and the inclusion boundary.

Theorem 3.2 *Let $\mathbf{D}(G)$ denote a Bayesian network represented by a DAG G. Let $\mathcal{N}_{\mathrm{NR}}(G)$, $\mathcal{N}_{\mathrm{CR}}(G)$, $\mathcal{N}_{\mathrm{NCR}}(G)$, $\mathcal{N}_{\mathrm{AR}}(G)$, $\mathcal{N}_{\mathrm{ENR}}(G)$ and $\mathcal{N}_{\mathrm{ENCR}}(G)$ be the sets of DAGs that form, respectively, the NR, CR, NCR, AR, ENR and ENCR neighborhoods of G. The following statements hold:*

1. *For all G:*

$$\mathcal{N}_{\mathrm{NR}}(G) \subseteq \mathcal{N}_{\mathrm{ENR}}(G) \text{ and } \{\mathbf{D}(G_i) \,|\, G_i \in \mathcal{N}_{\mathrm{ENR}}(G)\} = \mathcal{IB}(G)$$
$$\mathcal{N}_{\mathrm{NR}}(G) \subseteq \mathcal{N}_{\mathrm{NCR}}(G) \subseteq \mathcal{N}_{\mathrm{ENCR}}(G)$$
$$\mathcal{N}_{\mathrm{NR}}(G) \subseteq \mathcal{N}_{\mathrm{CR}}(G) \subseteq \mathcal{N}_{\mathrm{AR}}(G)$$
$$\mathcal{N}_{\mathrm{NR}}(G) \subseteq \mathcal{N}_{\mathrm{NCR}}(G) \subseteq \mathcal{N}_{\mathrm{AR}}(G)$$
$$\mathcal{N}_{\mathrm{ENR}}(G) \subseteq \mathcal{N}_{\mathrm{ENCR}}(G)$$

2. For all G:

$$\begin{aligned}
\{\mathbf{D}(G_i) \,|\, G_i \in (\mathcal{N}_{\mathrm{NCR}}(G) \backslash \mathcal{N}_{\mathrm{NR}}(G))\} \cap \mathcal{IB}(G) &= \emptyset \\
\{\mathbf{D}(G_i) \,|\, G_i \in (\mathcal{N}_{\mathrm{AR}}(G) \backslash \mathcal{N}_{\mathrm{CR}}(G))\} \cap \mathcal{IB}(G) &= \emptyset \\
\{\mathbf{D}(G_i) \,|\, G_i \in (\mathcal{N}_{\mathrm{ENCR}}(G) \backslash \mathcal{N}_{\mathrm{ENR}}(G))\} \cap \mathcal{IB}(G) &= \emptyset \\
\{\mathbf{D}(G_i) \,|\, G_i \in (\mathcal{N}_{\mathrm{CR}}(G) \backslash \mathcal{N}_{\mathrm{NR}}(G))\} \cap \mathcal{IB}(G) &= \emptyset
\end{aligned}$$

3. For more than two vertices, there exists a DAG G such that:

$$\{\mathbf{D}(G_i) \,|\, G_i \in \mathcal{N}_{\mathrm{AR}}(G)\} \not\supseteq \mathcal{IB}(G).$$

Proof *Statement 1.* The part $\mathcal{N}_{\mathrm{NR}}(G) \subseteq \mathcal{N}_{\mathrm{ENR}}(G)$ follows directly from the fact that ENR performs all operations that NR does. The same argument applies to the $\mathcal{N}_{\mathrm{NR}}(G) \subseteq \mathcal{N}_{\mathrm{NCR}}(G)$, $\mathcal{N}_{\mathrm{NCR}}(G) \subseteq \mathcal{N}_{\mathrm{ENCR}}(G)$, $\mathcal{N}_{\mathrm{NR}}(G) \subseteq \mathcal{N}_{\mathrm{CR}}(G)$, $\mathcal{N}_{\mathrm{CR}}(G) \subseteq \mathcal{N}_{\mathrm{AR}}(G)$, $\mathcal{N}_{\mathrm{NCR}}(G) \subseteq \mathcal{N}_{\mathrm{AR}}(G)$ and $\mathcal{N}_{\mathrm{ENR}}(G) \subseteq \mathcal{N}_{\mathrm{ENCR}}(G)$.

The equality $\{\mathbf{D}(G_i) \,|\, G_i \in \mathcal{N}_{\mathrm{ENR}}\} = \mathcal{IB}(G)$ follows from Lemmas 2.1, 3.2 and Theorem 3.1. By Lemma 2.1 we can reach any member of the equivalence class of G (and no DAG outside this equivalence class) by a sequence of covered arc reversals. Every member of $\mathcal{N}_{\mathrm{ENR}}(G)$ is obtained by adding or removing an arc from one of the DAGs Markov equivalent to G. By Chickering's Theorem 3.1 G precedes every member of $\mathcal{N}_{\mathrm{ENR}}(G)$ under the inclusion order. By Lemma 3.2, under the inclusion order, there are no Bayesian networks *between* G and those from $\mathcal{N}_{\mathrm{ENR}}(G)$, and therefore $\{\mathbf{D}(G_i) \,|\, G_i \in \mathcal{N}_{\mathrm{ENR}}(G)\} \subseteq \mathcal{IB}(G)$.

By Lemma 3.2 every member of $\mathcal{IB}(G)$ has one edge more or one edge less than G. From this fact and Chickering's Theorem 3.1 it follows that $\mathcal{IB}(G) \subseteq \{\mathbf{D}(G_i) \,|\, G_i \in \mathcal{N}_{\mathrm{ENR}}(G)\}$, thus concluding that $\{\mathbf{D}(G_i) \,|\, G_i \in \mathcal{N}_{\mathrm{ENR}}(G)\} = \mathcal{IB}(G)$.

Statement 2. The DAGs in the difference sets $(\mathcal{N}_{\mathrm{NCR}}(G) \backslash \mathcal{N}_{\mathrm{NR}}(G))$, $(\mathcal{N}_{\mathrm{AR}}(G) \backslash \mathcal{N}_{\mathrm{CR}}(G))$ and $(\mathcal{N}_{\mathrm{ENCR}}(G) \backslash \mathcal{N}_{\mathrm{ENR}}(G))$ are created by the reversal of a non-covered arc in G. This statement says that for any given Bayesian network $\mathbf{D}(G)$, if we reverse a non-covered arc of G obtaining a new DAG G', then $\mathbf{D}(G') \notin \mathcal{IB}(G)$. We prove this as follows. If an arc is not covered in G, its reversal either introduces or destroys an immorality in G' (see Definition 2.3) that yields a non-equivalent model (see Lemma 2.1). Because the number of edges remains the same, Lemma 3.1 applies, i.e., $\mathbf{D}^I(G) \not\subseteq \mathbf{D}^I(G')$ and $\mathbf{D}^I(G) \not\supseteq \mathbf{D}^I(G')$, and therefore $\mathbf{D}(G') \notin \mathcal{IB}(G)$.

The difference set $(\mathcal{N}_{\mathrm{CR}}(G) \backslash \mathcal{N}_{\mathrm{NR}}(G))$ contains only DAGs equivalent to G and hence the intersection $\{\mathbf{D}(G_i) \,|\, G_i \in \{\mathcal{N}_{\mathrm{CR}}(G) \backslash \mathcal{N}_{\mathrm{NR}}(G)\}\} \cap \mathcal{IB}(G)$ is the empty set.

Statement 3. Consider the situation described in Figure 3, where there is a DAG labeled as *current*, which we denote here by G, and its NR neighborhood depicted, which we denote here by $\mathcal{N}_{\mathrm{NR}}(G)$. The AR neighborhood of G, denoted by $\mathcal{N}_{\mathrm{AR}}(G)$, consists of those DAGs in $\mathcal{N}_{\mathrm{NR}}(G)$ plus the DAG resulting of reversing the only one edge that G has, and which we call it G'. Clearly, G' is Markov equivalent to G and not to the DAG labeled as *true*, which is part of the inclusion boundary of G. Therefore, we have found a DAG G with three vertices for which $\{\mathbf{D}(G_i) \,|\, G_i \in \mathcal{N}_{\mathrm{AR}}(G)\} \not\supseteq \mathcal{IB}(G)$. For any larger number of vertices, consider the previous example on three vertices, adding as many disconnected vertices as it is required. ■

The previous theorem proves two important facts: first and foremost, the set of Bayesian networks determined by the ENR neighborhood coincides with the inclusion boundary, while the set of Bayesian networks determined by any of the NR, AR, CR or NCR neighborhoods, neither coincides with it nor covers it. Second, non-covered edge reversal always leads to a Bayesian network out of the inclusion boundary.

As it follows from its definition (and formally stated in Theorem 3.2), the ENCR neighborhood contains the ENR neighborhood plus other models that are not part of the inclusion boundary. The ENR neighborhood already satisfies the inclusion boundary condition; however we shall see in our experiments that a particular implementation of the ENCR neighborhood is useful as well.

To conclude this section, we prove that the inclusion boundary condition (see Definition 3.2) is sufficient to avoid local maxima in structure learning of Bayesian networks under the following two assumptions:

1. The dataset D, on which we perform structure learning, is an independent and identically distributed, fully observed sample from a probability distribution P which is *faithful* to some DAG G. A probability distribution P is *faithful* (Spirtes et al., 1993) to a DAG G if *all and only* the CI restrictions in P make P Markov over G.[4] A Bayesian network $\mathbf{D}(G)$ is called the *true* Bayesian network, with respect to a probability distribution P, or a dataset D sampled from P, when P is faithful to G. The Bayesian network $\mathbf{D}(G)$ can be represented by any DAG Markov equivalent to G. All these DAGs can be called true DAGs, or true Bayesian network structures, too.

2. The number of records in D is unbounded.

From these assumptions it follows that the probability distribution estimated from D converges to P as the number of records in D increases. Under these premises, Chickering (2002b, Lemma 7) proves that a Bayesian scoring metric, such as the BDe (Heckerman et al., 1995), is *locally consistent*. A scoring metric is locally consistent if it:

1. *increases* as the result of adding any edge that eliminates a CI restriction that does not hold in P.

2. *decreases* as the result of adding any edge that does not eliminate a CI restriction that does not hold in P.

The local consistency of a scoring metric reveals that there exists a path in the search space ending in the true Bayesian network, through which the score always increases. In particular, such a path is an inclusion path, as we shall prove in the following theorem.

Theorem 3.3 *Let D^∞ be a dataset of unbounded length sampled from a probability distribution P which is faithful to some DAG G^* that determines a Bayesian network $\mathbf{D}(G^*)$. Let $\mathrm{sc}(G_i; D^\infty)$ be a locally consistent, and score equivalent, scoring metric. Let $G_1, \ldots, G_n$ be a sequence of DAGs such such that $\mathbf{D}(G_i) \neq \mathbf{D}(G_j)$ for $i \neq j$ and $\mathbf{D}(G_n) = \mathbf{D}(G^*)$.*

4. One also says that G is a perfect map of P (Pearl, 1988).

If $\mathbf{D}(G_1), \ldots, \mathbf{D}(G_n)$ *form an inclusion path* $\mathbf{D}^I(G_1) \supset \mathbf{D}^I(G_2) \supset \ldots \supset \mathbf{D}^I(G_n)$*, or* $\mathbf{D}^I(G_1) \subset \mathbf{D}^I(G_2) \subset \ldots \subset \mathbf{D}^I(G_n)$*, then*

$$\mathrm{sc}(G_1; D^\infty) < \mathrm{sc}(G_2; D^\infty) < \ldots < \mathrm{sc}(G_n; D^\infty).$$

Proof If $\mathbf{D}^I(G_i) \supset \mathbf{D}^I(G_{i+1}), i = 1, \ldots, n-1$, then by Chickering's Theorem 3.1, the sequence $G_1, \ldots, G_n$ can be constructed by applying a sequence of additions and covered edge reversals starting on G_1. Since $\mathrm{sc}(G_i; D^\infty)$ is score equivalent the score will not change after a covered edge reversal. Finally, because $\mathrm{sc}(G; D)$ is locally consistent and $\mathbf{D}^I(G_i) \supset \mathbf{D}^I(G_{i+1}), i = 1, \ldots, n-1$, all the necessary additions to obtain G_{i+1} from G_i will increase the score $\mathrm{sc}(G; D)$ and therefore $\mathrm{sc}(G_i; D^\infty) < \mathrm{sc}(G_{i+1}; D^\infty)$ for $i = 1, \ldots, n-1$.

If $\mathbf{D}^I(G_i) \subset \mathbf{D}^I(G_{i+1}), i = 1, \ldots, n-1$, the previous argument works analogously taking into account that first, by Chickering's Theorem 3.1 we can apply a sequence of removals and covered edge reversals to obtain $G_1, \ldots, G_n$ starting on G_1. Second, from the definition of local consistency of a scoring metric it follows that $\mathrm{sc}(G_i; D^\infty)$ will increase when removing an edge that creates a CI restriction that holds in P, and hence $\mathrm{sc}(G_i; D^\infty) < \mathrm{sc}(G_{i+1}; D^\infty)$ when $\mathbf{D}^I(G_i) \subset \mathbf{D}^I(G_{i+1})$ for $i = 1, \ldots, n-1$. ■

This theorem leads us to our final result in this section.

