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), a new type of probabilistic graphical model, somewhat similar to Bayesian networks and influence diagrams. We present an algorithm for evaluating DANs with two criteria, cost and effectiveness, 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 and then present 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; decision trees; probabilistic graphical models; influence diagrams; decision analysis networks.

Declaration of interest: none.

1. Introduction

Due to the rapid increase of medical expenditures in all countries, every health system must analyze whether the benefit of each intervention outweighs its economic cost [13, 26]. 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

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

is positive, do test B, and if it is also positive, apply drug D; if A is negative then. . . ". Cost-effectiveness analysis is a particular type of economic evaluation that tries to select the interventions having a positive *net monetary benefit* [32],

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

where e is the average effectiveness and c the average cost of the intervention for the subpopulation of interest. 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) [35]. As the willingness to pay is specific for each decision maker—in this case, for each health system—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.

In spite of the increasing importance of economic evaluations for medical decision making, artificial intelligence has not addressed this problem until very recently, due to the lack of appropriate tools. The main method that health economists use for medical decision analysis are decision trees, which were proposed around 60 years ago [30]. Their main drawback is that the number of branches grows very fast with the number of decisions and chance variables in the problem. The growth is even faster for CEA than for the unicriterion case, because the standard algorithm [30] cannot evaluate decision trees with embedded decision nodes, which are those other than the root of the tree [19, 2]—see Appendix A.1.

Probabilistic graphical models, which include Bayesian networks [27], influence diagrams (IDs) [15, 8], and several types of Markov models, have been extensively studied in the field of artificial intelligence [18]. IDs, like decision trees, have three types of nodes: chance, decision, and value. Their main advantage is 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 [21, 25]. The ID IctNeo is even larger [14]. An ID can be evaluated by converting it into a decision tree, but there are much more efficient algorithms [16, 17, 24], not only for the unicriterion case, but also for CEA [3]. 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) [11] as a new type of probabilistic graphical model 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. However, so far DANs could only be applied to unicriterion problems.

The main contribution of this paper is a CEA algorithm for DANs, which combines the method for evaluating unicriterion DANs [11] and the CEA algorithms for decision trees with embedded decision nodes [1] and for IDs [3]. The basic idea is straightforward, but the integration of those algorithms is not trivial. We show that this method can solve several problems that could not be addressed with previous techniques. The first example is a fictitious case 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 modeled this problem with an ID [25], which we later converted into a DAN.

2. Methods

We introduce here the n-test problem as an example involving unordered decisions and then show how to solve it for $n = 2$ using DANs.

2.1. The n-test problem

In this problem there are two mutually-exclusive therapies for a disease X; there are also 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, therapy 2 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, as shown in the rightmost column of Table 1.

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.

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

In Appendix A.1 we discuss why, in spite of the apparent simplicity of this example, it is difficult to determine the optimal intervention (as a function of λ).

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

Table 2: Cost, sensitivity, and specificity of each test.

2.2. Representation of decision problems with DANs

This section presents 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 $V_1 \rightarrow V_2$ 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 $V_1 \rightarrow V_2$ 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 Díez et al. [11] 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

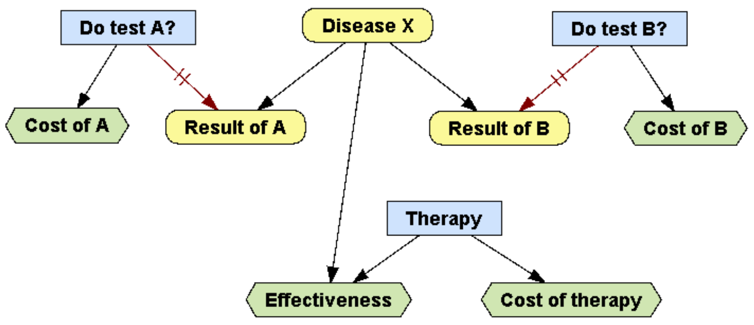
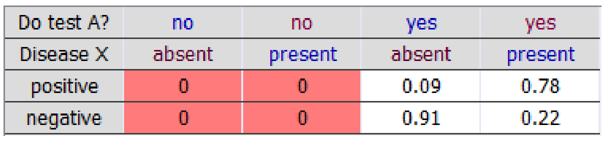


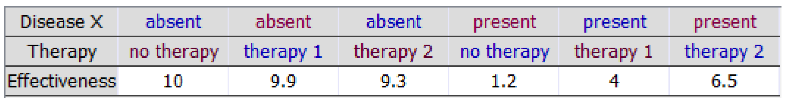
Figure 1: A DAN for the 2-test problem. Rectangles represent decisions. Rounded rectangles represent chance variables. Hexagons represent values, i.e., cost and effectiveness. Links represent causal influences. Links colored in red represent revelation conditions; for example, doing a test reveals the 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.

