Approximate Factorisation of Probability Trees*

Irene Martínez[1], Serafín Moral[2], Carmelo Rodríguez[3], and Antonio Salmerón[3]

[1] Dept. Languages and Computation, University of Almería, Spain
irene@ual.es
[2] Dept. Computer Science and Artificial Intelligence, University of Granada, Spain
smc@decsai.ugr.es
[3] Dept. Statistics and Applied Mathematics, University of Almería, Spain
{crt, Antonio.Salmeron}@ual.es

Abstract. Bayesian networks are efficient tools for probabilistic reasoning over large sets of variables, due to the fact that the joint distribution factorises according to the structure of the network, which captures conditional independence relations among the variables. Beyond conditional independence, the concept of asymmetric (or context specific) independence makes possible the definition of even more efficient reasoning schemes, based on the representation of probability functions through probability trees. In this paper we investigate how it is possible to achieve a finer factorisation by decomposing the original factors for which some conditions hold. We also introduce the concept of approximate factorisation and apply this methodology to the Lazy-Penniless propagation algorithm.

1 Introduction

Bayesian networks have been successfully used as efficient tools for knowledge representation and reasoning under uncertainty. The uncertainty is quantified in terms of a probability distribution over the domain variables, and the reasoning process conveys the computation of the posterior distribution for some variables given that the value of other variables is known. This task is called *probability propagation*.

There are several exact and approximate algorithms for probability propagation [2, 3, 6, 8, 10, 11], but the fact that it is an NP-hard problem [4, 5], 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 methods that incorporate the ability of dealing with factorised representations of the potentials that represent the probabilistic information. These algorithms are Lazy propagation [8] and Lazy-penniless propagation [3].

A particular feature of the Lazy-penniless algorithm is that it uses probability trees [1] to represent probabilistic potentials. Probability trees are usually more

* This work has been supported by the Spanish Ministry of Science and Technology, projects TIC2001-2973-C05-01,02, TIN2004-06204-C03-01 and by FEDER funds.

compact than probability tables and, what is more important, they 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 be just an approximation of the original potential.

2 Bayesian Networks and Probability Trees

We will use the concept of potential to represent any probabilistic information in a Bayesian network (including 'a priori', conditional and 'a posteriori' distributions and intermediate results of operations between them). A *potential* ϕ for a set of variables $\mathbf{X}$ is a mapping $\phi : \Omega_{\mathbf{X}} \rightarrow \mathbb{R}_0^+$, where $\mathbb{R}_0^+$ is the set of non-negative real numbers and $\Omega_{\mathbf{X}}$ is the set of possible cases of the set of variables $\mathbf{X}$. We will consider only discrete variables with a finite number of cases.

Probability propagation is usually carried out over an auxiliary structure called join tree. A *join tree* is a tree where each node is a subset of the variables in the network, and such that if a variable is in two distinct nodes, then it is also in every node in the path connecting them. Every potential in the original Bayesian network (i.e. every conditional distribution) is assigned to a node 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 there will be a potential ϕ_V defined over the set of variables V and which is equal to the product of all the potentials assigned 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 [8, 3]. Probability propagation is carried out by a flow of messages through the edges of the join tree. 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 removing from the potentials attached to V_i all the variables not in V_j. A variable is removed 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 above mentioned 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, either exact and approximate, that can help to overcome this problem.

A *probability tree* [1, 10] 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 $\phi : \Omega_{\mathbf{X}_I} \rightarrow \mathbb{R}_0^+$ 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 labeled with X_i.

A probability tree is usually a more compact representation of a potential than a table. Furthermore, trees allow to obtain even more compact representations in exchange of loosing accuracy. This is achieved by pruning some leaves and replacing them by the average value.

The basic operations (*combination* and *marginalisation*) over potentials required for probability propagation can be carried out directly over probability trees. The combination is done recursively and basically consists of selecting an initial node and multiplying each of its children by the other tree. A variable is marginalised out from a probability tree by replacing it by the sum of its children. We refer to [2] for the details.