Theorem 3.4 *Let* D^∞ *be a dataset of unbounded length sampled from a probability distribution* P *which is faithful to some DAG* G^* *that determines a Bayesian network* $\mathbf{D}(G^*)$*. Let* $\mathrm{sc}(G; D^\infty)$ *be a locally consistent, and score equivalent, scoring metric. Let* G *be any given DAG and* $\mathcal{N}(G)$ *its neighborhood created by a traversal operator that satisfies the inclusion boundary condition, i.e.,* $\{\mathbf{D}(G_i) \,|\, G_i \in \mathcal{N}(G)\} \supseteq \mathcal{IB}(G)$*. There exists at least one DAG* $G' \in \mathcal{N}(G)$ *such that* $\mathrm{sc}(G'; D^\infty) > \mathrm{sc}(G; D^\infty)$ *unless* $\mathbf{D}(G) = \mathbf{D}(G^*)$*.*

Proof If $\mathbf{D}^I(G) \supset \mathbf{D}^I(G^*)$, or $\mathbf{D}^I(G) \subset \mathbf{D}^I(G^*)$, it follows immediately from Theorem 3.3 that for some $\mathbf{D}(G') \in \mathcal{IB}(G)$, $\mathrm{sc}(G; D^\infty) < \mathrm{sc}(G'; D^\infty) \leq \mathrm{sc}(G^*; D^\infty)$.

When $\mathbf{D}^I(G) \not\supset \mathbf{D}^I(G^*)$ and $\mathbf{D}^I(G) \not\subset \mathbf{D}^I(G^*)$, consider a $\mathbf{D}(G') \in \mathcal{IB}(G)$ such that $\mathbf{D}^I(G') \subset \mathbf{D}^I(G)$ and $\{\mathbf{D}^I(G) \backslash \mathbf{D}^I(G')\} \notin \mathbf{D}^I(G^*)$. Note that if $\mathbf{D}(G')$ would not exist, then $\mathbf{D}(G)$ and $\mathbf{D}(G^*)$ would be in inclusion and the first case would apply. The transformation from G to G' is removing some CI restriction that does not hold in P. Since the scoring metric is locally consistent, then $\mathrm{sc}(G; D^\infty) < \mathrm{sc}(G'; D^\infty)$. ■

The intuition behind the previous theorem is that there is always some inclusion path that permits traversing the search space towards some Bayesian network that is in inclusion with the true Bayesian network, e.g., the fully connected Bayesian network. In the first path the score will increase because we are removing CI restrictions that do not hold in the true Bayesian network. In the second path, that ends in the true Bayesian network, the score will increase as it is shown in Theorem 3.3.

This result is equivalent to Lemmas 8 and 9 from Chickering (2002b) where the optimality of the GES (Meek, 1997) algorithm for structure learning of Bayesian networks is proved. However, the inclusion boundary condition provides us with a general policy for the design of effective traversal operators for any given class of GMMs.

In fact, the inclusion boundary condition has been implicitly taken into consideration by most of the learning algorithms for undirected and decomposable models (Havránek, 1984, Edwards and Havránek, 1985, Giudici and Green, 1999) and surprisingly ignored by most authors in the context of Bayesian networks.

4. Inclusion-driven structure learning

In this section we describe an efficient implementation of the ENR and ENCR neighborhoods. This implementation is used in the following two subsections to build two new algorithms for structure learning of Bayesian networks.

4.1 The RCARNR and RCARR Neighborhoods

From their definitions, one realizes that the ENR and ENCR neighborhoods are not computationally efficient to handle. More concretely, the effort to enumerate the members of an equivalence class is prohibitive since there is no *cheap* graphical characterization of these members.

However, because the average ratio of DAGs per equivalence class seems to be bounded by some constant (see Observation 2.1), it may suffice to *simulate* somehow the ENR neighborhood. In order to do that, we introduce the *repeated covered arc reversal* algorithm, or RCAR algorithm, that allows us to reach any member of the equivalence class with certain probability. We detail the RCAR algorithm in Figure 5.

```
algorithm g.rcar(int r) is
01    int rr ← rnd(0, r)
02    for i ← 0 to rr do
03        vector ce ← g.covered_edges()
04        int j ← rnd(0, ce.size() − 1)
05        edge e ← ce[j]
06        g.reverse_edge(e)
07    endfor
endalgorithm
```

Figure 5: The RCAR algorithm implemented as a method for an object g that embodies a DAG and implements a method that returns a vector of the covered edges and an another method that reverses a given edge.

The algorithm in Figure 5 takes a positive integer r as parameter and iterates some random number of times between 0 (no iteration) and r. At each iteration, it picks at random a covered edge and reverses it. Lemma 2.1 guarantees that the RCAR algorithm reaches any member of the equivalence class with a positive probability for a *sufficiently large* maximum number r of iterations. The bounded ratio of DAGs per equivalence class suggests that a small number between 4 and 10 should be *sufficiently large*.

Note that when the number of undirected edges in the corresponding EG is less than or equal to the number of iterations of RCAR, then RCAR is able to reach any Bayesian

network in its equivalence class. Gillispie and Perlman (2001) show that the distribution of the sizes of the equivalence classes represented by EGs follows a very particular pattern. In particular, they make the following observation (Gillispie and Perlman, 2001):

> The pattern of the distribution shows that certain sizes appear more frequently than others. In particular, larger compound numbers occur more often than larger prime numbers. This is probably due to separate sets of undirected edges in the EG acting independently to produce class sizes that are products of the sizes of their independent components.

In a nutshell, EGs with many maximal disjoint sets of undirected edges represent equivalence classes with a large number of members. However, the size of these sets of undirected edges is inversely proportional to the number of them. This fact permits RCAR to reach a substantial fraction of the members of those equivalence classes. Moreover, not all the members of an equivalence class are strictly required to reach the whole inclusion boundary. For a particular local transformation, some of them may lead to the same equivalence class within the inclusion boundary. Using the RCAR algorithm, we may define the following two new concepts of neighborhood for Bayesian networks:

- **RCARNR** (RCAR+NR) Perform the RCAR algorithm and then create a NR neighborhood, denoted by $\mathcal{N}_{\mathrm{RCARNR}}(G)$.
- **RCARR** (RCAR+NCR) Perform the RCAR algorithm and then create a NCR neighborhood, denoted by $\mathcal{N}_{\mathrm{RCARR}}(G)$.

The RCARNR neighborhood may be seen as a simulation, or an approximation, of the ENR neighborhood, and analogously between the RCARR and ENCR neighborhoods, as it follows from the next lemma.

Lemma 4.1 *Let* $\mathbf{D}(G)$ *be a Bayesian network. For a* sufficiently large *maximum number* r *of iterations of the RCAR algorithm, if* $G' \in \mathcal{N}_{\mathrm{ENR}}(G)$ *then* $G' \in \mathcal{N}_{\mathrm{RCARNR}}(G)$ *with positive probability, and if* $G' \in \mathcal{N}_{\mathrm{ENCR}}(G)$ *then* $G' \in \mathcal{N}_{\mathrm{RCARR}}(G)$ *with positive probability.*

Proof It follows directly from Lemma 2.1 and the definitions of ENR, ENCR, RCARNR and RCARR neighborhoods. ■

By the previous lemma, a traversal operator creating either the RCARNR, or the RCARR, neighborhoods satisfies the inclusion boundary condition with positive probability.

4.2 Heuristic Search

The usual learning algorithm used in heuristic search consists of a hill-climber that iterates until the scoring metric does not improve. The scoring metric is evaluated typically throughout an NR or an AR neighborhood at each iteration. Such a setup works reasonably well in many domains where the complexity of the interactions between the variables is not very high. However, when applied to complex domains, where the outcome often matches

poorly the domain theory, some algorithms assume that a causal ordering between the variables is known (Cooper and Herskovits, 1992), or they search for a *good* causal ordering that may help in providing later a better result (Bouckaert, 1992, Singh and Valtorta, 1993, Larrañaga et al., 1996, Friedman and Koller, 2000).

However, the causal ordering reduces the already small part of the inclusion boundary that was reachable from the NR and AR neighborhoods. Therefore, errors in the ordering may easily lead to very bad local maxima, as shown by Chickering et al. (1995). Heuristic algorithms that use EG-space (Spirtes and Meek, 1995, Chickering, 1996, 2002a,b) do not assume that any form of causal ordering is known probably because, in general, they can work better with complex domains.

We introduce here a new heuristic algorithm which works in DAG-space and accounts for the inclusion order producing reasonably good results when applied to complex domains, as we shall see in Section 5. This will be achieved by the use of the RCAR algorithm (see Figure 5) that allows us to create the RCARNR and RCARR neighborhoods.

```
algorithm hcmc(int r, bool ncr) returns dag
01  dag g ← emptydag
02  bool local_maximum ← false
03  int trials ← 0
04  while not local_maximum do
05    g.rcar(r)
06    set nh ← g.neighborhood(ncr)
07    dag g' ← g.score_and_pick_best(nh)
08    local_maximum ← (g'.score() < g.score())
09    if not local_maximum then
10      g ← g'
11      trials ← 0
12    else if trials < MAXTRIALS then
13      g.rcar(r)
14      local_maximum ← false
15      trials ← trials + 1
16    endif
17  endwhile
18  return g
endalgorithm
```

Figure 6: Hill-Climber Monte Carlo algorithm

The algorithm we propose is shown in Figure 6. It consists of a usual hill-climber that iterates through lines 4 to 17. It contains two modifications. One, in line 5, is to perform the RCAR algorithm on the current DAG. The other, in line 13, consists of calling again the RCAR algorithm when a local maximum is reached. This last step is performed a maximum number of times (MAXTRIALS). If within this maximum number of times, it has not been possible to escape from that local maximum, then the RCAR algorithm is not called again and the hill-climber will stop iterating.

The first of the two modifications, in line 5, is followed by the creation of an NR or a NCR neighborhood, depending on the truth value of the parameter *ncr*. In this way, a RCARNR or a RCARR neighborhood is employed. Afterwards, in line 7, the method $g.score\_and\_pick\_best(nh)$ scores all members of the neighborhood and returns the one that provides the highest score, which is assigned to g'.

The second of the two modifications, in lines 12 to 16, resembles in a way the *iterative hill-climber* introduced by Chickering et al. (1995), which consists of perturbing randomly the current DAG once a local maximum is reached. This is done in line 13 as well but the perturbation is constrained to a move within the same equivalence class of the current DAG. Due to the random nature of the new steps introduced in the hill-climber, we call this algorithm the Hill-Climber Monte Carlo, or HCMC algorithm. Note that the HCMC algorithm may learn different models through different runs, and this permits trading time for multiple local maxima.

4.3 The Markov Chain Monte Carlo Method

The need to account for the uncertainty of models (Draper, 1995) has led to the development of computational methods that implement the full Bayesian approach to modeling. Recall the Bayes' theorem:

$$p(M|D) = \frac{p(D|M)p(M)}{p(D)}\,, \tag{3}$$

where $p(D)$ is known as the normalizing constant which is computed as follows

$$p(D) = \sum_{M \in \mathcal{M}} p(D|M)p(M)\,. \tag{4}$$

Once we account for the uncertainty of the models, it is possible to compute the posterior distribution of some quantity of interest Δ by averaging over all the models in the following way:

$$p(\Delta|D) = \sum_{M \in \mathcal{M}} p(\Delta|M, D)p(M|D)\,. \tag{5}$$

As we saw in Figure 1, the size of DAG-space prohibits enumerating all the models. Therefore, it is not computationally feasible to carry out the sums in (4) and (5).

The method of Markov Chain Monte Carlo (MCMC hereafter) solves this problem by sampling directly from the posterior distributions $p(M|D)$ and $p(\Delta|D)$, thus performing the summations implicitly. The MCMC method had its origins in a sampling method introduced by Metropolis et al. (1953) within the context of statistical physics. However, it was in the work of Hastings (1970), that this sampling method was generalized for statistical problems, by using the theory of Markov chains, introducing the well-known *Metropolis-Hastings* algorithm.

The Metropolis-Hastings algorithm was adapted for structure learning of GMMs by Madigan and York (1995), who called it the *Markov Chain Monte Carlo Model Composition*, or MC^3 algorithm.

For each Bayesian network $M \equiv \mathbf{D}(G)$, let $\mathcal{N}(G)$ be the set of neighbors of G. Let q be a transition matrix such that for some other Bayesian network $M' \equiv \mathbf{D}(G')$, $q(M \rightarrow M') = 0$ if $G' \notin \mathcal{N}(G)$ and $q(M \rightarrow M') > 0$ if $G' \in \mathcal{N}(G)$. Using the transition matrix q we build a Markov chain $M(t,q)$, $t = 1, 2, \ldots, n$, with state space $\mathcal{M}$.

