Bayesian Inference in Treewidth-Bounded Graphical Models  
Without Indegree Constraints

Daniel J. Rosenkrantz  
University at Albany – SUNY  
Albany, NY 12222  
drosenkrantz@gmail.com

Madhav V. Marathe  
Virginia Tech  
Blacksburg, VA 24061  
mmarathe@vbi.vt.edu

S. S. Ravi  
University at Albany – SUNY  
Albany, NY 12222  
ravi@cs.albany.edu

Anil K. Vullikanti  
Virginia Tech  
Blacksburg, VA 24061  
akumar@vbi.vt.edu

Abstract

We present new polynomial time algorithms for inference problems in Bayesian networks (BNs) when restricted to instances that satisfy the following two conditions: they have bounded treewidth and the conditional probability table (CPT) at each node is specified concisely using an $r$-symmetric function for some constant $r$. Our polynomial time algorithms work directly on the unmoralized graph. Our results significantly extend known results regarding inference problems on treewidth bounded BNs to a larger class of problem instances. We also show that relaxing either of the conditions used by our algorithms leads to computational intractability.

1 INTRODUCTION

Bayesian Networks (BNs) represent dependencies among a collection of probabilistic domain variables (Darwiche, 2009; Koller and Friedman, 2009; Pearl, 1988). Structurally, a BN $G(V,E)$ is a directed acyclic graph (dag) in which each node $v \in V$ represents a stochastic variable $x_v$; a directed edge $(u,v)$ in $E$ indicates that variable $x_v$ depends on $x_u$. Each node $v$ is also associated with a table which gives the probability distribution of $x_v$ conditioned on the variables on which $x_v$ depends. Thus, a BN provides a simple graphical representation of the dependencies among domain variables.

BNs can be used to formulate and solve many problems in the context of stochastic decision support systems. For example, in the inference problem, the input is an observation (i.e., the observed values of a nonempty subset of variables) and the goal is to compute the conditional probability distribution for one specified variable. Such problems are useful in many application domains including medical diagnosis, weather forecasting, design of diagnosis-and-repair modules in computer systems, etc. (Darwiche, 2009; Koller and Friedman, 2009; Pearl, 1988).

Formal definitions of inference problems for BNs are provided in Section 2. In general, obtaining exact or approximate solutions to these problems is known to be computationally intractable (Abdelbar et al., 2000; Cooper, 1990; Dagum and Luby, 1993; Darwiche, 2009; de Campos (2011)). Given the practical importance of these problems, researchers have tried to identify restricted versions of the problems which are useful in practice and which can be solved in polynomial time. An important development in this direction is the result of Lauritzen and Spiegelhalter (1988) who showed that for BNs of bounded treewidth, the inference problems can be solved in polynomial time using dynamic programming. Their approach uses the moralized form of the network, where for any node $v$, the parents of $v$ are connected together as a clique. As a consequence, a moralized BN has bounded treewidth only when the maximum indegree in the unmoralized BN is also bounded.

Our Contributions: Our main result is a new dynamic programming approach that extends the class of BNs for which various inference problems can be solved in polynomial time. In particular, our approach does not use moralization. Instead, it works with the given BN and its tree decomposition. Thus, our algorithms are applicable to treewidth-bounded BNs, even when the indegrees of nodes are not bounded. Allowing nodes of unbounded indegree introduces a difficulty, namely that a fully specified CPT at a node may be exponentially large. To overcome this difficulty, we require the conditional probability tables at each node to be specified concisely using certain restricted classes of functions, called $r$-symmetric functions for some fixed integer $r$. As will be explained in Section 2, any CPT for a BN with maximum indegree $d$ can be specified as a $d$-symmetric function. In other words, CPTs for BNs with bounded indegrees are a restricted form of $r$-symmetric functions. Thus, our approach identifies a larger class of BNs for which inference problems can be solved efficiently. Our results also extend the earlier results in (Bacchus et al., 2003; Courcelle et al., 2001; Fischer et al., 2008; Samer and Szeider, 2007) as discussed below. The results in (Courcelle et al., 2001; Fischer et al., 2008; Samer and Szeider, 2007) when combined with those of (Bacchus
et al., 2003) can be used to obtain polynomial time results for treewidth bounded BNs, with no a priori bound on indegree but whose CPTs are specified using certain threshold functions. As will be explained in Section 2.3, the class of \( r \)-symmetric functions is a strict superset of the class of threshold functions.

We also present hardness results that provide an indication of the tightness of our efficient solvability results. In particular, we show that if the conditional probability tables are not necessarily \( r \)-symmetric, then the inference problems remain computationally intractable (\#P-hard) even when the BN is a directed tree (whose treewidth is 1). We also show that if the treewidth of the BN is not bounded, the inference problems remain computationally intractable even when each conditional probability table is expressed as a symmetric function. In other words, relaxing either of the assumptions (treewidth boundedness or \( r \)-symmetric functions) makes the problems computationally intractable. We note that the necessity of bounded treewidth for efficient inference in BNs was proven in Kwisthout et al. (2010) under a different assumption regarding complexity classes, namely, the Exponential Time Hypothesis (Impagliazzo and Paturi, 2001).

The remainder of this paper is organized as follows. Section 2 defines the necessary graph theoretic terms and presents formulations of several computations problems for BNs. It also discusses some related work on these problems. For space reasons, Section 3 presents our algorithm assuming that each CPT is a 1-symmetric function. Extensions of the result to other inference problems and \( r \)-symmetric functions (for any \( r \geq 2 \)) are discussed in a longer version of this paper (Rosenkrantz et al., 2014). Section 4 mentions our hardness results whose proofs also appear in Rosenkrantz et al. (2014). Section 5 summarizes the paper and provides directions for future work.

## 2 Definitions and Previous Work

### 2.1 Bayesian Networks

As mentioned earlier, a Bayesian network (BN) consists of a directed acyclic graph \( G(V, E) \), where nodes represent stochastic domain variables and directed edges represent dependencies between variables. For simplicity, we will assume that each node represents a Boolean variable; the results in this paper can be extended to variables that assume values from a finite domain. Also, we do not distinguish between a node of the graph and the corresponding Boolean variable. When there is a directed edge \((u, v) \in E\), we say that \( u \) is a parent of \( v \). The indegree of a node \( v \) is the number of parents of \( v \).

