Factorisation of Probability Trees and its Application to Inference in Bayesian Networks

Irene Martínez
Dept. Languages and Computation
University of Almería
La Cañada de San Urbano s/n
E-04120 Almería (Spain)
irene@ual.es

Serafín Moral
Dept. Computer Science and A.I.
University of Granada
Daniel Saucedo Arana s/n
E-18071 Granada (Spain)
smc@decsai.ugr.es

Carmelo Rodríguez and Antonio Salmerón
Dpt. Statistics and Applied Mathematics
University of Almería
La Cañada de San Urbano s/n
E-04120 Almería (Spain)
{crt,Antonio.Salmeron}@ual.es

Abstract

Bayesian networks can be seen as a factorisation of a joint probability distribution over a set of variables, based on the conditional independence relations amongst the variables. In this paper we show how it is possible to achieve a finer factorisation decomposing the orignal factors when certain conditions hold. The new ideas can be applied to algorithms able to deal with factorised probabilistic potentials, such as Lazy Propagation, Lazy-Penniless as well as Monte Carlo methods based on Importance Sampling.

1 Introduction

A Bayesian network can be understood as a representation of a multivariate probability distribution, such that the conditional independence relations induced by the distribution are encoded by the structure of the network, according to the d-separation criterion (Pearl, 1988). Usage of Bayesian networks commonly involve a task known as *probability propagation*, which consists of the computation of the posterior probability distribution for some variables of interest given that some other variables have been observed.

Several algorithms have been developed for efficient probability propagation, most of which rely on the use of an auxiliary structure called a *join tree* (Jensen et al., 1990; Lauritzen and Spiegelhalter, 1988; Madsen and Jensen, 1999; Shenoy, 1997; Shenoy and Shafer, 1990). However, these algorithms may become unfeasible if the size of the join tree obtained from the Bayesian network is too large. This fact has motivated the development of approximate algorithms able to provide estimations of the posterior probabilities in complex problems (Cano et al., 2000; Cano et al., 2002; Cheng and Druzdzel, 2000; Dagum and Luby, 1997; Fung and Chang, 1990; Hernández et al., 1998; Jensen and Andersen, 1990; Salmerón et al., 2000; Salmerón and Moral, 2001; Shachter and Peot, 1990).

It is known that probability propagation (exact and approximate) is an NP-hard problem (Cooper, 1990; Dagum and Luby, 1993). This justifies investing effort in the study of new algorithms with the aim of enlarging the class of affordable problems. The most recent advances in propagation have come along with algorithms that incorporate the ability of dealing with factorised representations of the potentials associated with the join tree. These algorithms are

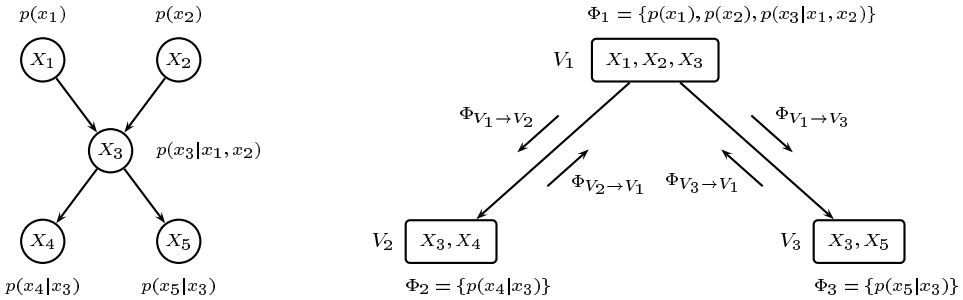

Figure 1: A Bayesian network and a join tree associated with it. Probability propagation is carried out sending messages throughout the edges.

Lazy propagation (Madsen and Jensen, 1999) and Lazy-penniless propagation (Cano et al., 2002).

The Lazy-penniless algorithm has the particular feature that it uses probability trees (Cano and Moral, 1997) to represent probabilistic potentials. Probability trees are usually more compact than probability tables and, what is more important, provide a flexible way to reduce the space required to store a probabilistic potential, by pruning some of the branches of the trees. Of course, it can happen that the resulting tree will be just an approximation of the original potential. In this paper we show how it is possible to achieve a factorisation finer than the one used in the Lazy and Lazy-penniless algorithms, and thus the efficiency of this kind of algorithms can be significantly increased.

