Decision analysis networks

Francisco Javier Díez, Manuel Luque, Caroline Leonore König, Iñigo Bermejo

***Abstract*—This paper presents decision analysis networks (DANs) as a new type of probabilistic graphical model. Like influence diagrams (IDs), DANs are much more compact and easier to build than decision trees, and are able to represent conditional independencies. Both IDs and DANs can represent symmetric problems, but DANs can also represent problems involving restrictions between the values of the variables (structural asymmetry) and partial orderings of the decisions (order asymmetry). Therefore, DANs can easily model and solve many real-world problems that IDs cannot. We argue that DANs compare favorably with other formalisms proposed for modeling asymmetric decision problems. Additionally, we offer OpenMarkov as an open source software tool for editing and evaluating DANs.**

I. Introduction

THE two formalisms most widely used for the representation and analysis of decision problems are decision trees (DTs) [1] and influence diagrams (IDs) [2]. DTs have the advantage of almost absolute flexibility, but also have three drawbacks: their size grows exponentially with the number of variables, they cannot represent conditional independencies, and they require in general a preprocessing of the probabilities [2], [3]; for example, medical diagnosis problems are usually stated in terms of direct probabilities, namely the prevalence of the diseases and the sensitivity and specificity of the tests, while DTs are built with inverse probabilities, i.e., the positive and negative predictive values of the tests. Even in cases with only a few chance variables, this preprocessing of probabilities is a difficult task. Moreover, IDs have the advantages of being very compact, easily representing conditional independence, and using direct probabilities, but they can only represent symmetric decision problems. We say that there is *structural asymmetry* when the value taken on by a variable restricts the domain of other variables. There is *order asymmetry* when several orderings of the decisions are possible [3], [4].

In practice, virtually all real-world problems are asymmetric, in particular all those that involve the possibility of getting additional information at a cost; for example, the variable that represents the result of a test can take some values only when the decision is to do the test. Several formalisms have been proposed for representing and solving asymmetric decision problems, but all of them have drawbacks, as we discuss in Section V; in fact, none of them has been used to build any real-world application. For this reason, we present a new formalism, called *decision analysis networks* (DANs), which can represent all symmetric problems and, in our opinion, can also represent real-world asymmetric decision problems more naturally than any other formalism. We developed it when trying to solve a complex medical decision problem: the mediastinal staging of non-small cell lung cancer.

The DAN for that medical problem, together with other DANs for the most famous asymmetric decision problems proposed in the literature, can be found in www.ProbModelXML.org/networks; they are encoded in ProbModelXML, a format for probabilistic graphical models [5]. OpenMarkov,[1] an open-source tool for probabilistic graphical models, can be used to view,

Technical Report CISIAD-14-01, UNED, Madrid, Spain, 2014.

F. J. Díez, M. Luque and I. Bermejo are with the Department of Artificial Intelligence, UNED, Madrid, 28040 Spain (e-mail: {fjdiez,mluque,ibermejo}@dia.uned.es).

Submitted to *IEEE Transaction on Pattern Analysis and Machine Intelligence.*

[1]See www.openmarkov.org.

edit, and evaluate DANs.

The rest of the paper is structured as follows: first, we introduce the n-test problem, which will serve us to illustrate the properties of DANs and to compare different formalisms. Then Section II presents the definition of DANs, Section III explains how to convert any DAN into a DT, Section IV shows that symmetric DANs are equivalent to IDs, Section V compares DANs with other formalisms, and Section VI contains the conclusions and some proposals for future work.

Example 1. The *n-test problem* consists in deciding how to treat a patient that may suffer from a certain disease. After an initial examination of the symptoms, the doctor may order one or several of n available tests, each one having a cost. Each test can be performed once at most and its result will be known immediately. The doctor has to decide which tests to perform and in which order.

In the simplest version of the problem, we assume that there is only one symptom, and all the variables are dichotomous, i.e., that the disease and the symptom are either present or absent, and the result of each test is either positive or negative. The diabetes problem [6] is an instance of the two-test problem.

II. Definition of a DAN

In this section we define DANs by describing the elements that compose them and the mechanisms that indicate the availability of information. We also discuss the meaning of the different types of links.

A. Graph and variables of a DAN

In this paper we represent variables with capital letters (X) and their values with lower-case letters (x). A bold upper-case letter ($\mathbf{X}$) denotes a set of variables and a bold lower-case letter ($\mathbf{x}$) denotes a configuration of them, i.e., the assignment of a value to each variable in $\mathbf{X}$. The set of variables of a DAN, $\mathbf{V}$, is partitioned into three disjoint subsets: chance variables $\mathbf{C}$, decisions $\mathbf{D}$, utility variables $\mathbf{U}$. Chance variables represent real-world properties that are not under the direct control of the decision maker, decisions correspond to actions that are under the direct control of the decision maker, and utilities represent their preferences. A DAN also has an acyclic directed graph such that each node represents a variable; hence we will speak indifferently of nodes and variables. When the graph has a link $X \rightarrow Y$, we say that X is a parent of Y and Y is a child of X. The set of parents of a node X is denoted by $Pa(X)$, and $pa(X)$ represents a configuration of them. When there is a directed path from X to Y, we say that X is an ancestor of Y and Y is a descendant of X. In this paper we will assume that utility nodes do not have children.[2] We also assume that all the decisions and chance variables are discrete.

The DAN in Figure 1 contains 4 chance variables, drawn as rounded rectangles, 3 decisions, drawn as rectangles, and 3 utility nodes, drawn as hexagons.

B. Restrictions

A *restriction* associated to a link $X \rightarrow Y$, such that X and Y are chance or decision variables, is a pair (x, y), where x is a value of X and y is a value of Y. It means that variable Y cannot take the value y when X takes the value x. In OpenMarkov the restrictions between X and Y are represented by a table with a column for each value of X and a row for each value of Y; when the values x and y are compatible, i.e., when there is no restriction (x, y), the corresponding cell contains a 1 and is colored in green; otherwise it contains a 0 and is colored in red—see Figure 2. When all the values of Y are incompatible with a value x, as in this figure, we say that there is a *total restriction* and denote it by (x, Y). It implies that in some scenarios the variable Y does not exist; we will see it more clearly when expanding

[2]In standard IDs the parents of a utility node can only be decisions and chance nodes. Tatman and Shachter [7] introduced super-value nodes, i.e., utility nodes whose parents are all utility nodes. This can be further generalized by allowing utility nodes to have any combination of the three types of nodes as their parents. Thus, if a utility node U_1 has another utility node U_2 as a child, then U_2 must be absorbed before applying the inference algorithm described in this paper, as follows: if the utility functions for these nodes are $u_1(\mathbf{x})$ and $u_2(u_1, \mathbf{y})$, respectively, U_1 is removed, its parents become parents of U_2, and the new utility function for this node is $u_2(\mathbf{x}, \mathbf{y}) = u_2(u_1(\mathbf{x}), \mathbf{y})$.

