Efficient Bayesian Inference by Factorizing Conditional Probability Distributions

Marcel A. J. van Gerven

Institute for Computing and Information Sciences, Radboud University
Toernooiveld 1, 6525 ED Nijmegen, The Netherlands
E-mail: marcelge@cs.ru.nl

Abstract

Bayesian inference becomes more efficient when one makes use of the structure that is contained within the conditional probability tables that together constitute a joint probability distribution over a set of discrete random variables. Such structure can be represented in the form of probability trees or Boolean polynomials. However, in order to make use of such representations in Bayesian inference, one needs to be able to represent them compactly within the inference engine. Recently, it was shown that potentials that exhibit functional dependence can be factorized efficiently by means of the introduction of hidden variables. Here we demonstrate how these techniques can be extended to represent compactly arbitrary distributions that do not exhibit this functional dependence.

Key words: Bayesian networks, inference, context-specific independence, probability trees, Boolean polynomials

1 Introduction

Efficient methods exist for factorizing conditional probability tables that together constitute a joint probability distribution over a set of discrete random variables in a Bayesian network [1,14]. These factorizations make use of the notion of context-specific independence and represent the conditional probability table by means of a set of smaller conditional probability tables that are directly representable within a Bayesian network structure. Recently, Vomlel, inspired by the work of Díez on the representation of the noisy max model has derived efficient factorizations for conditional probability tables that exhibit functional dependence [5,13]. In this paper we will exploit this factorization method in order to represent compactly conditional probability tables that do not necessarily contain this functional dependence structure. The method

relies on an extension of the factorization method that allows for a compact representation of a conditional probability table in the form of decision trees or Boolean polynomials. We show how the extension can be used to represent arbitrary conditional probability distributions and how this may lead to more efficient probabilistic inference in general.

The structure of this paper is as follows. In section 2 we introduce Bayesian networks, discuss probabilistic inference and review the factorization method for distributions that exhibit functional dependence. In section 3 we discuss the notion of context-specific independence that underlies the compact representation of arbitrary conditional probability distributions, examine the use of probability trees in this context and discuss the efficiency of the derived approach. Here, we will also discuss two alternative means of utilizing the factorization method, based on the factorization of multiplexer nodes and based on the use of Boolean polynomials. We end with some conclusions regarding the application of the factorization method in Bayesian network inference.

2 Preliminaries

Bayesian networks (also called belief networks or probabilistic networks), were introduced by Pearl in the 1980's [9] and provide for a compact factorization of a joint probability distribution over a set of random variables by exploiting the notion of *conditional independence*. In the following we discuss the necessary preliminaries.

2.1 Bayesian networks and probabilistic inference

Let P be a joint probability distribution defined over a set of random variables $\mathbf{X}$. Let $\mathbf{U}, \mathbf{V}, \mathbf{W} \subseteq \mathbf{X}$ be disjoint subsets of $\mathbf{X}$. Then, $\mathbf{U}$ is said to be conditionally independent of $\mathbf{V}$ given $\mathbf{W}$, denoted by $\mathbf{U} \perp\!\!\!\perp_P \mathbf{V} \mid \mathbf{W}$, iff

$$P(\mathbf{U} \mid \mathbf{V}, \mathbf{W}) = P(\mathbf{U} \mid \mathbf{W}).$$

Conditional independence can be represented by an acyclic directed graph (ADG) G consisting of vertices $V(G)$ and arcs $A(G)$ and relies on the notion of *d-separation* [9], where $\mathbf{U} \perp\!\!\!\perp_G \mathbf{W} \mid \mathbf{Z}$ denotes the d-separation of $\mathbf{U}$ and $\mathbf{W}$ by $\mathbf{Z}$ in the ADG G.

Let G be an ADG and P a joint probability distribution over a set of random variables $\mathbf{X} = \{X_1, \ldots, X_n\}$. We assume that there is a one-to-one correspondence between the vertices $V(G)$ and random variables $\mathbf{X}$. In general, we will use X_i to refer to the random variable that corresponds to a vertex i. We use

$X_{\mathbf{U}}$ to refer to the set of random variables $\{X_i \mid i \in \mathbf{U}, \mathbf{U} \subseteq V(G)\}$. The *parents* of a vertex $v \in V(G)$ is the set $\pi_G(v) = \{v' \mid (v', v) \in A(G)\}$ whereas the *children* of a vertex $v \in V(G)$ is the set $\rho_G(v) = \{v' \mid (v, v') \in A(G)\}$. We omit G when clear from context.

A *Bayesian network* is defined as a pair $\mathbf{B} = (G, P)$, such that

$$\mathbf{U} \perp\!\!\!\perp_G \mathbf{W} \mid \mathbf{Z} \Rightarrow \mathbf{U} \perp\!\!\!\perp_P \mathbf{W} \mid \mathbf{Z}. \tag{1}$$

If this property holds for G then we say that G is a *(directed) independence map* (I-map). The I-map property of G admits the following recursive factorization of the joint probability distribution:

$$P(\mathbf{X}) = \prod_{i \in V(G)} P(X_i \mid X_{\pi(i)}). \tag{2}$$

In the following, to simplify notation, we will use vertices $V(G)$ and random variables in $\mathbf{X}$ interchangeably, where the interpretation will be clear from context.

We will use lower-case letters x to denote an element of the sample space S_X of a random variable X. We will use bold-face lower-case letters $\mathbf{x}$ for an element $(x_1, \ldots, x_n) \in S_{\mathbf{X}}$ with $S_{\mathbf{X}} = S_{X_1} \times \cdots \times S_{X_n}$ for a set of random variables $\mathbf{X}$. We use the shorthand notation:

$$P(X_1 = x_1, \ldots, X_n = x_n \mid Y_1 = y_1, \ldots, Y_n = y_n) = P(\mathbf{x} \mid \mathbf{y}).$$

There exist various algorithms for probabilistic inference in Bayesian networks, which can be distinguished into exact and approximate inference schemes. Clustering methods such as the junction-tree algorithm [6] are well-known and relatively efficient methods for exact probabilistic inference which rely on the transformation of a Bayesian network's acyclic digraph into a graph that is more suitable as a computational architecture. This transformation is depicted in Fig. 1 .