From here onwards, the paper continues with a description of the relationship between Bayesian networks and probability trees in Section 2. Methods for factorisation are proposed in Section 3 and some aspects of its application in practice are considered in Section 4. The paper ends with conclusions in Section 5.

2 Bayesian networks and probability trees

A *Bayesian network* is defined as a directed acyclic graph where each node represents a random variable, and the topology of the graph encodes the independence relations among the variables, according to the d-separation criterion (Pearl, 1988). Along with the graph structure, a probability distribution is given for each node conditional on its parents, such that the joint distribution over all the variables in the network factorises as the product of the conditional distributions.

We will use the concept of potential to represent probabilistic information (including 'a priori', conditional and 'a posteriori' distributions). A *potential* ϕ for a set of variables $\mathbf{X}$ is a mapping $\phi : \Omega_{\mathbf{X}} \to I\!R_0^+$, where $\Omega_{\mathbf{X}}$ is the set of possible cases of the set of variables $\mathbf{X}$ and $I\!R_0^+$ is the set of non-negative real numbers. From now onwards, we will consider only discrete variables with a finite number of cases, and the *size* of a potential will be the highest number of values necessary to completely specify it; i.e. if ϕ is defined on $\Omega_{\mathbf{X}}$, its size is $|\Omega_{\mathbf{X}}|$.

Probability propagation is seldom carried out over the Bayesian network, but rather over an auxiliary structure called a join tree. A *join tree* is a tree where each node V is a subset of the variables in the network, and such that if a variable is on two distinct nodes, V_1 and V_2, then it is also on every node in the path between V_1 and V_2. Every potential in the original Bayesian network (i.e. every conditional distribution) is assigned to a node V_j containing the variables involved in the conditional distribution. A potential constantly equal to 1 (unity potential) is assigned to nodes which did not receive any conditional distribution. In this way, attached to every node V_i there will be a potential ϕ_{V_i} defined over the set of variables V_i and which is equal to the product of all the potentials as-

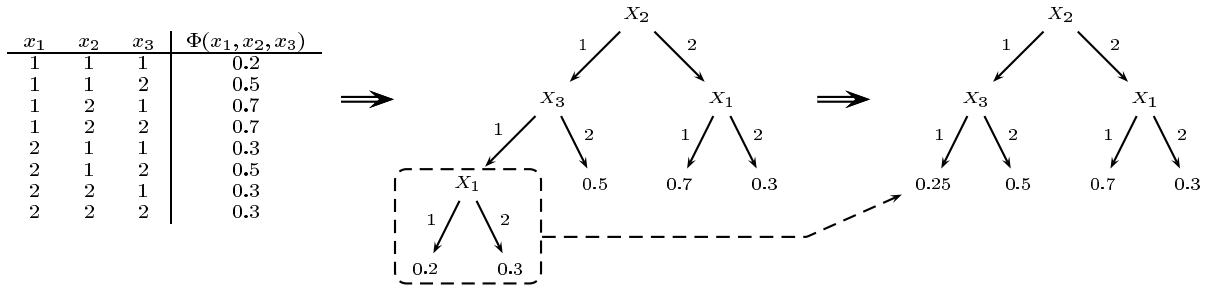

Figure 2: A potential ϕ, a probability tree representing it and an approximation of it after pruning some branches.

signed to it.

There are different ways to represent the potentials in the join tree (for instance, probability tables and probability trees) and it is possible to keep the potentials assigned to a node as a list instead of multiplying them initially (Madsen and Jensen, 1999; Cano et al., 2002). Figure 1 illustrates the process of probability propagation: A join tree (right side) is constructed from a Bayesian network (left side) and then propagation is carried out by a flow of messages through the edges of the join tree. Observe that the potentials in the nodes are kept as a list.

A message from one node V_i to one of its neighbours, V_j is a potential defined for the variables contained in $V_i \cap V_j$, and is obtained as the result of marginalising out from the potentials attached to V_i all the variables not in V_j. A variable is marginalised out by multiplying the potentials containing it and then summing the variable out. This is precisely the step in which the complexity of probability propagation arises: The domain of the potential resulting from the product mentioned above may become so large that a huge amount of memory would be necessary to store it.