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 [4]. Each value node also has an associated table, as in Figure 3.



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.



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 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 [25]. The main difference is that the decisions in the DAN are only partially ordered.

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 [30] assigns a numeric value to each node, the algorithm in [1] 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 them into a finite set of intervals, such that within each interval the cost, the effectiveness, and the intervention

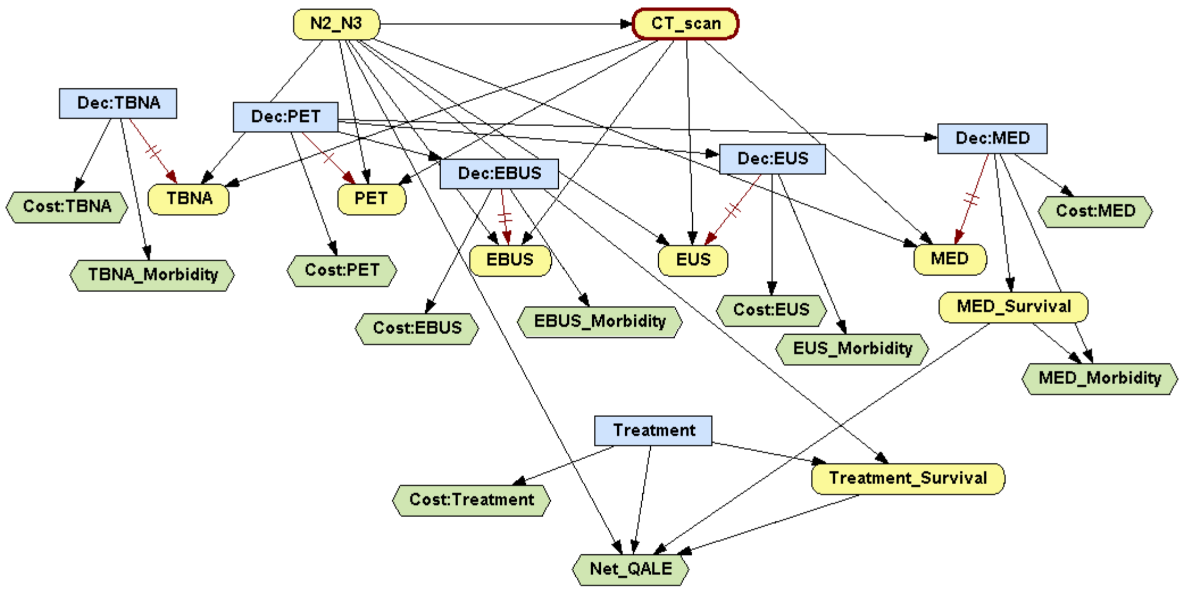


Figure 4: MEDIASTINET, a DAN for the mediastinal staging of non-small cell lung cancer (TBNA = transbronquial needle aspiration; PET = positron emission tomography; EBUS = endobronchial ultrasound; EUS = endoscopic ultrasound; MED = mediastinoscopy; QALY = quality-adjusted life expectancy).

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 [3] and DANs.

2.3.1. Cost-effectiveness partition (CEP)

Let us consider two mutually-exclusive interventions, A and B, such that the latter is more effective ($e_B > e_A$) but more expensive ($c_B > c_A$). If we define the incremental cost-effectiveness ratio (ICER),

$$ICER(A,B) = \frac{c_B - c_A}{e_B - e_A} \ , \tag{2}$$

then it follows from Equation 1 that

$$NMB_B > NMB_A \iff \lambda > ICER(A,B) \ . \tag{3}$$

Therefore, the ICER splits the possible values of λ into two subintervals: if $\lambda \in (0, ICER(A,B))$, the optimal intervention is A; if $\lambda \in (ICER(A,B), +\infty)$, then B has a higher net benefit. This result can be generalized to an arbitrary number of interventions, as following.