3 Exact Factorisation of Probability Trees

Probability propagation basically relies on the combination and marginalisation operations, but the complexity is mainly determined by the combination. For instance, consider the situation in which we are going to delete a variable X_i in order to send a message between two nodes of the join tree. The first step is to combine the potentials (probability trees in this case) containing X_i. The result will be, in the worst case, a potential of size equal to the product of the sizes of the trees that took part in the combination. A gain in efficiency could be achieved if we managed to decompose each tree containing X_i as a product of two trees (factors) of lower size, one of them containing X_i and the other not containing it [9]. Then, the product would be actually carried out over potentials (trees) with reduced domains and therefore, the complexity of probability propagation would decrease. Clearly, it would only be true if the next two conditions hold:

1. The product of the factors into which a tree is decomposed is equal to the original tree, in order to keep the correctness of the results.
2. The propagation algorithm is able to deal with lists of potentials, instead of single potentials in each node and separator of the join tree.

We will devote the rest of the paper to investigate situations in which the probability trees can be decomposed preserving the first condition above, and also situations in which that condition holds only approximately. In this case, the results of the propagation will not be exact, but it is compensated for by the fact that the reasoning can be carried out over very large networks. With respect to the second condition, it is fulfilled by the Lazy [8] and Lazy-penniless [3] algorithms.

We have found two main situations in which probability trees can be decomposed. One is achieved when the variable to marginalise out is only in a part of the tree and the other one is met when some sub-trees of the original one are proportional.

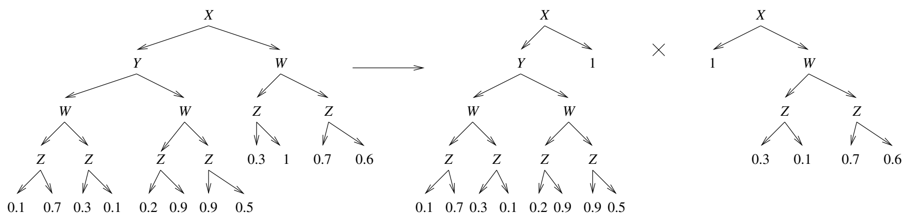

Fig. 1. A decomposition of a probability tree by splitting it with respect to Y

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 1. 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 operation carried out to reduce the size of a tree.

This fact allows to decompose the original tree as the product of two factors of lower size, as displayed in Figure 1. The advantage of this decomposition is that the second factor does not take part in the product previous to the deletion of Y, because it does not contain Y, and the first factor is simpler 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 the upper part of Figure 2. We can see that, within context $W = 0$, all the children of X are proportional. In this case, it is possible to factorise the tree as a product of two trees, where the size of each of the factors is lower than the size of the original tree (see the lower part of Figure 2), in such a way that one of the factor keeps the information regarding X and the other contains the information irrelevant to X. More formally, trees able to 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. 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 $x_j, x_i \neq x_j \in \Omega_X$, $\exists \alpha_j > 0$ such that*

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

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 $\boldsymbol{\alpha} = \{\alpha_j | j \neq i\}$ are called proportionality factors.

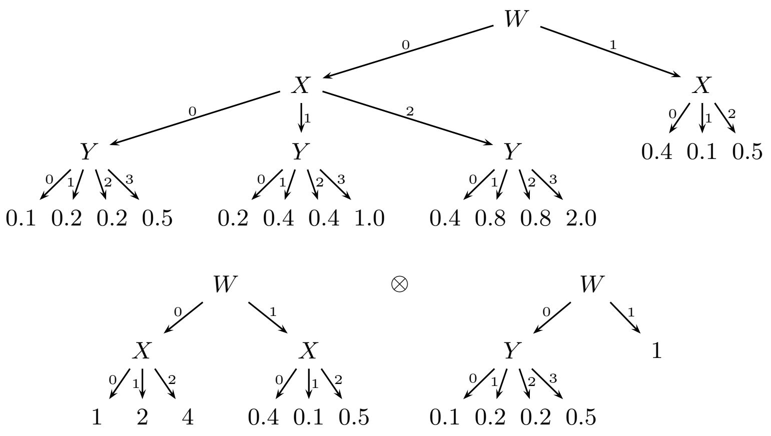

Fig. 2. A probability tree proportional below X for context $(W = 0)$ and its decomposition with respect to variable X

The following definition identify each one of the factors into which a tree verifying definition 1 can be decomposed.