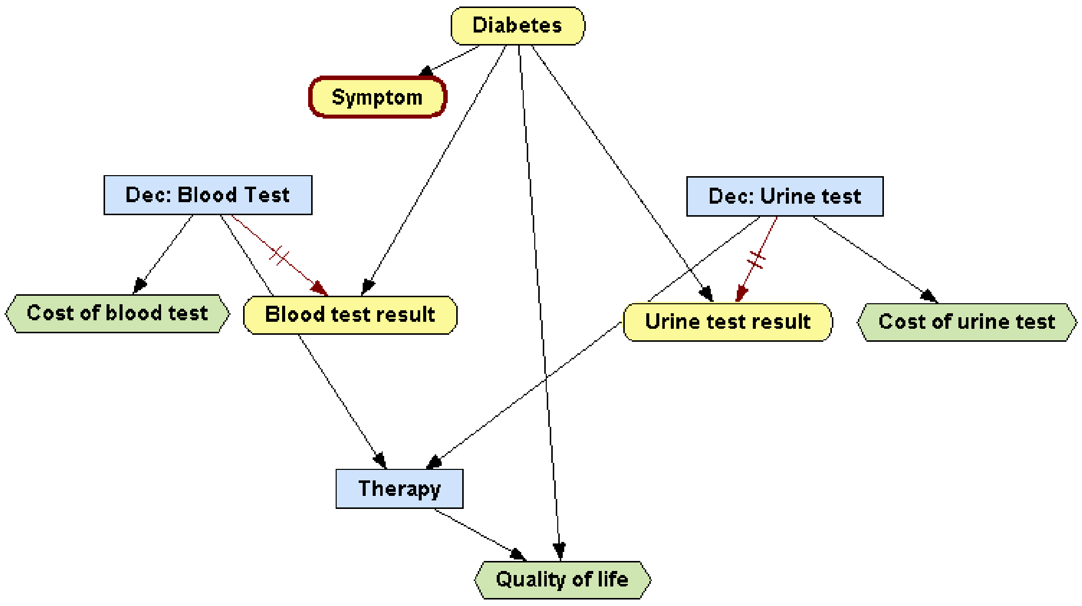
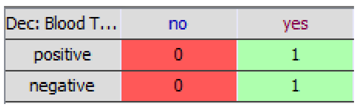


Figure 1. A DAN for the diabetes problem. The decision about a test may depend on the presence of the symptom and on the result of the other test. There is no constraint on the order of the tests.

the equivalent decision tree. In OpenMarkov, a link having a total restriction, such as the link *Dec: Blood Test* $\rightarrow$ *Blood test result* in Figure 1, is marked with a short perpendicular double line.



Dec: Blood T...	no	yes
positive	0	1
negative	0	1

Figure 2. Compatibility table for the link *Dec: Blood test* $\rightarrow$ *Blood test result* in Figure 1. It contains two restrictions, (*no*, *positive*) and (*no*, *negative*), which mean that when the test is not performed, it gives neither a positive nor a negative result.

When there is a restriction (x, y) but every value of X is compatible with at least one value of Y, we say that there is a *partial restriction*, and in OpenMarkov the corresponding link is marked with a short perpendicular line. Thus, the DAN for the reactor problem [8], [9] (Fig. 3) has a partial restriction for the link *Result of test* $\longrightarrow$ *Build decision*, because a bad test result prevents the construction of an advanced reactor, as shown in Fig. 4, but every value of *Result of test* is compatible with at least one value of *Build decision*.

If Y is a chance variable, the restriction (x, y)—associated to the link $X \rightarrow Y$, which implies that $X \in Pa(Y)$—means that $P(y|pa(Y)) = 0$ for all the configurations of $Pa(Y)$ in which $X = x$; we will explain this in further detail in the the next section. Therefore, when Y is a chance variable, there is not a significant difference between attaching a partial restriction to a link and setting to 0 the corresponding cells in the conditional probability table.[3]

In contrast, if D is a decision, the restriction (x, d) means that when $X = x$ the decision maker cannot choose the option d—see again the example in Figure 4. Therefore, a partial restriction between X and D, where D is a decision, means that when evaluating the DAN, we have to choose among the allowed values of Y the one that maximizes the expected utility. In summary, a partial restriction associated to a link

[3]However, from the point of view of knowledge engineering, it may be useful to declare a partial restriction when building the graph of the network (qualitative information), instead of just setting the probability to 0 in the corresponding cell of the probability table (quantitative information), because the more structural information we have about a model, the easier it will be to maintain it, avoiding inconsistencies in its numeric parameters. Additionally, restrictions can be useful when expanding an equivalent DT (see below), when explaining the reasoning, and when performing sensitivity analyses.

