Abductive Inference in Bayesian Networks: Finding a Partition of the Explanation Space

M. Julia Flores[1], José A. Gámez[1], and Serafín Moral[2]

[1] Departamento de Informática, Universidad de Castilla-La Mancha, 02071 Albacete, Spain
[2] Departamento de Ciencias de la Computación e I.A., Universidad de Granada, 18071 Granada, Spain

Abstract. This paper proposes a new approach to the problem of obtaining the most probable explanations given a set of observations in a Bayesian network. The method provides a set of possibilities ordered by their probabilities. The main novelties are that the level of detail of each one of the explanations is not uniform (with the idea of being as simple as possible in each case), the explanations are mutually exclusive, and the number of required explanations is not fixed (it depends on the particular case we are solving). Our goals are achieved by means of the construction of the so called *explanation tree* which can have asymmetric branching and that will determine the different possibilities. This paper describes the procedure for its computation based on information theoretic criteria and shows its behaviour in some simple examples.

1 Introduction

Although the most common probabilistic inference in Bayesian networks (BNs) is probability or evidence propagation [18, 1, 11], that is, computation of posterior probability for all non-observed variables given a set of observations ($X_O = x_O$) (the *evidence*), there are other interesting inference tasks. In this paper we are concerned with the inference task that attempts to generate explanations for a given evidence. Generating explanations in Bayesian networks can be understood in two (main) different ways:

1. *Explaining the reasoning process* (see [12] for a review). That is, trying to justify *how* a conclusion was obtained, *why* new information was asked, etc.
2. *Diagnostic explanations* or *abductive inference* (see [9] for a review). In this case the explanation reduces to *factual* information about the state of the world, and the best explanation for a given evidence is the state of the world (configuration) that is the most probable given the evidence [18].

In this paper we focus on the second approach. Therefore, given a set of observations or evidence ($X_O = x_O$ or x_O in short) known as the *explanandum*, we aim to obtain the best configuration of values for the explanatory variables (the *explanation*) which is consistent with the *explanandum* and which needs

L. Godo (Ed.): ECSQARU 2005, LNAI 3571, pp. 63–75, 2005.

to be assumed to predict it. Depending on what variables are considered as *explanatory*, two main abductive tasks in BNs are identified:

- *Most Probable Explanation* (MPE) or *total abduction*. In this case all the unobserved variables (X_U) are included in the explanation [18]. The *best* explanation is the assignment $X_U = x_U^*$ which has maximum a posteriori probability given the *explanandum*, i.e.,

$$x_U^* = \arg \max_{x_U \in \Omega_{X_U}} P(x_U|x_O). \tag{1}$$

  Searching for the best explanation has the same complexity (NP-hard [23]) as probability propagation, in fact the best MPE can be obtained by using probability propagation algorithms but replacing summation by maximum in the marginalisation operator [3]. However, as it is expected to have several competing hypothesis accounting for the *explanandum*, our goal usually is to get the K best MPEs. Nilsson [15] showed that using algorithm in [3] only the first three MPEs can be correctly identified, and proposed a clever method to identify the remaining $(4, \ldots, K)$ explanations.
  One of the main drawbacks of the MPE definition is that as it produces complete assignments, the explanations obtained can exhibit the *overspecification* problem [21] because some non-relevant variables have been used as explanatory.
- *Maximum a Posteriori Assignment* (MAP) or *partial abduction* [14, 21]. The goal of this task is to alleviate the overspecification problem by considering as target variables only a subset of the unobserved variables called the *explanation set* (X_E). Then, we look for the maximum a posteriori assignment of these variables given the explanandum, i.e.,

$$x_E^* = \arg \max_{x_E} P(x_E|x_O) = \arg \max_{x_E} \sum_{x_R} P(x_E, x_R|x_O), \tag{2}$$

  where $X_R = X_U \setminus X_E$. This problem is more complex than the MPE problem, because it can be NP-hard even for cases in which MPE is polynomial (e.g., polytrees) [17, 5], although recently Park and Darwiche [16, 17] have proposed exact and approximate algorithms to enlarge the class of *efficiently* solved cases. With respect to looking for the K best explanations, exact and approximate algorithms which combine Nilsson algorithm [15] with probability trees [19] has been proposed in [6].