At each node \( v \), there is a conditional probability table (CPT) \( T_v \) which specifies the probability values for the variable \( v \), conditioned on the parents of \( v \). For a node \( v \) with indegree \( t \), there are \( 2^t \) different combinations of Boolean values for the parents of \( v \). For each such combination, the table specifies the probability of \( v \) being 1 (or 0) conditioned on the parents assuming the given combination of values. Thus, the number of entries in \( T_v \) is \( 2^t \). For a node \( v \) with indegree 0 (i.e., a node which does not have any parent), the table \( T_v \) specifies simply the probability of \( v \) assuming the value 1 (or 0).

Our formulation of the computational problems for BNs follows the presentation in (Bodlaender, 2004). For a node \( v \), let \( P(v) \) denote the set of parents of \( v \). Given a BN \( G(V, E) \), a configuration \( c_v \) is an assignment of Boolean values to each variable in \( V \). Given a subset \( O \subseteq V \), a partial configuration \( c_O \) on \( O \) specifies a value for each variable in \( O \). Given a configuration \( c_v \) (or a partial configuration \( c_O \)), we use \( c_v(v) (c_O(v)) \) to denote the value of variable \( v \) in that configuration (partial configuration). With a slight abuse of notation, we also extend this notation to subsets of variables. Thus, given a configuration \( c_v \) (or a partial configuration \( c_O \)), and a subset \( W \subseteq V \) (\( W \subseteq O \)), \( c_v(W) (c_O(W)) \) denotes the combination of values assigned to the variables in \( W \). Given a configuration \( c_v \), its probability \( \Pr\{c_v\} \) is given by

\[
\Pr\{c_v\} = \prod_{v \in V} \Pr\{c_v(v) | c_v(P(v))\}.
\]

A configuration \( c_v \) is an extension of a partial configuration \( c_O \) if for each variable \( v \) that is assigned a value in \( c_O \), \( c_v(v) = c_O(v) \). Thus, an extension of a partial configuration \( c_O \) is obtained by specifying values for the variables that are not assigned a value in \( c_O \).

We now provide formal definitions of two commonly considered problems in the context of BNs. In all these problems, we are given a partial configuration \( c_O \) (also called an observation) on a set of nodes \( O \subseteq V \).

**Definition 2.1** Let \( G(V, E) \) denote the given BN.

1. **Inference Problem** (denoted by \( \text{INF} \)): Given an observation \( c_O \) and a variable \( v \), find \( \Pr\{v = 1 \mid c_O\} \), that is, the probability that \( v \) assumes the value 1 conditioned on the observation \( c_O \).

2. **Most Probable Explanation Problem** (denoted by \( \text{MPE} \)): Given an observation \( c_O \), find an extension of \( c_O \) which has the maximum probability among all the extensions of \( c_O \).

For ease of exposition, we will also consider the following problem, which we call the **Probability Computation Problem** (denoted by \( \text{PROB} \)): Given an observation \( c_O \), find the probability of \( c_O \), that is, the sum of the probabilities of all configurations that are extensions of \( c_O \). (Thus, when an observation does not specify a value for any variable, the answer to the \( \text{PROB} \) problem is 1.) The reasons for considering the \( \text{PROB} \) problem are twofold. First, the solution to \( \text{INF} \) problem for a node \( v \) and observation \( c_O \) can
be obtained using two calls to the PROB problem: compute
Pr\{c_O\} and Pr\{v = 1 \land c_O\} using the algorithm for PROB
and use the fact that
\[
Pr\{v = 1 \mid c_O\} = \frac{Pr\{v = 1 \land c_O\}}{Pr\{c_O\}}.
\]
Second, an algorithm for the MPE problem can be devised
along lines that are similar to that for the PROB problem.
The main modification is that while the algorithm for the
PROB problem computes sums of probability values at vari-
sous steps, the algorithm for the MPE problem computes
the maximum of the probability values.

2.2 TREE DECOMPOSITIONS

We now recall the standard definition of tree decomposition
and treewidth from (Bodlaender, 1993), which will be used
throughout this paper.

Definition 2.2 Given a BN \( G(V, E) \), a tree decomposi-
tion of \( G \) is a pair \(( \{X_i\mid i \in I\}, T = (I, F)\)\), where
\( \{X_i\mid i \in I\} \) is a family of subsets of \( V \) and \( T = (I, F) \) is
an undirected tree with the following properties:

1. \( \bigcup_{i \in I} X_i = V \).
2. For every directed edge \( e = (v, w) \in E \), there is a
subset \( X_i \), \( i \in I \), with \( v \in X_i \) and \( w \in X_i \).
3. For all \( i, j, k \in I \), if \( j \) lies on the path from \( i \) to \( k \) in
\( T \), then \( X_i \cap X_k \subseteq X_j \).

The treewidth of a tree decomposition \(( \{X_i\mid i \in I\}, T)\) is
\( \max_{i \in I} \{|X_i| - 1\} \). The treewidth of a graph is the mini-
\mum over the treewidths of all its tree decompositions.

A class of graphs is treewidth bounded if there is a con-
stant \( k \) such that the treewidth of every graph in the class
is at most \( k \).

A number of problems that are NP-hard on general graphs
can be solved efficiently when restricted to the class of
treewidth-bounded graphs. A considerable amount of work
has been done in this area (see for example (Bodlaen-
der, 1997, 1993; Courcelle and Mosbah, 1993; Gottlob and
Szeider, 2008; Robertson and Seymour, 1986) and the ref-
\erences therein).

As mentioned earlier, our approach works on the given (un-
moralized) BN. To illustrate the effect of moralization on a
BN, consider the class of directed star graphs defined as
follows. For each \( n \geq 2 \), a directed star graph has \( n \) nodes
and \( n - 1 \) directed edges; there is one center node and each of the
other \( n - 1 \) nodes has just one outgoing edge to the
center node. Thus, the center node has \( n - 1 \) parents and the
moralized graph has a clique of size \( n - 1 \). Consequently,
the moralized graph is not treewidth-bounded. On the other
hand, it can be seen that according to Definition 2.2, this
class of graphs has a treewidth of 1.