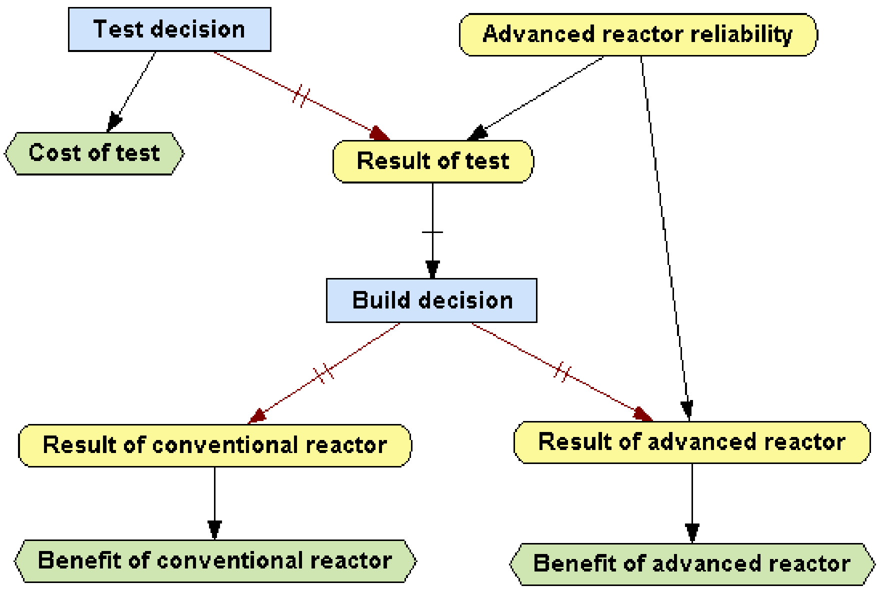
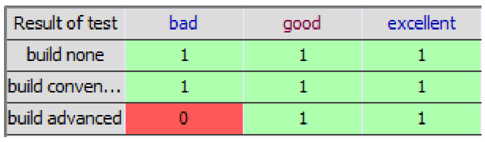
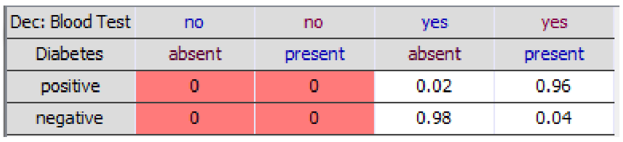

Test decision
Advanced reactor reliability
Cost of test
Result of test
Build decision
Result of conventional reactor
Result of advanced reactor
Benefit of conventional reactor
Benefit of advanced reactor

Figure 3. A DAN for the reactor problem. The main decision is which type of reactor to build, if any.



Result of test	bad	good	excellent
build none	1	1	1
build conven…	1	1	1
build advanced	0	1	1

Figure 4. Compatibility table for the link *Result of test* → *Build decision* in Figure 3. It means that s a bad result prevents the construction of an advanced reactor.



Dec: Blood Test	no	no	yes	yes
Diabetes	absent	present	absent	present
positive	0	0	0.02	0.96
negative	0	0	0.98	0.04

Figure 5. Probabilistic potential associated to the chance node *Blood test result*. The first two columns are 0 due to the restrictions shown in Figure 2. The third column defines a conditional probability distribution for *Blood test result* (0.98 is the specificity of the test). The fourth column defines another conditional probability distribution for the same variable (0.96 is the sensitivity of the test).

$X \to Y$ affects the edition of the network if Y is a chance variable and affects inference if Y is a decision.

C. Potentials

A potential ψ is a function that maps each configuration $\mathbf{x}$ of a set of variables $\mathbf{X}$ onto $\mathbb{R}$, i.e., $\psi(\mathbf{x})$ is a real number. Each chance node Y has an associated conditional probability potential, denoted by $\psi(y|pa(Y))$. When there is a restriction (x, y) and $X = x$ in the configuration $pa(Y)$, then $\psi(y|pa(Y)) = 0$, because y is incompatible with x. If at least one value of Y is compatible with all the values of its parents in the configuration $pa(Y)$, then $\psi(y|pa(Y))$ is a conditional probability distribution for Y given that configuration, as shown in Figure 5.

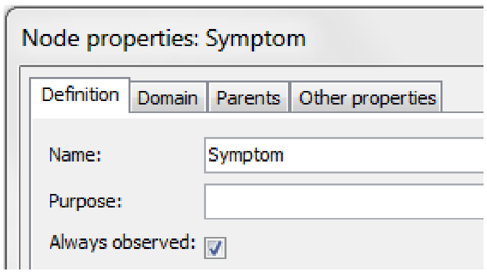
Similarly, each utility node U has an associated potential, $u(pa(U))$.

D. Representing the availability of information

In DANs there are two ways to indicate when a variable becomes observed: *always-observed variables* and *revelation arcs*. In both cases, we rely on the *no-forgetting hypothesis*, which means that when the value of a variable is known, then it is known for any subsequent decision.

1) Always observed variables: A chance variable declared as *always observed* means that its value is known before making any decision. In Figure 1, the node *Symptom* is marked with a red thick border to indicate that it is always observed (see also Figure 6).

2) Revelation links: Given a link $X \to Y$, such that Y is a chance variable, we can declare that



Node properties: Symptom

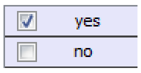
Name: Symptom

Purpose:

Always observed: ✓

Figure 6. The chance variable *Symptom* is declared as *always observed* because we know whether it is *present* or *absent* without performing any test.

certain values of X *reveal* the value of Y; we then say that $X \to Y$ is a *revelation link*. In general, X is a decision node, as in Figure 7, but it might also be a chance node; in this case, it means that Y is known only if a fortuitous event X occurs and the value of X is observed. The *revelation conditions* are the values of X that reveal Y. In OpenMarkov revelation links are colored in red. Figure 7 shows the revelation conditions for the link *Dec: Blood test* $\to$ *Blood test result* for the DAN of the diabetes problem.



✓	yes
☐	no

Figure 7. Revelation conditions for the link *Dec: Blood test* $\to$ *Blood test result*. The value *yes* (i.e., the decision to test) reveals the result of the test, but the value *no* (not to test) does not.

When Y is an always-observed node, it does not make sense to declare a revelation condition for any link $X \to Y$. It is also contradictory to declare x as a revelation condition for Y when there is a total restriction (x, Y). The current implementation of DANs in OpenMarkov does not check for these consistency properties, but in this paper we assume that all DANs satisfy them.

E. Summary: The meaning of links

In DANs a link $X \to Y$ may have five meanings:

1) *Causal influence:* For example, the links *Diabetes* $\to$ *Symptom* and *Diabetes* $\to$ *Blood test result* mean that the presence of the disease affects the probability of having the symptom and the outcomes of this test respect. Y must be a chance node.
2) *Functional dependence:* For example, the four links pointing at *Cost of blood test*, *Cost of urine test*, and *Quality of life* denote which variables affect directly the decision maker's preferences. Y must be a utility node.
3) *Temporal order:* For example, the link *Dec: Blood Test* $\to$ *Therapy* indicates which decision is made first. Y must be a decision.
4) *Revelation:* In general X is a decision, but it may also be a chance node, as explained above; Y must be a chance node.
5) *Restriction:* X must be a chance node or a decision, because utility nodes do not have children; Y must be a decision or a chance node.

The first three meanings are the same as in IDs. Revelation links in DANs replace information links in IDs. Restrictions are a novelty of DANs with respect to IDs.