The question now is which variables should be included in the explanation set. Many algorithms avoid this problem by assuming that the explanation set is provided as an input, e.g., given by the experts or users. Many others interpret the BN as a causal one and only ancestors of the *explanandum* are allowed to be included in the explanation set (sometimes only root nodes are considered) [13]. However, including all the ancestors in the explanation set does not seem to avoid the overspecification problem and even so, what happens if the network does not have a causal interpretation?, e.g., it has been learnt from a data base

or it represents an agent's beliefs [2]. Shimony [21, 22] goes one step further and describes a method which tries to identify the relevant variables (among the *explanandum* ancestors) by using independence and relevance based criteria. However, as pointed out in [2] the explanation set identified by Shimony's method is not as concise as expected, because for each variable in the *explanandum* all the variables in at least one path from it to a root variable are included in the explanation set. Henrion and Druzdzel [10] proposed a model called *scenario-based explanation*. In this model a tree of propositions is assumed, where a path from the root to a leaf represents a scenario, and they look for the scenario with highest probability. In this model, partial explanations are allowed, but they are restricted to come from a set of predefined explanations.

As stated in [2] *conciseness* is a desirable feature in an explanation, that is, the user usually wants to know only the most influential elements of the complete explanation, and does not want to be burdened with unnecessary detail. Because of this, a different approach is taken in [4]. The idea is that even when only the relevant variables to the *explanandum* are included in the explanation set, the explanations can be simplified due to context-specific irrelevance. This idea is even more interesting when we look for the K MPEs, because it allows us to obtain explanations with different number of literals. In [4] the process is divided into two stages: (1) the K MPEs are obtained for a given prespecified explanation set, and (2) then they are simplified by using different independence and relevance based criteria.

In this paper we try to obtain simplified explanations directly. The reason is that the second stage in [4] requires to carry out several probabilistic propagations and so its computational cost is high (and notice that this process is carried out after -a complex- MAP computation). Another drawback of the procedure in [4] is that it is possible, that after simplification, the explanations are not mutually exclusive, we can have even the case of two explanations such that one is a subset of the other. Here, our basic idea is to start with a predefined explanation set X_E, and them we build a tree in which variables (from X_E) are added in function of their explanatory power with respect to the *explanandum* but taken into account the current context, that is, the partial assignment represented by the path obtained from the root to the node currently analysed. Variables are selected based on the idea of *stability*, that is, we can suppose that our system is (more or less) stable, and that it becomes unstable when some (*unexpected*) observations are entered into the system. The instability of a variable will be measured by its entropy or by means of its (im)purity (GINI index). Therefore, we first select those variables that reduce most the uncertainty of the non-observed variables of the explanation set, i.e., the variables better determining the value of the explanation variables. Of course, the tree does not have to be symmetric and we can decide to stop the growing of a branch even if not all the variables in X_E have been included. In any case, our set of explanations will be mutually exclusive, and will have the additional property of being exhaustive, i.e., we will construct a true partition of the set of possible configurations or scenarios of the values of the variables in the explanation set.

The subsequent sections describe our method in detail and illustrate it by using some (toy) study cases. Finally in Section 4 we present our conclusions and outline future works.

2 How to Obtain an *Explanation Tree*

Our method aims to find the best explanation(s) for the observed variables that do not necessarily have a fixed number of literals. The provided explanations will adapt to the current circumstances. Sometimes that a variable X takes a particular value it is an explanation by itself (Occam's razor) and including other variables to this explanation will not add any new information. We have then decided to represent our solutions by a tree, the *Explanation Tree* (ET).

In the ET, every node will denote a variable of the explanation set and every branch from this variable will indicate the instantiation of this variable to one of its possible states. Each node of the tree will determine an assignment for the variables in the path from the root to it: each variable equal to the value on the edge followed by the path. This assignment will be called the configuration of values associated to the node. In the explanation tree, we will store for each leaf the probability of its associated configuration given the evidence. The set of explanations will be the set of configurations associated to the leaves of the explanation tree ordered by their posterior probability given the evidence. For example, in Fig. 5.a we can see three variables $A1$, $A2$ and $N2$ that belong to the explanation set, since they are nodes in the ET. In this particular example there are four leaves nodes, i.e., four possible explanations. What this ET indicates is that, given the observed evidence, $A1 = f$ is a valid explanation for such situation (with its probability). But if it is not the case then we should look into other factors, in this case $N2$. For example, we can see that adding $N2 = f$ to the current path ($A1 = ok$) will be enough to provide an explanation. Otherwise, when $N2 = ok$ the node needs to be *expanded* and we will look for other involved factors in order to find a valid explanation (in this example, by using $A2$).

