Cost-effectiveness analysis with unordered decisions

Francisco Javier Díez, Manuel Luque, Manuel Arias, Jorge Pérez-Martín

Dept. Artificial Intelligence, Universidad Nacional de Educación a Distancia (UNED), Madrid, Spain.

Abstract

Introduction: *Cost-effectiveness analysis (CEA) is used increasingly in medicine to determine whether the health benefit of an intervention is worth the economic cost. Decision trees, the standard decision modeling technique for non-temporal domains, can only perform CEAs for very small problems. Influence diagrams can model much larger problems, but only when the decisions are totally ordered.*
Objective: *To develop a CEA method for problems with unordered or partially ordered decisions, such as finding the optimal sequence of tests for diagnosing a disease.*
Methods: *We explain how to model those problems using decision analysis networks (DANs), present an algorithm for evaluating them, and perform some experiments to study its computational efficiency. We illustrate the representation framework and the algorithm using a hypothetical example involving two therapies and several tests. We also show a DAN for a real-world problem, the mediastinal staging of non-small cell lung cancer.*
Results: *The evaluation of a DAN with two criteria, cost and effectiveness, returns a set of intervals for the willingness to pay, separated by incremental cost-effectiveness ratios (ICERs). The cost, the effectiveness, and the optimal intervention are specific for each interval, i.e., they depend on the willingness to pay.*
Conclusion: *Problems involving several unordered decisions can be modeled with DANs and evaluated in a reasonable amount of time. OpenMarkov, an open-source software tool developed by our research group, can be used to build the models and evaluate them using a graphical user interface.*

Keywords: *cost-effectiveness analysis; diagnostic tests; decision trees; influence diagrams; decision analysis networks.*

`https://www.evise.com/evise/faces/pages/submission/Submission.jspx`

1. Introduction

The goal of cost-effectiveness analysis (CEA) is to determine whether the effectiveness, i.e., the health benefit of an intervention, outweighs its economic cost [1, 2]. By "intervention", we mean either a single action, such as applying a therapy, or a whole strategy, such as "Do the test A; if it is positive, do test B, and if it is also positive, apply drug D; if A is negative then…". When there are several interventions, the problem is finding the one that maximizes the *net monetary benefit* [3],

$$NMB(\lambda) = \lambda \cdot e - c\,, \tag{1}$$

where e is the effectiveness and c the cost of the intervention considered. The parameter λ, usually called *willingness to pay* or *cost-effectiveness threshold*, converts effectiveness into a monetary scale. It takes values on the set of positive real numbers, i.e., on the interval $(0, +\infty)$, and is measured in effectiveness units divided by cost units; for example, in dollars per death avoided or euros per quality-adjusted life year (QALY) [4]. As the willingness to pay is specific for each decision maker and often imprecise or uncertain, CEA must consider all the possible values of λ, i.e., the optimal intervention must be given as a function of λ.

The most common tool for medical decision analysis are decision trees [5]. Their main drawback is that the number of branches grows very fast with the number of decisions and chance variables in

Email address: `{fjdiez,mluque,marias,jperezmartin}@dia.uned.es` (Francisco Javier Díez, Manuel Luque, Manuel Arias, Jorge Pérez-Martín)

the problem. The growth is even faster for CEA than for the unicriterion case, because the standard algorithm cannot evaluate decision trees with embedded decision nodes, which are those other than the root of the tree [6, 7]—see Appendix Appendix A.1.

Influence diagrams (IDs) are an alternative representation for decision problems [8]. Like decision trees, they have three types of nodes: chance, decision, and value. Their main advantage with respect to decision trees is their compactness. An ID whose chance and decision nodes represent n binary variables is equivalent to a decision tree with around 2^n leaves. Using IDs, our research group has built two models for medical problems whose equivalent decision trees would contain more than 10,000 branches [9, 10]. The ID IctNeo is even larger [11]. An ID can be evaluated by converting it into a decision tree, but there are much more efficient algorithms [12, 13, 14], not only for the unicriterion case, but also for CEA [15]. The main drawback of IDs is that they require a total ordering of the decisions. Therefore, a typical problem that IDs cannot solve is finding the optimal sequence of tests for diagnosing a disease.

We have recently proposed decision analysis networks (DANs) [16] as a new type of probabilistic graphical model [17] for representing and solving decision problems. Every influence diagram (ID) can be transformed automatically into a DAN, which implies that DANs, like IDs, can represent problems that would require a decision tree with millions of branches. But for many DANs there is no equivalent ID. In particular, DANs can represent problems whose decisions are unordered or partially ordered, while IDs cannot. We could say that IDs are a subset of DANs, even though the way of representing the flow of information is different.

The main contribution of this paper is a CEA algorithm for DANs, which combines the method for unicriterion DANs and the CEA algorithms for decision trees with embedded decision nodes [18] and for IDs [15]. The basic idea is straightforward, but the integration of those algorithms is not trivial. We illustrate the method with some examples. The first one is a fictitious problem involving two mutually-exclusive therapies and n tests. The goal is to find, for each value of λ, the optimal intervention, i.e., to determine which test must be done first, if any, and depending on its result, whether to do a second test, etc., and finally to decide what therapy to apply, if any. We solve in detail a numerical example, with two tests ($n = 2$) and examine empirically the computational time required to solve this problem for different values of n. We then show how to solve a real-world problem: finding the optimal sequence of six tests available for the mediastinal staging of non-small cell lung cancer. We solved this problem with an ID [10], which we have now converted into a DAN.

In this paper we assume that the reader is familiar with decision trees and the fundamentals of CEA.

2. Methods

We introduce here the n-test problem as an example involving unordered decisions. In Appendix Appendix A.1 we discuss why, in spite of its apparent simplicity, it is difficult to determine the optimal intervention—as a function of λ—even for only two tests ($n = 2$). Then we will show how to solve the problem using DANs.

2.1. The n-test problem

In this problem there are two mutually-exclusive therapies for a disease X and n tests, each having two possible outcomes, positive and negative. Every test can be performed once at most.

In order to solve the problem numerically, we will assume that the prevalence of X is 0.14. The effectiveness of the therapies depends on whether the disease is present or not, as shown in Table 1. The best situation for the patient occurs when the disease is absent and no therapy is applied (effectiveness = 10.0 QALY). The worst situation occurs when the disease is present and no therapy is applied (effectiveness = 1.2 QALY). For sick patients, the second therapy yields 6.5 QALY while therapy 1 only yields 4.0, but when a therapy is applied to healthy patients (by mistake) it reduces the effectiveness due to side effects, especially therapy 2.

We assume that are two tests, A and B, such that the latter is more accurate, but more expensive, as shown in Table 2.

We invite the reader to try to solve this problem—i.e., to find the intervention that maximizes the NMB for each value of λ—using a software tool, such as Excel, R, or TreeAge, before reading the analysis we offer in Appendix Appendix A.1 and the solution we propose in Section 3.

Therapy	Cost	Effectiveness	
		disease	no disease
no therapy	€0	1.2 QALY	10.0 QALY
therapy 1	€20 000	4.0 QALY	9.9 QALY
therapy 2	€70 000	6.5 QALY	9.3 QALY

Table 1: Cost and effectiveness of each intervention for the 2-test example.

Test	Cost	Sensitivity	Specificity
A	€18	78%	91%
B	€150	90%	93%

Table 2: Cost and effectiveness of each intervention for the 2-test example.

2.2. Representation of decision problems with DANs

In this section we present decision analysis networks (DANs) as a framework for CEA in problems with unordered or partially ordered decisions. We explain first how to model the problem using a DAN and then how to evaluate it.

A DAN contains three types of nodes, the same as in decision trees. Decision nodes are represented by rectangles, chance nodes by rounded rectangles, and value nodes by hexagons. In multicriteria DANs, each value node has one associated criterion. The DAN in Figure 1 contains three decision nodes: two for the tests (each with two options: "do test" and "do not test") and one for the therapy (with three options: "no therapy", "therapy 1", and "therapy 2"). It also contains three chance nodes: one for the disease (with states "present" and "absent") and two for the outcomes (each with the states "positive" and "negative"). This DAN also has four value nodes; three of them represent economic cost and the other one effectiveness. Given that each node represents a variable, we speak indistinctly of nodes and variables.

