International Journal of Approximate Reasoning 52 (2011) 49–62

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

journal homepage: www.elsevier.com/locate/ijar

Approximate inference in Bayesian networks using binary probability trees

Andrés Cano *, Manuel Gómez-Olmedo, Serafín Moral

Dept. Computer Science and Artificial Intelligence, Higher Technical School of Computer Science and Telecommunications, University of Granada, Granada 18071, Spain

ARTICLE INFO

Article history:
Available online 15 June 2010

Keywords:
Bayesian networks inference
Approximate computation
Variable elimination algorithm
Deterministic algorithms
Probability trees

ABSTRACT

The present paper introduces a new kind of representation for the potentials in a Bayesian network: binary probability trees. They enable the representation of context-specific independences in more detail than probability trees. This enhanced capability leads to more efficient inference algorithms for some types of Bayesian networks. This paper explains the procedure for building a binary probability tree from a given potential, which is similar to the one employed for building standard probability trees. It also offers a way of pruning a binary tree in order to reduce its size. This allows us to obtain exact or approximate results in inference depending on an input threshold. This paper also provides detailed algorithms for performing the basic operations on potentials (restriction, combination and marginalization) directly to binary trees. Finally, some experiments are described where binary trees are used with the variable elimination algorithm to compare the performance with that obtained for standard probability trees.

1. Introduction

Bayesian networks are graphical models that can be used to handle uncertainty in probabilistic expert systems. They provide an efficient representation of joint probability distributions. It is known that exact computation [1] of the posterior probabilities, given certain evidence, may become unfeasible for large networks. As a consequence, improved algorithms and methods are continuously proposed to enable exact inference on larger Bayesian networks. For example, in [2] it is presented an alternative method for improving the time required for accessing the values stored in potentials (and producing substantial savings in computation time when performing combination, marginalization or addition operations on them); the paper in [3] describes some improvements to message computation in Lazy propagation. Unfortunately, even with these improvements inference on complex Bayesian networks may be still unfeasible. This has led to the proposal of different approximate algorithms. These algorithms provide results in shorter time, albeit inexact. Some of the methods are based on Monte Carlo simulation, and others rely on deterministic procedures. Some of the deterministic methods use alternative representations for potentials, such as *probability trees* [4–6]. This representation offers the possibility to take advantage of *context-specific independences*. Probability trees can be pruned and converted into smaller trees when potentials are too large, thus facilitating approximate algorithms. In the present paper, we introduce a new kind of probability trees in which the internal nodes always have two children. They will be called *binary probability trees*. These trees allow the specification of fine-grained context-specific independences in more detail than standard trees, and should work better than standard probability trees for Bayesian networks containing variables with a large number of states.

The remainder of this paper is organized as follows: in Section 2 we describe the problem of probability propagation in Bayesian networks. Section 3 explains the use of probability trees to obtain a compact representation of the potentials and

* Corresponding author. Tel.: +34 958 240803; fax: +34 958 243317.
E-mail addresses: acu@decsai.ugr.es (A. Cano), mgomez@decsai.ugr.es (M. Gómez-Olmedo), smc@decsai.ugr.es (S. Moral).

0888-613X/$ - see front matter
doi:10.1016/j.ijar.2010.05.006

presents the related notation. In Section 4, we introduce binary probability trees and describe the procedure to build them from a potential, and how they can be approximated by pruning terminal trees; we also show the algorithms for direct application of the basic operations with potentials to binary probability trees. These algorithms are very similar to the algorithms for performing operations in *mixed trees* (trees with continuous and discrete variables) [7]. Section 5 provides details of the experimental work. Finally, Section 6 gives the conclusions and future work.

2. Probability propagation in Bayesian networks

Let $\mathbf{X} = \{X_1, \ldots, X_n\}$ be a set of variables. Let us assume that each variable X_i takes values on a finite set of states Ω_{X_i} (the domain of X_i). We shall use x_i to denote one of the values of X_i, $x_i \in \Omega_{X_i}$. If I is a set of indices, we shall write $\mathbf{X}_I$ for the set $\{X_i | i \in I\}$. $N = \{1, \ldots, n\}$ will denote the set of indices of all the variables in the network; thus $\mathbf{X} = \mathbf{X}_N$. The Cartesian product $\times_{i \in I} \Omega_{X_i}$ will be denoted by $\Omega_{\mathbf{X}_I}$. The elements of $\Omega_{\mathbf{X}_I}$ are called configurations of $\mathbf{X}_I$ and will be represented as $\mathbf{x}_I$. We denote by $\mathbf{x}_I^{\downarrow \mathbf{X}_J}$ the *projection* of the configuration $\mathbf{x}_I$ to the set of variables $\mathbf{X}_J$, $\mathbf{X}_J \subseteq \mathbf{X}_I$.