Although the underlying idea is simple, how to obtain this tree is not so evident. There are two major points that have to be answered:

- As the ET is created in a top-down way, given a branch of the tree, how to select the next variable?
- Given our goals, i.e. allow asymmetry and get concise explanations, how to decide when to stop branching?

To solve the two previous questions we have used information measures. For the first one, we look for the variable that once instantiated the uncertainty of the rest explanation variables is reduced at maximum. In other words, given the context provided by the current branch, we identify the most explicative as the one that helps to determine the values of the other variables as much as possible.

Algorithm 1 (Create-New-Node) recursively creates our ET. In this algorithm we assume the existence of an inference engine that provides us with the probabilities needed during tree growing. We comment on such engine in Section 2.1. The algorithm is called with the following parameters:

1. The evidence/observations to be explained x_O.
2. The *path* corresponding to the branch we are growing. In the first call to this algorithm, i.e. when deciding the root node, this parameter will be null.
3. The current explanation set (X_E). That is, the set of explanatory variables already available given the context (path). In the first call X_E is the original explanation set. Notice also that if $X_E = X_U$ in the first call, i.e., all non-observed variables belong to the explanation set, then the method has to select those variables relevant to the explanation without prior information.
4. Two real numbers α and β used as thresholds (on information and probability respectively) to stop growing.
5. The final explanation tree that will be recursively and incrementally constructed as an accumulation of branches (paths). Empty in the initial call.

Algorithm 1. Creates a new node for the explanation tree

```
1: procedure CREATE_NEW_NODE(x_O,path,X_E,α,β,ET)
2:     for all X_j, X_k ∈ X_E do
3:         Info[X_j, X_k]  = Inf(X_j, X_k|x_O, path)
4:     end for
5:     X_j^* = arg max_{X_j ∈ X_E} Σ_{X_k} Info[X_j, X_k]
6:     if CONTINUE(Info[],X_j^*,α) and P(path|x_O) > β then
7:         for all state x_j of X_j^* do
8:             new_path ← path + X_j^* = x_j
9:             CREATE_NEW_NODE(x_O,new_path,X_E \ X_j^*,α,β,ET)
10:        end for
11:    else
12:        ET ← ET ∪ <path,P(path|x_O) >          ▷ update the ET adding path
13:    end if
14: end procedure
```

In algorithm 1, for each variable in the explanation set, X_j, we compute the sum of the amount of information that this variable provides about all the current explanation variables conditioned to the current observations x_O^*. We are interested in the variable that maximises this value. In our study we have considered two classical measures: *mutual information* ($Inf(X_j, X_k|x_O^*) = \sum_{x_j,x_k} P(x_j, x_k|x_O^*) \log\left(\frac{P(x_j,x_k|x_O^*)}{P(x_j|x_O^*).P(x_k|x_O^*)}\right)$) and *GINI index* ($Inf(X_j, X_k|x_O^*) = 1 - \sum_{x_j,x_k} P(x_j, x_k|x_O^*)^2$). Thus, there are different instances of the algorithm depending on the criterion used as *Inf*.

Once we have selected the next variable to be placed in a branch, we have to decide whether or not to expand this node. Again, we will use the measure *Inf*. The procedure CONTINUE is the responsible to take this decision by considering the vector Info[]. This procedure considers the list of values $\texttt{Info}[X_j^*, X_k]$ for $X_k \neq X_j^*$, then it computes the maximum, minimum, or average of them, depending on the particular criterion we are using. If this value is greater than α it decides to continue. Of course the three criteria give rise to different behaviour, being minimum the most restrictive, maximum the most permissive and having average and intermediate behaviour.