The graph of a DAN also contains directed links. When there is a link $V_1 \to V_2$ we say that V_1 is a parent of V_2 and V_2 is a child of V_1. The graph must by acyclic, which means that if we depart from a node and follow the links from parent to child, we never arrive at the node of departure. Links usually represent causal influence. For example, disease X affects the result of the tests, making a positive outcome more probable in the presence of the disease than in its absence. Links may have associated restrictions. A total restriction associated with link $V_1 \to V_2$ means that at least one value of V_1 is incompatible with all the values of V_2. For example, the value "do not test" of the node "Do test A?" is incompatible with the two values of "Result of A", because in this case the test is neither positive nor negative. This restriction is denoted by a double line on the link that connects these nodes. There may be a partial restriction between V_1 and V_2, denoted by a single line on the corresponding link, meaning that a certain value of V_1 is compatible with some values of V_2 and incompatible with others. In this example there is no partial restriction. Links colored in red represent revelation conditions. For example, the decision of doing a test reveals the outcome of the test, and this information is available for all subsequent decisions.

In some cases, the value of a chance variable is known before making any decision—for example, a symptom spontaneously reported by the patient. We then say that this variable is "always-observed" and denote it by drawing a red border around the node. The paper by **(author?)** [16] contains examples of DANs with partial restrictions and always-observed nodes.

This is the structural information contained in the DAN. There is also qualitative information, given by the parameters of the model. Each chance node has an associated conditional probability distribution for each configuration of its parents (except when there is a total restriction). When a variable and its parents are discrete, the probabilities are usually represented by a table, as shown in Figure 2. It can also be represented by a probability tree or by other types of potentials [19]. Each value node also has an associated table, as in Figure 3.

Figure 4 shows a DAN for a real-world problem: the mediastinal staging of non-small cell lung cancer. It is a slightly modified version of an ID built for the same purpose [10]. The main difference is that the decisions in the DAN are only partially ordered.

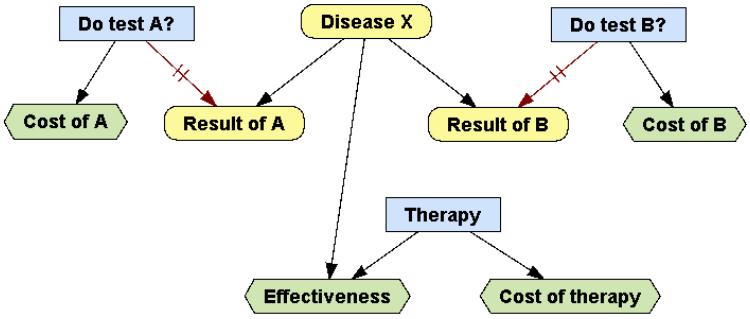

Figure 1: A DAN for the 2-test problem. Rectangles represent decisions. Rounded rectangles represent chance variables. Hexagons represent values, i.e., cost and effectiveness. Links represent causal influences. Links colored in red represent revelation conditions; for example, doing a test reveals the result of the test. A double line on a link represents a restriction; for example, the result of a test is available only when the test is performed.

Do test A?	no	no	yes	yes
Disease X	absent	present	absent	present
positive	0	0	0.09	0.78
negative	0	0	0.91	0.22

Figure 2: Conditional probability table for the node "Result of test A". Each column represents the conditional probability distribution for a configuration of the parents of this node, except for the columns corresponding to total restrictions. The sensitivity and specificity of the test are taken from Table 2.

2.3. Algorithms for cost-effectiveness evaluation of DANs

The algorithms we propose for the evaluation of DANs are based on cost-effectiveness partitions (CEPs), which were introduced for the evaluation of decision trees with embedded decision nodes: in the same way as the roll-back algorithm assigns a numeric value to each node, the algorithm in [18] assigns a CEP to each node. The CEP contains, for every value of λ, a cost-effectiveness pair and the optimal intervention. This way the algorithm evaluates the tree for all the values of λ, grouping a finite set of intervals, such that within each interval the cost, the effectiveness, and the intervention are the same for all the values of λ. The thresholds that delimit the intervals are determined dynamically when evaluating the tree. This way the new roll-back algorithm can evaluate cost-effectiveness decision trees with embedded decision nodes. The same idea can be applied to the evaluation of IDs [15] and DANs.

2.3.1. Cost-effectiveness partition (CEP)

Definition 1. A *cost-effectiveness partition* of n intervals (with $n \geq 1$) consists of the following elements:

- $\Theta = \{\theta_1, \ldots, \theta_{n-1}\}$ – a set of $n-1$ values (thresholds), such that $0 < \theta_1 < \ldots < \theta_{n-1}$,
- $C = \{c_0, \ldots, c_{n-1}\}$ – a set of n values (costs),
- $E = \{e_0, \ldots, e_{n-1}\}$ – a set of n effectiveness values, and
- $I = \{I_0, \ldots, I_{n-1}\}$ – a set of n interventions.

In practice, every CEP results from a CEA whose input is a set of interventions, each one having a known cost and effectiveness. In the resulting CEP, $\{I_0, \ldots, I_{n-1}\}$ is the subset of non-dominated interventions, c_i and e_i are the cost and effectiveness of intervention I_i respectively, and the θ_i's are the incremental cost-effectiveness ratios (ICERs):

$$\theta_i = ICER(I_{i-1}, I_i) = \frac{c_i - c_{i-1}}{e_i - e_{i-1}} \,. \tag{2}$$

For example, the CEP in Figure 5 is the result of a CEA for the 2-test problem when disease X is known to be present. It has three non-dominated interventions and two thresholds, which delimit three intervals for λ. When the disease is absent, "no therapy" dominates the other two interventions, because these are more expensive and less effective, so the resulting CEP has no threshold and only one interval, with I_0 = "no therapy", $c_0 = €0$, $e_0 = 10$ QALY.

Disease X	absent	absent	absent	present	present	present
Therapy	no therapy	therapy 1	therapy 2	no therapy	therapy 1	therapy 2
Effectiveness	10	9.9	9.3	1.2	4	6.5

Figure 3: Table for the node "Effectiveness". Its values are taken from Table 1.

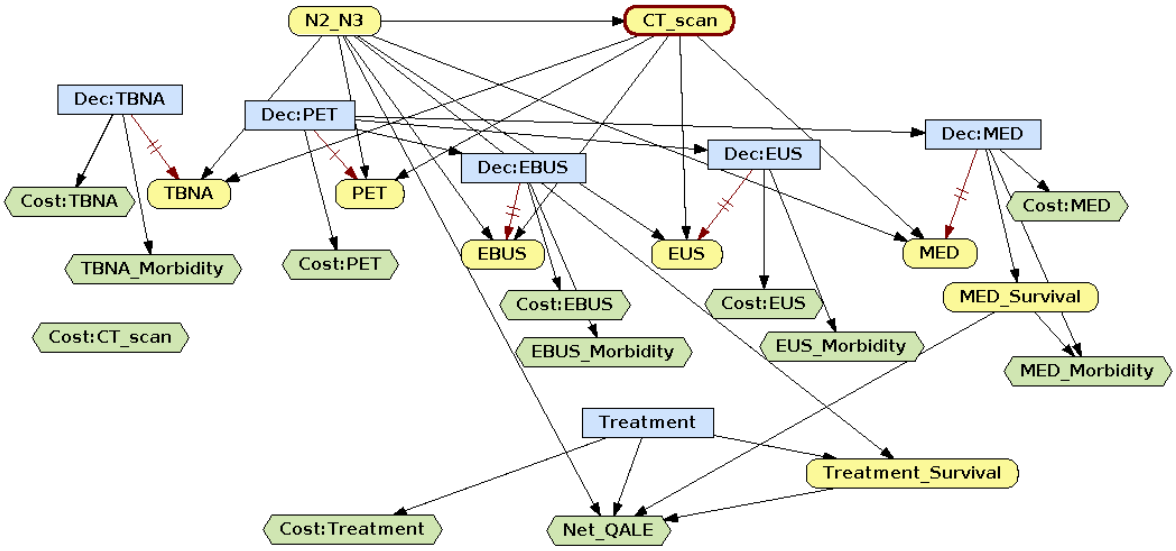

Figure 4: MEDIASTINET, a DAN for the mediastinal staging of non-small cell lung cancer.

2.3.2. Evaluation of the DAN

Cost-effectiveness DANs can be evaluated with Algorithm 1, which recursively decomposes the original network into DANs without decision nodes. We illustrate its performance for 2-test DAN (cf. Fig. 1). We denote the variables by capital letters and their values (states) by the corresponding lowercase letters. Thus, X represents the disease and $+x$ and $\neg x$ mean that it is present or absent respectively, R_A represents the outcome of test A and $+r_A/\neg r_A$ a positive/negative result, etc.