In this paper we are concerned with the representation of probabilistic potentials by means of probability trees. We will introduce some factorisation techniques that can help to overcome this problem.

A *probability tree* (Boutilier et al., 1996; Cano and Moral, 1997; Salmerón et al., 2000) is a directed labeled tree, where each internal node represents a variable and each leaf node represents a probability value. Each internal node has one outgoing arc for each state of the variable associated with that node. Each leaf contains a non-negative real number. The *size* of a tree $\mathcal{T}$, denoted as $size(\mathcal{T})$, is defined as its number of leaves.

A probability tree $\mathcal{T}$ on variables $\mathbf{X}_I = \{X_i | i \in I\}$ represents a potential $\phi : \Omega_{\mathbf{X}_I} \rightarrow I\!R_0^+$ if for each $\mathbf{x}_I \in \Omega_{\mathbf{X}_I}$ the value $\phi(\mathbf{x}_I)$ 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 with X_i.

A probability tree is usually a more compact representation of a potential than a table. This fact is illustrated in Figure 2, which displays a potential ϕ and its representation using a probability tree. The tree contains the same information as the table, but using five values instead of eight. Besides, locating a value in a probability tree is more efficient that in a probability table. The complexity of this operation is $O(n)$ in the worst case (the same as in a probability table), where n is the number of variables in the potential. However, the complexity decreases as the tree is pruned and becomes $O(m)$ where m is the number of variables actually contained in the tree.

Furthermore, trees allow an even more compact representation in exchange of loosing accuracy. This is achieved by pruning some leaves and replacing them by the average value, as shown in the second tree in Figure 2. The ad-

vantage of this representation with respect to others is that pruning can be done following information criteria, as in decision trees. The order in which the nodes are placed in the tree determines the efficiency of the pruning (the most informative variables should be placed at the top of the tree). This issue was discussed in (Salmerón et al., 2000).

The basic operations (*combination* and *marginalisation*) over potentials required for probability propagation can be carried out directly over probability trees, as described in the next algorithms.

The combination is done recursively and basically consists of selecting a starting node and multiplying each of its children by the other tree. This operation is illustrated in Figure 3, and the details of the algorithm can be found in (Cano et al., 2000).

Marginalisation is equivalent to summing out variables. A variable is summed out from a probability tree by replacing it by the sum of its children, as described in Figure 4. The details can be found in (Cano et al., 2000).

3 Factorisation of Probability Trees

Consider the situation in which we are about to marginalise out a variable X_i. The first step is to multiply the potentials (probability trees in this case) containing X_i. A gain in efficiency could be achieved if we managed to decompose each tree containing X_i as a product of two trees of smaller size, one of them containing X_i and the other not containing it. Then, the product would be actually carried out over potentials (trees) with reduced domains and thus, the complexity of probability propagation would decrease. Clearly, it would only be true if the propagation algorithm is able to deal with lists of potentials, as Lazy propagation and Lazy-penniless do. In the next subsections we consider two situations in which such a decomposition is feasible.

3.1 Tree splitting

Assume that probability propagation is being carried out and that Y is the next variable to marginalise out, and that it is contained in a potential represented by the tree in the left side of Figure 5. Observe that Y is in the sub-tree corresponding to the first case of variable X, but not in the sub-tree corresponding to the second case. This is a very common situation in Lazy-penniless propagation, where it is possible that a variable disappears from a part of a tree after a pruning.

This fact allows to decompose the original tree as the product of two smaller trees, as displayed in Figure 5. The advantage of this decomposition is that the second tree in the decomposition does not take part in the product previous to marginalising out Y, because it does not contain Y, and the first factor is more simple than the original tree; Therefore, the complexity of the deletion of variable Y is reduced and thus the efficiency of Lazy propagation increased.

3.2 Proportional sub-trees

Now assume that the next variable to marginalise out is X, and we find it in the tree shown in Figure 6. We can see that, within context $W = 0$, the two children of X are proportional. In this case, it is possible to factorise the tree as a product of two trees smaller than the original one (see Figure 7), in such a way that one of the factors keeps the information affected by X and the other contains the information not affected by X. More formally, trees that can be factorised in this way can be characterised by the next definition.