Let $M(t,q)$ be in state M and let us draw a candidate model M' using $q(M \rightarrow M')$. The proposed model M' is accepted with probability

$$\alpha(M', M) = min\left\{1, \frac{|\mathcal{N}(G)|p(M'|D)}{|\mathcal{N}(G')|p(M|D)}\right\}, \tag{6}$$

where $|\mathcal{N}(G)|$ refers to the cardinality of the set of neighbors of G. If M' is not accepted, $M(t,q)$ remains in state M. The idea behind this is that a Markov chain $M(t,q)$ built in this way has $p(M|D)$ as its equilibrium distribution. This means that, after the chain has run for *enough* time, the draws can be regarded as a sample from the target density $p(M|D)$, and one says that the chain has *converged*.

The *convergence* of the Markov chain to the target density $p(M|D)$ is guaranteed under two mild regularity conditions: irreducibility and aperiodicity. The reader may find a discussion in depth of these conditions in the paper of Smith and Roberts (1993).

Given output $M(t,q) = \{M_{t=1}, M_{t=2}, \ldots, M_{t=n}\}$ of the Markov chain, the regularity conditions allow us to derive the following asymptotic results (Chung, 1967, Hastings, 1970, Smith and Roberts, 1993, Madigan and York, 1995):

$$M_{t=n} \stackrel{n\rightarrow\infty}{\longrightarrow} M \sim p(M|D) \tag{7}$$

$$\frac{1}{n}\sum_{t=1}^{n} f(M(t,q)) \stackrel{n\rightarrow\infty}{\longrightarrow} E(f(M)) \tag{8}$$

These imply that, when the Markov chain $M(t,q)$ converges:

- the draws from the Markov chain mimic a random sample from $p(M|D)$. Therefore, in order to get an estimate of $p(M|D)$, it suffices to account for the frequency of visits of each model M and divide it by the number iterations n,

and

- the average of the realizations of any function of the model, $f(M)$, is an estimator of the expectation of $f(M)$. Therefore, by setting $f(M) = p(\Delta|M, D)$, and Δ as a quantity of interest derived from M, we approximate the sum in (5) (Madigan and York, 1995).

In this setting, the transition matrix q choses randomly the Bayesian network structure typically from NR or AR neighborhoods. The enhancement we introduce here consists of simply modifying q in order to use the RCARNR and RCARR neighborhoods. In Figure 7 we see the pseudocode of the modified MC^3 algorithm, which we will call the *enhanced* MC^3, or eMC^3 algorithm. For later comparison in the experiments, we have tuned the algorithm to work also with CR and NCR neighborhoods.
The algorithm, in Figure 7, requires the specification of the following six parameters:

```
algorithm eMC3(dag init, int n, int(dag) f, int r, bool ncr,
               int burn_in) returns {dbl[],dbl[]}
01  dag g ← init
02  int i ← 0
03  vector dbl p
04  vector dbl d
05  while i < n do
06    g.rcar(r)
07    set nh ← g.neighborhood(ncr)
08    dag g' ←pick_at_random(nh)
09    dbl x ← U(0,1)
10    if x ≤ α(g,g') then g ← g' endif
11    if i > burn_in then
12      p[g] ← p[g] + 1
13      d[f(g)] ← d[f(g)] + 1
14    endif
15    i ← i + 1
16  endwhile
17  for g ← p.first() to p.last() do p[g] ← p[g]/n enddo
18  for i ← 1 to d.size() do d[i] ← d[i]/n enddo
19  return {p,d}
endalgorithm
```

Figure 7: The eMC^3 algorithm.

- *init*: some arbitrary Bayesian network structure from which the Markov chain starts.
- n: number of iterations that the Markov chain will perform.
- f: function that takes a Bayesian network structure as input and gives an integer number as output, which indexes some quantity of interest of the model.
- r: maximum number of iterations for the RCAR algorithm.
- *ncr*: flag set to true when the algorithm should use a RCARR neighborhood, and false for a RCARNR neighborhood.
- *burn_in*: number of iterations we want to discard before the algorithm starts accounting for the frequency of visits to the models.

The algorithm uses two vectors p and d to store the estimated posteriors $p(M|D)$ and $p(\Delta|D)$, which are the results that the algorithm returns.

The Markov chain iterates through lines 5 to 16. On line 6, it performs the RCAR operation (see Figure 5) on the current Bayesian network structure g. From g, an NR or NCR neighborhood is created, depending on the truth value of the parameter *ncr*. In line 8 a Bayesian network structure is picked randomly from this neighborhood. These three lines of code implement the transition matrix q.

In line 9 a random number x is generated from a uniform distribution between 0 and 1. This random number x is used to accept the candidate Bayesian network structure g' with

probability as specified in (6). If the current iteration i is beyond those to be discarded (line 11) then, in lines 12 and 13, the current state of the Markov chain is stored.

Finally, in lines 17 and 18, the averages of the stored quantities p and d are computed and they are returned in line 19.

5. Experimental Comparison

We have performed an experimental comparison of the HCMC and eMC^3 algorithms with respect to the standard hill-climber and MC^3 algorithms using an AR neighborhood. These experiments show empirically that:

1. the HCMC algorithm performs significantly better in terms of the quality (score) of the learned models.
2. the eMC^3 algorithm accelerates the rate of convergence of the Markov chain.
3. the use of "inclusion-friendly" traversal operators is crucial in learning Bayesian networks.
4. the computational overhead introduced in the learning process by the RCAR operation is small, while the improvement in learning is large.

The rest of this section is organized as follows. First we describe the data used in the experiments. In Subsection 5.2 we show the results for heuristic learning, and in Subsection 5.3 we show the results for MCMC learning.

5.1 Synthetic and Real-World data

We have used two kinds of synthetic data. One is the Alarm dataset (Beinlich et al., 1989), which has become a standard benchmark dataset for the assessment of learning algorithms for Bayesian networks on discrete data. The Alarm dataset was sampled from the Bayesian network in Figure 8, which was designed for a monitoring system in the context of intensive care unit ventilator management.

The Alarm dataset, originally employed by Herskovits (1991), contains 20000 records. From this dataset the first 10000 records were used by Cooper and Herskovits (1992) to assess the K2 learning algorithm. We will use here the same dataset of 10000 records used by Cooper and Herskovits (1992). From these 10000 records we have sampled six different datasets of two different sizes: three of 1000 records and three of 5000 records. The experiments reported on 10000 records regard only the single dataset of the first 10000 records. As reported by Cooper and Herskovits (1992), this dataset of 10000 records does not support the arc between vertices 12 and 32 (see Figure 8).

The other kind of synthetic data is employed only to assess heuristic learning and consists of the following datasets. Using a recent method[5] by Ide and Cozman (2002), we have generated a collection of one hundred Bayesian network structures at random. This collection has been sampled from the space of DAGs with 25 vertices, restricting the maximum

5. In particular, we have used the software kindly provided by Jaime S. Ide and Fabio G. Cozman at `http://www.pmr.poli.usp.br/ltd/Software/BNGenerator`.

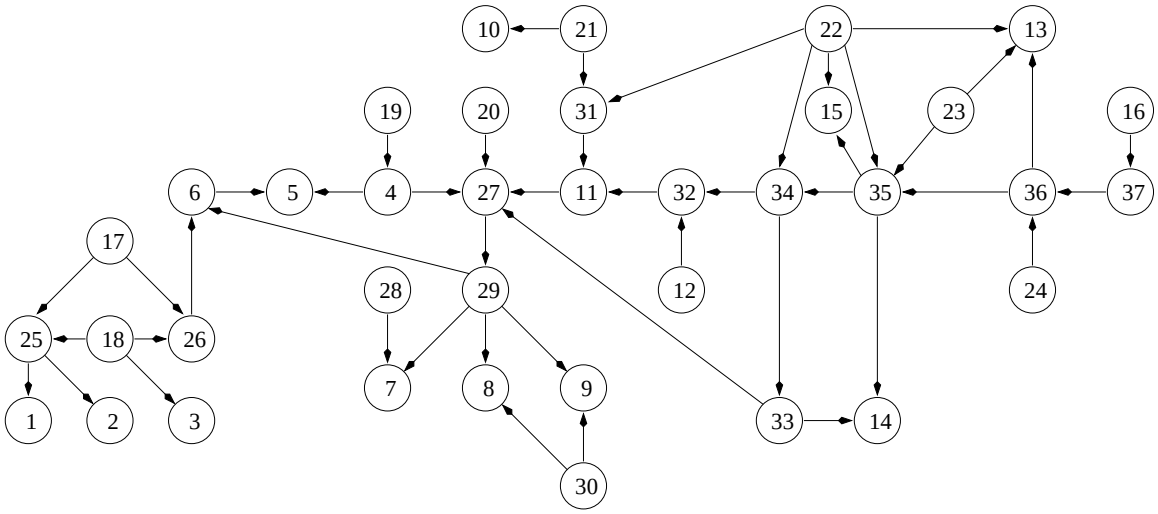


Figure 8: The Alarm network (Beinlich et al., 1989): 37 vertices and 46 edges.

number of parent vertices to 4. This constraint led to an average of 44 edges per network throughout the collection. The method by Ide and Cozman (2002) is able to create an uniformly distributed sample of Bayesian network structures using tools from the theory of Markov chains. For each Bayesian network structure, the corresponding conditional distributions are generated to be uniformly distributed, and we refer the interested reader to the given reference for further details. We have considered a categorical domain with two possible values for all the 25 variables.

From each of those one hundred Bayesian network structures, we have sampled eleven datasets of increasing size: ten of them ranging from 1000 to 10000 records, increasing by 1000, and one of them of 100000 records. Thus in total, we examined 1100 datasets.

As real-world data we have used the following three datasets. Each of them has been pre-processed previously by other authors to ensure that the resulting data is categorical and has no missing values.

- **Anonymous Microsoft Web Test Data**. This dataset, introduced by Breese et al. (1998) and available from the UCI Machine Learning Repository,[6] consists of 5000 instances of anonymous, randomly selected users of the web site `www.microsoft.com` recording which area, out of 294, each user visited in a one-week timeframe in February 1998. Giudici and Castelo (2001) transformed this data, removing all instances of users that visited only one page and classifying all 294 areas into 8 groups. This reduced the degree of sparseness producing a dataset of 3452 records (users) and 8 binary variables. We used this latter dataset.

- **Credit Data**. This dataset, introduced by Fahrmeir (1994) and available from the data repository of the department of Statistics at the University of Munich,[7] consists of 1000 records from credit borrowers of a German bank. For each borrower 21

6. `http://www.ics.uci.edu/~mlearn/MLRepository.html`
7. `http://www.stat.uni-muenchen.de/service/datenarchiv/welcome_e.html`

attributes were originally recorded from which we have discarded 3 in order to have an entire categorical sample. The cardinalities of the domains for the 18 categorical variables range from 2 to 10.

- **Insurance Company Benchmark Data (COIL 2000)**. This dataset, used in the COIL 2000 Challenge and available from the UCI KDD Archive,[8] consists of 86 variables and includes product usage and socio-demographic data on 5822 customers of an insurance company. Castelo et al. (2001) transformed this data obtaining a set of 22 binary variables over the 5822 records. We used this latter dataset.

5.2 Heuristic Learning

5.2.1 Synthetic Data

Concerning the Alarm data, on each of the seven datasets, the HCMC was run ten times for four different cardinalities of RCAR (2, 4, 7 and 10) and three different neighborhoods (AR, RCARR and RCARNR). The standard hill-climber that uses an AR neighborhood will be referred here as RCAR 0.

The reason for running the HCMC several times is obvious since this new hill-climber performs random moves that may lead to different results in different runs. The maximum number of trials for escaping local maxima (MAXTRIALS) was set to 50. The results have been averaged over ten runs and confidence intervals, at a level of 95% for the means of score and structural difference, have been included. We see these results in Table 1.

The information in Table 1 conveys two important types of information, the performance of the algorithm and the accuracy of the learned models with respect of the true one (Figure 8). The performance is provided in terms of number of steps (moves between equivalence classes) that the HCMC performs before stopping and the average time per step. The experiments have been carried out in a Pentium-III processor machine with 512MB of main memory running Linux.

The accuracy of the learned models is measured in terms of the scoring metric, where higher is better, and the structural difference between the EG of the learned DAG and the EG of the true model of Figure 8. The structural difference between two EGs with the same vertex set is computed as the number of adjacencies with different configurations (undirected edge, directed edge, no edge) in the two EGs. In summary, these are the measures taken:

- *steps*: number of steps ($g \leftarrow g'$ in the algorithm) of the HCMC algorithm.
- *sec/st*: speed of the HCMC in seconds per step.
- *score*: confidence interval of the mean of the score.
- *struct diff*: confidence interval of the mean of the structural difference.

As we may appreciate, there is a substantial difference in using RCAR within the hill-climber. Throughout all the seven samples, the structural differences for the standard hill-climber (RCAR 0) fall far outside the confidence intervals for any of the different cardinalities of RCAR, which are centered at much lower values than the values for RCAR 0.