The algorithm is first invoked with the original DAN and no evidence, because no variable has yet been observed. The first decomposition generates two new DANs, as shown in Figure 6; in one of them the first decision is whether to do test A or not, and in the other the first decision is whether to do test B. The decomposition of the former generates two new DANs, in which D_A disappears because every decision node is deleted when making the decision; in one of these DANs the option chosen is to do the test, $+d_A$, which reveals the value of R_A; in the other the option chosen is not to do the test, $\neg d_A$, which is incompatible with the two values of R_A (because in this case the test is neither positive nor negative), so node R_A does not appear in this network. The DAN for $+d_A$ is then decomposed into two new DANs: one for the positive result of the test ($+r_A$) and another one for a negative result ($\neg r_A$). The process continues until the network contains no decisions and every observable variable has been assigned a value. In this case the algorithm returns a probability (a single number) and a CEP, and the recursion continues backwards. At the end of the process the probability returned is 1 and the CEP contains the optimal intervention for each value of λ. The process is explained in more detail in Appendix Appendix A.2.

Every node in the decomposition tree (Fig. 6) corresponds to a node in a decision tree (Fig. 7), which implies that Algorithm 1 can be adapted to generate a decision tree if we wish, provided that our computer has enough memory to store it.

2.4. Implementation

Algorithm 1 can be implemented in any object-oriented language, such as C++, R, or MATLAB. We have implemented it in Java, within an open-source tool called OpenMarkov (see `www.openmarkov.org`), which has an advanced graphical user interface (GUI) for editing several types of probabilistic graphical models [17], including Bayesian networks, influence diagrams (IDs), Markov influence diagrams (MIDs), partially-observable Markov decision processes (POMDPs), DANs, etc. Models having several criteria

λ inf.	λ sup.	Cost	Effectiveness	Intervention
0.0	7142.86	0.0	1.2	Therapy = no therapy
7142.86	20000.0	20000.0	4.0	Therapy = therapy 1
20000.0	+∞	70000.0	6.5	Therapy = therapy 2

Figure 5: Cost-effectiveness partition (CEP) for the 2-test problem when disease X is present. The two thresholds, $\theta_1 =$ €7,142.86/QALY and $\theta_2 =$ €20,000.00/QALY, obtained with Equation 2, delimit three intervals for λ.

can be evaluated by joining them into a single criterion; for those having two criteria it is also possible to perform a CEA with just a few clicks. OpenMarkov's tutorial explains how to build and evaluate these models.

3. Results

Figure 8 shows the conclusion of the CEA for the 2-test DAN, which solves the numeric problem stated in Section 2.1. The CEP consists of 6 intervals, each having an optimal interventions. When the willingness to pay is very low, i.e., $\lambda <$ €7.718.95/QALY, it is not worth doing either test because, regardless of its result, no therapy will be applied. For all the other intervals, a therapy is applied only when the last test performed is positive. In the second interval, i.e., €7,718.95/QALY $< \lambda <$ €21,385.50/QALY, test A should be done; if it is positive, the first therapy should be applied. The optimal intervention for the third interval is similar, with therapy 2 instead of therapy 1. In the fourth interval test B should be done first; if it is positive, test A must be done to determine the optimal treatment: therapy 1 when A is negative and therapy 2 when it is positive. In the fifth interval the optimal intervention is to do both tests, in any order, even if the first one is negative; if one of them is positive, therapy 1 must be applied; if both are positive, the most beneficial treatment is therapy 2. In the sixth interval, the willingness to pay exceeds €113.139.00/QALY; a positive result in test B leads to the direct application of therapy 2, while a negative result requires doing test A and applying therapy 1 when this is positive.

It is possible to solve the n-test problem for a higher number of tests. Adding a new test to the DAN takes less than 2 minutes in OpenMarkov's graphical user interface, which is negligible compared to the time required to add a new test in a decision tree. On a desktop computer, Algorithm 1 can evaluate the DAN for $n \leq 3$ in less than a second, for $n = 4$ in 7 seconds, for $n = 5$ in 2 minutes, for $n = 6$ in 40 minutes, and for $n = 7$ in 17.5 hours.

Algorithm 1 in this paper is an adaptation for CEA of Algorithm 1 in [16]. The CEA version of Algorithm 2 in that paper, which is much more efficient (we do not show the pseudocode here due to space limits), can solve the case $n = 7$ in 9 minutes and $n = 8$ in 3 hours. With parallel computation the evaluation would be much faster. These are remarkable results because the decision trees (with embedded decision nodes) have more than 6 million leaves for $n = 7$ and more than 100 million for $n = 8$—see Appendix Appendix A.1.

The DAN MEDIASTINET, which contains 5 partially-ordered tests, can be evaluated in less than 3 seconds. The resulting CEP contains 18 intervals. Figure 9 shows the optimal intervention for $\lambda =$ €30,000/QALY, which is the shadow cost-effectiveness threshold estimated for the Spanish public health system [20, 21]. This intervention differs from the one we obtained from the ID that used the same parameters, which applies chemotherapy when the endobronchial ultrasound test (EBUS) is positive—see Figure 5 in [10]—while the DAN recommends doing transesophageal ultrasound-guided fine needle aspiration (EUS) and mediastinoscopy (MED) in this case.

All the networks presented in this paper are available at `www.probmodelxml.org/networks`. They can be evaluated with OpenMarkov's graphical user interface, which uses the more efficient algorithm. The web site `www.openmarkov.org` contains a compiled version of the program, as well as indications for installing and executing the Java code.

4. Discussion

Using DANs it is possible to solve many decision problems that cannot be modeled with traditional techniques. One reason is that the standard roll-back algorithm for CEA cannot evaluate decision trees containing embedded decision nodes [7]. It would be necessary to build a tree containing one decision