Definition 1 *Let $\mathcal{T}$ be a probability tree. Let $(\mathbf{X}_C = \mathbf{x}_C)$ be a configuration of variables leading from the root node in $\mathcal{T}$ to a variable X_i. We say that $\mathcal{T}$ is* proportional below X within context $(\mathbf{X}_C = \mathbf{x}_C)$ *if there is a $x_i \in \Omega_X$ such that for every $x_i \neq x_j \in \Omega_X$, $\exists \alpha_j > 0$ such that*

$$\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_i)} = \alpha_j \cdot \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_j)} \ , \quad (1)$$

where $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x)}$ denotes the sub-tree of $\mathcal{T}$ reached following the path determined by configuration $(\mathbf{X}_C = \mathbf{x}_C, X = x)$. The values $\alpha = \{\alpha_j | j \neq i\}$ are called proportionality factors.

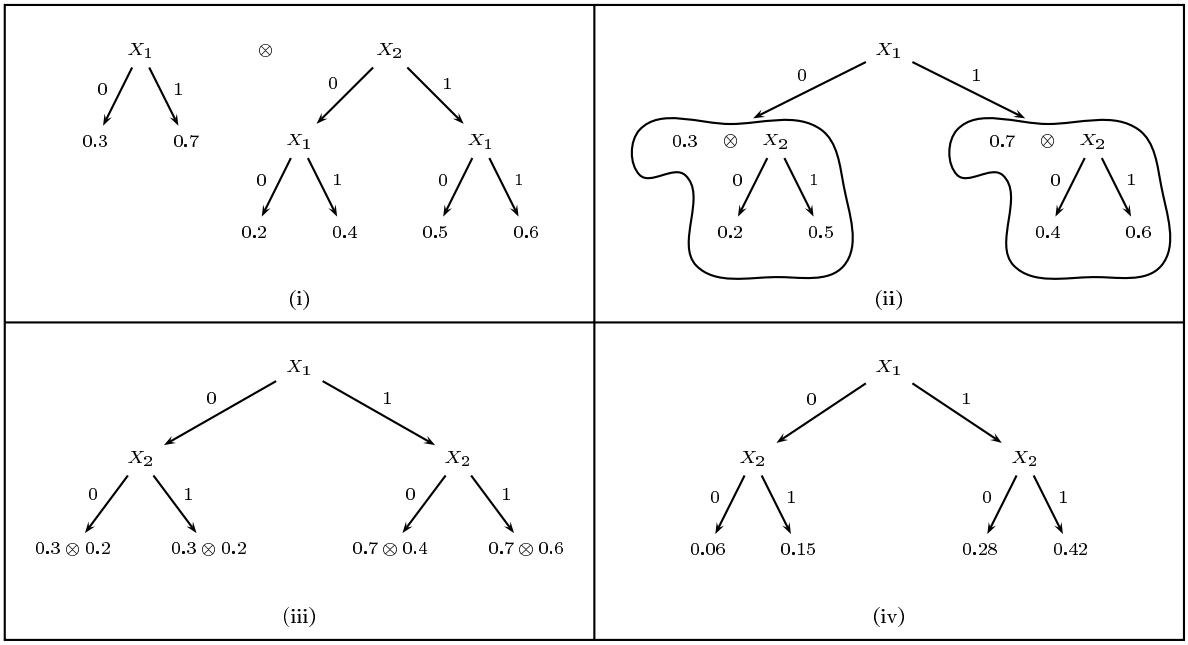

Figure 3: Combination of two trees.

Definition 2 *Let $\mathcal{T}$ be a probability tree proportional below X within context $(\mathbf{X}_C = \mathbf{x}_C)$, with proportionality factors α. We define the* core term *of $\mathcal{T}$, denoted by $\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, X = x, \alpha)$ as the tree obtained from $\mathcal{T}$ by replacing sub-tree $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_i)}$ by constant 1 and any other sub-tree $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_j)}$ by constant α_j.*

Definition 3 *Let $\mathcal{T}$ be a probability tree proportional below X within context $(\mathbf{X}_C = \mathbf{x}_C)$, with proportionality factors α. We define the* free term *of $\mathcal{T}$, denoted by $\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, X = x)$ as the tree obtained from $\mathcal{T}$ by replacing sub-tree $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C)}$ by $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_i)}$ and any other sub-tree $\mathcal{T}^{R(\mathbf{X}_D=\mathbf{x}_D)}$ by a constant 1 for any context $(\mathbf{X}_D = \mathbf{x}_D)$ inconsistent with $(\mathbf{X}_C = \mathbf{x}_C)$.*

Observe that the core and free terms are both probability trees with size smaller than $\mathcal{T}$. In this point, the next proposition can be proved.