Definition 2. *Let $\mathcal{T}$ be a probability tree which is proportional below X within context $(\mathbf{X}_C = \mathbf{x}_C)$, with proportionality factors $\boldsymbol{\alpha}$. We define the* core term *of $\mathcal{T}$, denoted by $\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, X = x_i, \boldsymbol{\alpha})$ 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. We define the* free term *of $\mathcal{T}$, denoted by $\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, X = x_i)$ 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 have size smaller than $\mathcal{T}$. Furthermore, the free term does not contain variable X. This, together with the result in the next proposition, show that factorisation increases the efficiency of probability propagation, in the sense that the amount of memory required is reduced.

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

$$\mathcal{T} = \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, X = x, \boldsymbol{\alpha}) \times \mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, X = x) \ . \qquad (2)$$

3.3 Partially Proportional Sub-trees

Still there is another situation in which some regularities can be found in a probability tree that can be used to reduce the complexity of the operations involved in the process of marginalising out a variable. The scenario is very similar to the case of proportional sub-trees described above, but instead of all

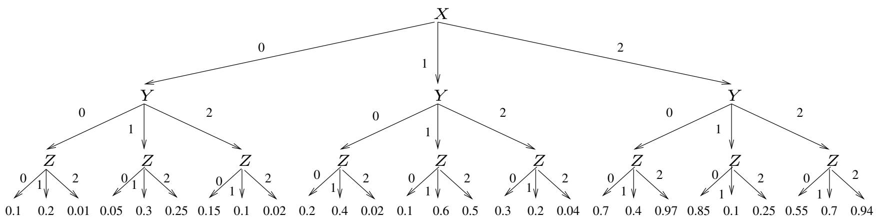

Fig. 3. A probability tree partially proportional below variable X

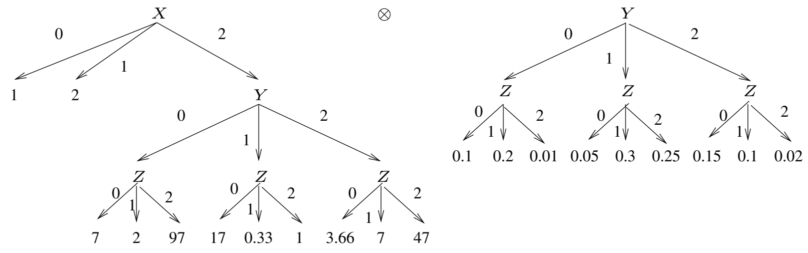

Fig. 4. Factorisation of the tree in figure 3

the children of the variable to delete, only some of them are proportional. This situation is illustrated in the next example.

Example 1. Assume we have three variables X, Y and Z, each one of them taking values on the set $\{0, 1, 2\}$. Consider the conditional distribution for X given Y and Z represented by the probability tree in figure 3. Observe that the tree is not proportional below X, because the sub-trees corresponding to $X = 0$ and $X = 1$ are proportional, but the sub-tree for $X = 2$ is not. However, even though the conditions in definition 1 are not met in this case, the tree can be decomposed in the way described in figure 4. Notice that the resulting factorisation is able to represent the conditional distribution for X using just 20 numbers instead of 27. □

Formally, a probability where this kind of proportionality occurs can be defined as follows.

Definition 3. *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. We say that $\mathcal{T}$ is* partially proportional below X within context $(\mathbf{X}_C = \mathbf{x}_C)$ *if there is a $x_i \in \Omega_X$ and a set $L \subset \Omega_X \setminus \{x_i\}$ such that for every $x_j \in L$, $\exists \alpha_j > 0$ such that*

$$\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_i)} = \alpha_j \cdot \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C, X=x_j)} \quad . \tag{3}$$

In this setting, the concept of core term given in definition 2 must be modified in order to guarantee that the product of the core and free terms is equal to the original tree. However, the free term needs not be re-defined.