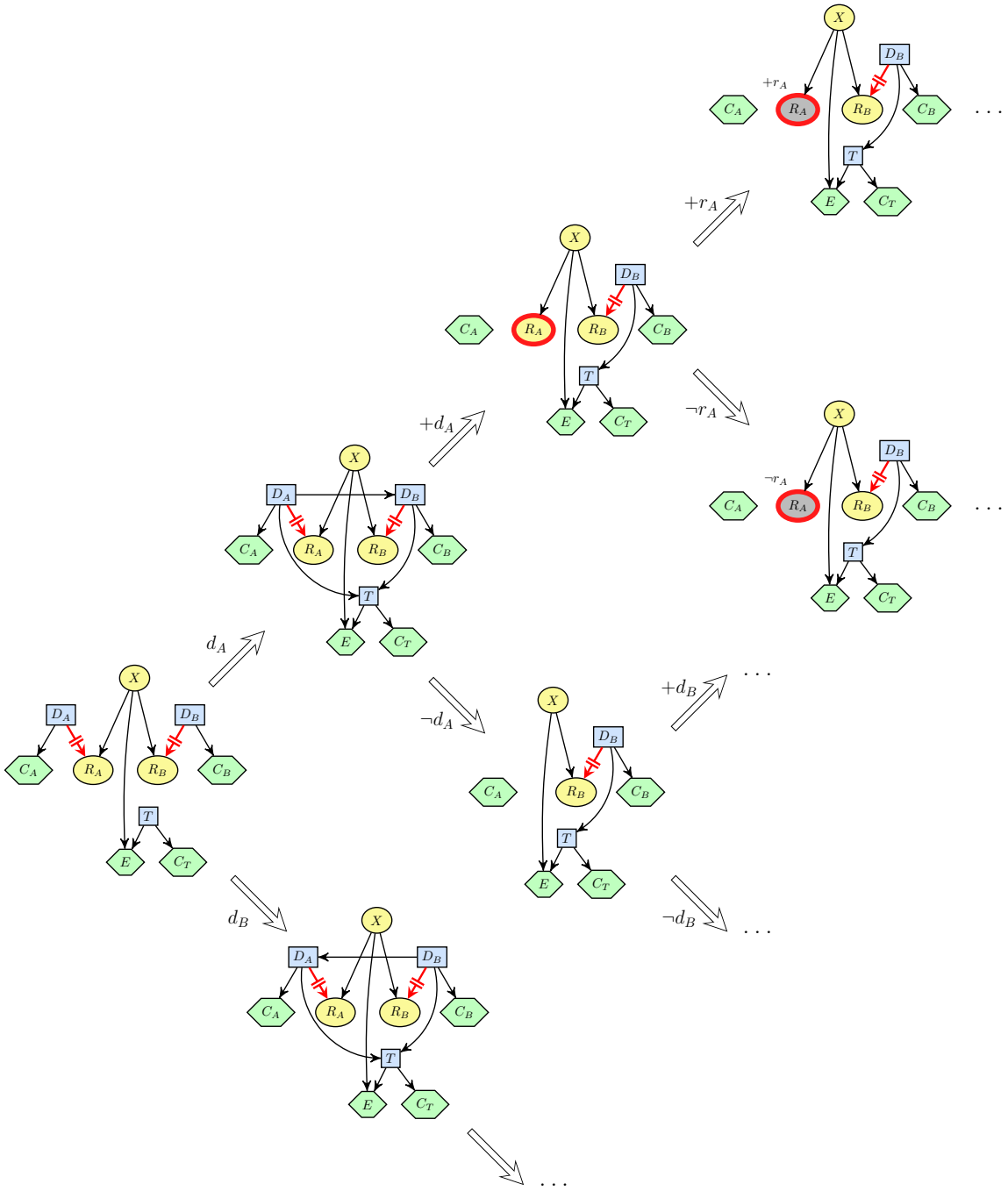

Figure 6: Recursive decomposition of the DAN for the 2-test problem. Every node in this tree corresponds to an invocation of evaluateDAN (Algorithm 1). In the DANs, chance nodes having associated evidence are colored in gray.

Algorithm 1: Cost-effectiveness analysis for a DAN

function: `evaluateDAN`
input : *DAN* – a decision analysis network,
$\mathbf{e}$ – evidence (a configuration of a set of variables $\mathbf{E}$)
output : $P(\mathbf{e})$ – the probability of the evidence,
CEP$(\mathbf{e})$ – a cost-effectiveness partition

if *DAN has decision nodes* **then**
  $\mathbf{O} \leftarrow$ always-observed variables in *DAN*, except those in $\mathbf{E}$;
else
  $\mathbf{O} \leftarrow$ all the chance variables in *DAN*, except those in $\mathbf{E}$;
if $\mathbf{O} = \varnothing$ *and DAN has no decisions* **then**
  $P(\mathbf{e}) \leftarrow$ product of the projected probability potentials;
  CEP$(\mathbf{e}) \leftarrow$ sum of the projected utility potentials;
else if $\mathbf{O} \neq \varnothing$ **then**
  select $X \in \mathbf{O}$ such that X has no ancestor in $\mathbf{O}$;
  foreach x *of* X **do**
    $DAN_x \leftarrow$ `instantiate` (*DAN*, X, x);
    $\{P(\mathbf{e} \circ x), CEP(\mathbf{e} \circ x)\} \leftarrow$ `evaluateDAN(`DAN_x, $\mathbf{e} \circ x$`)`;
  $P(\mathbf{e}) \leftarrow \sum_x P(\mathbf{e} \circ x)$;
  $P(x \mid \mathbf{e}) \leftarrow P(\mathbf{e} \circ x)/P(\mathbf{e})$;
  $CEP(\mathbf{e}) \leftarrow$ `averageCEP`$(X, P(x \mid \mathbf{e}), \{CEP(\mathbf{e} \circ x_1), \ldots, CEP(\mathbf{e} \circ x_m)\})$;
else if *exactly one decision* D *can be made first* **then**
  foreach d *of* D **do**
    $DAN_d \leftarrow$ `instantiate` (*DAN*, D, d);
    $\{P_d(\mathbf{e}), CEP_d(\mathbf{e})\} \leftarrow$ `evaluateDAN(`DAN_d, $\mathbf{e}$`)`;
  $P(\mathbf{e}) \leftarrow P_d(\mathbf{e})$ for an arbitrary value d;
  $CEP(\mathbf{e}) \leftarrow$ `optimalCEP`$(D, \{CEP_{d_1}(\mathbf{e}), \ldots, CEP_{d_m}(\mathbf{e})\})$;
else
  $\mathbf{D}_I \leftarrow$ *decisions that can be made first*;
  foreach D *in* $\mathbf{D}_I$ **do**
    $DAN_D \leftarrow$ `prioritize` (*DAN*, D);
    $\{P_D(\mathbf{e}), CEP_D(\mathbf{e})\} \leftarrow$ `evaluateDAN(`DAN_D, $\mathbf{e}$`)`;
  $P(\mathbf{e}) \leftarrow P_D(\mathbf{e})$ for an arbitrary decision D;
  $OD \leftarrow$ new decision variable (meta-decision), with one value for each decision D in $\mathbf{D}_I$;
  $CEP(\mathbf{e}) \leftarrow$ `optimalCEP`$(OD, \{CEP_{D_1}(\mathbf{e}), \ldots, CEP_{D_m}(\mathbf{e})\})$;
return $\{P(\mathbf{e}), CEP(\mathbf{e})\}$

node, at the root, with an outgoing branch for each intervention, as proposed by **(author?)** [6]. In Appendix Appendix A.1 we show that when no test is available ($n = 0$), there are 3 intervention, one for each therapy, so the tree has only 6 leaves. When $n = 1$ there are 9 interventions—after excluding some that are clearly suboptimal—and the tree has 30 leaves. When $n = 2$ there are 147 interventions and the tree grows up to 966 leaves. For $n \geq 3$ the interventions are so complex that it is very difficult to estimate their number, but the increase from 30 to 966 leaves shows us the impressive growth of the tree for every test we add.

Due to the complexity of the problem, economic evaluations involving unordered decisions use expert knowledge either to impose a total ordering among them or to select a very small number of interventions. For example, a CEA of four tests for the diagnosis of coronary heart disease only examined 8 out of the thousands of possible interventions [22]. It is likely that the shortlist contained the optimal intervention, but it is not necessarily so.

In fact, if we reexamine the six interventions resulting from the evaluation of the 2-test problem, which we presented in Section 3, we can see that the first three ones are straightforward, but those for the fourth and the last intervals are not so obvious. Some decision analysts, including ourselves, might

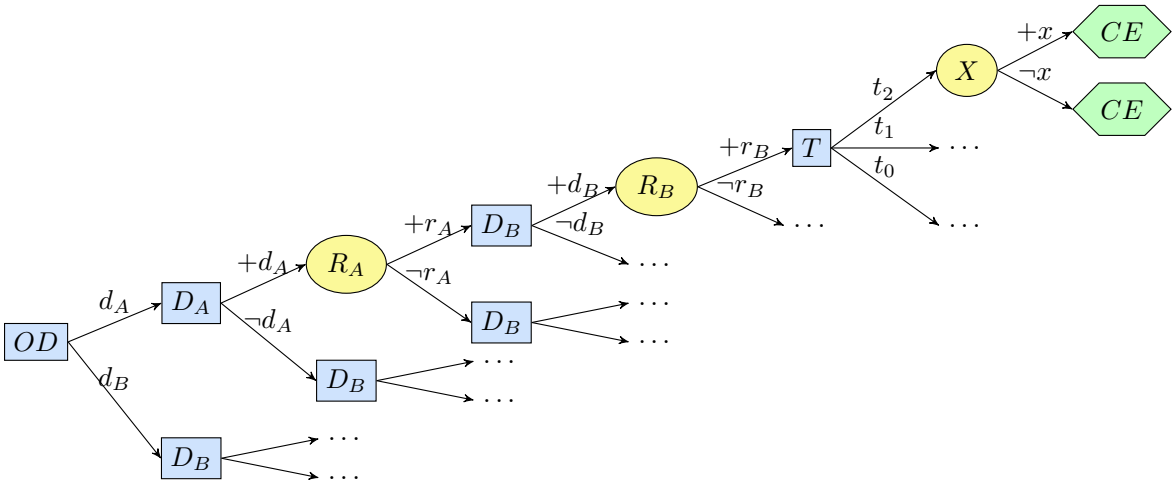

Figure 7: Decision tree generated from the DAN for the 2-test problem. Every node in this tree corresponds to a node in the decomposition tree (Fig. 6), i.e., to an invocation of evaluateDAN. The "meta-decision" node *OD* (order of the decisions) determines which decision to make first.