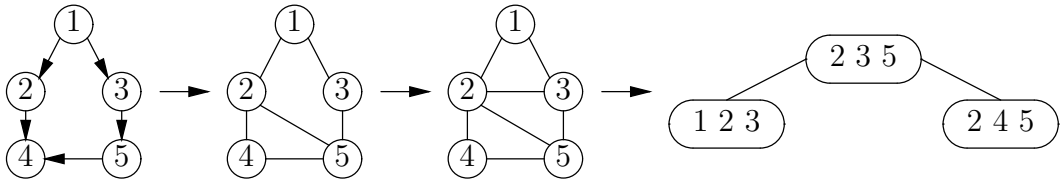

Fig. 1. Transforming an acyclic digraph to a moralized and triangulated undirected graph and finally into a junction tree that satisfies the running intersection property.

The transformation proceeds as follows. First, we *moralize* the graph, which means that we add an undirected edge between the parents of each vertex and convert the digraph into an undirected graph by dropping the direction of the arrows. Then, we *triangulate* the graph, which means that every simple cycle

with at least four vertices has at least one chord. The transformed graph now consists of a number of *cliques*. A *clique* is a subset of vertices in the graph such that the vertices and the arcs between them form a complete graph and no other vertices can be added such that the graph remains complete. When we are to apply join-tree propagation the cliques in this graph are connected in the form a tree. This tree should have the *running intersection property*, which means that any two vertex sets that are part of two cliques C_i and C_j must also be found in the cliques found on the chain between C_i and C_j.

For such clustering methods, probabilistic inference scales linearly with the size of the join-tree and exponentially with the clique size. For each clique C that contains variables $\mathbf{X} = \{X_1, \ldots, X_n\}$ a potential $\psi(X_1, \ldots, X_n)$ is constructed. The size of the potential is given by the cardinality $|S_{\mathbf{X}}|$ of the sample space $S_{\mathbf{X}}$. Consider for instance the two network structures in Fig. 2.

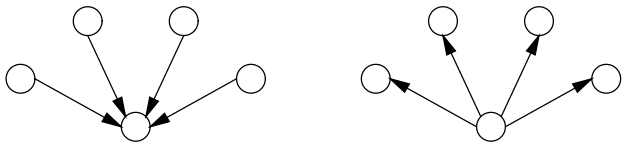

Fig. 2. Two Bayesian network structures.

After the transformation we obtain the two structures of Fig. 3.

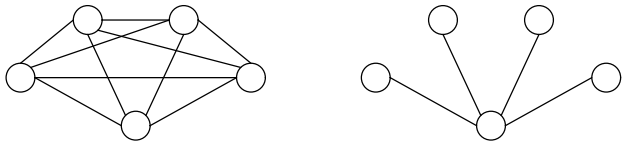

Fig. 3. The transformed graph.

For the left structure we have a clique size of 5 where the potential consists of n^5 elements, whereas for the right structure we have 4 cliques of size 2 where the potential consists of n^2 elements, for n-ary random variables. Hence, inference within the right structure is much more efficient. The left structure represents the relationship between a vertex and its parents, which form the basic elements in the factorization of a Bayesian network. In a generalization of [12,5] it was shown that if the probability distribution for this structure represents a functional relationship, then often a more efficient factorization exists, which relies on the introduction of a hidden variable [13].

2.2 Factorizations for functional dependence

Functional dependence is defined as follows.

Definition 1 *Let* $\mathbf{X} = \{X_1, \ldots, X_n\}$ *and let* ψ *be a probability potential defined on* $S_Y \times S_{X_1} \times \cdots \times S_{X_n}$. *We say that* Y *is* functionally dependent *on*

$\mathbf{X}$ *if there is a function* $f\colon S_{\mathbf{X}} \to S_Y$ *such that* $\psi(y, \mathbf{x}) = 1$ *if* $y = f(\mathbf{x})$ *and* $\psi(y, \mathbf{x}) = 0$ *otherwise.*

The transformation of a distribution that exhibits functional dependence to a more efficient form is based on selecting an appropriate basis $\mathbf{R}$ of *hyperrectangles* from $S_{\mathbf{X}}$.

Definition 2 *A* hyperrectangle R *is a set* $V_1 \times \cdots \times V_n$ *with* $\varnothing \neq V_i \subseteq S_{X_i}$.

This basis $\mathbf{R} = \{R_1, \ldots, R_m\}$ of hyperrectangles is associated with the sample space $S_B = \{b_1, \ldots, b_m\}$ of a hidden variable B that takes part in the factorization (Fig. 4).

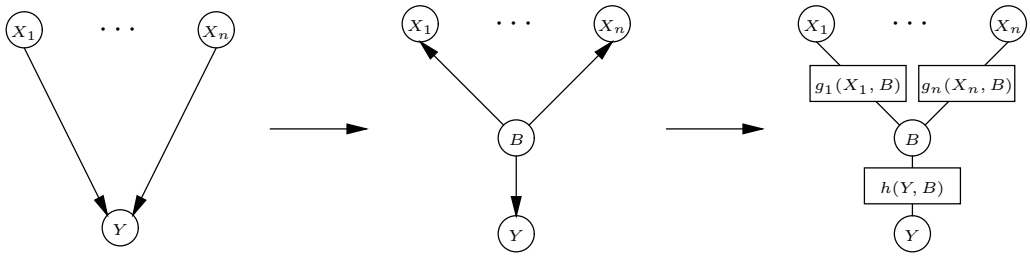

Fig. 4. Factorization of distributions that exhibit functional dependence.

From a basis $\mathbf{R}$, we may build legal expressions e using the following two operators.

Definition 3 *If* $\mathbf{X} \subseteq \mathbf{Y}$ *or* $\mathbf{Y} \subseteq \mathbf{X}$ *then the* proper difference *of* $\mathbf{X}$ *and* $\mathbf{Y}$ *is defined as* $\mathbf{X} \ominus \mathbf{Y} = (\mathbf{X} \setminus \mathbf{Y}) \cup (\mathbf{Y} \setminus \mathbf{X})$.