Definition 4. *Let $\mathcal{T}$ be a probability tree which is partially proportional below X within context $(\mathbf{X}_C = \mathbf{x}_C)$, with proportionality factors $\boldsymbol{\alpha}$ and let x_i and L be as in definition 3. We define the* partial core term *of $\mathcal{T}$, denoted by $\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, X = x_i, \boldsymbol{\alpha}, L)$ as the tree obtained from $\mathcal{T}$ by replacing:*

1. *Sub-tree $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_i)}$ by constant 1.*
2. *Any sub-tree $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_j)}$, $x_j \in L$, by constant α_j.*
3. *Any sub-tree $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_k)}$, $x_i \neq \mathbf{x}_k \notin L$, by $\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_k)}/\mathcal{T}(\mathbf{X}_C = \mathbf{x}_C, X = x_i)$.*

It can be shown that a partially proportional tree can be decomposed as the product of its core and free terms.

4 Approximate Factorisation of Probability Trees

There are situations in which the ways of decomposing trees described in the former section may be of interest, even if the conditions of proportionality or partial proportionality are not met. For instance, assume that we have three variables X, Y and Z, and that the actual distribution of X given Y and Z is the one given in figure 3, but that, due to sampling error, the learnt distribution is not exactly the same, but very close to it. Another scenario in which one could be interested in decomposing a tree even if the exact factorisation is not possible is when space limitations do not allow for exact probability propagation, and then it is necessary to tradeoff accuracy for space requirements.

The problem of approximate factorisation can be stated as follows. Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two sub-trees which are siblings for a given context (i.e. both sub-trees are children of the same node), such that both have the same size and their leaves contain only positive numbers. The goal of the *approximate factorisation* is to find a tree $\mathcal{T}_2^*$ with the same structure than $\mathcal{T}_2$, such that $\mathcal{T}_2^*$ and $\mathcal{T}_1$ become proportional, under the restriction that the potential represented by $\mathcal{T}_2^*$ must be as close as possible to the one represented by $\mathcal{T}_2$. Then, $\mathcal{T}_2$ can be replaced by $\mathcal{T}_2^*$ and the resulting tree that contains $\mathcal{T}_1$ and $\mathcal{T}_2^*$ can be decomposed, as it would become proportional or partially proportional for the given context.

Approximate factorisation involves: (1) The determination of the proportionality factor, α, and (2) Measuring the accuracy of the approximation. Both issues are connected, since it seems sensible to select the proportionality factor in such a way that the chosen divergence measure is minimised. In general, different divergence measures would result in different values for α. The problem of approximate factorisation is formalised in the next definition.

Definition 5. *We say that a probability tree $\mathcal{T}$ is* δ-factorisable *within context $(\mathbf{X}_C = \mathbf{x}_C)$, with proportionality factors $\boldsymbol{\alpha}$ with respect to a divergence measure $\mathcal{D}$ if there is an $x_i \in \Omega_X$ and a set $L \subset \Omega_X \setminus \{x_i\}$ such that for every $x_j \in L$, $\exists \alpha_j > 0$ such that*

$$\mathcal{D}(\mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_i)}, \alpha_j \cdot \mathcal{T}^{R(\mathbf{X}_C=\mathbf{x}_C,X=x_j)}) \leq \delta \ .$$

Parameter $\delta > 0$ is called the tolerance of the approximation.

Observe that proportional and partially proportional trees for context $(\mathbf{X}_C = \mathbf{x}_C)$ are δ-factorisable, with $\delta = 0$.

Now we will consider how to factorise δ-decomposable trees, analysing different divergence measures and computing the optimum α. We will impose the next consistency restriction to all the approximate factorisation methods that we will propose: A method is said to be consistent if it introduces no error when the tree is proportional or partially proportional below the considered context (see definitions 1 and 3).

4.1 Computing the Proportionality Factor

Consider a probability tree $\mathcal{T}$. Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be sub-trees of $\mathcal{T}$ below a variable X, for a given context $(\mathbf{X}_C = \mathbf{x}_c)$ with leaves $\mathcal{P} = \{p_i : i = 1, \ldots, n; p_i \neq 0\}$ and $\mathcal{Q} = \{q_i : i = 1, \ldots, n;\}$ respectively. As we described before, approximate factorisation is achieved by replacing $\mathcal{T}_2$ by another tree $\mathcal{T}_2^*$ such that $\mathcal{T}_2^*$ is proportional to $\mathcal{T}_1$. It means that the leaves of $\mathcal{T}_2^*$ will be $\mathcal{Q}^* = \{\alpha p_i : i = 1, \ldots, n;\}$, where α is the proportionality factor between $\mathcal{T}_1$ and $\mathcal{T}_2$. Let us denote by $\{\pi_i = q_i/p_i, i = 1, \ldots, n;\}$ the ratios between the leaves of $\mathcal{T}_2$ and $\mathcal{T}_1$.