Notice that when only two variables remain in the explanation set, the one selected in line 5 is in fact that having greater entropy ($I(X,X) = H(X)$) if mutual information is used. Also, when only one variable is left, it is of course the selected one, but it is necessary to decide whether or not it should be expanded. For that purpose, we use the same information measure, that is, $\mathrm{I}(X,X)$ or $\mathrm{GINI}(X,X)$, and only expand this variable if it is at least as uncertain (unstable) as the distribution $[1/3, 2/3]$ (Normalising with more than two states). That is, we only add a variable if it has got more uncertainty than a given threshold.

2.1 Computation

Our inference engine is (mainly) based on Shenoy Shafer running over a binary join tree [20]. Furthermore, we have forced the existence of a single clique (being a leaf) for each variable in X_E, i.e. a clique which contains only a variable. We use these cliques to enter *as evidence* the value to which an explanatory variable is instantiated, as well as to compute its posterior probability.

Here we comment on the computation of the probabilities needed to carry out the construction of the explanation tree. Let us assume that we are considering to expand a new node in the tree which is identified by the configuration (path) $C = c$. Let x^*_O be the configuration obtained by joining the observations $X_O = x_O$ and $C = c$. Then, we need to calculate the following probabilities:

- $P(X_i, X_j | x^*_O)$ for $X_i, X_j \in X_E \setminus C$. To do this we use a two stage procedure:
  1. Run a full propagation over the join tree with x^*_O entered as evidence. In fact, many times only the second stage (i.e., *DistributeEvidence*) of Shenoy-Shafer propagation is needed. This is due to the single cliques included in the join tree, because if only one evidence item (say X) has changed[1] from the last propagation, we locate the clique containing X, modify the evidence entered over it and run *DistributeEvidence* by using it as root.
  2. For each pair (X_i, X_j) whose joint probability is required, locate the two closest cliques (C_i and C_j) containing X_i and X_j. Pick all the potentials in the path between C_i and C_j and obtain the joint probability by using variable elimination [7]. In this process, we can take as basis the deletion sequence implicit in the join tree (but without deleting the required variables) and then the complexity is not greater than the complexity of sending a series of messages along the path connecting C_i with C_j for each possible value of X_i. But, the implicit triangulation has been optimized to compute marginal distributions for single variables, and it is possible to improve it to compute the marginal of two variables as in our case. The complexity of this phase is also decreased by using caching/hashing techniques, because some sub-paths can be shared between different pairs, or even a required potential can be directly obtained by marginalisation over one previously cached.

[1] Which happens frequently because we build the tree in depth, and (obviously) the create-node algorithm and the probabilistic inference engine are synchronised.

– $P(C = c|x_O) = \frac{P(C=c,x_O)}{P(x_O)}$. This probability can be easily obtained from previously described computations. We just use $P(x_O)$ that is computed in the first propagation (when selecting the variable to be placed in the root of our explanation tree) and $P(x_O^*) = P(C = c, x_O)$ which is computed in the current step (full propagation with x_O^* as evidence).

Though this method requires multiple propagations, all of them are carried out over a join tree obtained without constraining the triangulation sequence, and so it (generally) has a size considerably smaller than the join tree used for partial abductive inference over the same explanation set [17, 5]. Besides, the join tree can be pruned before starting the propagations [5].

3 Cases of Study: Explanation and Diagnosis

Because we are in an initial stage of research about the ET method, in order to show how it works and the features of the provided explanations, we found interesting to use some (toy) networks having a familiar meaning for us, to test whether the outputs are reasonable. We used the following two cases:

1. academe network: it represents the evaluation for a subject in an academic environment, let us say, university, for example. This simple network has got seven variables, as Fig. 3 shows. Some of them are intermediate or auxiliary variables. What this network tries to model is the final mark for a student, depending on her practical assignments, her mark in a theoretical exam, on some possible extra tasks carried out by this student, and on other factors such as behaviour, participation, attendance... We have chosen this particular topic because the explanations are easily understandable from an intuitive point of view.
   In this network we consider as evidence that a student has failed the subject, i.e., $x_O \equiv$\{*finalMark*=failed\}, and we look for the best explanations that could lead to this fact. We use \{*Theory, Practice, Extra, OtherFactors*\} as the explanation set. In this first approach we run our ET-based algorithm with $\beta = 0.0$, α=0.05|0.07 and criterion = max|min|avg. Figure 3 summarises the obtained results (variables are represented by using their initials).
2. gates network: this second net represents a logical circuit (Fig. 2.a). The network (Fig. 2.b) is obtained from the circuit by applying the method described in [8]. The network has a node for every input, output, gate and intermediate output. Again, we use an example easy to follow, since the original circuit only has got seven gates (two NOT-gates, two OR-gates and three AND-gates) and the resulting network has 19 nodes.
   In this case, we consider as evidence one possible input for the circuit (ABCDE=01010) plus an erroneous output (given such input), KL=10. Notice that the correct output for this case is KL=00, and also notice that from the transformation carried out to build the network, even when some gates are wrong the output could be correct (see [8]). So our evidence is ABCDEKL =

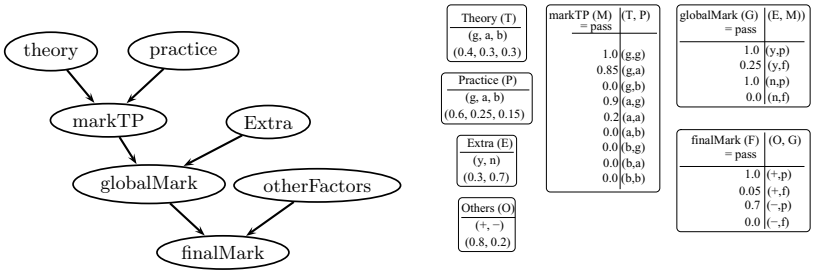

Fig. 1. Case of study 1: academe network

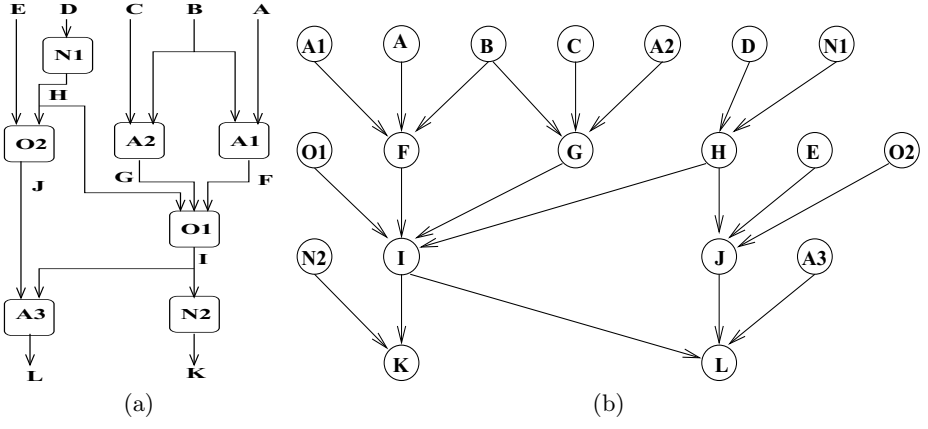

Fig. 2. (a) Original logic circuit. (b) Network gates obtained from (a) by using the transformation described in [8]

0101010 and we consider $X_E = \{A1, A2, A3, O1, O2, N1, N2\}$ as the explanation set with the purpose of detecting which gate(s) is(are) faulty. Figures 4 and 5 show the trees obtained for MI and GINI respectively. The same parameters as in the previous study case are used but $\beta = 0.05$.

3.1 Analysis of the Obtained Trees

The first thing we can appreciate from the obtained trees is that they are *reasonable*, i.e., the produced explanations are those that could be expected.

Regarding the academe network, when a student is failed, it seems reasonable that the most explicative variable is *theory* because of the probability tables introduced in the network. Thus, in all the cases *Theory* is the root node, and also in all the cases {*theory*=bad} constitutes an explanation by itself, being in fact the most probable explanation (0.56).