λ inf.	λ sup.	Cost	Effectiveness	Intervention
0.0	7718.95	0.0	8.768	OD = Do test A? -> Do test A? = no -> Do test B? = no -> Therapy = no therapy
7718.95	21385.5	2119.95	9.04264	OD = Do test A? -> Do test A? = yes -> IF Result of A = negative -> Do test B? = no -> ...
21385.5	24361.7	7304.85	9.28509	OD = Do test A? -> Do test A? = yes -> IF Result of A = negative -> Do test B? = no -> ...
24361.7	71550.3	9062.25	9.35723	OD = Do test B? -> Do test B? = yes -> IF Result of B = negative -> Do test A? = no -> ...
71550.3	113139.0	10734.9	9.38061	OD = Do test A? -> Do test A? = yes -> IF Result of A = negative -> Do test B? = yes -> ...
113139.0	+∞	14856.7	9.41704	OD = Do test B? -> Do test B? = yes -> IF Result of B = negative -> Do test A? = yes -> ...

Figure 8: CEP for the 2-test problem. It is the final result of evaluating the DAN in Figure 1 with Algorithm 1.

have overlooked the intervention that turned out to be optimal for $\lambda >$ €113.139.00/QALY: "do test B first; if positive, apply therapy 2; if negative, do test A and if this is positive apply therapy 1". That strategy is counterintuive, because apparently no therapy should be applied when test B, which is more sensitive and specific than A, has ruled out the presence of the disease. However, a numerical analysis shows that $P(+x|+r_A, \neg r_B) = 0.1317$, and for this posterior probability therapy 1 is more effective than both no therapy and therapy 2. This situation reminds us of the application of artificial intelligence to games, such as go and chess: even the best human players examine only a very small amount of combinations, while computers, which evaluate many more possibilities, sometimes find excellent moves that no human would have figured out. Similarly, when we built the ID MEDIASTINET, the pneumologist who helped us did not know whether endobronchial ultrasound (EBUS) should be done before EUS or vice versa. This questions has been solved with the DAN version of the same model (cf. Fig. 9).

DANs are not only superior to the traditional way of doing CEA with decision trees. They also outperform the methods we have proposed previously. One of these is the algorithm for evaluating cost-effectiveness decision trees with embedded decision nodes [18]. These trees do not need one branch for each intervention, because their evaluation implicitly compares all the possible interventions. So in the n-test problem they only need 18 leaves (instead of 30) for $n = 1$ and 78 leaves (instead of 966) for $n = 2$. However, it is still impossible to manually build the trees for a higher number of tests, which require 474 leaves for $n = 3$, 2,845 leaves for $n = 4$, etc. By contrast, DANs can evaluate up to the case $n = 8$, for which the decision tree would contain more than 100 million branches, as mentioned above.

DANs are also superior to IDs. These can perform CEA for large problems [15], but only if the decisions are totally ordered. So when building an ID for the 2-test problem we must decide which test to do first, either A or B, and it would therefore be impossible to obtain the CEP shown in Figure 8.

Additionally, the use of restrictions allows DANs to usually avoid dummy states, such as "test not done", which in an ID must be added to the outcomes of a test to indicate that when we decide not to do the test the result is neither "positive" nor "negative". Dummy states unnecessarily increase the size of the probability tables and the number of parameters, some of which are 0 or 1 [23, 16]. For this reason even when a problem can be modeled with an ID, the DAN representation is cleaner and more elegant.

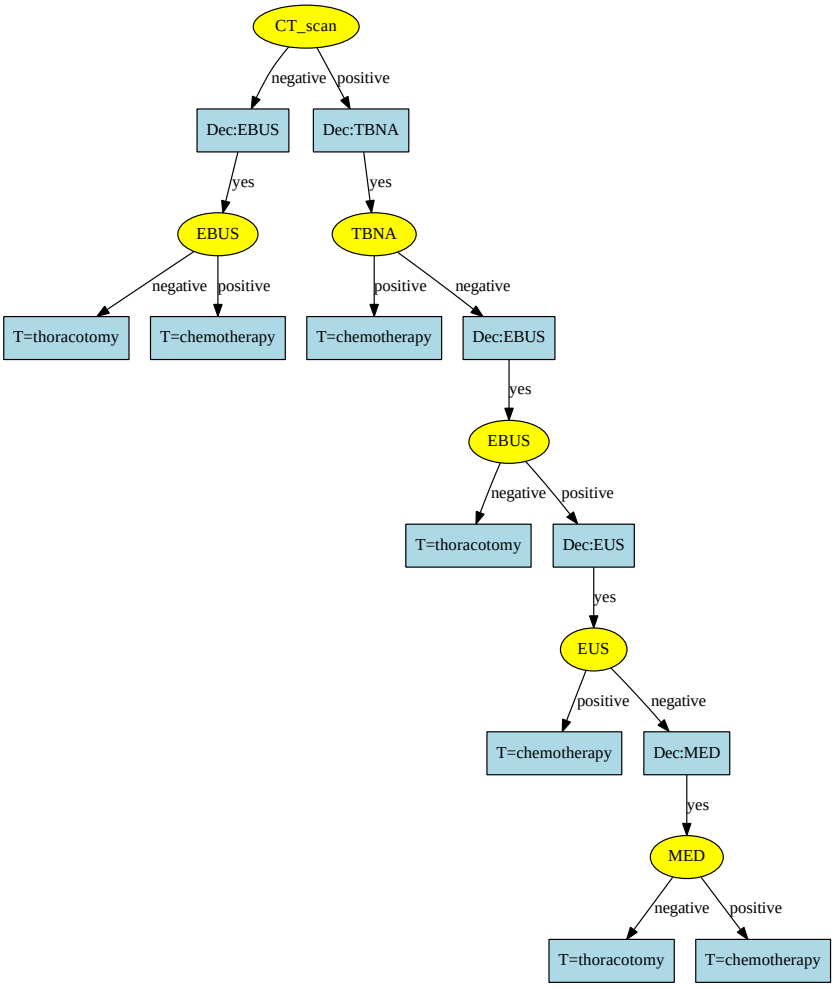

Figure 9: Optimal intervention for $\lambda = €30{,}000$/QALY obtained from then DAN MEDIASTINET (cf. Fig. 4).

The main drawback of DANs is that health economists, who are all familiar with decision trees, would need to learn a new formalism. However, the effort is small for someone acquainted with the fundamentals of decision analysis, and the possibility of easily solving much larger problems much more easily will soon compensate for the time invested. Additionally, the availability of an open-source tool for DANs will avoid the need to acquire software licenses for building and evaluating decision trees. In fact, OpenMarkov can automatically convert a DAN into a decision tree; if the tree is too big, it can be expanded only to the desired depth. This implies that the decision modeler who builds a decision tree only has a decision tree, while the one who builds a DAN has both the DAN and the tree, with much less effort.

We conclude this section with a comment about efficiency. Using DANs it is possible to model problems involving many decisions. Adding a new test to a decision problem takes less than two minutes in OpenMarkov, so we can model the n-test problem for arbitrary large values of n. The problem lies in the time complexity, which in general increases much faster than exponentially with the number of nodes in the DAN. Fortunately, Algorithm 1, which is equivalent to the evaluation of a decision tree, can easily be implemented in parallel, using a negligible amount of memory for each branch. Alternatively, it is possible to use a more efficient decomposition of the DAN (cf. Algorithm 2 in [16]), which can take profit of tensor-processing libraries that run on GPUs and TPUs.

5. Conclusions and future work

In this paper we have shown how to perform CEA in problems involving unordered or partially-ordered decisions. The proposed method is based on our previous CEA algorithms for decision trees with embedded decision nodes [18] and for IDs [15], and those for unicriterion DANs [16]. Like IDs, DANs can model some large problems for which a decision tree would contain millions of branches, but do not require a total ordering of the decisions. This way DANs combine the flexibility of decision trees with the compactness of IDs. The existence of OpenMarkov, an open-source tool for building and evaluating probabilistic graphical models, may be an additional reason for using DANs instead of decision trees.

One line for future research is the development of more efficient algorithms for DANs, including parallel implementations. Another line is to design sensitivity analysis options and explanation facilities similar to those existing for ID [24]. A third line is to develop Markov DANs, which would combine the advantages of DANs with those of Markov IDs [25]. And fourth, we might try to compact the interventions returned by the algorithms [26].

6. Acknowledgments

This work has been supported by grant TIN2016-77206-R of the Spanish Government, co-financed by the European Regional Development Fund. JP received a predoctoral grant from the Spanish Ministry of Education.

Appendix A. Appendices

Appendix A.1. Complexity of modeling the n-test problem with decision trees

We have introduced in Section 2.1 the n-test problem. We analyze here the complexity of modeling it with decision trees, first for the unicriterion case and then for CEA.