III. Equivalent decision tree

In this section we explain how to convert a DAN into a DT, with the main purpose of providing a formal semantics for DANs. In the first phase the algorithm builds the structure of the tree and in the second it assigns the probability and utility values. Then the DT can be evaluated with the standard roll-back algorithm [10].

We denote each node in the tree by the variable that it represents, for instance X, and each branch departing from this node by its associated value, x. In the first phase, the algorithm operates recursively on a tree whose leaf node represents DANs or utility values. The tree is initialized by placing the original DAN as the only node in the tree. Each node representing a DAN is expanded until all the leaves in the tree are utility nodes (values).

When expanding a DAN, there are five possible cases; if one holds, the next cases are ignored:

Case 1. The DAN has always-observed nodes: Then, select one of these nodes, say X, that is not a descendant of other always-observed nodes. Put this variable as a node in the tree, instead of the DAN. For each value x, create a copy of the DAN and:

1) delete the node X from the new DAN;
2) if x reveals the value of other variables, mark them as always observed in the new DAN;
3) if there is a total restriction (x, Y), remove Y and all the descendants of Y that are not descendants of another decision;
4) if there is a restriction (x, y), remove y from the domain of Y; and
5) put the new DAN in a branch x outgoing from node X.

Case 2. A decision D is an ancestor of all the other decisions (i.e., D is the first decision to make): Then, put D as a node in the tree, create a copy of the DAN for each value d, and execute the same 5 steps as in the previous case, with one exception: in step 2, if d reveals the variable Y that is a descendant of other decisions $\{D'_1, \ldots D'_n\}$, draw a link $D'_i \to Y$ for $i \in \{1, \ldots, n\}$ and declare that each value d'_i of D'_i reveals Y.[4]

If d reveals a variable Y that is not a descendant of any other decision, then declare Y as always observed, as in case 1.

Case 3. There are n decisions $\{D_1, \ldots, D_n\}$ such that there is no directed path between any two of them (i.e., any of them can be made first): Then, replace the DAN with a meta-decision node indicating which decision will be made first. Create n copies of the DAN. In the i-th copy, draw $n-1$ links $D_i \to D_j$ $(j \neq i)$ to indicate that D_i will be the first decision. Put each new DAN in a branch outgoing from the meta-decision node.

Case 4. The network has chance nodes (which are not always-observed; otherwise we would be in case 1): Then, select one of them, say X, and replace the DAN with a node X in the tree. For each value x of X, create a copy of the DAN, remove X, and put the new DAN at a branch x.[5]

Case 5. Otherwise (i.e., the network only has utility nodes): Then, replace the DAN with a utility node whose value is the sum of the utility nodes remaining in the DAN.

As an example, consider the conversion of the diabetes DAN into a DT. The tree is initialized by making this network the root of the tree. In the first iteration, we are in case 1: the DAN has one always-observed node, *Symptom*; then this node replaces the original DAN as the root of the tree (it is the S node in Fig. 1); two branches are added, one for *present* $(+s)$ and one for *absent* $(\neg s)$. Each leaf is a DAN in which the node *Symptom* has been removed, and has two decisions, *Dec: Blood Test* and *Dec: Urine Test*, with no directed path between them (case 3). Therefore we put in the tree a meta-decision node OD (order of the decisions) with two branches. The DAN at the bt branch has a link *Dec: Blood Test* $\to$ *Dec: Urine Test*, which indicates which is the first decision (case 2). This DAN is later replaced with a node BT (*Dec: Blood Test*) having two branches: $+bt$ (do test) and $\neg bt$ (do not test). In the DAN at the $\neg bt$ branch the node *Dec: Blood Test* has been removed when putting this variable in the tree; *Blood test result* has been removed because of the constraint in Figure 2. In the DAN at the $+bt$ branch the node *Dec: Blood Test* has been also removed, but the node *Blood test result* remains; it is now marked as always observed because of the revelation condition in Figure 7. This DAN gives rise to the branches $+b$ (positive test result) and $\neg b$ (negative) in Figure 8. Each of the three UT nodes in this figure is the result of expanding a DAN in which *Dec: Urine Test* was the first decision to make (case 2). When the decision is $+ut$ (do test), the variable U is marked as always-observed and in the subsequent expansion of the DAN it is put in the corresponding branch of the tree. When the decision is $\neg ut$ (do not test), U is removed. The node Th (*Therapy*) is the result of expanding a DAN in which this was the only

[4]The reason for this exception is that if Y is descendant of D'_i then this decision is a cause of Y and therefore Y cannot be known before making D'_i.

[5]In this case, the restrictions and revelation conditions are ignored. In fact, if X can never be observed, then no link $X \to Y$ should have total restrictions or revelation conditions.

decision (again case 3). The node D (*Diabetes*) results from a DAN having one chance node and three utility nodes (case 4). The expansion of the DAN that was at this node, D, generates the two utility nodes shown in this figure (case 5).

The algorithm always terminates: in cases 1, 2, and 4 the expansion of a DAN with n nodes generates a finite number of DANs with $n-1$ nodes; in case 3, the new DANs still have n nodes, but every one is in case 2, which ensures the elimination of a decision node when expanding each new DAN; therefore, case 3 can occur only a finite number of times.

We should also take into account that a decision that can reveal a variable should be made earlier than the decisions that cannot. The formal justification is the same as in the case of unconstrained influence diagrams [11]. For example, in Figure 1, the decision *Therapy* does not reveal any variable. Therefore, even if the links *Dec: Blood Test* $\rightarrow$ *Therapy* and *Dec: Urine Test*$\rightarrow$ *Therapy* were not present in 1, the node *OD* in Figure 1 should have only two outgoing branches, *bt* and *ut*, because *Therapy* should always be made after *Dec: Blood Test* and *Dec: Urine Test*. Therefore, the above algorithm should be refined as follows: when we are in case 3 and the variables revealed by D_i are a proper subset of those revealed by D_j, then remove D_i from the set of candidates to be the first decision. If only one candidate remains, proceed as in case 2.

The second phase of the algorithm consists of assigning the numerical parameters of the DT. Each path from the root node to a leaf node defines a *scenario*, i.e., a configuration of the variables in that path. The utility of a leaf is the sum of the utility functions in that scenario; for example, the utility for the upper utility node in Figure 8 is $u_1(+bt) + u_2(+ut) + u_3(+tr,+d)$, where u_1, u_2, and u_3 are the utility functions for the nodes *Cost of blood test*, *Cost of urine test*, and *Cost of therapy*, respectively. The probability of a scenario is the product of the conditional probabilities involved in it; for example, the probability of the upper scenario is $P(+s|+d) \cdot P(+b|+d,+bt) \cdot P(+u|+d,+ut) \cdot P(+d)$. Initially we compute the probability of each branch as the sum of the probabilities of the scenarios containing it; then, we normalize the probabilities of the branches going out from each chance node. This way we obtain an equivalent DT for the DAN.