8. `http://kdd.ics.uci.edu`

Table 1: Results (1st part) of the HCMC algorithm over the Alarm dataset. Performance is averaged over RCARR and RCARNR.

smpl	rcar	performance		score		struct diff	
		steps	sec/st	RCARNR	RCARR	RCARNR	RCARR
1ka	0	55	0.27	-11480.47	-11480.47	29	29
	2	58	0.40	-11491.52±15.12	-11470.46±15.79	18.90±3.06	16.50±2.00
	4	55	0.44	-11484.69±19.29	-11473.41±14.18	18.00±3.28	16.30±2.18
	7	54	0.43	-11469.06±07.88	-11470.75±14.67	15.50±1.55	15.60±1.88
	10	53	0.43	-11470.43±15.94	-11464.03±11.00	14.80±2.15	15.20±1.78
1kb	0	60	0.28	-11115.13	-11115.13	28	28
	2	58	0.40	-11113.50±19.07	-11105.10±20.62	18.10±2.77	14.70±3.87
	4	56	0.42	-11121.49±24.64	-11090.15±08.51	17.90±4.41	11.10±2.35
	7	53	0.42	-11095.19±12.38	-11083.13±05.07	13.40±2.55	10.00±1.47
	10	53	0.43	-11095.87±11.25	-11094.17±18.72	12.40±2.05	11.50±2.11
1kc	0	62	0.61	-11530.80	-11530.80	37	37
	2	60	0.41	-11451.58±14.23	-11453.70±15.75	18.20±2.23	15.90±3.10
	4	59	0.43	-11438.31±08.01	-11436.65±07.46	14.90±2.95	13.40±1.94
	7	56	0.67	-11431.02±06.23	-11427.84±03.62	11.80±1.61	10.80±0.81
	10	53	0.86	-11440.88±10.78	-11428.89±08.95	13.70±2.13	11.00±1.17
5ka	0	69	1.54	-55249.43	-55249.43	46	46
	2	66	2.50	-55072.99±67.61	-54993.41±08.70	11.60±6.70	7.20±2.23
	4	57	2.09	-55051.93±40.93	-54992.40±10.92	7.90±2.12	5.20±1.38
	7	56	2.14	-55024.53±48.12	-54989.70±10.12	7.10±2.17	4.90±1.37
	10	56	2.08	-55025.19±43.49	-54985.99±06.68	6.10±1.95	5.10±2.49
5kb	0	57	0.92	-54732.19	-54732	33	33
	2	57	2.02	-54679.46±34.06	-54641.27±60.60	12.50±6.25	6.10±4.08
	4	56	1.35	-54610.82±15.24	-54607.60±14.34	5.20±3.93	3.80±1.38
	7	53	1.29	-54611.85±25.63	-54602.77±10.93	4.50±1.52	3.70±1.31
	10	52	1.28	-54602.98±10.85	-54606.47±12.48	4.00±1.26	4.10±1.32
5kc	0	59	0.88	-54454.16	-54454.16	36	36
	2	63	1.19	-54340.02±16.48	-54335.27±32.25	10.20±3.97	8.00±2.18
	4	59	1.20	-54335.49±19.99	-54326.25±11.15	8.60±1.91	8.40±2.05
	7	55	1.25	-54331.19±12.09	-54315.06±07.33	8.50±1.55	7.70±1.35
	10	55	1.28	-54363.17±52.63	-54329.40±11.13	10.00±2.18	8.50±1.85
10k	0	56	1.86	-108697.78	-108697.78	21	21
	2	56	2.23	-108495.65±68.33	-108463.65±46.17	4.90±2.20	5.40±4.10
	4	54	2.28	-108549.53±63.63	-108437.83±35.72	6.80±2.25	**1.60±0.90**
	7	50	2.29	-108477.50±52.06	-108485.55±58.14	5.50±3.22	**2.80±1.11**
	10	50	2.41	-108468.56±53.07	-108477.98±51.65	4.20±1.34	**3.30±1.17**

Regarding the score on samples of 1000 records, RCAR 7 and 10 show a significantly higher score than RCAR 0 since the latter does not fall into their confidence intervals. At 5000 and 10000 records, the score for RCAR 0 falls outside the intervals for any of the cardinalities of RCAR.

The highest accuracy of the learned models is achieved when a RCARR neighborhood is used. The most striking evidence lies in the case of 10000 records and RCARR. There, an average of only 1.6 structural differences is achieved in about 54 steps. As its confidence interval suggests, on *eight* out of ten of those runs there was only one structural difference, corresponding to the missing arc not supported by the data. In some of those eight runs the result was reached in 49 steps and the same result was reached for RCAR 7 and 10 even in 48 steps, which is extremely close to the optimal path of the right result (46 additions).

In order to gain further insight into the HCMC algorithm and discuss its performance we have taken further measures that we see in Table 2 and we describe as follows:

- *considered models*: accumulated number of models that improve the score at each step of the learning algorithm.
- *escapes/trials*: number of times HCMC succeeded in escaping from a local maximum and average number of trials per escape.

Table 2: Results (2nd part) of the HCMC learning algorithm over the Alarm dataset.

rcar	sample 1ka	sample 1kb	sample 1kc	sample 5ka	sample 5kb	sample 5kc	sample 10k
	considered models (RCARNR)						
0	6505	6460	6760	8307	7418	7560	7774
2	6037±137	5940±80	5943±102	6956±154	6881±171	7165±150	6915±185
4	5856±117	5933±97	5903±128	6849±83	6552±159	6931±160	6865±199
7	5810±71	5800±90	5904±95	6784±98	6414±116	6873±122	6695±111
10	5723±64	5787±59	5823±53	6760±198	6305±115	6846±45	6682±140
	escapes/trials (RCARNR;RCARR)						
2	2.4/7.7;2.3/6.8	1.7/5.1;2.2/11.4	5.4/8.7;2.5/2.0	4.7/9.1;6.1/6.8	4.2/7.1;2.1/15.1	4.7/6.5;3.7/7.7	2.8/14.3;2.2/8.9
4	1.6/6.6;2.0/3.8	2.0/10.5;1.6/6.2	3.7/4.7;3.5/6.1	1.3/7.9;1.8/8.0	2.7/6.2;2.5/7.7	1.7/5.1;2.4/3.4	1.3/5.6;2.6/12.6
7	1.6/4.5;1.7/2.9	0.2/4.0;0.7/6.0	1.8/3.0;1.40/3.1	1.6/10.4;1.5/5.9	1.6/7.6;1.5/7.3	1.3/7.9;0.7/7.2	0.8/4.8;1.3/7.9
10	1.2/3.0;1.7/3.5	0.4/1.3;0.6/16.3	1.7/3.5;1.9/2.0	0.9/7.2;1.0/4.0	0.8/7.7;1.3/6.0	1.2/3.6;0.6/2.3	1.0/6.6;1.2/7.2

As we did in Table 1 with the scores, confidence intervals at a level of 95% are provided in Table 2 for the number of considered models which are shown only for RCARNR as they were not significantly different from RCARR. Although all of them are quite wide, they show that the HCMC algorithm considers significantly fewer models to achieve a better result. This means that the HCMC algorithm makes better choices during the search process which, given its greedy nature, implies that the way HCMC traverses the search space is definitely better.

The number of escapes and trials provide an idea of how effective the HCMC algorithm is in avoiding local maxima. We observe that a higher cardinality of RCAR implies a lower number of times that the HCMC escapes from a local maximum achieving even better results (see Table 1). This means that larger values of RCAR permit traversing the search space better. However, the escaping mechanism is still useful for large values of RCAR since we observe values larger than 0 for all averages.

Finally, let us highlight the computational trade-off that this approach affords. We can assess the overhead of using the RCAR operation within the hill-climber by comparing the seconds per step in RCAR 0 with respect to any other cardinality of RCAR. This information is in Table 1, and we see that the HCMC algorithm is, at most, *two times* slower than the usual hill-climber.

Concerning the data generated from the collection of random Bayesian network structures we have proceeded as follows. We run, for each dataset, the standard hill-climber with an AR neighborhood (HC-AR) and the HCMC algorithm using a RCARR neighborhood with a cardinality of 10 for the RCAR operation. The HCMC algorithm itself was run ten times for each dataset and we picked the learned model that maximized the score. For both learned models, from HC-AR and HCMC, we computed the structural difference of the corresponding EGs with respect to the EG of the randomly generated, true Bayesian network structure from which the dataset was sampled.

In Figure 9 we have plotted the structural differences as a function of the sample size, averaging them over the one hundred Bayesian network structures. In the same figure, we

have included vertical bars indicating the proportion of learned Bayesian network structures that were Markov equivalent to the corresponding true one, i.e., with null structural differences. We observed that the HC-AR algorithm was not able to learn the true Bayesian network from any of the 1100 datasets, and that is the reason why the vertical bars concern only the HCMC algorithm.

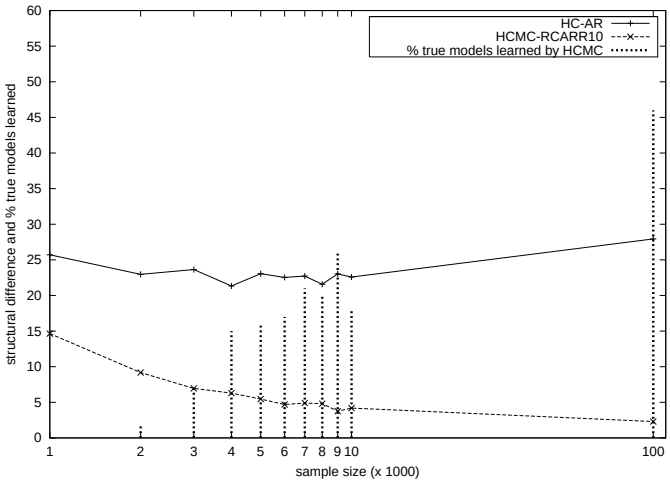


Figure 9: Average structural difference, and proportion of null structural differences, for the Bayesian network structures learned from datasets of increasing size sampled from a set of 100 randomly generated Bayesian network structures with an average of 44 edges.

Note that the average Bayesian network structure from this collection has a rather high degree of complexity if we consider that the Alarm network from Figure 8 had a similar number of edges but twelve more variables.

The results in Figure 9 allow us to see that the proportion of true Bayesian networks correctly learned increases, and the average structural difference decreases, as the available data becomes larger. This fact confirms empirically the expected asymptotic benefit of using a traversal operator that satisfies the inclusion boundary condition.

Observe further, that HC-AR does not show a trend towards a lower average structural difference as the sample size increases, which emphasizes the crucial role of an inclusion-driven approach in structure learning.

The earlier work by Chickering (1996) and Spirtes and Meek (1995) using EGs reported the ability to recover the true Alarm network from the Alarm dataset, but it made an intensive use of the transformation from DAG to EG which becomes a severe computational burden for the learning algorithm. The recent work by Chickering (2002a) improves the computational performance of the EG-based approach by providing a set of efficient traversal operators, but it does not follow any inclusion-driven strategy.

The very recent work by Chickering (2002b) provides an inclusion-driven strategy for the EG-based approach of Chickering (2002a), achieving similar results to the ones presented in this section.

5.2.2 Real-World Data

We have assessed the HCMC algorithm against real-world data by computing the *predictive performance*, i.e., measuring how well the learned models predict future observations. Each of the three datasets has been randomly split five times in two subsets. One subset D_L containing 50% of the data was used to learn the Bayesian network while the other subset D_T was used to compute the predictive performance. As Madigan et al. (1996, Section 4.4), we have computed the predictive performance as the average predictive probability on D_T:

$$\text{pperf} = \frac{1}{|\mathrm{D_T}|} \sum_{\mathrm{r} \in \mathrm{D_T}} \mathrm{p}(\mathrm{r}|\mathrm{M}, \mathrm{D_L}), \tag{9}$$

where $M \equiv \mathbf{D}(G)$ is the Bayesian network learned from D_L. The intuition behind assessing the predictive performance is that better Bayesian networks should assign higher probabilities to the records in D_T.

As with synthetic data, we have set MAXTRIALS to 50, but for each sample we have run the HCMC algorithm 20 times. We have used both RCARNR and RCARR neighborhoods with a value of 10 for the RCAR operation. From the set of 20 learned models, we have chosen the one with the highest score, and use it to compute the predictive performance. This is, when applied to real-world data, a nice feature of HCMC because when we are confronted with data from which we have little knowledge *a priori*, we may want to compare different alternatives and pick the model that suits best our criteria as, for instance, the score in this case.

Table 3: Results of the HCMC learning algorithm over three real-world datasets.