Definition 4 *Let* $\mathbf{X}, \mathbf{Y} \subseteq \mathbf{Z}$. *If* $\mathbf{X} \cap \mathbf{Y} = \varnothing$ *then the* disjunctive union *of* $\mathbf{X}$ *and* $\mathbf{Y}$ *is defined as* $\mathbf{X} \oplus \mathbf{Y} = (\mathbf{X} \setminus \mathbf{Y}) \cup (\mathbf{Y} \setminus \mathbf{X})$.

Definition 5 *A legal expression* e *is defined as follows:*

(1) $R \in \mathbf{R}$ *is a legal expression,*
(2) if e *and* e' *are legal expressions then* $(e \oplus e')$ *and* $(e \ominus e')$ *are legal expressions,*

We use legal expressions e_i to represent subsets $Y_i \subseteq S_{\mathbf{X}}$ such that $\mathbf{x} \in Y_i \Leftrightarrow f(\mathbf{x}) = y_i$. Hence, we require that each Y_i is the result of a legal expression e_i.

We use the notation e_{ij} to denote the outcome of the following simultaneous substitutions in the legal expression e_i:

- $\oplus$ is replaced by $+$.
- $\ominus$ is replaced by $-$.
- R_j is replaced by 1.
- R is replaced by 0 for all $R \neq R_j$.

In this way, we may express a potential ψ that exhibits functional dependence as follows:

$$\psi(y_i, \mathbf{x}) = \sum_{j=1}^{m} h(y_i, b_j) \prod_{u=1}^{n} g_u(x_u, b_j) \tag{3}$$

where

$$h(y_i, b_j) = e_{ij} \tag{4}$$

and

$$g_u(x_u, b_j) = \begin{cases} 1 & \text{if } \exists_{\mathbf{x}' \in R_j} \ [x'_u = x_u] \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

We will sometimes replace $\prod_{u=1}^{n} g_u(x_u, b_j)$ by the indicator function $g_j(\mathbf{x})$ that checks whether $\mathbf{x} \in R_j$, since both are equivalent. For further details we refer to [13]. Finding an efficient factorization for a conditional probability distribution that exhibits functional dependence now reduces to:

(1) finding a minimal basis $\mathbf{R}$ such that each Y_i can be expressed by a legal expression e_i,
(2) constructing $h(Y, B)$ according to Equation (4),
(3) constructing $g_u(X_u, B)$ with $1 \leq u \leq n$ according to Equation (5).

3 Factorizing arbitrary distributions

The factorization method described by Equation (3) only holds for distributions that exhibit functional dependence. However, most distributions do no exhibit such functional dependence and then the method can not be applied. In this paper, we show that we can benefit from the factorization method, even for distributions that do not exhibit functional dependence. This is accomplished by utilizing the local structure that is present within the conditional probability distributions.

3.1 A factorization method for arbitrary distributions

The structure of a Bayesian network's acyclic digraph can only capture independence relations that hold for every assignment $\mathbf{y}$ to the variables in Y. However, there may very well exist independencies that hold in particular contexts. This is formalized by the notion of *context-specific independence* [1].

Definition 6 *Suppose* $\mathbf{X}$, Y, $\mathbf{Z}$ *and* $\mathbf{C}$ *are disjoint sets of variables.* $\mathbf{X}$ *and* Y *are* contextually independent *given* $\mathbf{Z}$ *and a* context $\mathbf{c} \in S_{\mathbf{C}}$, *denoted by* $\mathbf{X} \perp\!\!\!\perp_P Y \mid \mathbf{Z}, \mathbf{c}$, *if*

$$P(\mathbf{X} \mid Y, \mathbf{Z}, \mathbf{c}) = P(\mathbf{X} \mid \mathbf{Z}, \mathbf{c})$$

whenever $P(Y, \mathbf{Z}, \mathbf{c}) > 0$.

We can adapt the factorization method for functional dependencies to accommodate probability distributions that exhibit such local structure. The method relies on the definition of less restrictive legal expressions, where $p_j R_j$ with $p_j \in [0,1]$ is also considered to be a legal expression. In this way, the potential $h(y_i, b_j)$ evaluates to either p_j or $-p_j$ depending on the structure of the legal expression e_i, such that we may write:

$$\psi(y_i, \mathbf{x}) = \sum_{j=1}^{m} e_{ij} \cdot g_j(\mathbf{x}) = k \cdot p_j \text{ with } k \in [-m, m]. \tag{6}$$

In this way, the potentials are no longer required to evaluate to either zero or one. In the following, we will use *probability trees* in order to show how the above method can be used to represent distributions that exhibit local structure. Although other canonical representations of probability distributions [14,4,2] can utilize Equation (6), probability trees are an often used and well-understood means for representing probability distributions.

3.2 Probability trees

In [1] probability trees are employed in order to represent context-specific independence. These trees provide for a concise representation of a conditional probability table.

Example 7 *Consider a conditional probability table for the distribution $P(Y \mid U, V, W, Z)$. If variables are binary then we need to specify 2^5 conditional probabilities. However, it may very well be the case that a considerable amount of structure is present in the table, which is represented by the probability tree in Fig. 5. This probability tree represents the context-specific independencies:*

- $\{Y\} \perp\!\!\!\perp_P \{V, W\} \mid u \wedge z$
- $\{Y\} \perp\!\!\!\perp_P \{W, Z\} \mid \neg u \wedge v$
- $\{Y\} \perp\!\!\!\perp_P \{Z\} \mid \neg u \wedge \neg v \wedge w$

The construction of a probability tree amounts to the construction of a decision tree, for instance using the ID3 algorithm [11], where instead of using the gain ratio, we use the conditional relative entropy:

$$\sum_{\mathbf{x}} P(y \mid \mathbf{x}) \log \frac{P(y \mid \mathbf{x})}{P_{\mathcal{T}}(y \mid \mathbf{x})}$$

between the actual distribution P and the distribution $P_{\mathcal{T}}$ represented by the tree $\mathcal{T}$ as the information measure [3]. Hence, we repeatedly add that node to the tree that minimizes this expression until some stopping criterion is satisfied.