This tree can be evaluated with the standard roll-back algorithm, which proceeds from the leaves to the root: the utility of a decision node is the maximum of the utilities of the nodes at its branches, and the utility of a chance node is the average of the utilities of the nodes at its branches, weighted by their probabilities.

The above algorithm may generate different DTs for one DAN because in cases 1 and 4 there may be several nodes to select. However, these trees only differ in the order of the set of chance variables placed between two consecutive decisions. These DTs are equivalent in the sense that there is a one-to-one correspondence between the decision nodes of each pair of trees, and the optimal policies and the maximum expected utility are the same for all the trees.

IV. Symmetric DANs

In this section we analyze the relation between symmetric DTs, IDs, and symmetric DANs.

Definition 2. A DT is *symmetric* if

1) every path from the root to a leaf has the same variables and in the same order, and
2) every node representing the variable X has an outgoing branch for each value x.

Definition 3. A DAN is *symmetric* if

1) it has no restrictions,
2) if a value of a variable X reveals Y, then all the other values of X reveal Y, and
3) a directed path connects all the decisions.

Proposition 4. *The algorithm in Section III applied to a symmetric DAN generates a symmetric DT (provided that when it has several chance nodes to select—in cases 1 and 4—it always selects the same node).*

Proof: The third condition in Definition 3 induces a total ordering of the decisions in the DAN:

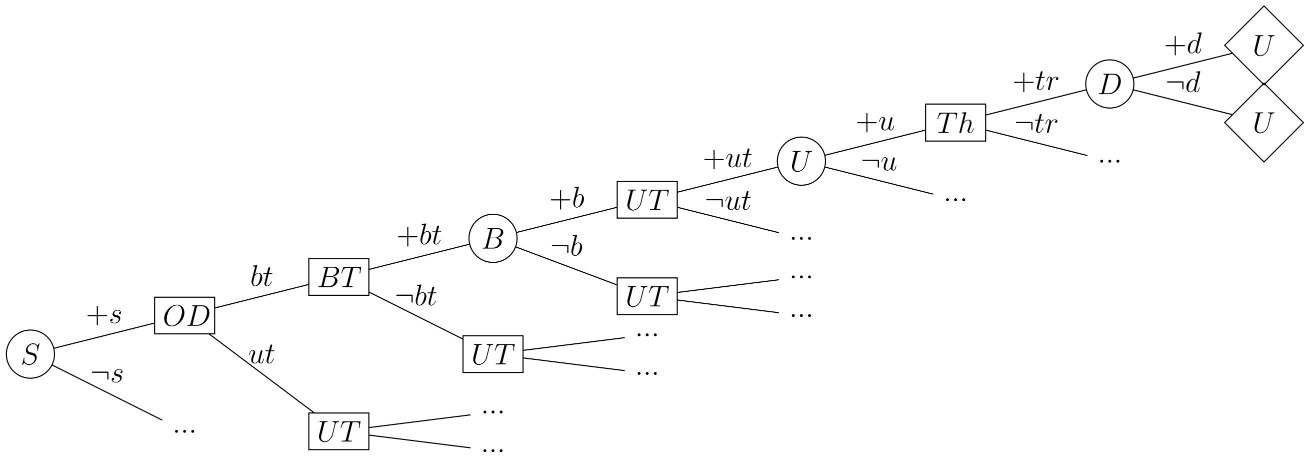


Figure 8. A decision tree for the diabetes DAN (Fig. 1).

$D_1 < \ldots < D_n$. The first and second conditions imply that when expanding the DT, the set of candidate chance nodes is the same for every branch of the tree. Therefore the first condition for a symmetric DT is always met. The second is guaranteed by the first condition in the definition of a symmetric DAN. ■

A symmetric DAN[6] having n decisions $\{D_1, \ldots, D_n\}$ induces a partition of $\mathbf{C}$, the set of chance variables, into $n+1$ disjoint subsets, $\{\mathbf{C}_0, \ldots, \mathbf{C}_n\}$, such that $\mathbf{C}_0$ is the subset of always-observed variables; a chance variable X belongs to $\mathbf{C}_i$, with $1 \leq i < n$, if and only if the decision D_i reveals X and no other decision D_j with $j < i$ reveals it; $\mathbf{C}_n$ is the set of the variables that are not revealed by any decision.

Theorem 5. *The maximum expected utility for a symmetric DAN is*

$$MEU = \sum_{\mathbf{c}_0} \max_{d_0} \ldots \sum_{\mathbf{c}_{n-1}} \max_{d_n} \sum_{\mathbf{c}_n} \prod_{X \in \mathbf{C}} P(x|pa(X)) \sum_{U_i \in \mathbf{U}} u_i(pa(U_i)) \; , \qquad (1)$$

and the optimal policy for decision D_i is

$$d_i^{opt}(\mathbf{c}_0, d_1, \ldots, \mathbf{c}_{i-1}) = \arg\max \sum_{\mathbf{c}_i} \max_{d_{i+1}} \ldots \sum_{\mathbf{c}_{n-1}} \max_{d_n} \sum_{\mathbf{c}_n} \prod_{X \in \mathbf{C}} P(x|pa(X)) \sum_{U_i \in \mathbf{U}} u_i(pa(U_i)) \; . \qquad (2)$$

Proof: These equations are a consequence of the algorithm in Section III, which builds the DT, and the roll-back algorithm, which evaluates it. ■

The properties of symmetric DANs are very similar to those of IDs. Namely, an ID with n decisions $\{D_1, \ldots, D_n\}$ induces a partition of $\mathbf{C}$, the set of chance variables, into $n+1$ disjoint subsets, $\{\mathbf{C}_0, \ldots, \mathbf{C}_n\}$ such that a chance variable X belongs to $\mathbf{C}_i$, with $0 \leq i < n$, if and only if there is an information link $C \rightarrow D_{i+1}$ and there is no link $X \rightarrow D_j$ for a previous decision D_j $(j \leq i)$. Both in a DAN and in an ID, $\mathbf{C}_0$ is the set of variables observed before making the first decision, $\mathbf{C}_i$ is the set of variables observed after decision D_i and before D_{i+1}, and $\mathbf{C}_n$ is the set of unobservable variables. Therefore the transformation of a symmetric DAN into an ID is straightforward: for every chance variable X, if X is always-observed, i.e., if $X \in \mathbf{C}_0$, draw a link $X \rightarrow D_1$; if X is revealed by decision D_i (and not revealed by any previous decision), i.e., if $X \in \mathbf{C}_i$, draw a link $X \rightarrow D_{i+1}$. The conversion of an ID into a symmetric DAN is analogous. Additionally, Equations 1 and 2 are valid both for IDs and symmetric DANs.