The other common point for the obtained ETs is that the branch with *theory* as good is always expanded. It is clear that being *theory* ok another reason must

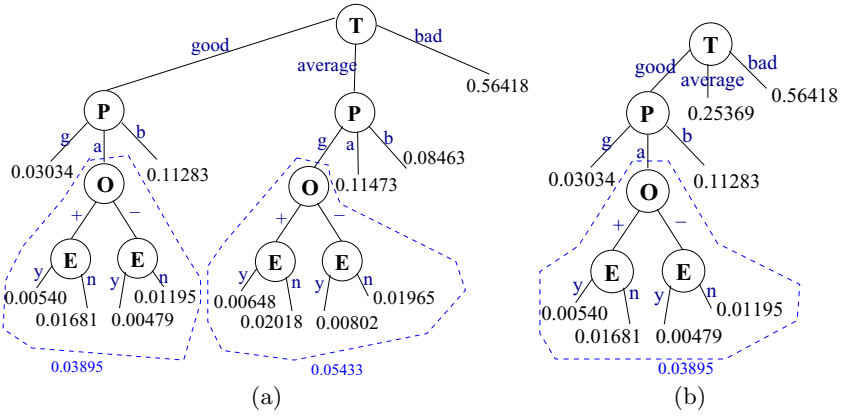

Fig. 3. Results for academe: (a) is the obtained tree for all MI cases except (MI,α=0.05,min) which produces tree (b) together with all (gini,α=0.05) cases and (gini,α=0.07,max). Finally it is necessary to remark that (gini,α=0.07,min|avg) leads to an empty tree, $\emptyset$, that is no node is expanded. β is 0.0

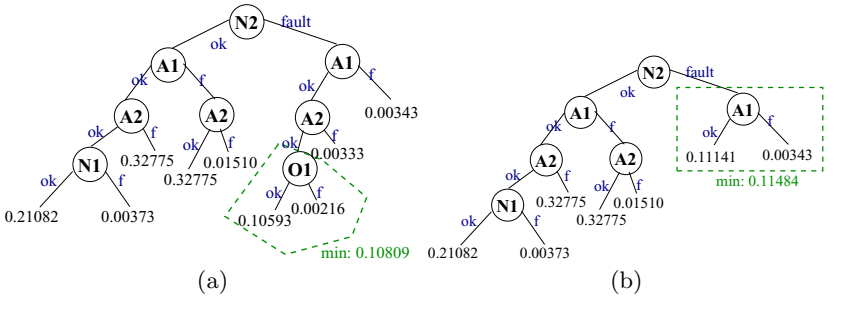

Fig. 4. Results for gates and MI: (a) is the obtained ET for (MI,α=0.05,max|avg) and also (MI,α=0.07,max); (b) is for (MI,α=0.07,avg). In both cases min prunes more the tree than avg, so the dotted area would not be expanded. β is 0.05

explain the failure. On the other hand, the main difference between the two ETs is that 3.(a) expands the branch {*theory*=average} and (b) does not. It is obvious that a bigger α makes the tree more restrictive. If this tree is expanded, as α=0.05 does, is because when *theory* is average it can be interesting to explore what happens with the *practical* part of the subject.

It is possible that variables that are not part of an explanation and that change their 'a priori' usual value or that have an important change in its 'a priori' probability distribution could be added to the explanation as this could be useful to the final user to fully understand some situations. An example can be the case of academe network with $\{theory = \text{good}, practice = \text{good}\}$. This branch is not expanded. The reason is that in this situation, the other variables have small entropy: *Extra* should be 'no' and *OtherFactors* '-', with high proba-

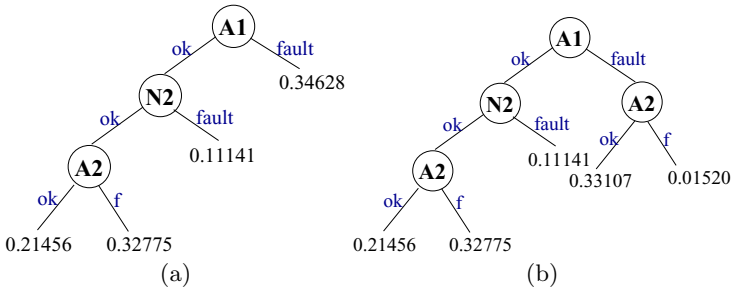

Fig. 5. Results for gates and GINI: (a) represents the tree for all gini cases, except (gini,α=0.05,max) which produces tree in part (b). β is 0.05

bility. This implies an important change with respect to 'a priori' probabilities for these values, and then these variables with their respective values could be added to the explanation $\{theory = \text{good}, practice = \text{good}\}$, making its meaning more evident.