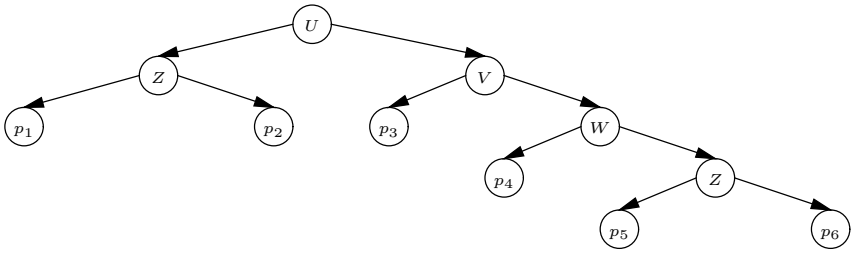

Fig. 5. A probability tree for $P(Y \mid U, V, W, Z)$.

The standard approach for representing a probability tree within a Bayesian network structure is to apply the following recursive decomposition. We decompose Y according to the parent X which is at the root of the tree (variable U in case of Example 7), construct new nodes Y_x for each $x \in S_X$ (variables Y_{u_0} and Y_{u_1} in case of Example 7) and create the distribution $P(Y \mid Y_{x_1}, \ldots, Y_{x_k}, X)$. The distributions $P(Y \mid Y_{x_1}, \ldots, Y_{x_k}, X)$ represent a so-called *multiplexer*, for which $P(y \mid y_{x_1}, \ldots, y_{x_k}, x_i) = 1$ if $y = y_{x_i}$ and 0 otherwise. For each of the newly constructed nodes Y_{x_i} we take the subtree that is reached by traversing from X according to the assumption that $X = x_i$ to be the new probability tree and repeat the decomposition. For our example, we would obtain the structure given in Fig. 6, which may then be used in order to perform probabilistic inference.

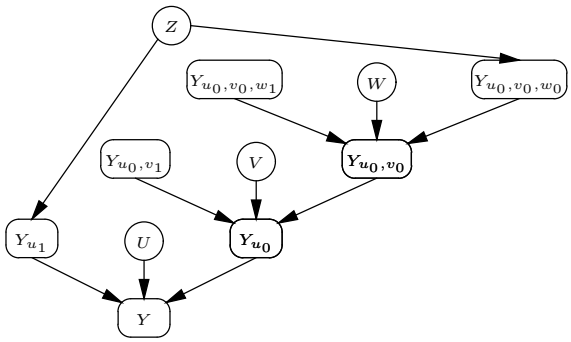

Fig. 6. A Bayesian network representation of a probability tree, where x_1 stands for *true* and x_0 stands for *false*.

For large conditional probability tables that contain considerable structure, inference will in general be faster due to the smaller clique sizes that are obtained. If we look at the structure of a probability tree then this structure can be interpreted as a set $\mathbf{F} = \{f_i \mid 1 \leq i \leq r\}$ of propositional formulas with which a set of probabilities $\mathbf{P} = \{p_f \mid f \in \mathbf{F}\}$ is associated.

Example 8 *The probability tree of Fig. 5 can be represented as the set* $\mathbf{F} = \{f_1, \ldots, f_6\}$ *where*

$$f_1 = u \wedge z \qquad f_3 = \neg u \wedge v \qquad f_5 = \neg u \wedge \neg v \wedge \neg w \wedge z$$
$$f_2 = u \wedge \neg z \qquad f_4 = \neg u \wedge \neg v \wedge w \qquad f_6 = \neg u \wedge \neg v \wedge \neg w \wedge \neg z$$

with probabilities $\mathbf{P} = \{p_{f_1}, \ldots, p_{f_6}\}$.

In the following we will show that we may use the factorization given by Equation (6) in order to represent probability trees more compactly.

3.3 Factorizing probability trees

The above example dealt with binary variables, but in general random variables are non-binary. In order to accommodate such variables, we generalize the preceding notation somewhat. We include the child variable Y within the representation of the probability tree. We assume that $|S_{X_i}| = m$ with $1 \leq i \leq n$ and $|S_Y| = k$. Figure 7 depicts a simple example for non-binary variables.

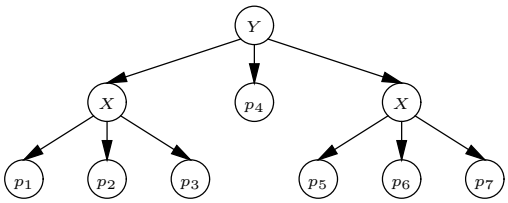

Fig. 7. A probability tree for Y with $k = 3$ and its parent X with $m = 3$.

Again, we represent the probability tree as a set of propositional formulas $\mathbf{F}$ and use

$$\mathbf{F}_y = \{f \mid f \in \mathbf{F}, f \vDash (Y = y)\}$$

to obtain those formulas where $Y = y$. We also select a state $y_d \in S_Y$ as the *distinguished state* and set $\mathbf{F}_{y_d} = \{\top\}$. We can use this element to represent compactly the largest probability tree $\mathbf{F}_y$ for some value $y \in S_Y$.

Due to the properties of a probability tree, each formula represents a hyperrectangle. We therefore construct for each formula $f \in \mathbf{F}_y$ a hyperrectangle

$$R_f = \{\mathbf{x} \mid f(\mathbf{x}) = \top\}$$

that represents the formula.

Example 9 *Returning to our running example, we assume that* $S_Y = \{0, 1\}$ *where* 0 *represents* false *and* 1 *represents* true *and set the distinguished element* $y_d = 0$ *such that we obtain* $R_\top = S_\mathbf{X}$ *and*

$$\begin{aligned}
R_{f_1} &= \{(1, v, w, 1) \mid (v, w) \in S_{\{V,W\}}\} & R_{f_4} &= \{(0, 0, 1, z) \mid z \in S_Z\} \\
R_{f_2} &= \{(1, v, w, 0) \mid (v, w) \in S_{\{V,W\}}\} & R_{f_5} &= \{(0, 0, 0, 1)\} \\
R_{f_3} &= \{(0, 1, w, z) \mid (w, z) \in S_{\{W,Z\}}\} & R_{f_6} &= \{(0, 0, 0, 0)\}
\end{aligned}$$

for $f_i \in \mathbf{F}_1$.