We have considered several possibilities for computing the proportionality factor, α. First we will derive the value of the proportionality factor under the restriction of minimising different measures of divergence:

1. The χ^2 divergence, defined as

$$\mathcal{D}_\chi(\mathcal{T}_2, \mathcal{T}_2^*) = \sum_{i=1}^{n} \frac{(q_i - \alpha p_i)^2}{q_i} \ ,$$

is minimised for α equal to $\alpha_\chi = \frac{\sum_{i=1}^n p_i}{\sum_{i=1}^n p_i/\pi_i}$. Instead of using $\mathcal{D}_\chi$, we can consider its normalised version

$$\mathcal{D}_{\chi^*}(\mathcal{T}_2, \mathcal{T}_2^*) = \sqrt{\frac{\mathcal{D}_\chi}{\mathcal{D}_\chi + n}} \ ,$$

which takes values between 0 and 1 and is minimised for the same α.

2. The mean squared error

$$\mathcal{D}_{mse}(\mathcal{T}_2, \mathcal{T}_2^*) = \frac{1}{n} \sum_{i=1}^{n} (q_i - \alpha p_i)^2$$

is minimised for $\alpha_{mse} = \frac{\sum_{i=1}^n \pi_i p_i^2}{\sum_{i=1}^n p_i^2}$.

In case of using a weighted MSE as divergence measure, i.e.

$$\mathcal{D}_{wmse}(\mathcal{T}_2, \mathcal{T}_2^*) = \sum_{i=1}^{n} h_i (q_i - \alpha p_i)^2$$

with $\{h_i \geq 0, i = 1, \ldots, n; \sum h_i = 1\}$, the optimum proportionality factor is

$$\alpha_{wmse} = \frac{\sum_{i=1}^n h_i \pi_i p_i^2}{\sum_{i=1}^n h_i p_i^2} \ .$$

A possible selection of the weights h_i is $h_i = \frac{q_i}{\sum_{i=1}^n q_i}$, in which case $\mathcal{D}_{mse}$ would be the expected MSE with respect to $\mathcal{T}_2$ (actually, with respect to a probability distribution proportional to the potential represented by $\mathcal{T}_2$).

3. The Kullback-Leibler divergence, defined as

$$\mathcal{D}_{kl}(\mathcal{T}_2, \mathcal{T}_2^*) = \sum_{i=1}^n q_i \log\left(\frac{q_i}{\alpha p_i}\right) \ ,$$

reaches its minimum at $\alpha_{kl} = 2^{\frac{\sum_{i=1}^n q_i \log(\pi_i)}{\sum_{i=1}^n q_i}}$. The problem of using $\mathcal{D}_{kl}$ is that it requires that the sum of the values of its arguments coincide [7]. Otherwise, $\mathcal{D}_{kl}$ can take negative values. This renders this criterion useless for our purposes.

But it is also possible to obtain the proportionality factor independently of any divergence measure. For instance, one restriction could be to ensure that the weight of the original and the approximate tree coincide, that is:

$$\text{sum}(\mathcal{T}_2^*) = \sum_{i=1}^n \alpha p_i = \sum_{i=1}^n q_i = \text{sum}(\mathcal{T}_2) \ .$$

We will refer to this as the *weight preserving method*, and the proportionality factor that corresponds to this restriction is

$$\alpha_{wp} = \frac{\sum_{i=1}^n q_i}{\sum_{i=1}^n p_i} = \frac{\sum_{i=1}^n \pi_i p_i}{\sum_{i=1}^n p_i} \ .$$

Perhaps the more straightforward way to obtain a value for α is the so-called *weighted average method*, which computes it as a weighted average of the ratios between the leaves of $\mathcal{T}_1$ and $\mathcal{T}_2$. The resulting proportionality factor is

$$\alpha_{wa} = \sum_{i=1}^n h_i \pi_i \ ,$$

with $\{h_i \geq 0, i = 1, \ldots, n; \sum h_i = 1\}$. Observe that α_{wp} and α_{mse} are particular cases of α_{wa} with $h_i = \frac{p_i}{\sum_{i=1}^n p_i}$ and $h_i = \frac{p_i^2}{\sum_{i=1}^n p_i^2}$ respectively.