HC–AR			HCMC–RCARNR10					HCMC–RCARR10				
#e	score	pperf	#e	score	pperf	sd	%gain	#e	score	pperf	sd	%gain
Web Data												
13	-6885.69	4.723503e-02	9	-6876.80	4.846662e-02	10	2.61	9	-6876.80	4.846662e-02	10	2.61
9	-6790.48	4.921026e-02	8	-6788.07	4.831944e-02	3	-1.81	8	-6788.07	4.831944e-02	3	-1.81
10	-6807.02	4.854368e-02	10	-6795.01	4.890523e-02	6	0.74	10	-6795.01	4.890523e-02	6	0.74
9	-6880.69	4.779160e-02	10	-6873.73	4.734724e-02	11	-0.93	9	-6874.58	4.778755e-02	10	-0.01
10	-6835.94	4.776003e-02	12	-6825.34	4.944954e-02	10	3.54	12	-6824.33	4.865120e-02	9	1.87
Credit Data												
14	-8562.06	8.523136e-07	14	-8562.06	8.523136e-07	0	0.00	14	-8562.06	8.523136e-07	0	0.00
11	-8522.61	1.022170e-06	13	-8514.13	1.067448e-06	2	4.43	13	-8514.13	1.067448e-06	2	4.43
10	-8480.03	1.205717e-06	10	-8480.03	1.205717e-06	0	0.00	10	-8480.03	1.205717e-06	0	0.00
14	-8499.15	1.084386e-06	15	-8494.85	1.124413e-06	1	3.69	15	-8494.85	1.124413e-06	1	3.69
11	-8496.36	9.574894e-07	13	-8481.30	1.017876e-06	4	6.31	13	-8481.30	1.017876e-06	4	6.31
Insurance Data												
41	-9491.43	8.924253e-02	41	-9481.39	8.976264e-02	10	0.56	44	-9482.30	8.989139e-02	17	0.73
37	-9646.09	8.828531e-02	35	-9625.94	8.927191e-02	15	1.12	36	-9623.33	8.920739e-02	17	1.04
39	-9835.66	8.861103e-02	39	-9831.05	8.945385e-02	17	0.95	39	-9828.98	8.920012e-02	15	0.66
38	-9729.18	8.877596e-02	38	-9723.81	8.906750e-02	6	0.32	40	-9726.03	8.879126e-02	12	0.02
36	-9641.69	8.891946e-02	33	-9636.72	8.914069e-02	21	0.25	37	-9636.09	8.920335e-02	23	0.32

In Table 3 we see the results, which are organized as follows:

- *#e*: number of edges.
- *score*: BDeu score.

- *pperf*: predictive performance computed by expression (9).

- *sd*: structural difference between the EGs equivalent to the Bayesian networks learned with the standard hill-climber, and the HCMC algorithm, respectively.

- *%gain*: proportion of gain in predictive probability by using the HCMC algorithm.

The quantities *#e*, *score* and *pperf* are measured for the standard hill climber with an AR neighborhood, HC-AR hereafter, and the HCMC algorithm. The most conclusive evidence lies on the *%gain* column where we see how larger the predictive performance of HCMC is, with respect to the predictive performance of HC-AR. For instance, for the first sample of the web data, the HCMC algorithm is able to learn a Bayesian network that assigns probabilities, on average, 2.61% higher than the Bayesian network learned with the HC-AR algorithm. This is the case for both RCARNR and RCARR, and in fact, the results are quite similar for both neighborhoods.

The HCMC algorithm learns the same model for two samples of the credit data and learns models with a higher score in the other 13 samples. This, however, does not guarantee a better predictive performance in all of them. We find two samples from the web data for which the model with a worse score, learned by HC-AR, outperforms the model learned by HCMC.

While the Bayesian networks learned with the two algorithms have a similar number of edges in all 15 samples, they are quite different in the case of the web and insurance data, as seen from the structural differences (*sd* column). These two datasets are less sparse than the credit dataset, and this might be the reason why both algorithms end in quite different local maxima.

However, while the Bayesian networks learned in the credit dataset differ much less, the gain in predictive performance, for those that are different, is much larger. This is probably due to the fact that the more sparse the data is, the worse estimates we obtain for the conditional probabilities and, therefore, mistakes in the model have a larger influence.

5.3 MCMC Learning

5.3.1 Synthetic Data

When we reviewed the MC^3 algorithm, we saw that the acceptance ratio (6) is the product of two ratios. One is the ratio of the posteriors $p(M'|D)/p(M|D)$, which nicely cancels the normalizing constant (4) and therefore the data D is involved only through the Bayes' factor $p(D|M')/p(D|M)$. The other is the ratio of the cardinalities of the neighborhoods $|\mathcal{N}(G)|/|\mathcal{N}(G')|$ which is known as the candidate-generating ratio. In our experimentation we have assumed a *symmetric* candidate-generating density (Chib and Greenberg, 1995), where $|\mathcal{N}(G)| = |\mathcal{N}(G')|$. This is reasonable in our context since G and G' will differ in a single adjacency.

The *e*MC^3 algorithm of Figure 7 needs the specification of some Bayesian network as a starting point (the *init* parameter). It is generally agreed within the MCMC literature that the run of the Markov chain is sometimes sensitive to the starting point. Therefore it makes sense to try several runs from several starting points chosen at random.

However, within the context of random generation of acyclic digraphs, Melançon et al. (2000) point out that starting the process from the empty graph gives an effective way of achieving a good mixing rate of the chain. They explain this effect as follows:

> Observe that the maximal distance between any two acyclic digraphs is bounded by $n(n-1)$, since an obvious (but far from optimal) path connecting them goes through the empty graph (by first deleting all edges from the first graph and then adding the edges of the second graph).

We consider that in our context, where the distribution of the DAGs that determine the Bayesian networks is not uniform, it still makes sense to follow this advice because one may reason analogously in terms of GMM inclusion.

In addition, we consider as good starting points those Bayesian networks with a high marginal likelihood, as they will be close to the mode of the distribution and therefore it may accelerate the convergence of the chain. In particular, we will use the output of the HCMC algorithm which was the original Alarm network structure of Figure 8 with one arc missed (the one not supported by the data), and to which we will refer as the *almost true* Alarm network. When such a starting point is used, it will be denoted by an asterisk in the legends, next to the name of the neighborhood.

We ran the $e\text{MC}^3$ algorithm for 10^5 iterations over each of the seven samples of the Alarm dataset, and for the 10000-record sample we additionally ran the chain starting from the almost true Alarm network. We do not provide all the results for every sample, as for some combinations the conclusions are the same.

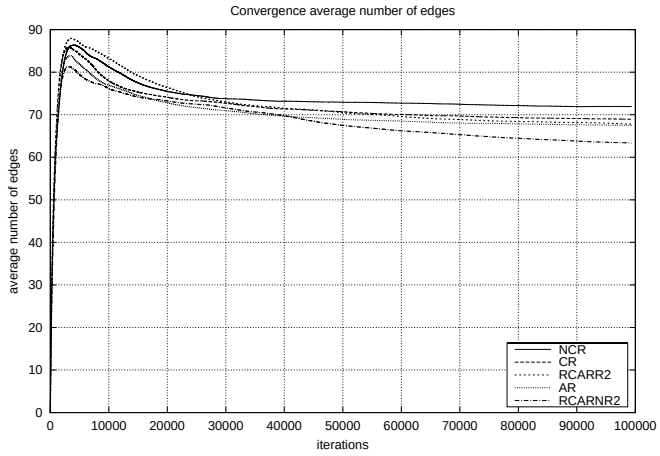
We assess the convergence rates of the $e\text{MC}^3$ and MC^3 algorithms by using several convergence diagnostics introduced by Giudici and Castelo (2003). The first convergence diagnostic is the behavior of the running average number of edges during the run. The rationale behind monitoring this average is that those Bayesian networks with higher posterior will be sampled more often, and therefore the average number of edges of these networks should be approached by the running average number of edges. If this running average shows some slope at some point of assessment, it is most likely that the Markov chain has not converged at that point.

We monitored this diagnostic starting from the empty Bayesian network structure using the standard neighborhoods AR, CR and NCR, and the newly introduced RCARNR and RCARR. In the case of these latter two, we ran the experiments with three different cardinalities for the RCAR algorithm (see Figure 5), namely 2,4 and 10.

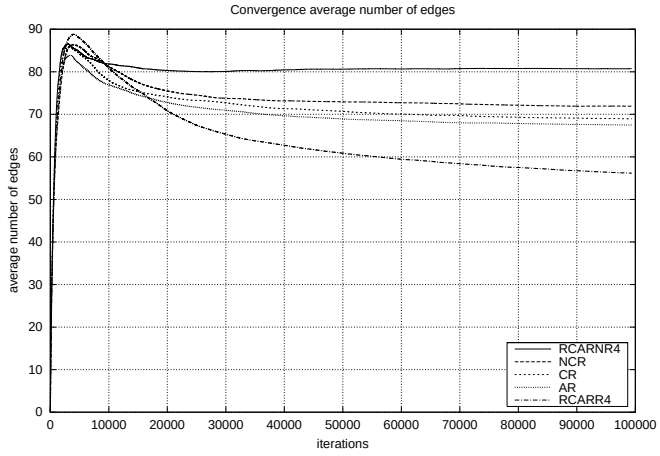
We plotted the AR, CR and NCR neighborhoods against RCARNR and RCARR, for each of the three cardinalities, 2, 4 and 10. We see these plots in Figure 10. They correspond to the runs using the 10000-record dataset.

Recall that the true Alarm network has 46 edges (see Figure 8). Therefore we should expect that the average number of edges converges towards that number. We ran the Markov chain for 10^5 iterations and we see in all three plots that all lines still have some slope downwards at the last iteration. According to this convergence diagnostic, it implies that the chain has not converged. However, it is possible to observe which approach affords a faster convergence.

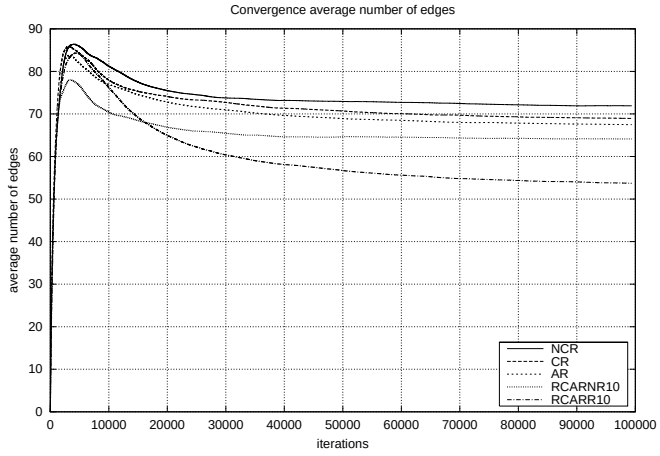
In all three plots, faster convergence is achieved by either RCARR, or RCARNR, neighborhoods. We may appreciate that the larger the cardinality of RCAR is, the larger the



(a)



(b)



(c)

Figure 10: Convergence of the average number of edges for the 10k dataset comparing RCARR and RCARNR neighborhoods with different RCAR cardinalities: 2 (a), 4 (b) and 10 (c). Legends are ordered with lines.

difference between the use of RCAR and any of the other *non-RCAR* neighborhoods. In the particular case of cardinality 4, although RCARR4 clearly outperforms the rest, RCARNR4 shows the slowest convergence. This may be due to a sequence of jumps that led the Markov chain into an area of local maxima from which escape is very improbable.

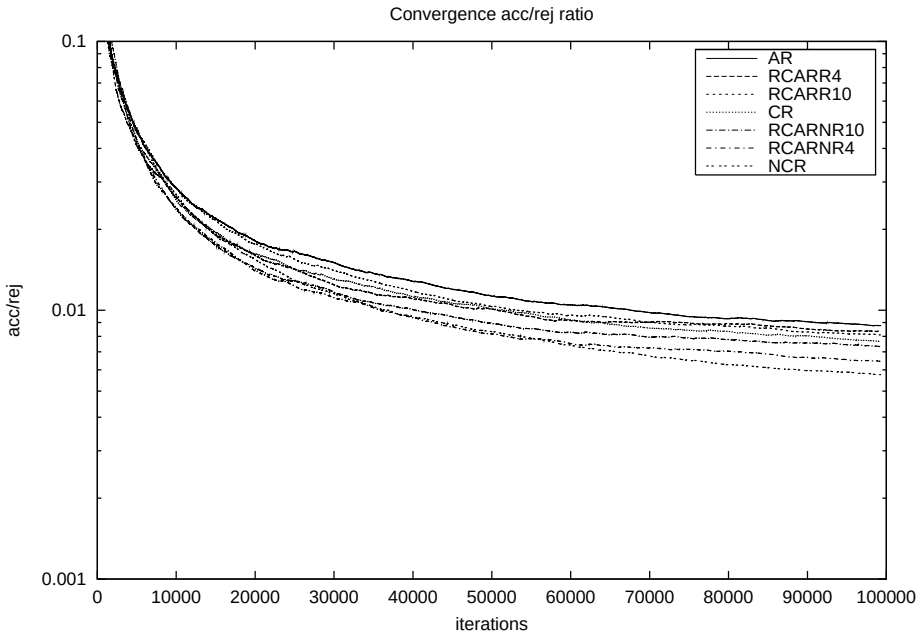


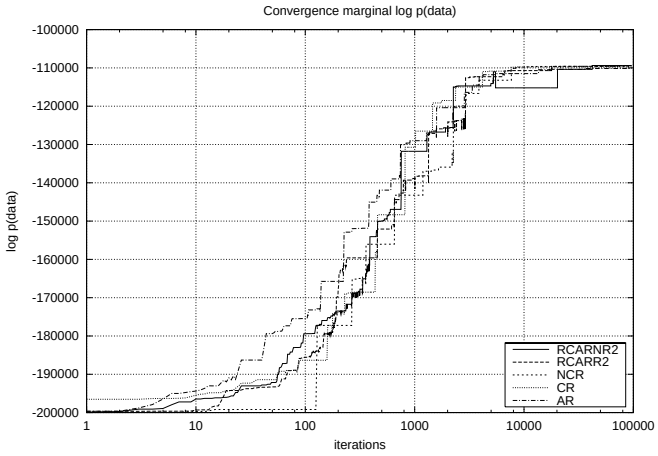
Figure 11: Convergence of the accepts/rejects ratio for the 10k dataset. Legends are ordered with lines.