We will use $\mathbf{R}_y$ to denote the set of hyperrectangles that are constructed for a set of propositional formulas $\mathbf{F}_y$ such that the basis $\mathbf{R}$ equals $\bigcup_{y \in S_Y} \mathbf{R}_y$. Let $\bigoplus_{R \in \{R_1,\ldots,R_k\}}$ denote $R_1 \oplus \cdots \oplus R_k$. We now construct $Y_i = \bigoplus_{R_f \in \mathbf{R}_{y_i}} (p_f \cdot R_f)$ for $y_i \neq y_d$ and $Y_d = R_\top \ominus \left(\bigoplus_{y_i \in S_Y \setminus \{y_d\}} Y_i\right)$.

We then use the substitution of Equation (4) in order to obtain the new potentials h and g_i.

Example 10 *For our running example, we equate $R_\top$, R_{f_1}, …, R_{f_6} with states b_0, b_1, …, b_6 of B respectively. Potentials $h(Y, B)$, $g_1(U, B)$, $g_2(V, B)$, $g_3(W, B)$ and $g_4(Z, B)$ are then given by Tables 1 and 2 respectively.*

Table 1
Representation of $h(Y, B)$.

h	b_0	b_1	b_2	b_3	b_4	b_5	b_6
y_0	1	$-p_{f_1}$	$-p_{f_2}$	$-p_{f_3}$	$-p_{f_4}$	$-p_{f_5}$	$-p_{f_6}$
y_1	0	p_{f_1}	p_{f_2}	p_{f_3}	p_{f_4}	p_{f_5}	p_{f_6}

Table 2
Representation of $g_1(U, B)$, $g_2(V, B)$, $g_3(W, B)$ and $g_4(Z, B)$.

g_1	b_0	b_1	b_2	b_3	b_4	b_5	b_6	g_2	b_0	b_1	b_2	b_3	b_4	b_5	b_6
u_0	1	0	0	1	1	1	1	v_0	1	1	1	0	1	1	1
u_1	1	1	1	0	0	0	0	v_1	1	1	1	1	0	0	0
g_3	b_0	b_1	b_2	b_3	b_4	b_5	b_6	g_4	b_0	b_1	b_2	b_3	b_4	b_5	b_6
w_0	1	1	1	1	0	1	1	z_0	1	0	1	1	1	0	1
w_1	1	1	1	1	1	0	0	z_1	1	1	0	1	1	1	0

We may for instance compute $\psi(y_0, \mathbf{x})$ with $\mathbf{x} = (u_1, v_1, w_0, z_1)$ as follows:

$$\begin{aligned}
\psi(y_0, \mathbf{x}) &= h(y_0, b_0)g(\mathbf{x}, b_0) + h(y_0, b_1)g(\mathbf{x}, b_1) + h(y_0, b_2)g(\mathbf{x}, b_2) + \\
&\quad h(y_0, b_3)g(\mathbf{x}, b_3) + h(y_0, b_4)g(\mathbf{x}, b_4) + h(y_0, b_5)g(\mathbf{x}, b_5) + \\
&\quad h(y_0, b_6)g(\mathbf{x}, b_6) \\
&= 1 + -p_{f_1} \cdot 1 \cdot 1 \cdot 1 \cdot 1 + -p_{f_2} \cdot 1 \cdot 1 \cdot 1 \cdot 0 + -p_{f_3} \cdot 0 \cdot 1 \cdot 1 \cdot 1 + \\
&\quad -p_{f_4} \cdot 0 \cdot 0 \cdot 0 \cdot 1 + -p_{f_5} \cdot 0 \cdot 0 \cdot 1 \cdot 1 + -p_{f_6} \cdot 0 \cdot 0 \cdot 1 \cdot 0 \\
&= 1 - p_{f_1}
\end{aligned}$$

which agrees with the original distribution.

Due to the fact that the potentials may contain negative values, it is possible to represent the distinguished element in the manner described above.

3.4 Efficiency of factorizations

Let $\mathcal{J}$ be the junction tree that is built from the factorized representation of a probability tree $\mathcal{T}$. It is important to have some insight into the total clique size

$$tcs(\mathcal{J}) = \sum_{\mathcal{C} \in cliques(\mathcal{J})} \prod_{X \in \mathcal{C}} |S_X|.$$

that is required to accommodate $\mathcal{T}$. Let n be the number of parents nodes in the distribution, k the number of values that the child variable Y can assume and m the number of values that the parent variables X_1 to X_n can assume. Let α denote the number of leaf probabilities in the probability tree. For the direct representation of the tree in terms of the states of a hidden node, we have one clique of size $|S_Y| \cdot |S_B|$ and n cliques of size $|S_{X_i}| \cdot |S_B|$, such that we have a total clique size of

$$tcs(\mathcal{J}) = \alpha(nm + k),$$

compared with one clique of size km^n for the standard representation in terms of a CPT. Figure 8 depicts the clique sizes for the derived representation compared with the standard representation.

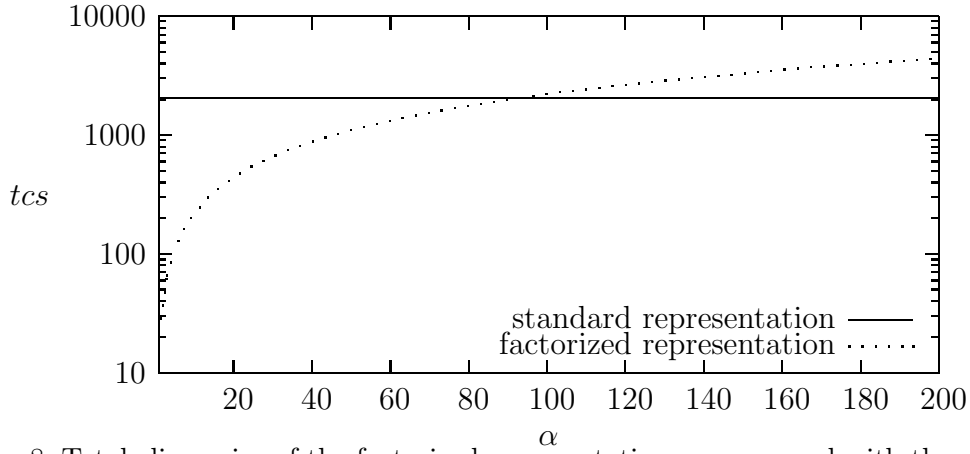

Fig. 8. Total clique size of the factorized representation as compared with the standard representation for $n = 10$, $k = m = 2$ and increasing values of α.