A mapping from a set $\Omega_{\mathbf{X}_I}$ into $\mathbb{R}_0^+$ will be called a *potential* p for $\mathbf{X}_I$. Given a potential p, we denote by $s(p)$ the set of variables for which p is defined. The process of inference in probabilistic graphical models requires the definition of two operations on potentials: *combination* $p_1 \otimes p_2$ (multiplication) and *marginalization* $p^{\downarrow \mathbf{X}_J}$ (by summing out all the variables not in X_J). Given a potential p, we denote by sum (p) the addition of all the values of the potential p.

A *Bayesian network* is a directed acyclic graph, where each node represents a random event X_i, and the topology of the graph shows the independence relations between variables according to the d-separation criterion [8]. Each node X_i has a conditional probability distribution $p_i(X_i | \Pi(X_i))$ for that variable, given its parents $\Pi(X_i)$.

A Bayesian network determines a joint probability distribution:

$$p(\mathbf{x}) = \prod_{i \in N} p_i(x_i | \pi(x_i)) \quad \forall \mathbf{x} \in \Omega_{\mathbf{X}}, \tag{1}$$

where $\pi(x_i)$ is the configuration $\mathbf{x}$ marginalized on the parents of X_i: $\Pi(X_i)$.

Let $\mathbf{E} \subset \mathbf{X}_N$ be the set of observed variables and $\mathbf{e} \in \Omega_{\mathbf{E}}$ the instantiated value. An algorithm that computes the posterior distributions $p(x_i | \mathbf{e})$ for each $x_i \in \Omega_{X_i}$, $X_i \in \mathbf{X}_N \backslash \mathbf{E}$, is called a *propagation algorithm* or *inference algorithm*.

3. Probability trees

Probability trees [9] have been used as a flexible data structure that enables the specification of *context-specific independences* (see [6]) and provides exact or approximate representations of probability potentials. A *probability tree* $\mathcal{T}$ is a directed labelled tree, in which each internal node represents a variable and each leaf represents a non-negative real number. Each internal node has one outgoing arc for each state of the variable that labels that node; each state labels one arc. The *size* of a tree $\mathcal{T}$, denoted by *size*$(\mathcal{T})$, is defined as its node count.

A probability tree $\mathcal{T}$ on variables $\mathbf{X}_I = \{X_i | i \in I\}$ represents a potential $p : \Omega_{\mathbf{X}_I} \rightarrow \mathbb{R}_0^+$ if, for each $\mathbf{x}_I \in \Omega_{\mathbf{X}_I}$, the value $p(\mathbf{x}_I)$ is the number stored in the leaf node that is reached by starting from the root node and selecting the child corresponding to coordinate x_i for each internal node labelled X_i. We use L_t to denote the *label of node t* (a variable for an internal node, and a real number for a leaf). A subtree of $\mathcal{T}$ is a *terminal tree* if it contains only one node labelled with a variable name, and all the children are numbers (leaf nodes).

A probability tree is usually a more compact representation of a potential than a table, because it allows an inference algorithm to take advantage of context-specific independences. This is illustrated in Fig. 1, which displays a potential p and its representation, using a probability tree. This tree shows that the potential is independent of the value of A in the context $\{B = b_1, C = c_2\}$. The tree contains the same information as the table, but only requires five values, while the table contains eight values. Furthermore, trees enable even more compact representations. This is achieved by pruning certain leaves and replacing them with the average value, as shown in the second tree shown in Fig. 1. The tradeoff is a loss of accuracy.

If $\mathcal{T}$ is a probability tree on $\mathbf{X}_I$ and $\mathbf{X}_J \subseteq \mathbf{X}_I$, we use $\mathcal{T}^{R(\mathbf{x}_J)}$ (probability tree restricted to the configuration $\mathbf{x}_J$) to denote the *restriction operation* which consists of returning the part of the tree which is consistent with the values of the configuration $\mathbf{x}_J \in \Omega_{\mathbf{X}_J}$. For example, in the first probability tree shown in Fig. 1, $\mathcal{T}^{R(B=b_1, C=c_1)}$ represents the terminal tree enclosed by the dashed square. This operation is used to define combination and marginalization operations, as well as for conditioning.

A	B	C	p(A,B,C)
a_1	b_1	c_1	0.2
a_1	b_1	c_2	0.5
a_1	b_2	c_1	0.7
a_1	b_2	c_2	0.7
a_2	b_1	c_1	0.3
a_2	b_1	c_2	0.5
a_2	b_2	c_1	0.3
a_2	b_2	c_2	0.3

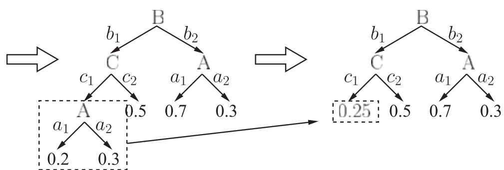

Fig. 1. Potential p, its representation as a probability tree and its approximation after pruning several branches.