Proposition 1 *Let $\mathcal{T}$ be a probability tree proportional below X within context $(\mathbf{X}_C = \mathbf{x}_C)$, with proportionality factors α. It holds that*

$$\begin{aligned} \mathcal{T} \;=\; & \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, X = x, \alpha) \\ & \times \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, X = x) \;. \end{aligned} \tag{2}$$

Proof: The proof is straightforward following the combination (multiplication) algorithm described in Section 2. □

This proposition indicates that any probability tree matching Definition 1 can be decomposed as a product of smaller trees, with the special feature that one of them (the free term) by definition does not contain X.

4 Factorisation in Practice

As we pointed out earlier, practical applications of factorisation arise when using algorithms as Lazy propagation (Madsen and Jensen, 1999) and Lazy-penniless propagation (Cano et al., 2002). Splitting is specially appropriate for the second one. But factorisation is not always recommended, since it may be too costly and provide no benefit if the trees are not decomposable. We can say that then next three considerations must be taken into account before applying factorisation (either splitting or proportional sub-trees):

- *Factorisation must only be used when we are going to marginalise out a variable that appears in more than one probability tree.*

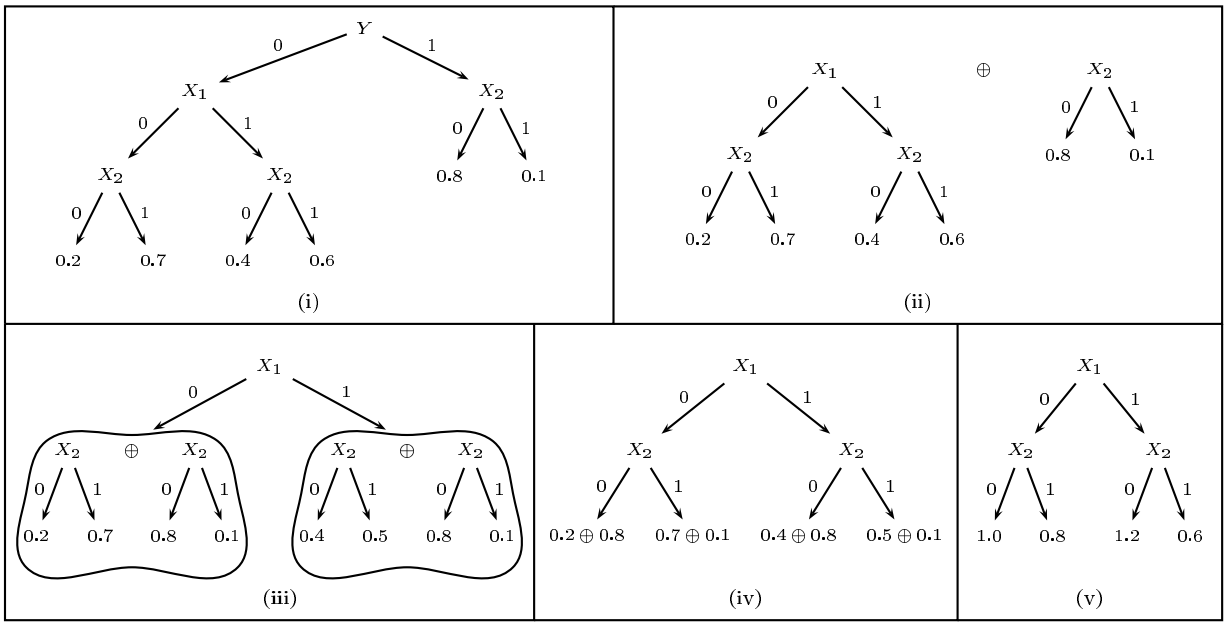

Figure 4: Process of marginalising out variable Y.