The figure shows that for small values of α (i.e. small probability trees) we can achieve substantially smaller total clique sizes.

A difficulty associated with the factorization method is the fact that due to the large sample spaces of the hidden variables the found cliques may become prohibitively large when the factorization method is used. Hence, triangulation algorithms must explicitly need to take into account the cardinality of the domains of random variables by using some kind of minimum size heuristic [7].

3.5 Alternative representations

The described factorization where we represent a probability tree directly in terms of the states of a hidden node is not the only way in which we can apply the factorization method to represent arbitrary distributions. In this section we will introduce two alternatives to the representation adopted in the above. The first alternative relies on using the factorization to represent the multiplexers that are used in the standard decomposition of a probability tree while the second alternative relies on the representation of arbitrary distributions by means of Boolean polynomials.

3.5.1 Factorizing multiplexer nodes

An alternative to the direct representation of a probability tree in terms of the states of the hidden nodes is to apply the factorization method to the multiplexer nodes that are used in the approach of [1]. These multiplexer nodes exhibit functional dependence such that we may apply the factorization as discussed in section 2.2. The goal then is to find a minimal basis $\mathbf{R}$ for arbitrary multiplexers.

We assume in the following that $|S_{X_i}| = m$ and $|S_Y| = 2$. The set $\mathbf{F}_1$, that represents those conditioning cases where $P(Y = 1 \mid Y_1, \ldots, Y_m, X) = 1$ can be written as:

$$\mathbf{F}_1 = \{(1, y_2, \ldots, y_m, x_1), \ldots, (y_1, \ldots, y_{m-1}, 1, x_m) \mid y_i \in S_Y, 1 \leq i \leq m\}.$$

We construct the following hyperrectangles in order to form a basis:

$$\begin{aligned} R_0 &= S_{Y_1} \times \cdots \times S_{Y_m} \times S_X \\ R_1 &= \{(1, y_2, \ldots, y_m, x_1) \mid y_i \in S_Y, 1 < i \leq m\} \\ &\vdots \\ R_m &= \{(y_1, y_2, \ldots, 1, x_m) \mid y_i \in S_Y, 1 \leq i < m\}, \end{aligned}$$

In this way, we can represent Y_1 by $R_1 \oplus \cdots \oplus R_m$ and Y_0 by $R_0 \ominus (R_1 \oplus \cdots \oplus R_m)$. The moralized graph of the network structure of Fig. 6 with (a) and without (b) the factorization is shown in Fig. 9.

We have a slightly more complex situation when a node Y_{x_i} that takes part in a multiplexer does not have parent nodes. In that case, the node represents some leaf probability p. However, here we can use Equation (6) in order to represent the leaf probability within the potentials, as is demonstrated in the following.

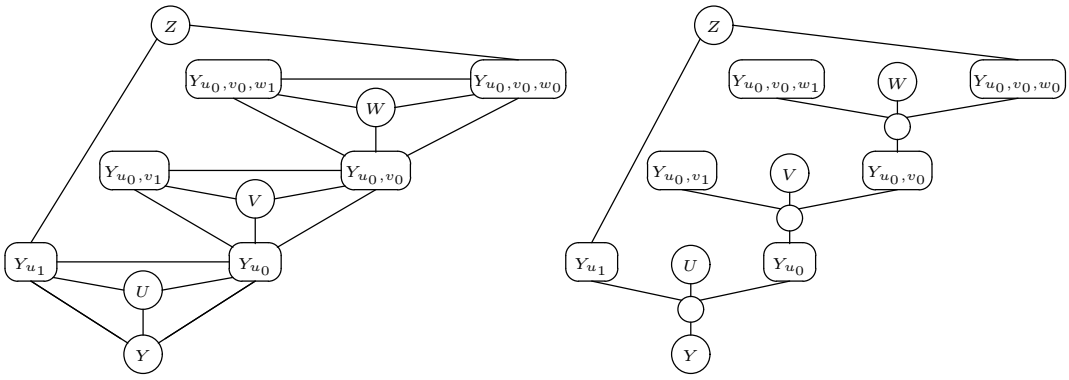

Fig. 9. Moralization of the graph of a probability tree, with (a) and without (b) the factorization of multiplexer nodes.

Table 3
Representation of $h(Y,B)$, $g_1(Y_{x_0},B)$, $g_2(Y_{x_1},B)$ and $g_3(X,B)$.

h	b_0	b_1	b_2	g_1	b_0	b_1	b_2
$Y=0$	1	-1	- p	$Y_{x_0}=0$	1	0	1
$Y=1$	0	1	p	$Y_{x_0}=1$	1	1	1
g_2	b_0	b_1	b_2	g_3	b_0	b_1	b_2
$Y_{x_1}=0$	1	1	0	$X=0$	1	1	0
$Y_{x_1}=1$	1	1	1	$X=1$	1	0	1