2.3 SPECIFYING CPTs CONCISELY

For a node \( v \) with \( q \) parents, the CPT \( T_v \) has \( 2^q \) entries. For
BNs in which the maximum indegree is bounded, the CPTs
can be given explicitly, since the size of each table is just
a constant. However, when we consider BNs in which node
indegrees may not be bounded, the size of a fully specified
table may be exponential in the size of the BN. Thus, we
need a method of specifying the CPTs concisely. We do
this by identifying restricted classes of functions to specify
the tables.

Consider a node \( v \) with \( q \) parents \( w_1, w_2, \ldots, w_q \). A
CPT \( T_v \) for \( v \) specifies a probability value (i.e., the value
\( Pr\{v = 1\} \)) for each combination of values of the parents
of \( v \). Thus, \( T_v \) represents a function from \( \{0, 1\}^q \) to the set
of real values in \([0, 1]\). By restricting the class of functions,
we can specify \( T_v \) concisely. We will now present some
examples of such restrictions.

Definition 2.3 Let \( q \) be an integer \( \geq 1 \). A function \( f \) from
\( \{0, 1\}^q \) to the set of real values in \([0, 1]\) is said to be sym-
metric if the value of \( f \) depends only on the number of in-
puts which are 1.

Thus, a symmetric function \( f \) of \( q \) variables can be con-
cisely described by specifying \( q + 1 \) probability values \( p_0, p_1, \ldots, p_q \), where \( p_i \) is the probability value when \( i \) of
\( p \) inputs are 1, 0 \( \leq i \leq q \).

Example: Consider the BN shown in Figure 1. In that
figure, nodes \( v_1, v_2 \) and \( v_3 \) don’t have any parents. So, the
probability values assigned to them can be thought of
as symmetric functions where the only possible value for
the number of parents is zero. Node \( v_4 \) has three parents.
Hence, the CPT for \( v_4 \) shows the value of \( Pr\{v_4 = 1\} \) when
0, 1, 2 or 3 parents of \( v_4 \) are assigned the value 1.

For any integer \( t \geq 0 \), a \( t \)-threshold Boolean function on \( q \)
inputs takes on the value 1 iff at least \( t \) of the inputs are 1. It
is easy to see that each \( t \)-threshold function is a symmetric
function. Thus, the class of symmetric functions contains
all threshold functions.

One can define a further generalization of the class of sym-
metric functions as follows (Barrett et al., 2007b).

Definition 2.4 Let \( r \geq 1 \) be a fixed integer. Let \( q \) be an
integer \( \geq 1 \). A function \( f \) from \( \{0, 1\}^q \) to the set of real
values in \([0, 1]\) is said to be \( r \)-symmetric if the set of in-
puts can be partitioned into \( r \) classes such that the value of
\( f \) depends only on the number of 1-valued inputs in each
class.

Note that any symmetric function is 1-symmetric. It can
also be seen that for any \( r \geq 1 \), any \( r \)-symmetric function
\( f \) of \( q \) variables can be concisely described by specifying
\( O(q^r) \) probability values. Since \( r \) is fixed, the size of the
specification of any \( r \)-symmetric function is a polynomial
in the size of the BN.
When a node has a bounded indegree, say $d$, the corresponding CPT can be thought of as a $d$-symmetric function, where each of the $d$ classes contains exactly one input. Thus, the class of BNs in which each node has a bounded indegree is a special case of BNs in which each CPT is specified by an $r$-symmetric function for some fixed $r$.

### 2.4 OTHER RELATED WORK

Motivated by the practical importance of inference problems (Darwiche (2009); Koller and Friedman (2009); Pearl (1988)), research in this area has proceeded along two primary directions. The first direction focuses on the development of efficient heuristics that can be used to obtain fast solutions to problems that arise in practice (see for example (Chavira, 2007; Dechter, 1999) and the references cited therein). The second direction is the identification of restricted versions of inference problems that can be solved efficiently. As mentioned earlier, an important step in that direction is the work of Lauritzen and Spiegelhalter (1988) which provides an efficient algorithm for inference problems for treewidth bounded (moralized) BNs. Other references that consider inference problems for restricted versions of BNs include (Bacchus et al., 2003; Boutilier et al., 1996; Dechter, 1999; Jensen et al., 1990). The notion of causal independence used in Zhang and Poole (1996) relies on conditional probability tables that are essentially symmetric functions. We note that symmetric functions have also been used in the context of lifted inference (Jha et al., 2010; Milch et al., 2008).

Another approach, called parent divorcing, for dealing with nodes of large indegrees was introduced in Olesen et al. (1989). The basic idea of this approach is to modify a given BN in the following manner: when a node has a large indegree, the subgraph consisting of the node and its predecessors is replaced by a directed tree in which each node has a small indegree. An example to illustrate this approach is shown in Figure 2. There are two main difficulties with this approach. The first is that the treewidth of the resulting BN can be much larger than that of the given BN. The second difficulty is that the size of domain from which newly added nodes take on values may become large. These two aspects can significantly increase the running time of the algorithms for the inference problems. Our algorithms can handle nodes with unbounded indegrees without modifying the given BN, provided all the CPTs are described by $r$-symmetric functions for some fixed integer $r$.

### 3 POLYNOMIAL TIME ALGORITHMS FOR TREEWIDTH-BOUNDED BNs WITH SYMMETRIC CPTs

#### 3.1 OVERVIEW

This section presents polynomial time algorithms for inference problems for treewidth-bounded BNs where each probability table is represented as symmetric function. We assume that a BN is given along with its tree decomposition of treewidth $k$, for some fixed integer $k \geq 1$. We will present the details of the algorithm for the PROB problem. The modifications needed to solve the INF and MPE problems and the extension of the algorithm to handle $r$-symmetric CPTs for any fixed $r \geq 1$ are presented in Rosenkrantz et al. (2014).