Besides, there may be other divergence measures that could be applied to our problem but that cannot be minimised with respect to α. Of special interest is the divergence measure computed as the *maximum absolute difference* between the leaves of $\mathcal{T}_2$ and $\mathcal{T}_2^*$, that we will use in the experiments:

$$\mathcal{D}_{mad}(\mathcal{T}_2, \mathcal{T}_2^*) = \max_{1 \leq i \leq n} |q_i - \alpha p_i| \ .$$

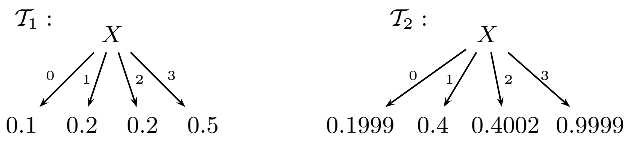

Fig. 5. Almost proportional trees

Table 1. Divergences between the tree $\mathcal{T}_2$ in Fig. 5 and the different approximations of it which are proportional to $\mathcal{T}_1$

	$\alpha_{wp} =$ 2.0	$\alpha_{\chi} =$ 1.9999998	$\alpha_{mse} =$ 1.9999412	$\alpha_{wmse} =$ 1.9998733	$\alpha_{kl} =$ 2.0000001	$\alpha_{wa} =$ 2.0000002
$\mathcal{D}_{mad}$	2E-4	2.00032E-4	2.11764E-4	2.25343E-4	1.99984E-4	1.99968E-4
$\mathcal{D}_{\chi}$	0.00039997005	0.00039997003	0.000402115	0.000409859	0.00039997005	0.00039997009
$\mathcal{D}_{\chi^*}$	0.00019998502	0.00019998501	0.000201057	0.000204929	0.00019998503	0.00019998504
$\mathcal{D}_{mse}$	0.000122474	0.000122467	0.000121267	0.000122872	0.000122477	0.000122481
$\mathcal{D}_{wmse}$	5.91671E-5	5.91549E-5	5.56270E-5	5.41362E-5	5.91732E-9	5.91793E-5

Example 2. The trees in figure 5 are "almost" proportional. It seems that they could be considered as proportional and the corresponding factorisation would not affect very much the results of the probability propagation algorithm. Table 1 shows the divergence between $\mathcal{T}_2$ and $\mathcal{T}_2^*$ using the different criteria for approximate factorisation described in this section. It can be seen from the results in that table how choosing $\alpha_{\chi}, \alpha_{mse}$ and α_{wmse} minimises the corresponding divergence measures with respect to which they were obtained. The maximum absolute divergence ($\mathcal{D}_{mad}$) is minimised, in this example, by choosing α_{wa} as proportionality factor.

If the trees in figure 5 are siblings below a given variable Y for a context $(\mathbf{X}_C = \mathbf{x}_C)$ of some tree $\mathcal{T}$, it can be said that, according to definition 5 that $\mathcal{T}$ is δ-factorisable within context $(\mathbf{X}_C = \mathbf{x}_C)$ for any $\delta > 0.001$, regardless the selected α and the divergence measure used.

For $\delta \leq 0.001$, $\mathcal{T}$ would not always be considered δ-factorisable. For example, if we selected a tolerance $\delta = 0.0002$ and the divergence measure $\mathcal{D}_{mad}$, $\mathcal{T}$ is δ-factorisable within context $(\mathbf{X}_C = \mathbf{x}_C)$ only for proportionality factors α_{wp}, α_{wa} and α_{kl}.