Appendix A.1.1. Decision trees for unicriterion analysis

If cost and effectiveness were combined into a single criterion, the NMB (cf. Eq. 1), the optimal intervention could be found by building the decision tree. When there is no test ($n = 0$), the tree contains the decision node "Therapy" at the root, with a "Disease X" node in each branch. If we denote by $l(n)$ the number of leaves in the tree for n tests, we have $l(0) = 3 \times 2 = 6$. When there is one test, the decision node at the root has two branches: "do the test" and "do no test". The first one has two branches, "positive" and "negative". Each of these three branches has a "Therapy" node, as in the previous case, so the tree has $l(1) = 3 \times 3 \times 2 = 18$ leaves. When there are n tests, the root has one branch, "do no test", and n branches "do test A", "do test B", etc. The "no test" branch has 6 leaves; each of the other branches has $l(n-1)$ leaves for a positive result of the test and $l(n-1)$ leaves for a

negative result. Therefore, $l(n) = 2n \cdot l(n-1) + 6$, and we have $l(2) = 78$, $l(3) = 474$, $l(4) = 3,798$, etc. Consequently the problem is hard for two tests and virtually impossible to solve for more than two tests. If there were more than two therapies and each test had more than two outcomes, the problem would be much harder even for only two tests.[1]

Appendix A.1.2. Decision trees for cost-effectiveness analysis

The standard roll-back algorithm for CEA evaluates decision trees by assigning a cost-effectiveness pair to each node, starting from the leaves, and then performing a CEA at the root, which returns a set of ICERs. This algorithm cannot be applied to the trees described in the previous section (except for $n = 0$) because the evaluation of an embedded decision node does not return a single cost-effectiveness pair.[2] The solution proposed by **(author?)** [6] consists in building a decision tree containing just one decision node, at the root, with an outgoing branch for each intervention. When no test is available ($n = 0$), there are only three possible interventions: "no therapy", "therapy 1", and "therapy 2". Each branch has a chance node, X, with two outgoing branches, "present" and "absent". Therefore, in this case the tree has 6 leaves, $l(0) = 6$.

When $n = 1$, there are three no-test interventions, as in the previous case, plus each with a plus 3×3 do-test interventions: "if test is negative, then no therapy; if positive, no therapy", "if test is negative, no therapy; if positive, therapy 1", "if test is negative, no therapy; if positive, therapy 2", "if test is negative, then therapy 1; if positive, therapy 2", ... , "if positive, therapy 2; if negative, therapy 2". The 3 interventions in which the therapy applied is the same regardless of the result of the test can be ruled out because it is not worth doing a test that will not guide the decision about the therapy. Therefore, there are 3×2 do-test interventions. Each one has 2×2 leaves, because the chance node that represents the result of the test has two outgoing branches, "positive" and "negative", and node for X has two outgoing branches, "present and absent". Therefore, the tree has $6 + 6 \times 4$ leaves, i.e., $l(1) = 30$.

When $n = 2$, there are 3 no-test interventions; each one corresponds to a branch outgoing from the root of the tree, and has two leaves. There are also some interventions that begin doing test A. When it is positive, there are 3 subsequent interventions that do not perform test B (as many as when there was not test available) and 6 interventions that perform it (as many as if there were only one test available), i.e., 9 sub-interventions. There are also 9 sub-interventions for a negative result of test A. This makes a total of 9×9 interventions that begin doing test A. However, it is not worth doing the test when it does entail a difference in the sub-intervention applied after it, so we can limit our analysis to $9 \times 8 = 72$ interventions. Each branch for an intervention in which both tests are done contain $2 \times 2 \times 2 = 8$ leaves. An intervention that performs only test A leads to $2 \times 2 = 4$ leaves. When test B is done only for one of the outcomes of test A, the branch contains 6 leaves. This makes a total of 480 leaves for the interventions that begin doing test A. There are also 72 interventions that begin doing test B. This makes a total of $3 + 72 + 72 = 147$ interventions and $6 + 480 + 480 = 966$ leaves.

For $n \geq 3$ the possible interventions are so complex that it is very difficult even to estimate their number, but the increase from $l(1) = 30$ to $l(2) = 966$ leaves shows the impressive increase of the size of the tree for every test we add.

We have excluded from these counts the interventions in which the result of a test does not guide the next decisions—for example, the interventions that apply a therapy regardless of the result of the test. Depending on the numerical parameters of the model we might discard more interventions. For example, in the one-test problem we can see that the incremental effectiveness of "therapy 1" with respect to "no therapy" grows with the probability of disease X. Therefore, the intervention "when test A is negative apply therapy 1; when positive, apply no therapy" is clearly suboptimal. However, when the number of decisions increases, the proportion of therapies that can be discarded is smaller and smaller.

In summary, building manually a decision tree with only one decision node for CEA, as proposed by

[1] The trees proposed in this paragraph are slightly more compact than those generated from the corresponding DANs (cf. Fig. 7) but the computational complexity is virtually the same. The main difference lies in the effort required to build them.

[2] Old versions of TreeAge evaluated embedded decision nodes by asking the user for a value of λ in order to select for each node the branch that maximizes the NMB. The question might disconcert the user, because it is not necessary: the tree must be evaluated for all the values of λ,not for a particular one. Recent versions use a default value, which can be set by the user. Using a single value of λ makes it is possible to assign a cost-effectiveness pair to each node, but the evaluation depends on the value arbitrarily chosen and the result is often wrong [7].

[6], is unfeasible for the 2-test problem and absolutely impossible for $n > 2$.

Appendix A.2. Evaluation of the DAN for the 2-test problem

In Section 2.3.2 we have used the 2-test DAN to explain the evaluateDAN algorithm, which recursively decomposes a DAN until it contains no decisions. In this section we offer the same explanation in much more detail.

Appendix A.2.1. Decomposition of the DAN

Every invocation of the Algorithm 1 takes two arguments: a DAN and a set of findings, denoted by **e** (evidence). The first invocation always takes the original network and no evidence. In our example, the DAN has no always-observed variable, so the assignment in line 4 makes $\mathbf{O} \leftarrow \varnothing$. The network has three decisions: D_A, D_B, and T. In principle any of them might be made first, but the algorithm detects that T cannot reveal any information, so it must not be the first one; in line 23 it makes $\mathbf{D}_I \leftarrow \{D_A, D_B\}$ and then decomposes the original network into two DANs, as shown in Figure 6: in one of them D_A is prioritized by drawing the links $D_A \rightarrow D_B$ and $D_A \rightarrow T$; in the other, D_B is prioritized. Line 28 adds a meta-decision variable OD (for "order of the decisions"), indicating which decision will be made first. If we modify Algorithm 1 in order to explicitly build the decision tree, this variable would be the root (see Fig. 7); in other examples there might be several meta-decisions.

The evaluation of the network containing the link $D_A \rightarrow D_B$ arrives at line 16 and generates two new DANs, one for $+d_A$ (do test A) and one for $\neg d_A$ (do not perform test A), which no longer contain the decision D_A. This decomposition corresponds to the branches $+d_A$ and $\neg d_A$ in Figure 7. In the new DANs the node D_A disappears because this decision has been made.

In the DAN for $+d_A$, the method instantiate marks R_A as always-observed. For this reason it has a red oval around it, which means that its value is known when making the next decisions. The tables $P(r_A|d_A)$ and $c_A(d_A)$ have been projected for $+d_A$. Line 4 of Algorithm 1 makes $\mathbf{O} \leftarrow \{R_A\}$ and line 12 invokes evaluateDAN twice: first with $\mathbf{e} = \{+r_A\}$ and then with $\mathbf{e} = \{\neg r_A\}$. In both cases the DAN is decomposed for D_B—as it was previously decomposed for D_A—and then for R_B, T, and X. The upper node CE in Figure 7 corresponds to an invocation of evaluateDAN with a DAN in which all the decisions have been removed and every chance node has been assigned a value , $\mathbf{e} = \{+r_A, +r_B, +x\}$.

In the DAN for $\neg d_A$ (see again Fig. 6), the node R_A has disappeared because the two values of this variable, "positive" and "negative", are incompatible with $\neg d_A$, due to the total restriction for link $D_A \rightarrow R_A$. This DAN is subsequently decomposed as in the previous case, and also the DAN containing the link $D_B \rightarrow D_A$ is evaluated in the same way.