Otherwise, no multiplications are carried out and thus the decomposition provides no benefit.

- *The closer to the root the variable to marginalise out is, the higher the benefits will be.* The basis of this statement is that the difference of size between the original tree and the core term increases as the variable is closer to the root.

- *If none of the trees are decomposable, factorisation is useless.* This is obvious, because the time spent on checking factorisation is not compensated at all.

Regarding this last consideration, it is possible to consider the possibility of decomposing a tree even if the sub-trees are not proportional, but "almost" proportional. In this case, the decomposition is not exact, but networks for which probability propagation is unfeasible, could be tackled by this method.

In order to see how far this concept can go in practice, we have implemented the factorisation as a feature of Lazy-penniless propagation in the Elvira system (`http://leo.ugr.es/~elvira`). We have performed tests with different large real-world networks. However, we have been unable to draw clear conclusions. In some of the networks we have observed benefits but in some other the use of factorisation clearly slowed down the programs. The method seems to be quite promising in very large networks, where the join tree has nodes with a high number of variables and involving several potentials in the same node. One of these networks is Munin1, that we borrowed from the Decision Support System Group at Aalborg University (Denmark) and for which the results where improved with respect to Lazy-penniless.

5 Conclusions

In this paper we have proposed an idea to increase the efficiency of propagation algorithms that deal with factorised representations of the potentials in a join tree. Though the idea is not yet thoroughly developed, the approach seems to be promising. It is important now to check the performance of the algorithms in very difficult problems. A criterion for approximate factorisation should be sought, and we are currently working on it. We think that a way to

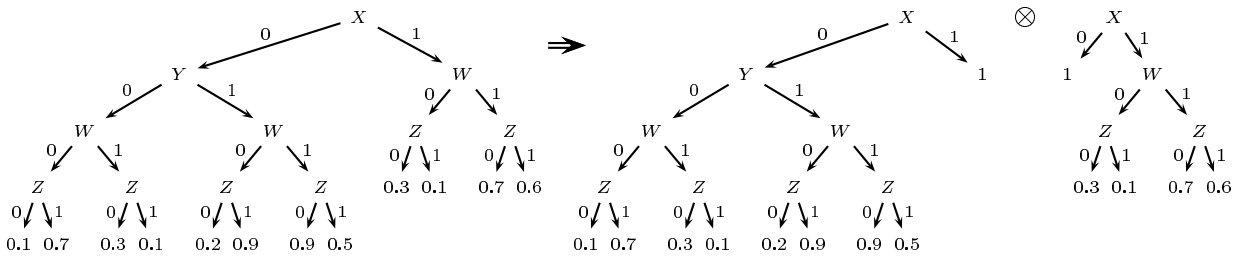

Figure 5: A decomposition of a probability tree by splitting with respect to variable Y.

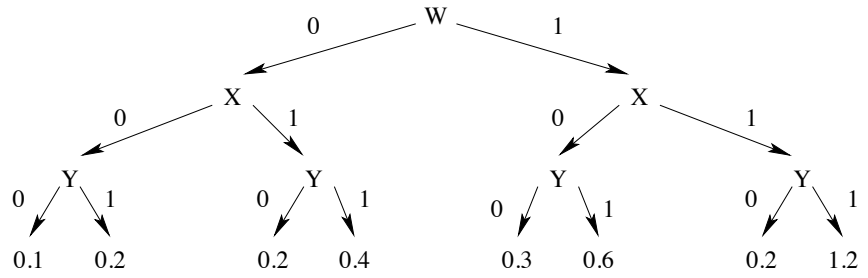

Figure 6: A probability tree where the sub-trees of X are proportional for context $W = 0$.

do it is to decompose as long as the increase in the Kullback-Leibler divergence from the original tree and the product of the core and free terms does not surpass a previously specified limit.

Furthermore, we believe that the ideas introduced in this paper can be of interest in the area of learning Bayesian networks from data: a database can be regarded as a probability tree, and the factorisation would provide a means to detect independences.

Acknowledgments

We are very grateful to Kristian G. Olesen for allowing us to use the Munin network in our experiments. This work has been supported by the Spanish Ministry of Science and Technology, through projects TIC2001-2973-C05-01,02.