Example 11 *Consider a multiplexer node Y with $P(Y \mid Y_{x_0}, Y_{x_1}, X)$ and assume that $P(Y_{x_1}=1)=p$. We construct a basis $\mathbf{R}$ according to the previously described method with*

$$R_0 = \{0,1\}^3$$
$$R_1 = \{(1,0,0),(1,1,0)\}$$
$$R_2 = \{(0,1,1),(1,1,1)\}.$$

We associate the leaf probability p with the hyperrectangle R_2 and construct

$$Y_0 = R_0 \ominus (R_1 \oplus p \cdot R_2)$$
$$Y_1 = R_1 \oplus p \cdot R_2$$

such that the potentials h and the potentials g_i are given by Table 3. We may

then compute $\psi(Y = 1, Y_{x_0} = 0, Y_{x_1} = 1, X = 1)$ *as*

$$\begin{aligned}\psi(1,0,1,1) &= \textstyle\sum_{j=1}^{3} h(1,b_j)g_1(0,b_j)g_2(1,b_j)g_3(1,b_j) \\ &= h(1,b_0)g_1(0,b_0)g_2(1,b_0)g_3(1,b_0)+ \\ &\quad h(1,b_1)g_1(0,b_1)g_2(1,b_1)g_3(1,b_1)+ \\ &\quad h(1,b_2)g_1(0,b_2)g_2(1,b_2)g_3(1,b_2) \\ &= 0\cdot 1\cdot 1\cdot 1 + 1\cdot 0\cdot 1\cdot 0 + p\cdot 1\cdot 1\cdot 1 = p\end{aligned}$$

which agrees with the definition.

The approach followed in Example 11 implies that the clique sizes of multiplexer nodes remain the same, regardless of whether the node represents leaf probabilities.

In order to analyze the obtained total clique sizes let us focus on one multiplexer node. Without factorization we obtain one clique of size mk^{m+1} where $k = |S_Y|$. With factorization we have one potential $g(X, B)$, where the number of states is equal to $m \cdot |S_B|$. We also have m potentials $g_i(Y_i, B)$ with $1 \leq i \leq m$ and a potential $h(Y, B)$ consisting each of $k \cdot |S_B|$ states. The states of the hidden variable B correspond to the hyperrectangles $R_0, R_1, \ldots, R_m$, such that $|S_B|$ is equal to $m+1$. Therefore, the number of states of the $g_i(Y_i, B)$ and $h(Y, B)$ equals $k(m+1)$ and the number of states for the potential $g(X, B)$ equals $m(m + 1)$. Hence, the total clique size for a factorized representation of multiplexer nodes is given by $(m + 1)(k(m + 1) + m)$. This implies that we have a more compact representation of a multiplexer node in all cases save the situation that $k = m = 2$. This particular representation has the advantage that the we do not obtain hidden nodes with an exceedingly large number of states, provided that k and m are sufficiently small.

3.5.2 Boolean polynomials

Previously it was demonstrated that we can benefit from the factorization method if we represent conditional probability distributions in the form of a probability tree given sufficient local structure. Here, we introduce an alternative method that relies on the representation of the local structure in terms of Boolean polynomials. We will assume here that the random variables are binary, but we remark that the theory can be easily extended to the non-binary case. We consider the conditional probability distribution $P(Y \mid X_1, \ldots, X_n)$. As in the case of probability trees we require that there is redundancy in the sense that for some instantiations $\mathbf{x}, \mathbf{x}' \in S_{\mathbf{X}}$ with $\mathbf{X} = \{X_1, \ldots, X_n\}$ the conditional probabilities $P(y \mid \mathbf{x})$ and $P(y \mid \mathbf{x}')$ are equal. We use $p_{\mathbf{x}}$ to denote $P(y_1 \mid \mathbf{x})$.

The first step in the transformation is to group all configurations into k groups $G_1, \ldots, G_k$ with associated probabilities p_i such that for all $\mathbf{x} \in G_i$ it holds that $p_{\mathbf{x}} = p_i$ where $1 \leq i \leq k$ and for all p_i, p_j it holds that if $i < j$ then $p_i < p_j$. We then write each group in the form of a Boolean polynomial (a disjunction of conjunctions). This is accomplished by transforming each instantiation $\mathbf{x} \in G_i$ into a conjunction $l_1 \wedge \cdots \wedge l_n$ where $l_j = x_j$ if X_j is true in $\mathbf{x}$ and where $l_j = \neg x_j$ if X_j is false in $\mathbf{x}$, with $1 \leq i \leq k$ and $1 \leq j \leq n$. Given these Boolean polynomials F_i with associated probabilities p_i, $1 \leq i \leq k$, we construct $F'_i \equiv F_i \vee \cdots \vee F_k$ and $p'_i = p_i - p'_{i-1}$ where $p'_1 = p_1$.

The compact representation of a probability distribution now relies on the compact representation of the Boolean polynomials F'_i. This can be achieved by using well-known minimization procedures such as Quine-McCluskey minimization [10,8]. These procedures construct equivalent Boolean polynomials for which it holds that the number of conjunctions that are used in the polynomial is less than the number of conjunctions that are used in the original polynomial. One restriction to the minimization procedure is that the conjunctions used in the minimized polynomial are mutually exclusive since these conjunctions act as the hyperrectangles in the factorization. The general procedure for using this representation in order to attain a compact factorization of a probability distribution is as follows:

(1) Construct groups $G_1, \ldots, G_k$ with associated probabilities p_i for the distribution.
(2) Construct Boolean polynomials F'_i with associated probabilities p'_i from the representation F_i of G_i.
(3) Minimize the Boolean polynomials F'_i to equivalent polynomials H_i, subject to the constraint that the conjunctions in H_i are mutually exclusive.
(4) Let $C_{i1}, \ldots, C_{im_i}$ denote the conjunctions contained in H_i. For each conjunction C_{ij} construct a hyperrectangle $R_{ij} = \{\mathbf{x} \mid \mathbf{x} \vDash C_{ij}\}$ with $\mathbf{x} \in S_{\mathbf{X}}$ such that $R_i = \{R_{i1}, \ldots, R_{im_i}\}$.
(5) Construct legal expressions $e_1 = p_1 R_{11} \oplus \cdots \oplus p_k R_{km_k}$ and $e_0 = R_{11} \ominus e_1$ and create the potentials h and g_i with $1 \leq i \leq n$ in the standard way.

This procedure is guaranteed to construct a factorized representation of the original distribution.