[6]If an always-observed variable X reveals Y in a symmetric DAN, we can declare Y as always-observed and remove the revelation conditions from X to Y. Similarly, if a decision D reveals X, and X reveals Y, we may declare that D reveals Y and remove the revelation conditions from X to Y. Additionally, it is irrelevant whether the last decision, D_n, reveals any variable or not, because this information has no effect on any decision. Therefore, in order to simplify the exposition in this section, we assume that in symmetric DANs only the decisions $\{D_1, \ldots, D_{n-1}\}$ can reveal any variables.

As a consequence of this close relation between the two types of networks, any algorithm for IDs can be used to evaluate symmetric DANs. In general, it is not even necessary to convert the DAN into an ID; it is enough to infer the partition $\{\mathbf{C}_0, \ldots, \mathbf{C}_n\}$ from the graph of the DAN. For example, the variable elimination algorithm [12], [13], which consists essentially of applying Equations 1 and 2, is identical in both cases. The arc reversal algorithm [14], [15] can also be applied to DANs: the only difference is that in an ID the first nodes to be removed (i.e., $\mathbf{C}_n$) are those that have no outgoing information links, while in a DAN they are those that have no incoming revelation links.

V. Comparison of different formalisms

In this section we examine seven formalisms for decision analysis. First we compare DANs with IDs, which are the standard framework for decision analysis, and then (in Sec. V-B) we analyze six formalisms for asymmetric decision problems.

A. DANs vs. IDs

We have shown that symmetric DANs are equivalent to IDs. Therefore, they are equally suited for representing symmetric problems.

Problems having only structural asymmetry can be symmetrized by using two modeling tricks [16]. First, a total restriction on a chance variable X can be modeled by adding a dummy state. For example, the one-test problem can be represented with an ID having three values for the result of the test: *positive*, *negative*, and (the dummy state) *not-performed*. The drawback of this trick is that it complicates the edition of the probability tables of the variables that have dummy states. It also makes the evaluation less efficient, due to the enlarged probability and utility tables, and complicates the interpretation of the results, because some policies contain configurations that can never occur, such as (*do not test*, *positive*), (*do not test*, *negative*), and (*do test*, *not-performed*). Even for this small problem it is advantageous to use a DAN, which does not need dummy states. Another modeling trick can be used in an ID to model a restriction (x, d), where D is a decision: to add a dummy utility node U such that $u(x, d) = -\infty$ and $u(x', d') = 0$ for the other configurations of X and D. This trick complicates the graph of the ID and makes inference less efficient than when using a DAN.

Order asymmetry poses a much more serious difficulty to IDs. For example, if we try to model the n-test problem with an ID, we need a n decision nodes for the tests, $\{T_1, \ldots T_n\}$, each having $n+1$ options; the extra option "do no test". We then need $n(n-1)/2$ dummy utility nodes to represent the restrictions that tests cannot be repeated, and that if the $i-th$ decision is not to test, then all subsequent test decisions are not to test. We also need n chance variables, $\{R_1, \ldots R_n\}$, for the results of the tests. The meaning of R_i depends which test has been done in the i-th place. Each R_i has $m+1$ possible states, where m is the maximum of the number of outcomes of the tests; the extra state is "not performed". If a test has fewer than m outcomes, some of the states of the R's are meaningless. Combining continuous and discrete tests in the same model is impossible. Specifying the conditional probabilty tables for the R's is cumbersome; in fact, it is the same table for every R_i, which complicates the maintenance of the model and makes sensitivity analysis virtually impossible. If a new test is added to the model, the domains of the all the T's and the R's must be revised, as well as the conditional probability tables of the R's and the tables of dummy utility nodes; the effort is comparable to building the new model from scratch.

In contrast, the DAN for the n-test problem does not need dummy states, nor dummy utility nodes, nor links between the T's; each T_i is dichotomic and represents the decision about a single test, the states of R_i correspond to the outcomes of the i-th test, thus allowing to combine continuous and discrete tests in the same model (see below); each parameter is encoded only once, the size of the model is proportional to n, and adding a new test to the model is straightforward.

B. DANs vs. other formalisms for asymmetric decision problems

Because of the difficulty of representing asymmetric problems with IDs, several alternative formalisms have been proposed. In this section we briefly compare six of them: influence diagrams with constraints (IDCs) [16], asymmetric influence diagrams (AIDs) [17], sequential valuation networks (SVNs) [6], [18], unconstrained influence diagrams (UIDs) [11], sequential influence diagrams (SIDs) [4], and decision analysis networks (DANs)—see also the comparisons in [3], [9]. Table I summarizes the main features of each method.

The first column indicates whether a formalism needs dummy states to "symmetrize" the problems containing structural asymmetries, as explained in the previous section. IDs and UIDs suffer from this drawback.

The second column tells us which models require a total ordering of the decisions. Obviously, these formalisms are inadequate for problems having order asymmetry.

Finally, the third column shows which methods need information links. These links, which indicate whether a variable is known for a decision, complicate the graph when there are many sequences of decisions that can reveal a variable. In the n-test problem, it leads to a link from each result-of-test variable to each test decision. König [19], who has built IDCs, AIDs, SVNs, UIDs, SIDs, and DANs for three of the asymmetric problems proposed in the literature, explains why the formalisms that use information links result in cumbersome diagrams even for the diabetes problem, let alone for the n-test problem.

UIDs and DANs share the advantage of not needing such links, but differ from each other in two important aspects. First, UIDs cannot represent restrictions and are therefore inappropriate for problems having structural asymmetries—see the first column. Second, in a UID an observable variable becomes observed when all its ancestral decisions have been made, independently of the option chosen for each decision; in a DAN a revelation link $X \rightarrow Y$ indicates which values of X reveal Y. For example, the decision to do a test reveals its result, while the decision not to do it does not reveal anything. This property cannot be modeled directly with a UID: it would be necessary to add a dummy state to the result-of-test variable, as explained in Section V-A.