References

J. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. 1996. Context-specific independence in Bayesian networks. In E. Horvitz and F.V. Jensen, editors, *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*, pages 115–123. Morgan & Kaufmann.

A. Cano and S. Moral. 1997. 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.

A. Cano, S. Moral, and A. Salmerón. 2000. Penniless propagation in join trees. *International Journal of Intelligent Systems*, 15:1027–1059.

A. Cano, S. Moral, and A. Salmerón. 2002. Lazy evaluation in Penniless propagation over join trees. *Networks*, 39:175–185.

J. Cheng and M.J. Druzdzel. 2000. AIS-BN: An adaptive importance sampling algorithm for evidential reasoning in large Bayesian networks. *Journal of Artificial Intelligence Research*, 13:155–188.

G.F. Cooper. 1990. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405.

P. Dagum and M. Luby. 1993. Approximating prob-

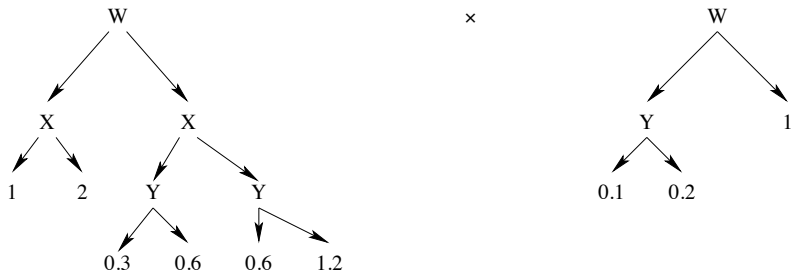

Figure 7: Factorisation of the tree in Figure 6 with respect to variable X.

abilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60:141–153.

P. Dagum and M. Luby. 1997. An optimal approximation algorithm for Bayesian inference. *Artificial Intelligence*, 93:1–27.

R. Fung and K.C. Chang. 1990. Weighting and integrating evidence for stochastic simulation in Bayesian networks. In M. Henrion, R.D. Shachter, L.N. Kanal, and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence*, volume 5, pages 209–220. North-Holland (Amsterdam).

L.D. Hernández, S. Moral, and A. Salmerón. 1998. A Monte Carlo algorithm for probabilistic propagation in belief networks based on importance sampling and stratified simulation techniques. *International Journal of Approximate Reasoning*, 18:53–91.

F. Jensen and S.K. Andersen. 1990. Approximations in Bayesian belief universes for knowledge-based systems. In *Proceedings of the 6th Conference on Uncertainty in Artificial Intelligence*, pages 162–169.

F.V. Jensen, S.L. Lauritzen, and K.G. Olesen. 1990. Bayesian updating in causal probabilistic networks by local computation. *Computational Statistics Quarterly*, 4:269–282.

S.L. Lauritzen and D.J. Spiegelhalter. 1988. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society, Series B*, 50:157–224.

A.L. Madsen and F.V. Jensen. 1999. Lazy propagation: a junction tree inference algorithm based on lazy evaluation. *Artificial Intelligence*, 113:203–245.

J. Pearl. 1988. *Probabilistic reasoning in intelligent systems*. Morgan-Kaufmann (San Mateo).

A. Salmerón and S. Moral. 2001. Importance sampling in Bayesian networks using antithetic variables. In *Lecture Notes in Artificial Intelligence*, volume 2143, pages 168–179.

A. Salmerón, A. Cano, and S. Moral. 2000. Importance sampling in Bayesian networks using probability trees. *Computational Statistics and Data Analysis*, 34:387–413.

R.D. Shachter and M.A. Peot. 1990. Simulation approaches to general probabilistic inference on belief networks. In M. Henrion, R.D. Shachter, L.N. Kanal, and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence*, volume 5, pages 221–231. North Holland (Amsterdam).

P.P. Shenoy and G. Shafer. 1990. Axioms for probability and belief function propagation. In R.D. Shachter, T.S. Levitt, J.F. Lemmer, and L.N. Kanal, editors, *Uncertainty in Artificial Intelligence 4*, pages 169–198. North Holland, Amsterdam.

P.P. Shenoy. 1997. Binary join trees for computing marginals in the Shenoy-Shafer architecture. *International Journal of Approximate Reasoning*, 17:239–263.