#### 3.2 NOTES ON TREE DECOMPOSITION

This section mentions some known facts about tree decompositions and also reviews some related terminology. We assume that one of the nodes of the tree decomposition is selected as the root so that the tree decomposition can be viewed as a rooted tree. When a graph $G$ has bounded treewidth, it is well known that a tree decomposition $(\{X_i | i \in I\}, T = (I, F))$ of $G$ can be constructed in time that is a polynomial in the size of $G$. Moreover, this can be done so that all of the following conditions hold (Barrett et al., 2007a,b; Bodlaender, 1997): (a) $T$ is a binary tree; that is, each node of $T$ has at most two children. (b) The number of nodes of $T$ with fewer than two children is at most $n$, the number of nodes in $G$. (c) The number of nodes of $T$ with two children is at most $n$. Our algorithm relies on this special form of tree decomposition.
The following terminology regarding nodes in tree decompositions is from (Barrett et al., 2007a,b). Let $T$ be the given tree decomposition of a BN $G$. For a given node $i$ of $T$, the nodes of $G$ in $X_i$ are called explicit nodes of $i$. If a given explicit node $v$ of $i$ is also an explicit node of the parent of $i$, then $v$ is referred to as an inherited node of $i$; and if $v$ does not occur in the parent of $i$, then $v$ is called an originating node of $i$.

We refer to the set of all explicit nodes occurring in the subtree of $T$ rooted at $i$ that are not explicit nodes of $i$ as hidden nodes of $i$. (Thus, the hidden nodes of $i$ are the union of the originating and hidden nodes of the children of $i$.) For any node $i$ in $T$, we use $Y_{i}^{\text{inh}}$, $Y_{i}^{\text{org}}$ and $Y_{i}^{\text{hid}}$ to denote respectively the set of inherited nodes, the set of originating nodes and the set of hidden nodes of $i$.

**Example:** A tree decomposition for the BN of Figure 1 is shown in Figure 3. The tree decomposition has three nodes. For each tree node, the set of explicit nodes is shown. The table in Figure 3 also shows the explicit, inherited, originating and hidden sets for each tree node.

**3.3 CONFIGURATIONS AND SIGNATURES**

For all of the computational problems we consider, we are given a BN $G(V,E)$ and an observation $c_O$. As mentioned earlier, we also assume that each CPT is specified as a symmetric function.

**Definition 3.1** (a) Let $Y$ be a set of nodes of the given BN $G$. We refer to a partial configuration on $Y$ as a $Y$–configuration.

(b) Let $Y$ and $W$ be not necessarily disjoint sets of nodes of the given BN $G$. We say that a given $Y$–configuration $\alpha$ and a given $W$–configuration $\beta$ are consistent if for all nodes $z$ in $Y \cap W$, $Y(z) = W(z)$.

(c) Given the observation (i.e., $O$–configuration) $c_O$, we say that a given $Y$–configuration $\alpha$ is valid if for all nodes $w$ assigned values in both $c_O$ and $\alpha$, $c_O(w) = \alpha(w)$, i.e., $\alpha$ and $c_O$ are consistent.

(d) $\Gamma_Y$ denotes the set of all valid $Y$–configurations.

(e) Let $Y$ and $W$ be sets of nodes of the given BN $G$ such that $W \subseteq Y$. Let $\alpha$ be a $Y$–configuration. We define the restriction of $\alpha$ to $W$, denoted as $\alpha|W$, to be the $W$–configuration obtained by restricting $\alpha$ to the members of $W$.

The concept of a signature, defined below for the case where each CPT is a symmetric function, plays an important role in our algorithm.

**Definition 3.2** Let $Y$ and $W$ be not necessarily disjoint sets of nodes of the given BN $G$. (a) Let $\alpha$ be a $Y$–configuration. The signature of $\alpha$ with respect to $W$, denoted as $\text{sig}(\alpha,W)$, specifies for each $w \in W$, the number of parents of $w$ that are set to $1$ by $\alpha$. We refer to such a signature as a $(Y,W)$–signature.

(b) Suppose $\Gamma$ is a set of $Y$–configurations. The signature of $\Gamma$ with respect to $W$ is the union of the signature of each $\gamma \in \Gamma$ with respect to $W$.

(c) We say that a given $(Y,W)$–signature is valid if it is $\text{sig}(\alpha,W)$ for some valid $Y$–configuration $\alpha$.

(d) $H_{Y,W}$ denotes the set of all valid $(Y,W)$–signatures.

**Example:** Consider the BN shown in Figure 1. Let $Y = \{v_2, v_3\}$ and $W = \{v_1, v_4\}$. Consider the $Y$–configuration $\gamma$ which sets $v_2 = 0$ and $v_3 = 1$. It is easy to see that $\gamma$ sets $0$ of $v_1$’s parents to $1$ and exactly one of $v_4$’s parents to $1$. So $\text{sig}(\gamma,W)$, the signature of $\gamma$ with respect to $W$, can be represented as $[v_1 : 0, v_4 : 1]$. Suppose that the given observation $c_O$ does not specify the value of any node. Then $\Gamma_Y$ contains four $Y$–configurations. By computing the union of the signatures of these four $Y$–configurations, it can be seen that $H_{Y,W}$ is the set $\{[v_1 : 0, v_4 : 0], [v_1 : 0, v_4 : 1], [v_1 : 0, v_4 : 2]\}$.

If $\sigma$ is $(Y,W)$–signature, then for any $w \in W$, we use $\sigma(w)$ to denote the value specified by $\sigma$ for $w$. Using this notation, we define some operations on signatures which produce new signatures. These operations are used by our algorithm.

**Definition 3.3** Let $G$ be a BN and let $W$ be a subset of nodes of $G$.

(a) Let $\sigma$ and $\sigma'$ be two signatures with respect to a node set $W$. The sum of the two signatures is another signature denoted by $\sigma + \sigma'$, such that for each $w \in W$, $(\sigma + \sigma')(w) = \sigma(w) + \sigma'(w)$.

(b) Let $\sigma$ be a signature with respect to a node set $W$ and let $Y$ be a subset of $W$. The restriction of $\sigma$ to $Y$ is another signature denoted by $\sigma|Y$, such that for each $y \in Y$, $(\sigma|Y)(y) = \sigma(y)$.