In summary, IDCs, AIDs, and SVNs were designed to represent structural asymmetries, while UIDs were designed only for order asymmetry. SIDs were designed for both; apparently this was a significant improvement, but the use of information links in SIDs makes this formalism unsuitable for representing order asymmetry. Moreover, a SID consists of two overlapping graphs, one for the probability and utility relations, and the other for modeling the sequence of decisions and observations, which makes the model difficult to build, to communicate to experts, and to maintain.

Furthermore, the five frameworks proposed previously for asymmetric problems have been illustrated only with toy problems, which in many cases were designed just to illustrate the strengths of the new formalism. None of these formalisms has tried to solve all the asymmetric problems proposed previously in the literature. Additionally, no software tool, either commercial or open-source, has implemented them any of these formalisms.[7] These limitations may explain why none of these frameworks, proposed as an alternative to IDs, has been used to build any real-world application.

In contrast, DANs were developed for a complex medical problem: the mediastinal staging of non-small cell lung cancer based on several tests: CT-scan, PET, TBNA, EBUS, EUS, and mediastinoscopy; this is very similar to the n-test problem. Later another DAN was built for deciding when to implant a knee prosthesis [23]. These medical networks are available at www.ProbModelXML.org/networks, together with other DANs for the problems proposed previously in

[7]UIDs are a partial exception to this assertion: they were implemented in Elvira [11], [20], [21], [22] with the goal of measuring the efficiency of some inference algorithms, but Elvira's graphical user interface for building UIDs is not mature, and no document explains how to build UIDs in Elvira.

Table I
Main features of several methods for representing decision problems.

	dummy states	total order	inform. links
IDs [2]	yes	yes	yes
IDCs [16]	no	yes	yes
AIDs [17]	no	yes	yes
SVNs [18]	no	yes	yes
UIDs [11]	yes	no	no
SIDs [4]	no	no	yes
DANs	no	no	no

the literature on asymmetric decision formalisms: the used car buyer problem [24], the reactor problem [8], the diabetes problem [6] (which is another instance of the n-test problem), the dating problem [17], and the king's problem [11]; all these networks were built using OpenMarkov, an open source software tool for editing and evaluating DANs. Finally, DANs can also solve other real-world problems, such as troubleshooting different types of devices, which is very similar to the n-test problem.

VI. Conclusion

In this paper, we have proposed a new type of probabilistic graphical model, called *decision analysis networks* (DANs), and explained how to convert each DAN into a DT. The main purpose of this conversion is to give a formal semantics for our representation language—in other formalisms proposed in the literature, the semantics is described only by means of an example. Any other algorithm for DANs should prove that it returns the same expected utility and optimal strategy as the expansion and evaluation of a DT.

A second reason for this conversion is that many decision analysts will understand DANs much better if they can examine the equivalent DT. In fact, the conversion of IDs into DTs was one of the explanation facilities proposed in [25]. In some fields, such as medicine, DTs are the standard analysis tool, while IDs are almost unknown and rarely used [26], let alone the frameworks for asymmetric decision problems.

In principle, DTs are more flexible than DANs. However, after reviewing 23 books that describe DTs, most of them about medical decision making, we have found no example that cannot be modeled with a DAN. On the contrary, we have mentioned in this paper several DANs whose size prevents the construction of a DT.

On the other hand, for every ID there is an equivalent symmetric DAN (cf. Sec. V-A). Problems having only structural asymmetry can be modeled with IDs by adding dummy states and/or dummy utility nodes, but these modeling tricks complicate the network and make inference less efficient than when using DANs. Problems having order asymmetry are very difficult to model with IDs—in fact it is unfeasible in practice—while it is straightforward with DANs.

We have also argued (in Sec. V-B) that DANs compare favorably with other probabilistic graphical models for asymmetric decision problems. All the formalisms, except UIDs, use information links, which makes them unsuitable for representing order asymmetry. But UIDs were designed only for order asymmetry. Therefore none of these formalisms is suitable for problems, such as the n-test problem, having both types of asymmetry. This might explain why none of them has been implemented in the graphical user interface of any software tool nor used to build any real-world application. In contrast, there is at least one software package, OpenMarkov, for editing and evaluating DANs, and we expect that in the future other tools, both open-source and commercial, will implement them as well.

In this paper the only method proposed for evaluating an asymmetric DAN is to convert it into an equivalent DT, but the space complexity of this method makes it impractical for large problems. For this reason, we have developed more efficient evaluation algorithms . One of them consists of building a tree of symmetric DANs; these DANs can be evaluated with any algorithm for IDs, such as variable elimination or arc reversal, as explained in Section IV. The other method consists of building an S-DAG, as in [11]. These algorithms have allowed us to evaluate the two medical DANs mentioned in Section V-B and the n-test problem for up to 12 tests.[8] A line for future research is the evaluation of DANs by converting them into decision circuits [27], [28]. It would also be interesting to develop algorithms for DANs containing continuous variables; a point of departure might be to adapt some of the algorithms for IDs [3], [29].

Another line for future research is to develop algorithms for the explanation of reasoning, similar to those available for Bayesian networks and IDs [25], [30], which have been very useful for debugging probabilistic models [31].

In summary, given the advantages of DANs over DTs and IDs and the fact that we have so far found no problem based on the non-forgetting assumption—also implicit in DTs—that cannot be easily represented with DANs, we conjecture that this formalism will play a significant role in the field of decision analysis.

Acknowledgments: We thank the anonymous reviewers for many useful comments.

This work has been supported by grants TIN2006-11152 and TIN2009-09158 of the Spanish Ministry of Science and Technology.

References

[1] H. Raiffa and R. Schlaifer, *Applied Statistical Decision Theory.* Cambridge, MA: John Wiley & Sons, 1961.

[2] R. A. Howard and J. E. Matheson, "Influence diagrams," in *Readings on the Principles and Applications of Decision Analysis* (R. A. Howard and J. E. Matheson, eds.), pp. 719–762, Menlo Park, CA: Strategic Decisions Group, 1984.

[3] C. Bielza, M. Gómez, and P. P. Shenoy, "A review of representation issues and modelling challenges with influence diagrams," *Omega*, vol. 39, pp. 227–241, 2011.