Appendix A.2.2. Probabilities and CEPs returned

Every recursive call to evaluateDAN (Algorithm 1) returns a probability, i.e., a single number between 0 and 1, and a CEP. The call corresponding to the upper CE node in Figure 7 returns the probability $P(\mathbf{e}) = P(+x, +r_A, +r_B) = P(+x) \cdot P(+r_A \mid +x) \cdot P(+r_B \mid +x) = 0.14 \cdot 0.78 \cdot 0.90 = 0.09828$, computed from the conditional probabilities of the chance nodes, just as in the case of Bayesian networks [27]. The CEP for this node consists of a single interval, $(0, +\infty)$, which implies that every leaf node has only one value of cost and one of effectiveness, just as in the standard CEA algorithm for decision trees. The cost and the effectiveness for this CEP is the sum of the values of the corresponding nodes: $c = c_A(+d_A) + c_B(+d_B) + c_T(t_2) = 18 + 150 + 70,000 = 70,168$ and $e = e(+x, t_2) = 6.5$—let us remember that decisions that led to this DAN are $D_A = +d_A$, $D_B = +d_B$, and $T = t_2$. The intervention is empty because no decision node has yet been evaluated.

For the other U node shown in Figure 7 we have $P(\mathbf{e}) = P(\neg x, +r_A, +r_B) = P(\neg x) \cdot P(+r_A \mid \neg x) \cdot P(+r_B \mid \neg x) = 0.86 \cdot 0.09 \cdot 0.07 = 0.005418$, $c = c_A(+d_A) + c_B(+d_B) + c_T(t_2) = 70,168$, $e = e(\neg x, t_2) = 9.3$, and the empty intervention.

The algorithm evaluateDAN steps back to node X in that figure. It corresponds to a call in which $\mathbf{e} = \{+r_A, +r_B\}$. Line 13 of Algorithm 1 computes $P(+r_A, +r_B) = P(+x, +r_A, +r_B) + P(\neg x, +r_A, +r_B) = 0.103698$ and line 14 computes $P(+x \mid +r_A, +r_B) = 0.94775$ and $P(\neg x \mid +r_A, +r_B) = 0.05225$. These are the probabilities for the two branches outgoing from node X in Figure 7. When averageCEP (Algorithm 2) is invoked in the next line, these probabilities are used to average the cost and the effectiveness in the resulting CEP, which still has an empty intervention and no threshold because no decision node has yet been evaluated.

The recursion steps back to node T in that figure. It corresponds to a call to evaluateDAN with a network that still contained the decision node T. The "for" loop in line 17 returns three probability

values and three CEPs, one from each branch of T in the tree. The three probabilities are identical, because $P(\mathbf{e}) = P(+r_A, +r_B)$ and the results of the tests do not depend on the therapy selected after doing them. So line 20 can arbitrarily select any of the $P_d(\mathbf{e})$'s. The three CEPs are defined on the same interval, $(0, +\infty)$, but each one has a different cost, effectiveness, and intervention. Then line 21 invokes the method optimalCEP (Algorithm 3), which performs a deterministic analysis (with Algorithm 4) for the only interval of the input CEPs and returns a CEP that, for the numerical parameters of this model, has three intervals.

Node R_B in Figure 7 corresponds to the top right DAN in Figure 1. The "for" loop in line 10 receives two probabilities, $P(+r_A, +r_B)$ and $P(+r_A, \neg r_B)$. Line 14 computes $P(+r_B \mid +r_A) = 0.5557$ and $P(\neg r_B \mid +r_A) = 0.4443$, which are the probabilities of its outgoing branches. The CEP for $+r_A$ contains two thresholds, €7,551.50/QALY and €21,385.50/QALY, while the CEP for $\neg r_A$ contains one, €70,923.70/QALY. The method averageCEP, invoked in the next line, computes the average cost and the average effectiveness for each of the four intervals. In the first one, i.e., when $\lambda <$ €7,551.50/QALY, the optimal intervention is not to do test B and not to apply any therapy, even though A has given a positive result. For the other intervals, test B is done. The optimal intervention for the second interval is: "if B is negative, apply no therapy; if it is positive, therapy 1 is applied". The optimal intervention for the third interval is similar, with therapy 2 instead of therapy 1. In the fourth interval, a positive result of B leads to therapy 1 and a negative result to therapy 2.

The recursion continues all the way back to the root of the trees, which—as we said above—corresponds to a call to evaluateDAN with the original DAN no evidence. The "for" loop in line 24 returns two probabilities, which are both equal to 1, and two CEPs, one for the DAN in which the decision D_A is made before D_B and another one for the DAN in which D_B is made first. Line 28 creates the auxiliary variable OD, i.e., the meta-decision that determines, for each λ-interval, which decision must be made first.

Appendix A.3. Auxiliary algorithms

In this section we present some auxiliary algorithms invoked by the main method, Algorithm 1. They were introduced to evaluate decision trees with embedded decision nodes [18] and later used for evaluating influence diagrams [15]. We reproduce them here for the sake of completeness.

The method instantiate, invoked in lines 11 and 18, takes as arguments a DAN, a variable X, and a value x, and generates a new network, DAN_x, as follows:

1. create DAN_x as a copy of the DAN;
2. if x reveals a variable Y that is not a descendant of any decision (other than X), then declare Y as observed in DAN_x; if Y is a descendant of some decisions (other than X), $\{D'_1, \ldots, D'_n\}$, draw a link $D'_i \rightarrow Y$ for $i \in \{1, \ldots, n\}$—if it did not exist—and declare that D'_i reveals Y unconditionally;
3. if there is a total restriction (x, Y), remove Y; then remove recursively all the chance and utility nodes that are children of Y;
4. if there is a restriction (x, y), remove y from the domain of Y;
5. if a chance node Y is a child of X, project the table $P(y|pa(Y))$ or $u_Y(pa(Y))$ by making $X = x$;
6. if X is a decision, remove node X and its outgoing links.

Line 15 of Algorithm 1 invokes the method averageCEP, given by Algorithm 2. Its input consists of a variable X having m values (states), a probability distribution $P(x)$, and m CEPs. It first gathers all the thresholds of the input CEPs and computes, for each interval, the average cost, the average effectiveness, and the intervention, taken the cost, effectiveness, and interventions of the input CEPs for those intervals. Again, an example can be found in [18].

Line 21 of Algorithm 1 invokes the method optimalCEP, given by Algorithm 3. Its input consists of a decision D having m values (options) and m CEPs. It first gathers all the thresholds of the input CEPs and performs, within each interval, a deterministic CEA, which may in turn generate new thresholds. Those thresholds that lie within the interval analyzed are added to the set of CEPs, Θ; the others are discarded. For each of the new intervals, the algorithm selects the value d_k of D that maximizes the NMB. It determines the cost and the effectiveness for that interval, just as when selecting the optimal branch outgoing from a decision node in a tree. The optimal intervention for that interval begins by selecting $D = d_k$ and continues with the partial intervention that the k-th CEP determined for that interval. Finally, the algorithm eliminates the thresholds that are not necessary, i.e., those that separate intervals having the same cost, effectiveness, and intervention. Those thresholds were generated in branches of the tree that later turned out to be suboptimal. An example can be found in [18].

Algorithm 2: Weighted average of CEPs

function: `averageCEP`
input : X – a chance variable whose domain is $\{x_1, \ldots, x_m\}$,
$P(x_j)$ – a probability distribution for X, and
$\{Q_1, \ldots, Q_m\}$ – a set of CEPs
output : *CEP* – a cost-effectiveness partition

1 $\Theta \leftarrow \bigcup_{j=1}^{m} \Theta_j$
2 $n \leftarrow$ card$(\Theta) + 1$
3 **for** $i \leftarrow 1$ **to** n **do**
4 $\quad c_i \leftarrow \sum_{j=1}^{m} P(x_j) \cdot cost_{Q_j}(\theta_i)$
5 $\quad e_i \leftarrow \sum_{j=1}^{m} P(x_j) \cdot \mathit{eff}_{Q_j}(\theta_i)$
6 $\quad I_i \leftarrow$ "If $X = x_1$, then $interv_{Q_1(\theta_i)}$; if $X = x_2$, then $interv_{Q_2(\theta_i)}$..."
7 **return** $((\theta_1, \ldots, \theta_{n-1}), (c_0, \ldots, c_n), (e_0, \ldots, e_n), (I_0, \ldots, I_n))$

Algorithm 3: Optimal CEP

function: `optimalCEP`
input : D – a decision node whose domain is $\{d_1, \ldots, d_m\}$ and
$\{Q_1, \ldots, Q_m\}$, a set of m CEPs with $Q_j = (\Theta_j, C_j, E_j, I_j)$
output : *CEP* – a cost-effectiveness partition

1 $\Theta \leftarrow \bigcup_{j=1}^{m} \theta_j$
2 $n \leftarrow$ card(Θ)+1
3 **for** $i \leftarrow 1$ **to** n **do**
4 $\quad$ perform a deterministic CEA analysis within the i-th interval
5 $\quad$ add to Θ the new thresholds that belong to this interval
6 $n \leftarrow$ card(Θ)+1
7 **for** $i \leftarrow 1$ **to** n **do**
8 $\quad k \leftarrow \arg\max_j \mathit{NMB}(\theta_i)$
9 $\quad c_i \leftarrow cost_{Q_k}(\theta_i)$
10 $\quad e_i \leftarrow \mathit{eff}_{Q_k}(\theta_i)$
11 $\quad I_i \leftarrow$ "$D = d_k$; $interv_{Q_k(\theta_i)}$..."
12 fuse contiguous intervals that have the same intervention, the same cost, and the same effectiveness.