We observe this effect examining the next convergence diagnostic, the ratio of accepts over rejects, in Figure 11. The RCARNR4 has one of the lowest ratios, only larger than NCR. We did not include RCARR2 and RCARNR2 but they are larger than RCARNR4. Note from this convergence diagnostic that the lines still have some slope, so we can also conclude from this diagnostic that the chain did not converge.

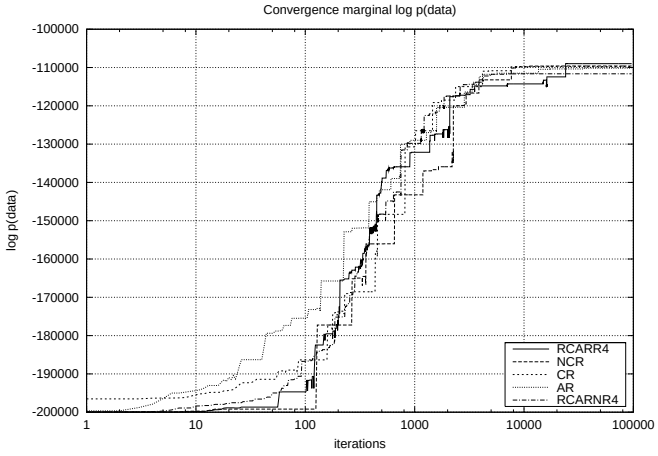
The third convergence diagnostic consists of monitoring the approximated marginal log-likelihood of the data $\log p(D)$. After swapping terms in Bayes' theorem (3) we obtain an explicit expression of this marginal:

$$p(D) = \frac{p(D|M)p(M)}{p(M|D)}\,.$$

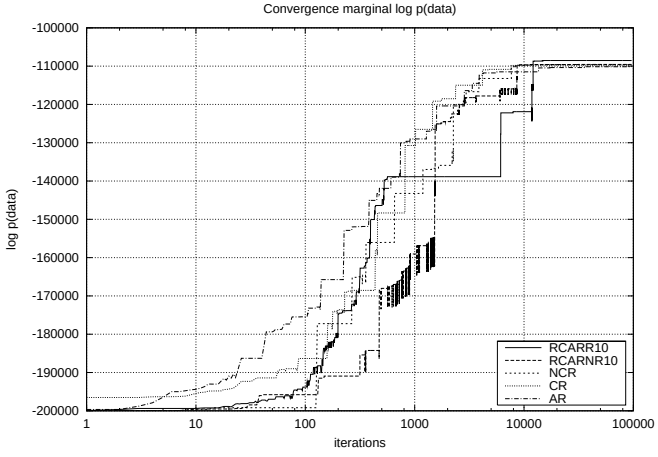
Note that this equality holds for any given Bayesian network M. Obviously, when the posterior $p(M|D)$ is approximated by MCMC, only an approximate marginal likelihood $\hat{p}(D)$ can be obtained. Such an approximation will be better for models within an area of high probability in the posterior distribution. Furthermore, as Kass and Raftery (1995) point out, small likelihoods may have large effects on the final approximation and make the resulting estimator $\hat{p}(D)$ very unstable. This leads us to compute the approximate marginal



(a)

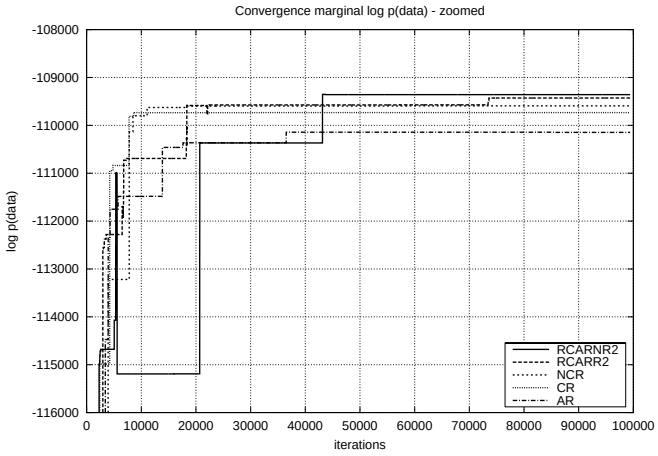


(b)

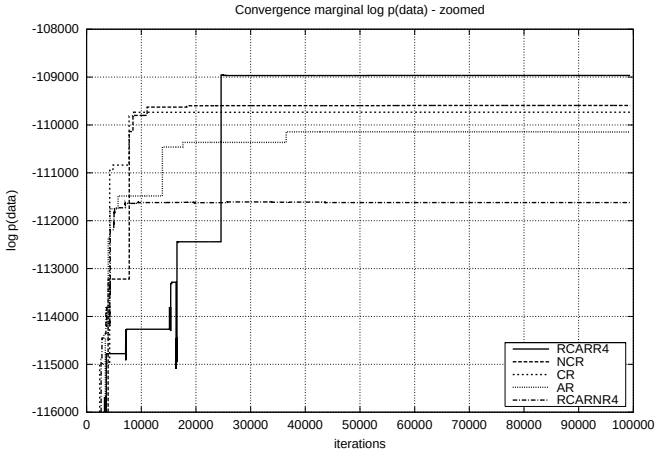


(c)

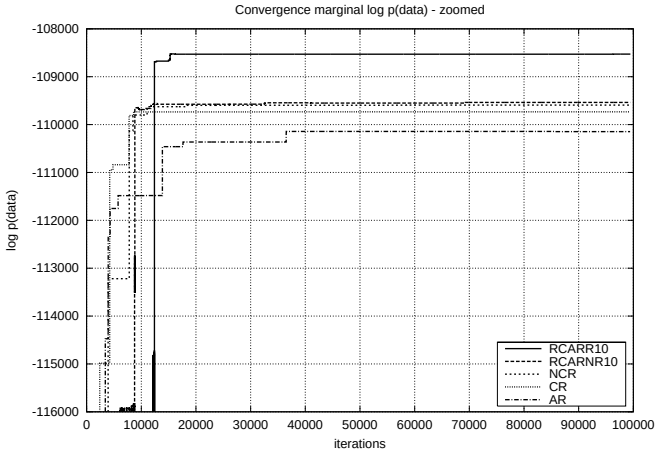
Figure 12: Convergence of the marginal of the data (entire marginal range). Comparison between AR, CR, NCR, RCARNR and RCARR for RCAR cardinalities of 2 (a), 4 (b) and 10 (c). Legend is ordered with lines.



(a)



(b)



(c)

Figure 13: Convergence of the marginal of the data (zoomed marginal range). Comparison between AR, CR, NCR, RCARNR and RCARR for RCAR cardinalities of 2 (a), 4 (b) and 10 (c). Legend is ordered with lines.

likelihood as an average of the approximations from the models with highest posteriors, as follows.

Let $\hat{p}(M|D)$ be the current estimated posterior for model M give data D. Let $p(D|M)$ be the current likelihood of the model M. The marginal likelihood $\hat{p}(D)$ can be estimated as:

$$\hat{p}(D) = \frac{1}{|\mathcal{B}|} \sum_{M \in \mathcal{B}} \frac{p(D|M)p(M)}{\hat{p}(M|D)}, \tag{10}$$

where $\mathcal{B}$ is a set formed by the Bayesian networks M with highest posterior at each iteration of the eMC^3 algorithm. In our experiments below, we have chosen five Bayesian networks to form $\mathcal{B}$.

We see this convergence diagnostic in Figure 12 for the 10000-record sample starting from the empty Bayesian network. In Figure 13 we find the same plots with the marginal range zoomed for the higher values. Here a larger $\log p(D)$ indicates that the Markov chain moves in an area of a higher posterior, hence providing a faster convergence. Again RCARR and RCARNR outperform CR, NCR and AR neighborhoods. We observe for the cardinality 4 of RCAR that the slow convergence shown in the previous plot 10(b) is in agreement with a low $\log p(D)$.

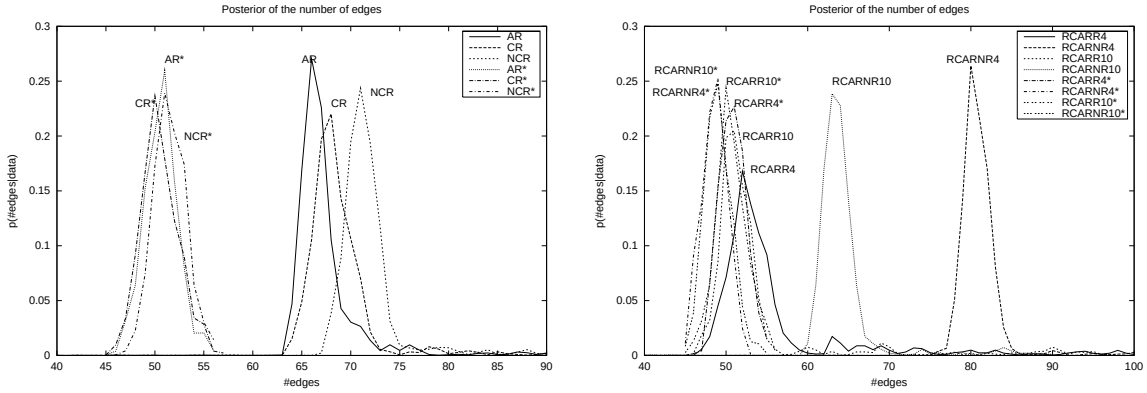
Another interesting fact emerges when examining the slope of the curve for the AR neighborhood when we look at the entire length of the Markov chain (Figure 12). We notice that using the AR neighborhood, $p(D)$ increases faster than any of the others. This means that the AR neighborhood allows the algorithm to perform larger steps in the search space, but then later seems to get stuck in worse local maxima than using RCARR or RCARNR neighborhoods.

Finally, the last convergence diagnostic corresponds to the posterior distribution of the total number of edges present. More formally, let n be the number of vertices of the DAG that determines the Bayesian network. Let W be the random variable that takes as its value the number of edges of the DAG at each iteration of the Markov chain. The integer random variable W will take values in the range $[0, 1, 2, \ldots, n(n-1)/2)]$, which are all possible cardinalities of the set of edges.

The quantity to monitor is $p(W|D)$, which is computed as in the general case of any quantity of interest Δ (see expression (5)). This posterior distribution has a normal shape and it is centered close to the cardinality of the model for which the Markov chain gives the highest posterior. If the center of the normal shape shifts through longer runs, it means that the Markov chain has not converged.

In Figure 14 we show the distribution over the number of edges. In this case, we have started the Markov chain from both the empty Bayesian network structure and the almost true Alarm network, which has been denoted by an asterisk next to the name of the corresponding neighborhood. In plot 14(a) we see the runs for the AR, CR and NCR neighborhoods. Clearly, those that started at the almost true Alarm network show a faster rate of convergence than those that did not.

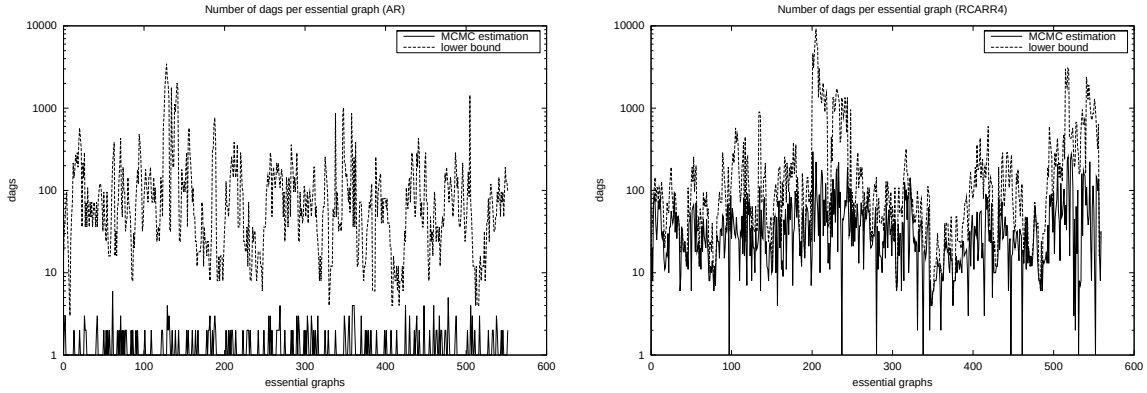
In plot 14(b) we see the runs for cardinalities 4 and 10 of the RCARNR and RCARR neighborhoods. In contrast with the previous case, here two of them, namely RCARR4 and RCARR10, are able to provide distributions quite similar to those of the ones that started



(a) (b)

Figure 14: Comparison convergence ability for the AR, CR and NCR neighborhoods (a) and the RCARNR and RCARR neighborhoods (b).

in the almost true alarm network. So far, from all these convergence diagnostics, we can conclude that the RCAR operation improves the convergence rate of the MC^3 algorithm.