706
(c) Let \( \sigma \) be a signature with respect to a node set \( W \) and let \( X \) be a superset of \( W \). The extension of \( \sigma \) to \( X \) is another signature denoted by \( \text{ext}(\sigma, X) \), which is defined as follows: for each \( x \in X \), if \( x \in W \), then \( \text{ext}(\sigma, X)(x) = \sigma(x) \); otherwise, \( \text{ext}(\sigma, X)(x) = 0 \).

Suppose \( w \) is a node of \( G \) and \( \eta \) is a partial configuration that specifies a value (namely, \( \eta(w) \)) for \( w \) and for every parent of \( w \). Given these values, the CPT for \( w \) specifies a probability value for \( \eta(w) \), which will be denoted by \( p_{\eta}(w) \). Suppose \( W \) is a subset of nodes of \( G \) and \( \eta \) is a partial configuration that specifies a value for every node \( w \in W \) and for every parent of every node \( w \in W \). Thus, for every node \( w \in W \), the value \( p_{\eta}(w) \) is defined. We define \( p_{\eta}(W) \) by

\[
p_{\eta}(W) = \prod_{w \in W} p_{\eta}(w). \tag{1}
\]

We also need a slight extension of the definition given by Equation (1). Let \( X \) and \( Y \) be disjoint sets of nodes of \( G \). Let \( \eta \) be a \( X \)-configuration, and \( \sigma \) be a \( (Y, X) \)-signature. Suppose that for a given node \( w \) in \( X \), all the parents of \( w \) are in \( X \cup Y \). Because the CPT for \( w \) is given by a symmetric function, given the values that \( \eta \) assigns to those parents of \( w \) that are in \( X \), and the value that \( \sigma \) assigns to \( w \), the CPT for \( w \) assigns a probability value to \( \eta(w) \). We denote this probability value as \( p_{\eta, \sigma}(w) \). Suppose that \( W \) is a subset of \( X \), such that for every node \( w \) in \( W \), all the parents of \( w \) are in \( X \cup Y \). (Thus, for every node \( w \) in \( W \), the value \( p_{\eta, \sigma}(w) \) is defined.) Now, we define \( p_{\eta, \sigma}(W) \) by

\[
p_{\eta, \sigma}(W) = \prod_{w \in W} p_{\eta, \sigma}(w). \tag{2}
\]

### 3.4 Algorithm for the Prob Problem

Recall that in the PROB problem, we are given a BN \( G(V, E) \) and an observation \( c_O \). The goal is to find the probability of \( c_O \), that is, the sum of the probabilities of all (complete) configurations that are extensions of \( c_O \). Let the constant \( k \) denote the treewidth of \( G \). We assume that a tree decomposition \( \{\{X_i | i \in I\}, T = (I, F)\} \) of \( G \) satisfying all the conditions mentioned in Section 3.2 is also given and that each CPT is specified as a symmetric function.

#### 3.4.1 Information Maintained by the Algorithm

Our algorithm solves the PROB problem for \( G \) by using bottom-up dynamic programming on the tree decomposition \( T \). The algorithm maintains information for each node of \( T \), as summarized in Table 1, and described below.

For each node \( i \) of \( T \), the algorithm maintains the two sets of signatures \( H_{Y^{\text{hid}}, X_i} \) and \( H_{(Y^{\text{hid}} \cup Y^{\text{org}}), Y^{\text{inh}}}, \) plus two tables of probability values, which we denote as \( Q^i \) and \( R^i \). We now provide a description of these signature sets and tables for each tree node \( i \).

\( a \) \( H_{Y^{\text{hid}}, X_i} \) is the set of all valid \( (Y^{\text{hid}}, X_i) \)-signatures. (Recall that \( Y^{\text{hid}} \) is the set of hidden nodes of \( i \), and \( X_i \) is the set of explicit nodes of \( i \).)

\( b \) \( H_{(Y^{\text{hid}} \cup Y^{\text{org}}), Y^{\text{inh}}}, \) is the set of all valid \( (Y^{\text{hid}} \cup Y^{\text{org}}, Y^{\text{inh}}) \)-signatures. (Recall that \( Y^{\text{inh}} \) is the set of inherited nodes of \( i \).

\( c \) Table \( Q^i \) contains a probability value for each pair in \( \Gamma_{X_i} \times H_{Y^{\text{hid}}, X_i} \).

Consider a given element of table \( Q^i \), say \( Q^i[\alpha, \sigma] \), where \( \alpha \) is a valid \( X_i \)-configuration and \( \sigma \) is a valid \( (Y^{\text{hid}}, X_i) \)-signature. The value of \( Q^i[\alpha, \sigma] \) is defined by

\[
Q^i[\alpha, \sigma] = \sum_{\beta} p_{\alpha \cup \beta}(Y^{\text{hid}}) \tag{3}
\]

where the summation is over all \( \beta \) such that \( \beta \) is a valid \( Y^{\text{hid}} \)-configuration and \( \text{sig}(\beta, X_i) = \sigma \).

Note that the definition of a tree decomposition ensures that every parent of a hidden node of \( i \) is either an explicit node or a hidden node of \( i \), so each probability value occurring in Equation (3) is well defined.

\( d \) Table \( R^i \) contains an entry for each pair in \( \Gamma_{Y^{\text{inh}}} \times H_{(Y^{\text{hid}} \cup Y^{\text{org}}), Y^{\text{inh}}} \).

Consider a given element of table \( R^i \), say \( R^i[\psi, \theta] \), where \( \psi \) is a valid \( Y^{\text{inh}} \)-configuration and \( \theta \) is a valid \( (Y^{\text{hid}} \cup Y^{\text{org}}, Y^{\text{inh}}) \)-signature. The value of \( R^i[\psi, \theta] \) is defined by

\[
R^i[\psi, \theta] = \sum_{\beta} p_{\psi \cup \beta}(Y^{\text{hid}} \cup Y^{\text{org}}) \tag{4}
\]

where the summation is over all \( \beta \) such that \( \beta \) is a valid \( (Y^{\text{hid}} \cup Y^{\text{org}}) \)-configuration and \( \text{sig}(\beta, Y^{\text{inh}}) = \theta \).