[4] F. V. Jensen, T. D. Nielsen, and P. P. Shenoy, "Sequential influence diagrams: A unified asymmetry framework," *International Journal of Approximate Reasoning*, vol. 42, pp. 101–118, 2006.

[5] M. Arias, F. J. Díez, M. A. Palacios-Alonso, and I. Bermejo, "ProbModelXML. a format for encoding probabilistic graphical models," in *Proceedings of the Sixth European Workshop on Probabilistic Graphical Models (PGM'12)* (A. Cano, M. Gómez, and T. D. Nielsen, eds.), (Granada, Spain), pp. 11–18, 2012.

[6] R. Demirer and P. P. Shenoy, "Sequential valuation networks for asymmetric decision problems," *European Journal of Operational Research*, vol. 169, pp. 286–309, 2006.

[7] J. A. Tatman and R. D. Shachter, "Dynamic programming and influence diagrams," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 20, pp. 365–379, 1990.

[8] Z. Covaliu and R. M. Oliver, "Representation and solution of decision problems using sequential decision diagrams," *Management Science*, vol. 41, pp. 1860–1881, 1995.

[9] C. Bielza and P. P. Shenoy, "A comparison of graphical techniques for asymmetric decision problems," *Management Science*, vol. 45, pp. 1552–1569, 1999.

[10] H. Raiffa, *Decision Analysis. Introductory Lectures on Choices under Uncertainty.* Reading, MA: Addison-Wesley, 1968.

[11] F. V. Jensen and M. Vomlelová, "Unconstrained influence diagrams," in *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI'02)* (A. Darwiche and N. Friedman, eds.), (San Francisco, CA), pp. 234–241, Morgan Kauffmann, 2002.

[12] F. V. Jensen and T. D. Nielsen, *Bayesian Networks and Decision Graphs.* New York: Springer-Verlag, second ed., 2007.

[13] M. Luque and F. J. Díez, "Variable elimination for influence diagrams with super-value nodes," *International Journal of Approximate Reasoning*, vol. 51, pp. 615 – 631, 2010.

[14] S. M. Olmsted, *On Representing and Solving Decision Problems.* PhD thesis, Dept. Engineering-Economic Systems, Stanford University, CA, 1983.

[15] R. D. Shachter, "Evaluating influence diagrams," *Operations Research*, vol. 34, pp. 871–882, 1986.

[16] J. E. Smith, S. Holtzman, and J. E. Matheson, "Structuring conditional relationships in influence diagrams," *Operations Research*, vol. 41, pp. 280–297, 1993.

[17] T. D. Nielsen and F. Jensen, "Representing and solving asymmetric bayesian decision problems," in *Proceedings*

[8]At first sight, in this problem there are $12! = 4.8 \times 10^8$ possible orderings of the tests, but the combinatorial explosion is much worse because once we have decided which test to do first, the ordering of the other $n-1$ tests may be different for each outcome of the first test, and so on. Therefore it would be infeasible to represent and solve this problem using DTs or IDs.

of the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI'00) (C. Boutilier and M. Goldszmidt, eds.), (San Francisco, CA), pp. 416–425, Morgan Kauffmann, 2000.

[18] P. P. Shenoy, "Valuation network representation and solution of asymmetric decision problems," *European Journal of Operational Research*, vol. 121, pp. 579–608, 2000.

[19] C. König, "Representing asymmetric decision problems with Decision Analysis Networks," Master's thesis, Dept. Artificial Intelligence, UNED, Madrid, Spain, 2012.

[20] K. S. Ahlmann-Ohlsen, F. V. Jensen, T. D. Nielsen, O. Pedersen, and M. Vomlelová, "A comparison of two approaches for solving unconstrained influence diagrams," *International Journal of Approximate Reasoning*, vol. 50, pp. 153–173, 2009.

[21] T. Elvira Consortium, "Elvira: An environment for creating and using probabilistic graphical models," in *Proceedings of the First European Workshop on Probabilistic Graphical Models (PGM'02)* (J. A. Gámez and A. Salmerón, eds.), pp. 1–11, 2002.

[22] M. Luque, T. D. Nielsen, and F. V. Jensen, "An anytime algorithm for evaluating unconstrained influence diagrams," in *Proceedings of the Fourth European Workshop on Probabilistic Graphical Models (PGM'08)* (F. V. Jensen and U. Kjærulff, eds.), (Hirtshals, Denmark), pp. 177–184, 2008.

[23] D. León, "A probabilistic graphical model for total knee arthroplasty," Master's thesis, Dept. Artificial Intelligence, UNED, Madrid, Spain, 2011.

[24] R. A. Howard, "The used car buyer," in *Readings on the Principles and Applications of Decision Analysis* (R. A. Howard and J. E. Matheson, eds.), pp. 689–718, Menlo Park, CA: Strategic Decisions Group, 1984.

[25] C. Lacave, M. Luque, and F. J. Díez, "Explanation of Bayesian networks and influence diagrams in Elvira," *IEEE Transactions on Systems, Man and Cybernetics—Part B: Cybernetics*, vol. 37, pp. 952–965, 2007.

[26] S. Pauker and J. Wong, "The influence of influence diagrams in medicine," *Decision Analysis*, vol. 2, pp. 238–244, 2005.

[27] M. Pittarelli and R. D. Shachter, "Evaluating influence diagrams with decision circuits," in *Proceedings of the Twenty-third Conference on Uncertainty in Artificial Intelligence (UAI'07)* (R. Parr and L. C. van der Gaag, eds.), (Corvallis, OR), pp. 9–16, AUAI Press, 2007.

[28] D. Bhattacharjya and P. Shenoy, "Formulating asymmetric decision problems as decision circuits," *Decision Analysis*, vol. 9, pp. 138–145, 2012.

[29] B. R. Cobb and P. P. Shenoy, "Decision making with hybrid and influence diagrams using mixtures of truncated exponentials," *European Journal of Operational Research*, vol. 186, pp. 261–275, 2008.

[30] C. Lacave and F. J. Díez, "A review of explanation methods for Bayesian networks," *Knowledge Engineering Review*, vol. 17, pp. 107–127, 2002.

[31] C. Lacave, A. Oniśko, and F. J. Díez, "Use of Elvira's explanation facilities for debugging probabilistic expert systems," *Knowledge-Based Systems*, vol. 19, pp. 730–738, 2006.