5 Experiments

In order to illustrate how the techniques above described can be used to tradeoff accuracy for space requirements, we have tested the Lazy-penniless algorithm [3] with the added feature of factorising the potentials before deleting a variable, using different real networks. In order to analyse the impact of the factorisation, we have used the simplest version of Lazy-penniless (no heuristic is used to select the order of combination of the potentials), and the trees are not pruned. Due to space limitations, we only report the results for two well known networks

Table 2. Experimental results for network Munin1

	D_χ divergence			Weight Preserving		
δ	Mean	MSE	nAp	Mean	MSE	nAp
0.025	27556.85	3.06E-6	3070	23704.26	1.49E-6	2788
0.050	27286.13	1.52E-4	3387	23609.30	2.68E-6	2982
0.075	26885.23	2.40E-4	3699	23300.01	1.33E-5	3443
0.1	26238.68	7.04E-4	4645	23499.51	1.42E-5	3655
0	31947.58	0	132	31947.58	0	132

Table 3. Experimental results for network Water

	D_χ divergence			Weight Preserving		
δ	Mean	MSE	nAp	Mean	MSE	nAp
0.025	1884.80	1.93E-5	368	1884.54	1.93E-5	367
0.050	1735.47	2.23E-5	435	1737.91	2.22E-5	419
0.075	1692.30	9.88E-6	530	1693.14	1.02E-5	512
0.1	1581.15	3.28E-5	570	1581.74	3.35E-5	558
0	1733.23	0	2	1733.23	0	2

(Munin1 and Water) borrowed from the Decision Support Systems Group at Aalborg University. The results are displayed in table 2 and table 3 respectively, where the first column δ indicates the error allowed when factorising (tolerance in terms of distance D_{χ^*}). The reason to use D_{χ^*} is that it is easier to control, since it is between 0 and 1. We have computed the mean of the sizes of the potentials used during the propagation, the average mean squared error (MSE) for all the unobserved variables after the propagation and the number of factorisations actually carried out. In the experiments, we have only searched for proportional subtrees which root is not located beyond half of the depth of the tree, in order to avoid useless factorisations (for instance, factorising only the leaves). With respect to the computing times, they are about a 20% higher compared with the Lazy propagation (or exact Lazy-penniless), but the space requirements are lower. The mean clique sizes for Lazy propagation are 31905.37 for Munin1 and 1733.2 for Water.

Even though the analysis is still rather preliminary, the results seem to indicate that approximate factorisation is a valid method for controlling the space requirements during propagation.

6 Conclusions

In this paper we have extended the factorisation technique presented in [9] by introducing the possibility of decomposing the trees that are approximately proportional. The results suggest that this method provides a valid tradeoff between space requirements and approximation error, and that it can be controlled by means of the δ parameter. A deeper experimental analysis is necessary to know how far this technique can go, and which of the proposed distance measure achieves the best results. Besides, we have not yet checked the joint behaviour of factorising and splitting, but we believe that the results must significantly improve. We are also implementing the use of factorisation in compilation time, in order to obtain smaller initial probability distributions for the propagation phase.

References

1. C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller. 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, 1996.
2. A. Cano, S. Moral, and A. Salmerón. Penniless propagation in join trees. *International Journal of Intelligent Systems*, 15:1027–1059, 2000.
3. A. Cano, S. Moral, and A. Salmerón. Lazy evaluation in Penniless propagation over join trees. *Networks*, 39:175–185, 2002.
4. G.F. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42:393–405, 1990.
5. P. Dagum and M. Luby. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, 60:141–153, 1993.
6. F.V. Jensen, S.L. Lauritzen, and K.G. Olesen. Bayesian updating in causal probabilistic networks by local computation. *Computational Statistics Quarterly*, 4:269–282, 1990.
7. S. Kullback and R. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:76–86, 1951.
8. A.L. Madsen and F.V. Jensen. Lazy propagation: a junction tree inference algorithm based on lazy evaluation. *Artificial Intelligence*, 113:203–245, 1999.
9. I. Martínez, S. Moral, C. Rodríguez, and A. Salmerón. Factorisation of probability trees and its application to inference in Bayesian networks. In J.A. Gámez and A. Salmerón, editors, *Proceedings of the First European Workshop on Probabilistic Graphical Models*, pages 127–134, 2002.
10. A. Salmerón, A. Cano, and S. Moral. Importance sampling in Bayesian networks using probability trees. *Computational Statistics and Data Analysis*, 34:387–413, 2000.
11. P.P. Shenoy. Binary join trees for computing marginals in the Shenoy-Shafer architecture. *International Journal of Approximate Reasoning*, 17:239–263, 1997.