Note that the definition of a tree decomposition ensures that every parent of an hidden or originating node of \( i \) is either an explicit node or a hidden node of \( i \), so each probability value occurring in Equation (4) is well defined.

Equation (3) represents the definition of each entry of \( Q^i \). However, for a given \( \alpha \) and \( \sigma \), one cannot use the equation directly to efficiently compute the value of \( Q^i[\alpha, \sigma] \), since the number of valid configurations to be considered may be exponential in the number of nodes of \( G \). A similar comment applies to the computation of \( R^i[\psi, \sigma] \) directly using Equation (4). How these values can be computed efficiently is discussed below.
Table 1: Notation used in Describing the Dynamic Programming Algorithm for the PROB Problem

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_i$</td>
<td>The set of explicit nodes of tree node $i$.</td>
</tr>
<tr>
<td>$Y_i^{inh}$</td>
<td>The set of inherited nodes of tree node $i$.</td>
</tr>
<tr>
<td>$Y_i^{org}$</td>
<td>The set of originating nodes of tree node $i$.</td>
</tr>
<tr>
<td>$Y_i^{hid}$</td>
<td>The set of hidden nodes of tree node $i$.</td>
</tr>
<tr>
<td>$\Gamma_X$</td>
<td>The set of valid partial configurations on the explicit nodes of $i$.</td>
</tr>
<tr>
<td>$H_{Y_i^{hid},X_i}$</td>
<td>The set of valid partial configurations on the inherited nodes of $i$.</td>
</tr>
<tr>
<td>$H_{(Y_i^{hid} \cup Y_i^{org}),Y_i^{inh}}$</td>
<td>The set of signatures of all valid partial configurations of the hidden nodes of $i$ with respect to the explicit nodes of $i$.</td>
</tr>
<tr>
<td>$Q^i[\Gamma_{X_i},H_{Y_i^{hid},X_i}]$</td>
<td>$Q^i[\alpha,\sigma]$ maps valid partial configuration $\alpha$ on the explicit nodes of $i$ and signature $\sigma \in H_{Y_i^{hid},X_i}$ to a probability value.</td>
</tr>
<tr>
<td>$R^i[\Gamma_{Y_i^{inh}},H_{(Y_i^{hid} \cup Y_i^{org}),Y_i^{inh}}]$</td>
<td>$R^i[\psi,\theta]$ maps valid partial configuration $\psi$ on the inherited nodes of $i$ and signature $\theta \in H_{(Y_i^{hid} \cup Y_i^{org}),Y_i^{inh}}$ to a probability value.</td>
</tr>
</tbody>
</table>

of the signature sets and tables for each node of the tree decomposition. We present the construction in the following order.

1. First, we describe the computation of the set $H_{Y_i^{hid},X_i}$ and the $Q^i$ table for a leaf node $i$ of the tree decomposition.

2. Next, we describe the computation of set $H_{(Y_i^{hid} \cup Y_i^{org}),Y_i^{inh}}$ and the $R^i$ table for an arbitrary node $i$ of the tree decomposition, given set $H_{Y_i^{hid},X_i}$ and table $Q^i$.

3. Then we describe the computation of set $H_{Y_i^{hid},X_i}$ and table $Q^i$ for a nonleaf node $i$ of the tree decomposition, given the $H_{(Y_i^{org} \cup Y_i^{hid}),Y_i^{inh}}$ sets and $R$ tables for the children of node $i$ in the tree decomposition.

4. Finally, we indicate how the solution for the PROB problem can be computed from the values computed for the root of the tree decomposition.

We now present the details for each of the four parts above. The operations on signatures defined in Section 3.3 are used in the following description.

**Part 1:** Consider a leaf node $i$ of the tree decomposition. Note that leaf node $i$ contains no hidden nodes. Consequently, $H_{Y_i^{hid},X_i}$ consists of a single signature $\sigma$, which maps each node of $X_i$ into the value 0.

For each valid $X_i$–configuration $\alpha$, the table entry $Q^i[\alpha,\sigma]$ is given the value 1. Pseudocode for Part 1 is presented in Figure 4.

**Part 2:** For any node $i$ of the tree decomposition, given set $H_{Y_i^{hid},X_i}$ and table $Q^i$, set $H_{(Y_i^{hid} \cup Y_i^{org}),Y_i^{inh}}$ and table $R^i$ can be constructed as follows.

1. $H_{Y_i^{hid},X_i} = \{\sigma\}$, where $\sigma$ is the signature that maps each node $x \in X_i$ into the value 0.

2. For each valid partial configuration $\alpha$ on $X_i$, $Q^i[\alpha,\sigma] = 1$.

Figure 4: Pseudocode for Part 1 of the Algorithm for the PROB Problem

Recall that $\Gamma_{Y_i^{org}}$ denotes the set of all valid $Y_i^{org}$–configurations, and that for any $\gamma \in \Gamma_{Y_i^{org}}$, $sig(\gamma,Y_i^{inh})$ denotes the signature of $\gamma$ with respect to $Y_i^{inh}$.

**Computation of $H_{(Y_i^{hid} \cup Y_i^{org}),Y_i^{inh}}$:** This quantity is computed using the following equation

$$H_{(Y_i^{hid} \cup Y_i^{org}),Y_i^{inh}} = \bigcup_{\gamma,\sigma'} sig(\gamma,Y_i^{inh}) + (\sigma' | Y_i^{inh})$$

where the union is over each pair $\gamma, \sigma'$ such that $\gamma \in \Gamma_{Y_i^{org}}$ and $\sigma' \in H_{Y_i^{hid},X_i}$.

**Computation of $R^i$:** Consider an entry in the $Q^i$ table, say $Q^i[\alpha,\sigma]$. The valid $X_i$–configuration $\alpha$ can be considered to be the disjoint union of the valid $Y_i^{inh}$–configuration $\psi = \alpha | Y_i^{inh}$ and the valid $Y_i^{org}$–configuration $\gamma = \alpha | Y_i^{org}$. Similarly, the $(Y_i^{hid},X_i)$–signature $\sigma$ can be considered to be the disjoint union of the $(Y_i^{hid},Y_i^{inh})$–signature $\sigma' = \sigma | Y_i^{inh}$ and the $(Y_i^{hid},Y_i^{org})$–signature $\sigma'' = \sigma | Y_i^{org}$. Let $\theta$ be the $(Y_i^{hid} \cup Y_i^{org},Y_i^{inh})$–signature $\sigma' + sig(\gamma,Y_i^{inh})$. The entry $Q^i[\alpha,\sigma]$ of the $Q^i$ table contributes to the value of the entry $R^i[\psi,\theta]$ of the $R^i$ table. The value of this contribution is the product $Q^i[\alpha,\sigma] * p_{\alpha,\sigma}(Y_i^{org})$. 