(a) (b)

Figure 15: DAGs per EG using the AR (a) and the RCARR4 (b) neighborhoods.

A further interesting outcome of our experimentation arises from looking at the number of members of each equivalence class of Bayesian network structures visited by the Markov chain during the 10^5 iterations. For comparison, we have computed a lower bound on the size of each equivalence class as follows.

Let $\mathbf{D}(G)$ be a Bayesian network. Let $\mathbf{E}(G^*)$ be its equivalent EG Markov model, i.e., $\mathbf{D}(G) = \mathbf{E}(G^*)$, where G^* is the EG representation Markov equivalent to G. Let G^* have m connected components, where each of them has ρ_i reversible edges. The lower bound on the number of members of the equivalence class represented by G^* is

$$\prod_{i=1}^{m} (\rho_i + 1)\,.$$

In Figure 15 we have the plots of these numbers for the runs that started on the almost true Alarm network over the 10000-record sample for the AR and RCARR4 neighborhoods. We have not included numbers from CR or NCR as they were not significantly different from AR.

We see that RCARR4, Figure 15(b), can estimate much better the number of members of each equivalence class than AR, Figure 15(a). This provides empirical evidence in support of Theorem 3.2, which explains why the RCAR operation enhances both heuristic search and the MCMC method.

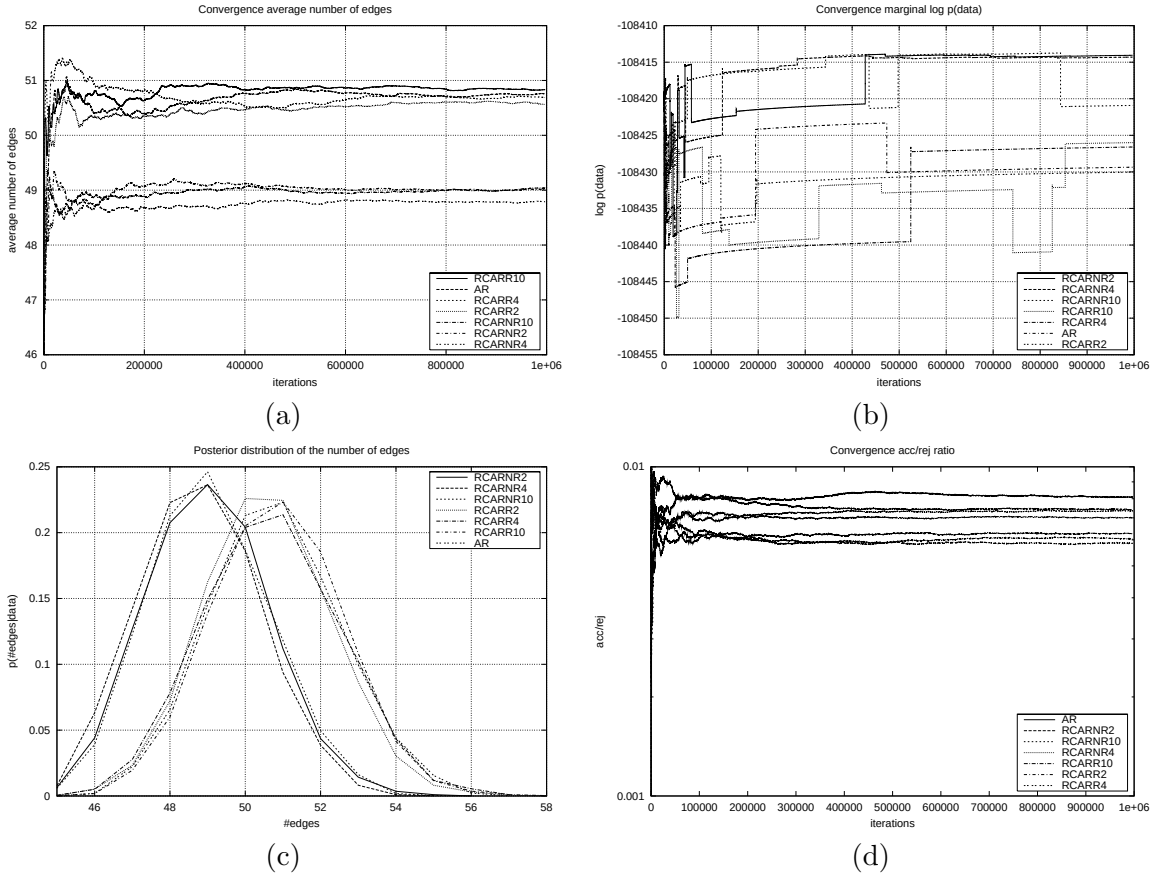


Figure 16: Convergence diagnostics for a Markov chain of length 10^6 iterations, starting from the almost true Alarm network: average number of edges (a), marginal of the data (b), distribution of the cardinalities of the sets of edges (c) and ratio of accepts over rejects (d). Legends are ordered with lines.

The length of Markov chain we have used (10^5 iterations) is long enough to clearly distinguish among different rates of convergence. However, all the diagnostics show lack of convergence of the chain. We ran a longer chain for 10^6 iterations starting from the almost true Alarm network. We see the convergence diagnostics in Figure 16.

In this case we do not include the CR and NCR neighborhoods for comparison. In plot 16(a) we show the average number of edges during the run. In plot 16(b) we show the convergence of the marginal of the data. In plot 16(c) we show the posterior distribution of the cardinalities of the corresponding sets of edges. In all three diagnostics we distinguish two groups that converge at a different rate; one formed by AR, RCARR10, RCARR4 and RCARR2, and another by RCARNR10, RCARNR2 and RCARNR4, the latter being the one that shows better convergence.

In a way, this is surprising because, as we had seen so far, RCARR outperforms RCARNR. Nevertheless, this behavior is consistent with the fact the ENR neighborhood represents all the models in the inclusion boundary, as shown in Theorem 3.2, and that the RCARNR neighborhood simulates it (see Lemma 4.1). In fact, we have no reason to believe that any of the other neighborhoods would improve the RCARNR neighborhood. We can see empirically that when starting from the empty Alarm network, the RCARR neighborhood converges faster.

We may conclude that when the Markov chain is close to the model with largest marginal likelihood, the non-covered reversal operation in the RCARR neighborhood leads the chain to local maxima that slows down the convergence. We see in plot 16(d) that the RCARR neighborhoods have a lower ratio of accepts over rejects, demonstrating precisely this effect due to the very low probability of escaping from where the chain is. This might be happening because the non-covered reversal makes the chain moving always to a model out of the inclusion boundary (see Theorem 3.2). Probably a better strategy would combine, during the run of the Markov chain, both the RCARR and RCARNR neighborhoods.

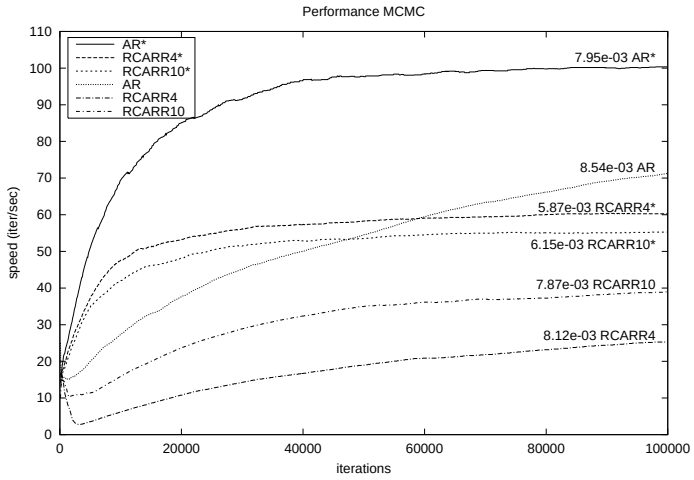


Figure 17: Comparison of the computational performance between using the AR neighborhood, and using the RCARR4 and RCARR10 neighborhoods.

Finally, we will take a look at the computational overhead of the RCAR operation within the MC^3 algorithm. In Figure 17 we have plotted the average number of iterations (lines 5 through 16 in Figure 7) per second. This plot corresponds to the runs of length 10^5 iterations. In addition to the legend we have labeled the lines with their corresponding

neighborhood and also included the total average ratio of accepts and rejects. A lower ratio speeds up the chain as it is making, on average, less moves.

For clarity we only report comparison between AR, RCARR4 and RCARR10, as the differences are similar for other combinations. Analyzing separately those runs that start from the empty Bayesian network structure from those that start from the almost true Alarm network, we see that in both cases, using the RCAR operation is between two and three times slower than not using it. This cost is similar to the one we observed for heuristic search, and we consider it to be a very good trade off.

5.3.2 Real-World Data

To assess the eMC^3 algorithm with real-world data we compute the predictive performance by Bayesian model averaging (BMA) as follows,

$$\text{pperf} = \frac{1}{|\mathrm{D_T}|} \sum_{\mathrm{r} \in \mathrm{D_T}} \sum_{\mathrm{M} \in \mathcal{B}} \mathrm{p}(\mathrm{r}|\mathrm{M}, \mathrm{D_L}) \mathrm{p}(\mathrm{M}|\mathrm{D_L}) \,. \tag{11}$$

Expression (11) is equivalent to (9) but the predictive probabilities are averaged across the models in a set $\mathcal{B}$. The MCMC method allows us to approximate the posterior probability of the models given a dataset and this permits weighting the competing models in a sensible way. Since we are using a score equivalent metric, after each run of the Markov chain we have summed up the posterior probabilities of equivalent Bayesian network structures in order to obtain a posterior distribution of EGs. Therefore, when we refer to the posterior of a model we are in fact referring to the posterior of the model represented by the corresponding EG.

In our experiments we have used a set $\mathcal{B}$ with the three models with highest posterior. For all datasets we have re-normalized the probabilities of the three models, as they were not accounting for the whole posterior distribution.

Table 4: Results of the eMC^3 learning algorithm over three real-world datasets.

MC^3–AR		eMC^3–RCARNR10				eMC^3–RCARR10			
score(m)	score(avg)	score(m)	score(avg)	cm	%gain	score(m)	score(avg)	cm	%gain
Web Data									
-6872.08	-6872.11+-3.95	-6871.42	-6872.18+-2.04	1	0.95	-6870.53	-6870.85+-2.93	1	-0.43
-6785.33	-6786.27+-3.61	-6785.33	-6787.11+-3.83	1	-0.47	-6787.94	-6787.07+-3.32	2	0.61
-6789.18	-6789.67+-1.43	-6789.18	-6789.04+-1.43	2	0.48	-6789.18	-6789.67+-1.43	3	0.01
-6873.73	-6875.55+-4.46	-6873.73	-6874.41+-1.52	1	-0.44	-6873.73	-6875.36+-4.52	2	-0.86
-6824.40	-6825.19+-1.80	-6824.40	-6825.44+-2.25	2	-0.15	-6826.08	-6824.94+-2.46	1	-1.36
Credit Data									
-8562.45	-8562.54+-1.21	-8562.74	-8562.54+-0.61	0	-1.58	-8562.11	-8562.41+-0.77	1	2.24
-8514.13	-8514.91+-2.00	-8514.13	-8514.70+-1.26	2	0.07	-8514.13	-8515.08+-2.66	2	-0.22
-8480.48	-8480.40+-0.85	-8480.03	-8480.40+-0.85	3	-0.28	-8480.48	-8480.51+-1.23	2	-0.45
-8493.83	-8494.15+-0.69	-8494.35	-8494.61+-0.99	1	-5.12	-8493.83	-8494.75+-1.99	1	0.19
-8481.29	-8481.76+-1.51	-8481.30	-8481.87+-1.58	1	-0.63	-8481.29	-8481.52+-0.59	2	-0.44
Insurance Data									
-9520.93	-9524.35+-7.38	-9506.56	-9508.13+-10.86	0	0.18	-9526.41	-9512.98+-28.87	0	0.18
-9655.85	-9658.81+-10.09	-9647.90	-9653.83+-12.94	0	0.05	-9658.76	-9655.74+-6.87	0	0.20
-9872.57	-9872.44+-8.59	-9849.94	-9862.78+-28.35	0	0.67	-9877.00	-9877.46+-27.64	0	0.02
-9750.62	-9760.32+-24.20	-9750.47	-9745.61+-23.57	0	-0.36	-9752.47	-9755.71+-25.67	0	-0.12
-9685.09	-9684.04+-21.61	-9671.74	-9672.31+-15.69	0	0.32	-9687.91	-9681.60+-29.84	0	0.12

Table 4 shows the predictive performance of BMA for the MC^3 algorithm using a standard AR neighborhood (MC^3-AR), and the eMC^3 algorithm using both RCARNR and RCARR neighborhoods with a value of 10 for the RCAR operation, just as in heuristic search. Each Markov chain in these experiments ran for 10^5 iterations, starting from the Bayesian network learned with its corresponding counterpart in heuristic search (HC-AR for MC^3-AR, and so on), and it had an initial "burn-in" period of 10^4 iterations. The quantities shown in Table 4 are the following:

- *score(m)*: score of the mode of the posterior distribution.
- *score(avg)*: confidence interval at 95% level for the mean of the score of the three models with highest posterior.
- *cm*: number of common models between the highest three from the distributions approximated by MC^3-AR and eMC^3.
- *%gain*: gain in predictive probability of BMA by eMC^3 over MC^3-AR.