The deterministic CEA performed in line 4 of this algorithm might be done with the traditional method, which consists in keeping the non-dominated interventions and discarding those dominated by others (single dominance or extended dominance) [28]. A slightly more efficient method, introduced in [18], is given by Algorithm 4. It begins by selecting the intervention with the lowest cost and then the one with the lowest ICER among those with higher effectiveness, and so on.

Algorithm 4: Deterministic cost-effectiveness analysis

function: `deterministicCEA`
input : a set of interventions $\{I_1, \ldots, I_m\}$
output : *CEP* – a cost-effectiveness partition

1 $\sigma(0) \leftarrow \arg\min_i \ cost(I_i)$
2 $R_0 \leftarrow \{i \mid I_i \in I \wedge \mathit{eff}(I_i) > \mathit{eff}(I_{\sigma(0)})\}$
3 $i \leftarrow 1$;
4 **while** $R_{i-1} \neq \varnothing$ **do**
5 $\quad \sigma(i) := \arg\min_{j \in R_i} \ ICER(I_{\sigma(i-1)}, I_j)$
6 $\quad \theta_i \leftarrow \min_{j \in R_i} \ ICER(I_{\sigma(i-1)}, I_j)$
7 $\quad c_i \leftarrow cost(I_{\sigma(i)})$
8 $\quad e_i \leftarrow \mathit{eff}(I_{\sigma(i)})$
9 $\quad R_i \leftarrow \{j \mid j \in R_{i-1} \wedge \mathit{eff}(I_j) > \mathit{eff}(I_{\sigma(i)})\}$
10 $\quad i \leftarrow i + 1$
11 **return** $((\theta_1, \ldots, \theta_{n-1}), (c_0, \ldots, c_n), (e_0, \ldots, e_n), (I_0, \ldots, I_n))$

Finally, the method prioritize, invoked in line 25 of Algorithm 1 with a DAN and a decision node D as arguments, is implemented by just drawing links to the other candidates to be the first decision, i.e., the nodes in $\mathbf{D}_I$.

[1] M. F. Drummond, M. J. Sculpher, K. Claxton, G. L. Stoddart, G. W. Torrance, Methods for the Economic Evaluation of Health Care Programmes, cuarta Edition, Oxford University Press, Oxford, UK, 2015.

[2] P. J. Neumann, G. D. Sanders, L. B. Russell, J. E. Siegel, T. G. Ganiats, Cost-Effectiveness in Health and Medicine, 2nd Edition, Oxford University Press, New York, 2016.

[3] A. A. Stinnett, J. Mullahy, Net health benefit: A new framework for the analysis of uncertainty in cost-effectiveness analysis, Medical Decision Making 18 (1998) S68–80.

[4] M. C. Weinstein, G. Torrance, A. McGuire, QALYs: The basics, Value in Health 12 (2009) S5–S9.

[5] H. Raiffa, R. Schlaifer, Applied Statistical Decision Theory, John Wiley & Sons, Cambridge, MA, 1961.

[6] K. M. Kuntz, M. C. Weinstein, Modelling in economic evaluation, in: M. F. Drummond, A. McGuire (Eds.), Economic Evaluation in Health Care, Oxford University Press, New York, 2001, Ch. 7, pp. 141–171.

[7] M. Arias, F. J. Díez, The problem of embedded decision nodes in cost-effectiveness decision trees, Pharmacoeconomics 32 (2014) 1141–1145. doi:10.1007/s40273-014-0195-1.

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

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

[10] M. Luque, F. J. Díez, C. Disdier, Optimal sequence of tests for the mediastinal staging of non-small cell lung cancer, BMC Medical Informatics and Decision Making 16 (2016) 1–14. doi:10.1186/s12911-016-0246-y.

[11] M. Gómez, C. Bielza, J. A. Fernández del Pozo, S. Ríos-Insua, A graphical decision-theoretic model for neonatal jaundice, Medical Decision Making 27 (2007) 250–265.

[12] F. Jensen, F. V. Jensen, S. L. Dittmer, From influence diagrams to junction trees, in: R. L. de Mántaras, D. Poole (Eds.), Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence (UAI'94), Morgan Kaufmann, San Francisco, CA, 1994, pp. 367–373.

[13] F. V. Jensen, T. D. Nielsen, Bayesian Networks and Decision Graphs, 2nd Edition, Springer-Verlag, New York, 2007.

[14] M. Luque, F. J. Díez, Variable elimination for influence diagrams with super-value nodes, International Journal of Approximate Reasoning 51 (2010) 615–631. doi:10.1016/j.ijar.2009.11.004.

[15] M. Arias, F. J. Díez, Cost-effectiveness analysis with influence diagrams, Methods of Information in Medicine 54 (2015) 353–358. doi:10.3414/ME13-01-0121.

[16] F. J. Díez, M. Luque, I. Bermejo, Decision analysis networks, International Journal of Approximate Reasoning 96 (2018) 1–17. doi:10.1016/j.ijar.2018.02.007.

[17] D. Koller, N. Friedman, Probabilistic Graphical Models: Principles and Techniques, The MIT Press, Cambridge, MA, 2009.

[18] M. Arias, F. J. Díez, Cost-effectiveness analysis with sequential decisions, Technical Report CISIAD-11-01, UNED, Madrid, Spain (2011).

[19] M. Arias, F. J. Díez, M. P. Palacios, ProbModelXML. a format for encoding probabilistic graphical models, Technical Report CISIAD-11-02, UNED, Madrid, Spain (2011).

[20] J. A. Sacristán, J. Oliva, J. del Llano, L. Prieto, J. L. Pinto, ¿Qué es una tecnología sanitaria eficiente en España?, Gaceta Sanitaria 16 (2002) 334–343.

[21] E. de Cock, M. Miravitlles, J. R. González-Juanatey, J. R. Azanza-Perea, Valor umbral del coste por año de vida ganado para recomendar la adopción de tecnologías sanitarias en España: evidencias procedentes de una revisión de la literatura, Pharmacoeconomics Spanish Research Articles 4 (2007) 97–107.

[22] S. Walker, F. Girardin, C. McKenna, S. Ball, J. Nixon, S. Plein, J. P. Greenwood, M. Sculpher, Cost-effectiveness of cardiovascular magnetic resonance in the diagnosis of coronary heart disease: an economic evaluation using data from the CE-MARC study, Heart 99 (2013) 873–881.

[23] C. Bielza, M. Gómez, P. P. Shenoy, Modeling challenges with influence diagrams: Constructing probability and utility models, Decision Support Systems 49 (2010) 354–364.

[24] C. Lacave, M. Luque, F. J. Díez, Explanation of Bayesian networks and influence diagrams in Elvira, IEEE Transactions on Systems, Man and Cybernetics—Part B: Cybernetics 37 (2007) 952–965. doi:10.1109/TSMCB.2007.896018.

[25] F. J. Díez, M. Yebra, I. Bermejo, M. A. Palacios-Alonso, M. Arias, M. Luque, J. Pérez-Martín, Markov influence diagrams: A graphical tool for cost-effectiveness analysis, Medical Decision Making 37 (2017) 183–195. doi:10.1177/0272989X16685088.

[26] L. Prada, Optimization of policy trees for inluence diagrams, Master's thesis, Dept. Artificial Intelligence, UNED, Madrid, Spain (2014).

[27] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, San Mateo, CA, 1988.

[28] M. C. Weinstein, H. V. Fineberg, A. S. Elstein, et al., Clinical Decision Analysis, Saunders, Philadelphia, PA, 1980.