Example 12 *Consider $P(Y, U, V, W)$, which is specified as in Table 4.*

For this table, we construct $F_1 \equiv c_1 \vee c_2 \vee c_5 \vee c_6$ with $p_1 = 0.1$, $F_2 \equiv c_3 \vee c_4 \vee c_7$ with $p_2 = 0.2$ and $F_3 \equiv c_8$ with $p_3 = 0.3$. Based on these Boolean polynomials we construct $F'_1 \equiv c_1 \vee \cdots \vee c_8$, $F'_2 \equiv c_3 \vee c_4 \vee c_7 \vee c_8$ and $F'_3 \equiv c_8$ with $p'_1 = p'_2 = p'_3 = 0.1$. After minimization we have the equivalent Boolean polynomials $H_1 \equiv \top$, $H_2 \equiv v$ and $H_3 \equiv u \wedge v \wedge w$. Now we only need to construct one

Table 4
CPT for $P(Y, U, V, W)$.

Configuration	U	V	W	$P(Y = 1 \mid U, V, W)$
c_1	0	0	0	0.1
c_2	0	0	1	0.1
c_3	0	1	0	0.2
c_4	0	1	1	0.2
c_5	1	0	0	0.1
c_6	1	0	1	0.1
c_7	1	1	0	0.2
c_8	1	1	1	0.3

hyperrectangle for each H_i, namely

$$
\begin{aligned}
R_{11} &= \{\mathbf{x} \mid \mathbf{x} \vDash \top\} = \{(u, v, w)\} \\
R_{21} &= \{\mathbf{x} \mid \mathbf{x} \vDash v\} = \{(u, 1, w)\} \\
R_{31} &= \{\mathbf{x} \mid \mathbf{x} \vDash u \wedge v \wedge w\} = \{(1, 1, 1)\}
\end{aligned}
$$

with $\mathbf{x} \in S_{\{U,V,W\}}$, such that

$$
\begin{aligned}
e_1 &= 0.1 \cdot R_{11} \oplus 0.1 \cdot R_{21} \oplus 0.1 \cdot R_{31} \\
e_0 &= R_{11} \ominus (0.1 \cdot R_{11} \oplus 0.1 \cdot R_{21} \oplus 0.1 \cdot R_{31}).
\end{aligned}
$$

The potentials are then as in Table 5.

Table 5
Representation of $h(Y, B)$, $g_1(U, B)$, $g_2(V, B)$ and $g_3(W, B)$.

h	b_1	b_2	b_3	g_1	b_1	b_2	b_3
y_0	0.9	0.9	0.9	u_0	1	1	0
y_1	0.1	0.1	0.1	u_1	1	1	1
g_2	b_1	b_2	b_3	g_3	b_1	b_2	b_3
v_0	1	0	0	w_0	1	1	0
v_1	1	1	1	w_1	1	1	1

In this way, we have again used the states of a hidden variable B in order to fully represent the original distribution. However, contrary to the representation that uses probability trees, we allow multiple states of B to take part in

the computation of the final probability. For instance, we may compute

$$\begin{aligned}\psi(y_1,(u_1,v_1,w_1)) = {} & 0.1 \cdot g((u_1,v_1,w_1),b_1)+ \\ & 0.1 \cdot g((u_1,v_1,w_1),b_2)+ \\ & 0.1 \cdot g((u_1,v_1,w_1),b_3) = 0.3,\end{aligned}$$

which agrees with the probability associated with configuration c_8.

4 Conclusions

In this paper, it has been demonstrated that the factorization method for functional dependencies can be extended to accommodate for arbitrary probability distributions. It has been shown that probability trees, a standard method for the compact representation of conditional probability distributions, can be represented directly in terms of the states of a hidden variable. Although substantial savings can be obtained when the factorization method is used we need to take into account the large number of states that the hidden variables may assume. The actual improvement is highly sensitive to the structure of the ADG underlying a Bayesian network as these hidden variables may become part of the same cliques.

The aim of this research was to demonstrate that the factorization of a probability distribution by means of a hidden variable can be used in principle to represent arbitrary probability distributions compactly. Next to the use of probability trees as the basis for the factorization method we have demonstrated the use of two alternative representations. One alternative is to use the factorization method to represent the multiplexer nodes that are used in the original representation of probability trees in a compact way. Another alternative is to represent distributions in the form of (minimal) Boolean polynomials subject to the constraint of mutual exclusiveness. We leave the comparison of the efficiency of these and various other representational schemes as a possible direction for future research.

Acknowledgements

We thank Jiří Vomlel for his valuable comments. This research was sponsored by the Dutch Science Foundation under grant number 612.066.201.

References

[1] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. Context-specific independence in Bayesian networks. In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence*, pages 115–123, San Francisco, CA, 1996. Morgan Kaufmann Publishers.

[2] Randal E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, August 1986.

[3] A. Cano and S. Moral. Propagación exacta y aproximada con árboles de probabilidad. In *Actas de la VII Conferencia de la Asociación Española para la Inteligencia Artificial*, pages 635–644, 1997.

[4] A. Darwiche. Decomposable negation normal form. *Journal of the ACM*, 48(4):608–647, 2001.

[5] F.J. Díez and S.F. Gálan. Efficient computation for the noisy max. *International Journal of Intelligent Systems*, 18(2):165–177, 2003.

[6] F. Jensen, S. Lauritzen, and K. Olesen. Bayesian updating in recursive graphical models by local computations. *Computational Statistics Quarterly*, 4:269–282, 1990.

[7] U. Kjaerulff. Triangulation of graphs - algorithms giving small total state space. Technical Report R 90-09, University of Aalborg, Denmark, 1990.

[8] E. McCluskey. Minimization of Boolean functions. *The Bell System Technical Journal*, 35:437–457, 1956.

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

[10] W. V. O. Quine. A way to simplify truth functions. *American Mathematical Monthly*, 62:627–631, 1955.

[11] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[12] M. Takikawa and B. D'Ambrosio. Multiplicative factorization of noisy-max. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence*, pages 622–630, San Francisco, CA, 1999. Morgan Kaufmann Publishers.

[13] J. Vomlel. Exploiting functional dependence in Bayesian network inference. In *Proceedings of The 18th Conference on Uncertainty in Artificial Intelligence*, pages 528–535, 2002.

[14] N. Zhang and D. Poole. Exploiting causal independence in Bayesian network inference. *Journal of Artificial Intelligence Research*, 5:301–328, 1996.