708
The value $R^i[ψ, θ]$ can be computed using the following equation.

$$R^i[ψ, θ] = \sum Q^i[α, σ] \cdot p_{α,σ}(Y^\text{org}_i)$$

where the summation is over each $α \in Γ_x$ and $σ \in H_{Y^\text{hid}_i, X_i}$ such that $(ψ = α|Y^\text{inh}_i) \land (θ = (σ|Y^\text{inh}_i) + sig(α_i Y^\text{org}_i, Y^\text{inh}_i))$.

Alternatively, the $R^i$ table can be computed by first setting all the entries in the table to zero, and then scanning the $Q^i$ table, adding the contribution of each entry in the $Q^i$ table to the appropriate entry in the $R^i$ table. Pseudocode for Part 2, using this approach to computing the $R^i$ table, is shown in Figure 5.

**Computation of $H_{Y^\text{hid}_i \cup Y^\text{org}_i, Y^\text{inh}_i}$:**

1. Initialization: $H_{Y^\text{hid}_i \cup Y^\text{org}_i, Y^\text{inh}_i} = \emptyset$.

2. for each valid $Y^\text{org}_i$-configuration $γ$ do
   (i) Compute $σ = sig(γ, Y^\text{inh}_i)$, the signature of $γ$ with respect to $Y^\text{inh}_i$.
   (ii) for each signature $σ \in H_{Y^\text{hid}_i, X_i}$ do
       (a) $σ'' = σ' + (σ|Y^\text{inh}_i)$.
       (b) $H_{Y^\text{hid}_i \cup Y^\text{org}_i, Y^\text{inh}_i} = H_{Y^\text{hid}_i \cup Y^\text{org}_i, Y^\text{inh}_i} \cup \{σ''\}$.

**Computation of $R^i$:**

for each valid $Y^\text{inh}_i$-configuration $ψ$ do
   for each signature $θ ∈ H_{Y^\text{hid}_i \cup Y^\text{org}_i, Y^\text{inh}_i}$ do
      $R^i[ψ, θ] = 0$.

for each valid $X_i$-configuration $α$ do
   1. $ψ = α|Y^\text{inh}_i$.
   2. for each signature $σ \in H_{Y^\text{hid}_i, X_i}$ do
      (a) Compute $θ = sig(α|Y^\text{org}_i, Y^\text{inh}_i)$, the signature of $α|Y^\text{org}_i$ with respect to $Y^\text{inh}_i$.
      (b) $θ = (σ|Y^\text{inh}_i) + θ'$.
      (c) $R^i[ψ, θ] = R^i[ψ, θ] + Q^i[α, σ] \cdot p_{α,σ}(Y^\text{org}_i)$.

**Figure 5:** Pseudocode for Part 2 of the Algorithm for the PROB Problem

**Part 3:** We now consider computing set $H_{Y^\text{hid}_i, X_i}$ and table $Q^i$ for a nonleaf node $i$ of the tree decomposition.

**Case 1:** Nonleaf node $i$ has only one child.

Let $i_1$ denote the child of $i$ in the tree decomposition. We compute $H_{Y^\text{hid}_i, X_i}$ as

$$H_{Y^\text{hid}_i, X_i} = \{ext(θ_1, X_i) \mid θ_1 \text{ is in } H_{Y^\text{hid}_1 \cup Y^\text{org}_1, Y^\text{inh}_1}\}.$$  

Given the table $R^{i_1}$ for $i_1$, the table $Q^i$ is constructed as follows. Consider a given entry $Q^i[α, σ]$, for $X_i$-configuration $α$ and $(Y^\text{hid}_i, X_i)$-signature $σ$. The value of this entry is set to the value of $R^{i_1}[α|Y^\text{inh}_i, σ|Y^\text{inh}_i]$.  

**Case 2:** Nonleaf node $i$ has two children.

Let $i_1$ and $i_2$ denote the children of $i$ in the tree decomposition. We compute $H_{Y^\text{hid}_i, X_i}$ as

$$H_{Y^\text{hid}_i, X_i} = \bigcup_{θ_1, θ_2} ext(θ_1, X_i) + ext(θ_2, X_i).$$

The tables $R^{i_1}$ and $R^{i_2}$ for tree nodes $i_1$ and $i_2$ are combined to produce table $Q^i$ for tree node $i$ as follows. For any $θ_i \in H_{Y^\text{hid}_i \cup Y^\text{org}_i, Y^\text{inh}_i}$ and $θ_2 \in H_{Y^\text{hid}_2 \cup Y^\text{org}_2, Y^\text{inh}_2}$, let

$$σ = ext(θ_1, X_i) + ext(θ_2, X_i).$$

For any valid $X_i$-configuration $α$, the table entries $R^{i_1}[α|Y^\text{inh}_i, θ_1]$ and $R^{i_2}[α|Y^\text{inh}_i, θ_2]$ together contribute to the value of $Q^i[α, σ]$. The value of this contribution is $R^{i_1}[α|Y^\text{inh}_i, θ_1] + R^{i_2}[α|Y^\text{inh}_i, θ_2]$.

Consider a given valid $X_i$-configuration $α$ and signature $σ$ in $H_{Y^\text{hid}_i, X_i}$. We can compute $Q^i[α, σ]$ as a sum of products:

$$Q^i[α, σ] = \sum_{θ_1, θ_2} R^{i_1}[α|Y^\text{inh}_i, θ_1] \cdot R^{i_2}[α|Y^\text{inh}_i, θ_2]$$

where the summation is over all pairs $θ_1$ and $θ_2$ such that $θ_1 \in H_{Y^\text{hid}_1 \cup Y^\text{org}_1, Y^\text{inh}_1}$, $θ_2 \in H_{Y^\text{hid}_2 \cup Y^\text{org}_2, Y^\text{inh}_2}$ and $σ = ext(θ_1, X_i) + ext(θ_2, X_i)$.