In contrast with heuristic search, the results do not show much difference between using MC^3 or eMC^3 with these three real-world datasets. Among all thirty runs, fifteen showed gain due to the use of eMC^3 and the other fifteen did not. Only in four of them was the gain, or the loss, larger than 1%, while for heuristic search this happened in fourteen out of the thirty runs. As we saw in heuristic search, the larger gain, and loss, lies in the most sparse data, the credit data. Among the confidence intervals for the mean of the score only one of them, in the first sample of the insurance data, was significantly different from MC^3, and in particular significantly larger; but this did not translate into a large gain in predictive performance (0.18%).

Moreover, observe that for the insurance data there was no sample where the posterior distributions had a common model among the three with highest posterior. This may be related to the fact that the Bayesian networks being learned are significantly more complex than in the other two datasets (see the #e column in Table 3). It is probably more difficult for the Markov chains to converge to the same posterior when there is a higher complexity in the interactions underlying the data.

The fact that the eMC^3 algorithm does not show an improvement in predictive performance over MC^3 with these real-world data is unfortunate and we think that it is a byproduct of the following two facts. On the one hand the MC^3 algorithm is, in a way, already doing the RCAR operation when uses the AR neighborhood, as it is able to move between Markov equivalent Bayesian network structures. This is happening, of course, at a much slower rate than in eMC^3 and that would explain the faster convergence of eMC^3 with synthetic data. On the other hand, the bias introduced by the sizes of the equivalence classes when using DAG-space may be overriding the benefit of using an inclusion-driven strategy, but it is an open question to what extent this effect influences the result.

6. Conclusion

In this paper we have shown the fundamental role that the graphical Markov model inclusion order plays in structure learning of Bayesian networks. In particular, we have introduced

the inclusion boundary condition that provides us with a general policy for the design of effective traversal operators for any given class of GMMs.

Following that policy, we designed two new concepts of neighborhood for DAG-space, the RCARNR and RCARR. These neighborhoods implement two new ways of traversing the search space and have led to two new learning algorithms in the context of heuristic and MCMC learning: the HCMC and eMC^3 algorithms.

We proved, under the assumptions of faithfulness of the underlying probability distribution to a DAG, and unbounded data length, that the inclusion boundary condition guarantees a search strategy the capability of avoiding local maxima in structure learning of Bayesian networks.

The experiments with heuristic learning and synthetic data show that such an asymptotic result is relevant for samples of limited size, because the accuracy of the HCMC algorithm improves substantially as more data becomes available. Note that this was not the case for the standard hill-climber using the AR neighborhood (see Figure 9), which shows how crucial the use of an inclusion-driven strategy in structure learning of Bayesian networks can be.

The experiments with real-world data show an improvement in score and predictive performance for the models learned with HCMC, but that is not the case for eMC^3, which does show an improvement in the rate of convergence with synthetic data. The possible reasons for the lack of improvement of eMC^3 in predictive performance may be the following. First, the MC^3 algorithm with the AR neighborhood already performs "in a way" the RCAR operation. Second, the approximated posterior distribution may be biased by the size of the equivalence classes, as pointed out by Madigan et al. (1996). It would be interesting to analyze carefully the extent to which the size of the equivalence classes biases the result when using eMC^3. Once this is known, it may be possible to correct such a bias within the algorithm.

Acknowledgments

This paper has benefited from discussions with Max Chickering, Steven Gillispie, Finn V. Jensen and Michael Perlman. The authors are grateful to the editor and the anonymous reviewers whose suggestions and remarks helped to improve this paper. Part of this work was carried out when the first author was working at the Institute of Information and Computing Sciences of the University of Utrecht, The Netherlands.

References

S. Andersson, D. Madigan, and M. Perlman. A characterization of Markov equivalence classes for acyclic digraphs. *Annals of Statistics*, 25:505–541, 1997a.

S. Andersson, D. Madigan, and M. Perlman. On the Markov equivalence of chain graphs, undirected graphs, and acyclic digraphs. *Scandinavian Journal of Statistics*, 24:81–102, 1997b.

S. Andersson, D. Madigan, M. Perlman, and C. Triggs. On the relation between conditional independence models determined by finite distributive lattices and by directed acyclic graphs. *Journal of Statistical Planning and Inference*, 48:25–46, 1995.

I. Beinlich, H. Suermondt, R. Chavez, and G. Cooper. The alarm monitoring system: A case study with two probabilistic inference techniques for belief networks. In J. Hunter, editor, *Proc. of the Second European Conference on Artificial Intelligence in Medicine*, pages 247–256. Springer-Verlag, 1989.

R. Bouckaert. Optimizing causal orderings for generating dags from data. In D. Dubois, M.P. Wellman, B. D'Ambrosio, and P. Smets, editors, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, pages 9–16. Morgan Kaufmann, 1992.

J.S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In G.F. Cooper and S. Moral, editors, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, pages 43–52. Morgan Kaufmann, 1998.

W. Buntine. Theory refinement on Bayesian networks. In P. Smets B. D'Ambrosio and P.P. Bonissone, editors, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, pages 52–60. Morgan Kaufmann, 1991.

R. Castelo, A. Feelders, and A. Siebes. Mambo: Discovering association rules based on conditional independencies. In F. Hoffmann, D.J. Hand, N.M. Adams, D. Fisher, and G. Guimarães, editors, *Proceedings 4th Symposium on Intelligent Data Analysis*, volume 2189 of *Lecture Notes in Computer Science*, pages 289–298. Springer, 2001.

R. Castelo and M.D. Perlman. Learning essential graph Markov models from data. In J.A. Gámez and A. Salmerón, editors, *First European Workshop on Probabilistic Graphical Models*, pages 17–24, November 2002.

S. Chib and E. Greenberg. Understanding the Metropolis-Hastings algorithm. *American Statistician*, 49(4):327–335, 1995.

D.M. Chickering. A transformational characterization of equivalent Bayesian networks. In P. Besnard and S. Hanks, editors, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, pages 87–98. Morgan Kaufmann, 1995.

D.M. Chickering. Learning equivalence classes of Bayesian network structures. In E. Horvitz and F. Jensen, editors, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, pages 150–157. Morgan Kaufmann, 1996.

D.M. Chickering. Learning equivalence classes of Bayesian-network structures. *Journal of Machine Learning Research*, 2:445–498, 2002a.

D.M. Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2002b.

D.M. Chickering, D. Geiger, and D. Heckerman. Learning Bayesian networks: Search methods and experimental results. In D. Fisher and H-J. Lenz, editors, *Proc. of the Intl. Workshop on Artificial Intelligence and Statistics*, pages 112–128, 1995.

K.L. Chung. *Markov Chains with Stationary Transition Probabilities (2nd ed).* Springer-Verlag, 1967.

G. Cooper and E.H. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–405, 1992.

A.P. Dawid. Conditional independence in statistical theory (with discussion). *Journal of the Royal Statistical Society B*, 41(1):1–31, 1979.

D. Draper. Assessment and propagation of model uncertainty. *Journal of the Royal Statistical Society B*, 57:45–97, 1995.

D. Edwards and T. Havránek. A fast procedure for model search in multidimensional contingency tables. *Biometrika*, 72(2):339–351, 1985.

L. Fahrmeir. *Multivariate Statistical Modelling Based on Generalized Linear Models.* Springer-Verlag, New York, 1994.

N. Friedman and D. Koller. Being Bayesian about network structure. In C. Boutilier and M. Goldszmidt, editors, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, pages 201–210. Morgan Kaufmann, 2000.

M. Frydenberg. The chain graph Markov property. *Scandinavian Journal of Statistics*, 17: 333–353, 1990.

S. Gillispie and M. Perlman. Enumerating Markov equivalence classes of acyclic digraph models. In J. Breese and D. Koller, editors, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, pages 171–177. Morgan Kaufmann, 2001.

P. Giudici and R. Castelo. Association models for Web Mining. *Journal of Data Mining and Knowledge Discovery*, 24(1):39–57, 2001.

P. Giudici and R. Castelo. Improving Markov Chain Monte Carlo model search for Data Mining. *Machine Learning*, 50(1/2):127–158, 2003.

P. Giudici and P.J. Green. Decomposable graphical gaussian model determination. *Biometrika*, 86(4):785–801, 1999.

W.K. Hastings. Monte carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

T. Havránek. A procedure for model search in multidimensional contingency tables. *Biometrics*, 40:95–100, 1984.

D. Heckerman, D. Geiger, and D.M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20:194–243, 1995.

E.H. Herskovits. *Computer-Based Probabilistic Network Construction.* PhD thesis, Medical Information Sciences, Stanford University, 1991.

J.S. Ide and F.G. Cozman. Random generation of Bayesian networks. In G. Bittencourt, editor, *Proc. of the Brazilian Symposium on Artificial Intelligence*, pages 366–375. Springer-Verlag, 2002.

R.E. Kass and A.E. Raftery. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995.

T. Kočka. *Graphical models - learning and applications*. PhD thesis, University of Prague, 2001.

T. Kočka, R. Bouckaert, and M. Studený. On characterizing inclusion of Bayesian networks. In J. Breese and D. Koller, editors, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, pages 261–268. Morgan Kaufmann, 2001.

T. Kočka and R. Castelo. Improved learning of Bayesian networks. In J. Breese and D. Koller, editors, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, pages 269–276. Morgan Kaufmann, 2001.

P. Larrañaga, C. Kuijpers, R. Murga, and Y. Yurramendi. Learning Bayesian network structures by searching for the best ordering with genetic algorithms. *IEEE Transactions on Systems, Man and Cybernetics*, 26(4):487–493, 1996.

S.L. Lauritzen. *Graphical Models*. Oxford University Press, Oxford, 1996.

S.L. Lauritzen, A.P. Dawid, B.N. Larsen, and H.G. Leimer. Independence properties of directed Markov fields. *Networks*, 20:491–505, 1990.

D. Madigan, S.A. Andersson, M. Perlman, and C.T. Volinsky. Bayesian model averaging and model selection for Markov equivalence classes of acyclic digraphs. *Communications in Statistics (theory and methods)*, 25(11):2493–2512, 1996.

D. Madigan and J. York. Bayesian graphical models for discrete data. *International Statistical Review*, 63:215–232, 1995.

C. Meek. *Graphical models, selecting causal and statistical models*. PhD thesis, Carnegie Mellon University, 1997.

G. Melançon, I. Dutour, and M. Bousquet-Melou. Random generation of DAGs for graphs drawing. Technical report, Centrum voor Wiskunde en Informatica, February 2000.

N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, San Mateo, California, 1988.

J. Pearl and T. Verma. The logic of representing dependencies by directed graphs. In K. Forbus and H. Shrobe, editors, *Proc. of the Conf. of the American Association of Artificial Intelligence*, pages 374–379, 1987.

R.W. Robinson. Counting labeled acyclic digraphs. In F. Harary, editor, *New Directions in the Theory of Graphs*, pages 239–273. Academic Press, New York, 1973.

M. Singh and M. Valtorta. An algorithm for the construction of Bayesian network structures from data. In D. Heckerman and A. Mamdani, editors, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, pages 259–265. Morgan Kaufmann, 1993.

A.F.M. Smith and G.O. Roberts. Bayesian computation via the gibbs sampler and related Markov Chain Monte Carlo methods. *Journal of the Royal Statistical Society B*, 55(1): 3–23, 1993.

P. Spirtes, C. Glymour, and R. Scheimes. *Causation, Prediction and Search*. Springer-Verlag, New York, 1993.

P. Spirtes and C. Meek. Learning Bayesian networks with discrete variables from data. In U.M. Fayyad and R. Uthurusamy, editors, *Proc. of the Intl. Conf. on Knowledge Discovery and Data Mining*, pages 294–299. AAAI Press, 1995.

T.S. Verma and J. Pearl. Influence diagrams and d-separation. Technical report, Cognitive Systems Laboratory, UCLA, March 1988.

T.S. Verma and J. Pearl. Equivalence and synthesis of causal models. In P.P. Bonissone, M. Henrion, L.N. Kanal, and J.F. Lemmer, editors, *Proc. of the Conf. on Uncertainty in Artificial Intelligence*, pages 255–268. Morgan Kaufmann, 1990.

J. Whittaker. *Graphical Models in Applied Multivariate Statistics*. Wiley, New York, 1990.