We also used this case to show the influence of β. As $\beta = 0.0$ was used, we can see that some branches represent explanations with a very low posterior probability (those in the dashed area in Fig. 3), and so they will not be useful. The dashed areas in Fig. 3 represent the parts of the tree that are not constructed if we use $\beta \simeq 0.05$, which apart of producing a simpler and more understandable tree is also of advantage to reduce the computational effort (probabilistic propagations) required to construct the tree.

With respect to the resulting trees for the gates case, we can appreciate two clear differences: (1) GINI produces simpler trees than MI, and (2) the most explicative variable is different depending on the used measure. Regarding this last situation, we can observe in the circuit that there are many independent causes[2] (faults) that can account for the erroneous output. Choosing the AND gate A1 as GINI does is reasonable (as well as choosing A2) because AND gates have (in our network) greater a priori fault probability. On the other hand, choosing N2 as MI does is also reasonable (and perhaps closer to human behaviour) because its physical proximity to the wrong output. If we were a technician this would probably be the first gate to test. In this way, it seems that MI manages in some way the fact that the impact a node has in the value of the remaining nodes is attenuated with the distance in the graph.

Once the first variable has been decided, the algorithm tries to grow the branches until they constitute a good explanation. In some cases, it seems that some branches could be stopped early (i.e. once we know that N2=`fault`), but these situations depend on the thresholds used and it is clear that studying how to fix them is one of the major research lines for this work.

[2] However, it is interesting to observe that applying probability propagation, the posterior probability of each gate given the evidence, e.g. $P(A1|x_O)$, indicates that that for all the gates it is more probable to be ok.

Perhaps an interesting point is to think about why O1 is not selected by MI when N2=ok as could be expected given the distance-based preference previously noticed. But, if we look carefully the circuit, we can see that output L (which is correct) also receives as input the output of gate O1, so it is quite probable that O1 is working properly.

Of course, we get different explanations depending on the used measure, the value of α or the criterion, but in general we can say that all the generated explanations are quite reasonable. Finally, in all the trees there is a branch, and so an explanation which indicates that a set of gates are ok. Perhaps this cannot be understood as an explanation to a fault, but we leave it in the tree in order to provide a full partitioning. Some advice about these explanations can be given to the user by indicating for example if such explanations raise or not the probability of the fault with respect to its prior probability.

4 Conclusions and Further Work

This paper has proposed a procedure providing explanations at different level of complexity for the same evidence. The method gives a partition of the different possible scenarios for the explanation variables. The partition can have different levels of granularity depending on the values of the some variables.

We have shown that the results are reasonable in some simple examples and that computations are feasible: though they involve several probabilistic propagations, they are carried out in any junction tree associated to the original Bayesian network, without any restriction. The complexity can be controlled with two parameters (α and β) which at the same time will determine the level of detail of the provided explanations. In fact, the number of explanations (number of leaves in the explanation tree) is bounded by $O(1/\beta)$. Also the expansion of each node of the explanation tree can involve a quadratic number (with respect to the size of the explanation set) of probabilistic propagations, but these are partial propagations and usually we need far less computations than in a complete propagation.

We are conscious that this is an initial step and that additional work is necessary. In the future, we plan to test different criteria to select the variable to branch and to stop branching, specially in the last point where we aim to integrate the two parameters into a single one. Also, we want to make experiments with large Bayesian networks and refine the algorithms to improve its performance. We are studying different ways in which the results can be presented to the user: for example it is possible that variables that are not part of an explanation and that change their usual value (without evidence) could be added to the explanation, as this can be useful to the final user. Finally, for the evaluation of the different procedures it would be necessary a set of experiments in which final users rank the solutions according to their degree of satisfaction with them.

Acknowledgements. This work has been supported by FEDER and Spanish MCYT and MEC: TIC2001-2973-CO5-{01,05} and TIN2004-06204-C03-{02,03}.