Alternatively, the $Q^i$ table can be computed by first setting all the entries in the table to zero, and then scanning the $R^{i_1}$ and $R^{i_2}$ tables, adding the contribution of each pair of entries in these tables to the appropriate entries in the $R^i$ table. Pseudocode for Part 3, using this approach to computing the $Q^i$ table, is shown in Figure 6.

**Part 4:** Let $r$ be the root node of the tree decomposition. The root node has no inherited nodes, so $Y^\text{inh}_r = \emptyset$. Consequently, $Γ_{Y^\text{inh}_r}$ contains only the empty partial configuration, which we denote as $ψ_0$. Also, set $H_{Y^\text{org}_r \cup Y^\text{hid}_r, Y^\text{inh}_r}$ contains only the empty signature, which we denote as $σ_0$. Table $R^r$ consists of a single entry, $R^r[ψ_0, σ_0]$. The value of $R^r[ψ_0, σ_0]$ is the solution to the PROB problem.

### 3.4.3 Running Time Analysis

We now state a result that gives the running time of the algorithm presented in the previous section. A proof of this result appears in Rosenkrantz et al. (2014).

**Lemma 3.4** The dynamic programming algorithm for the PROB problem runs in $O(n^{2k+3})$ time, where $n$ is the number of nodes in the given Bayesian network $G$ and $k$ is the treewidth of $G$.  

709
Case 1: Node $i$ has only one child $i_1$ in the tree decomposition.

**Computation of $H_{Y_i^{hid},X_i}$:**

1. Initialization: $H_{Y_i^{hid},X_i} = \emptyset$.
2. For each signature $\theta_1 \in H_{(Y_i^{hid} \cup Y_i^{org})},Y_i^{inh}}$ do
   $H_{Y_i^{hid},X_i} = H_{Y_i^{hid},X_i} \cup \{ext(\theta_1, X_i)\}$.

**Computation of $Q^i$:**

For each valid $X_i$-configuration $\alpha$ do
   For each signature $\sigma \in H_{Y_i^{hid},X_i}$ do
     $Q^i[\alpha, \sigma] = R_i[\alpha | Y_i^{inh}, X_i]$.

Case 2: Node $i$ has two children $i_1$ and $i_2$ in the tree decomposition.

**Computation of $H_{Y_i^{hid},X_i}$:**

1. Initialization: $H_{Y_i^{hid},X_i} = \emptyset$.
2. For each signature $\theta_1 \in H_{(Y_i^{hid} \cup Y_i^{org})},Y_i^{inh}}$ do
   For each signature $\theta_2 \in H_{(Y_i^{hid} \cup Y_i^{org})},Y_i^{inh}}$ do
     (a) $\sigma = ext(\theta_1, X_i) + ext(\theta_2, X_1)$.
     (b) $H_{Y_i^{hid},X_i} = H_{Y_i^{hid},X_i} \cup \{\sigma\}$.

**Computation of $Q^i$:**

For each valid $X_i$-configuration $\alpha$ do
   For each signature $\sigma \in H_{Y_i^{hid},X_i}$ do
     $Q^i[\alpha, \sigma] = 0$.

For each signature $\theta_1 \in H_{(Y_i^{hid} \cup Y_i^{org})},Y_i^{inh}}$ do
   For each signature $\theta_2 \in H_{(Y_i^{hid} \cup Y_i^{org})},Y_i^{inh}}$ do
     (a) $\sigma = ext(\theta_1, X_i) + ext(\theta_2, X_i)$.
     (b) For each valid $X_i$-configuration $\alpha$ do
       $Q^i[\alpha, \sigma] = Q^i[\alpha, \sigma] + R_i[\alpha | Y_i^{inh}, \theta_1]$.
       $R_i[\alpha | Y_i^{inh}, \theta_2]$.

Figure 6: Pseudocode for Part 3 of the Algorithm for the PROB Problem

Since $k$ is fixed, our algorithm for the PROB problem runs in polynomial time. Thus, the following theorem summarizes the main result of Section 3.4.

**Theorem 3.5** The PROB problem can be solved efficiently for the class of treewidth-bounded BNs where each CPT is specified as a symmetric function.

---

**4 HARDNESS RESULTS**

The results in the previous section show that inference problems for treewidth bounded BNs can be solved efficiently even when the indegrees of nodes are not bounded, provided each CPT is expressed as a symmetric function. The following result points out the tightness of these results; in particular, the result shows that the problems remain computationally intractable even if one of the conditions is violated. A proof of this result appears in Rosenkrantz et al. (2014).

**Proposition 4.1** (a) If CPTs are not required to be $r$-symmetric, then the PROB problem is #P-hard even when the BN is a directed tree (whose treewidth is 1).

(b) When the treewidth of the BN is not bounded, the PROB problem is #P-hard even when the CPT at each node is given by a symmetric function.

---

**5 CONCLUSIONS**

We presented efficient algorithms for exact inference problems for BNs when the unmoralized graph is treewidth-bounded and each CPT is an $r$-symmetric function for a fixed $r$. We also observed that if either of these conditions is relaxed, the inference problems are computationally intractable.

We conclude by mentioning two general directions for further research. First, dynamic programming algorithms for treewidth-bounded BNs require memory that grows exponentially with the treewidth. It will be useful to develop practical techniques that can significantly reduce the amount of memory needed. Second, it is of interest to identify additional restrictions on BNs (based on problem instances that arise in practice) that can lead to practical algorithms for large problem instances.

**Acknowledgments**

We thank the UAI 2014 reviewers for providing helpful comments. This work has been partially supported by the following grants: DTRA Grant HDTRA1-11-1-0016, DTRA CNIMS Contract HDTRA1-11-D-0016-0010, NSF Career CNS 0845700, NSF ICES CCF-1216000, NSF NETSE Grant CNS-1011769 and DOE DE-SC0003957.
References