References

1. E. Castillo, J.M. Gutiérrez, and A.S. Hadi. *Expert Systems and Probabilistic Network Models*. Springer-Verlag, 1997.
2. U. Chajewska and J. Y. Halpern. Defining explanation in probabilistic systems. In *Proc. of 13th Conf. on Uncertainty in Artificial Intelligence*, pages 62–71. 1997.
3. A.P. Dawid. Applications of a general propagation algorithm for probabilistic expert systems. *Statistics and Computing*, 2:25–36, 1992.
4. L.M. de Campos, J.A. Gámez, and S. Moral. Simplifying explanations in Bayesian belief networks. *International Journal of Uncertainty, Fuzziness and Knowledge-based Systems*, 9:461–489, 2001.
5. L.M. de Campos, J.A. Gámez, and S. Moral. On the problem of performing exact partial abductive inference in Bayesian belief networks using junction trees. In B. Bouchon, J. Gutierrez, L. Magdalena, and R.R. Yager, editors, *Technologies for Constructing Intelligent Systems 2: Tools*, pages 289–302. Springer Verlag, 2002.
6. L.M. de Campos, J.A. Gámez, and S. Moral. Partial abductive inference in Bayesian networks by using probability trees. In O. Camp, J. Filipe, S. Hammoudi, and M. Piattini, editors, *Enterprise Information Systems V*, pages 146–154. Kluwer Academic Publishers, 2004.
7. R. Dechter. Bucket elimination: A unifying framework for probabilistic inference. In *Proc. of the 12th Conf. on Uncertainty in Artificial Intelligence*, pages 211–219, 1996.
8. J. deKleer and B.C. Williams. Diagnosing multiple faults. *Artificial Intelligence*, 32(1):97–130, 1987.
9. J.A. Gámez. Abductive inference in Bayesian networks: A review. In J.A. Gámez, S. Moral, and A. Salmerón, editors, *Advances in Bayesian Networks*, pages 101–120. Springer Verlag, 2004.
10. M. Henrion and M.J. Druzdzel. Qualitative propagation and scenario-based schemes for explaining probabilistic reasoning. In P.P. Bonissone, M. Henrion, L.N. Kanal, and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence 6*, pages 17–32. Elsevier Science, 1991.
11. F.V. Jensen. *Bayesian Networks and Decision Graphs*. Springer Verlag, 2001.
12. C. Lacave and F.J. Díez. A review of explanation methods for Bayesian networks. *The Knowledge Engineering Review*, 17:107–127, 2002.
13. Z. Li and D'Ambrosio B. An efficient approach for finding the MPE in belief networks. In *Proc. of the 9th Conf. on Uncertainty in Artificial Intelligence*, pages 342–349. 1993.
14. R. E. Neapolitan. *Probabilistic Reasoning in Expert Systems. Theory and Algorithms*. Wiley Interscience, 1990.
15. D. Nilsson. An efficient algorithm for finding the M most probable configurations in Bayesian networks. *Statistics and Computing*, 9:159–173, 1998.
16. J.D. Park and A. Darwiche. Solving map exactly using systematic search. In *Proc. of the 19th Conf. in Uncertainty in Artificial Intelligence*, pages 459–468, 2003.
17. J.D. Park and A. Darwiche. Complexity results and approximation strategies for map explanations. *Journal of Artificial Intelligence Research*, 21:101–133, 2004.
18. J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann, 1988.
19. A. Salmerón, A. Cano and S. Moral. Importance sampling in Bayesian networks using probability trees. *Computational Statistics and Data Analysis*, 34:387–413, 2000.

20. P.P. Shenoy. Binary join trees for computing marginals in the shenoy-shafer architecture. *International Journal of Approximate Reasoning*, 17(2-3):239–263, 1997.
21. S.E. Shimony. Explanation, irrelevance and statistical independence. In *Proc. of the National Conf. in Artificial Intelligence*, pages 482–487, 1991.
22. S.E. Shimony. The role of relevance in explanation I: Irrelevance as statistical independence. *International Journal of Approximate Reasoning*, 8:281–324, 1993.
23. S.E. Shimony. Finding maps for belief networks is NP-hard. *Artificial Intelligence*, 68:399–410, 1994.