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

- $I = \{I_0, \ldots, I_{n-1}\}$ – a set of n interventions,
- $C = \{c_0, \ldots, c_{n-1}\}$ – a set of n costs, such that $c_0 < c_1 < \ldots < c_{n-1}$,
- $E = \{e_0, \ldots, e_{n-1}\}$ – a set of n effectiveness values, such that $e_0 < e_1 < \ldots < e_{n-1}$, and

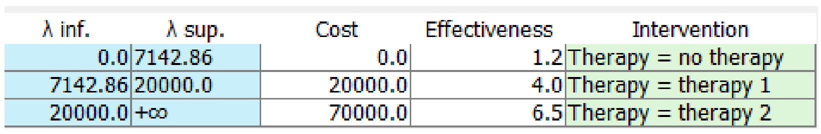
- $\Theta = \{\theta_1, \ldots, \theta_{n-1}\}$ – a set of $n-1$ thresholds, such that $0 < \theta_1 < \ldots < \theta_{n-1}$.

In practice, every CEP results from a deterministic CEA with Algorithm 2 (see Appendix A.2), which takes as input 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. The thresholds are given by the ICERs:

$$\theta_i = ICER(I_{i-1}, I_i) = \frac{c_i - c_{i-1}}{e_i - e_{i-1}} \ . \tag{4}$$

I_i is the optimal intervention when $\theta_i < \lambda < \theta_{i+1}$ (with $i \in \{0, 1, \ldots, n-1\}$) and I_{n-1} is the optimal intervention when $\lambda > \theta_{n-1}$.

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 is a deterministic CEA because in this case there is no uncertainty.) This CEP 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.



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, θ_1 = €7,142.86/QALY and θ_2 = €20,000.00/QALY, obtained with Equation 4, delimit three intervals for λ.

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. As an example, we illustrate its performance for the 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.[1] 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.[2] The decomposition of the former generates two new DANs, in which D_A disappears because

[1]The evidence $\mathbf{e}$ is a configuration of the variables observed so far; for example, if only test A has been done and it has given a positive result, then $\mathbf{e} = (+r_A)$. In Algorithm 1, $\mathbf{e} \circ x$ denotes the composition of two configurations; for example, if $\mathbf{e} = (+r_A)$ then $\mathbf{e} \circ \neg r_B = (+r_A, \neg r_B)$.

[2]There might be a third DAN in which the first decision would be T, i.e., which therapy to apply, if any. However, the decisions that can reveal some information (in this example, D_A and D_B may reveal the value of R_A and R_B respectively) should be made earlier than those that cannot——see [11, Sec. 4.1]—so Tmust be the last decision.

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
1  if $DAN$ has decision nodes then
2      $\mathbf{O} \leftarrow$ always-observed variables in $DAN$, except those in $\mathbf{E}$;
3  else
4      $\mathbf{O} \leftarrow$ all the chance variables in $DAN$, except those in $\mathbf{E}$;
5  if $\mathbf{O} = \varnothing$ and $DAN$ has no decisions then
6      $P(\mathbf{e}) \leftarrow$ product of the projected probability potentials;
7      $CEP(\mathbf{e}) \leftarrow$ sum of the projected utility potentials;
8  else if $\mathbf{O} \neq \varnothing$ then
9      select $X \in \mathbf{O}$ such that $X$ has no ancestor in $\mathbf{O}$;
10     foreach $x$ of $X$ do
11         $DAN_x \leftarrow$ instantiate ($DAN$, $X$, $x$);
12         $\{P(\mathbf{e} \circ x), CEP(\mathbf{e} \circ x)\} \leftarrow$ evaluateDAN($DAN_x$, $\mathbf{e} \circ x$);
13     $P(\mathbf{e}) \leftarrow \sum_x P(\mathbf{e} \circ x)$;
14     $P(x \mid \mathbf{e}) \leftarrow P(\mathbf{e} \circ x)/P(\mathbf{e})$;
15     $CEP(\mathbf{e}) \leftarrow$ averageCEP($X, P(x \mid \mathbf{e}), \{CEP(\mathbf{e} \circ x_1), \ldots, CEP(\mathbf{e} \circ x_m)\}$);
16 else if exactly one decision $D$ can be made first then
17     foreach $d$ of $D$ do
18         $DAN_d \leftarrow$ instantiate ($DAN$, $D$, $d$);
19         $\{P_d(\mathbf{e}), CEP_d(\mathbf{e})\} \leftarrow$ evaluateDAN($DAN_d$, $\mathbf{e}$);
20     $P(\mathbf{e}) \leftarrow P_d(\mathbf{e})$ for an arbitrary value $d$;
21     $CEP(\mathbf{e}) \leftarrow$ optimalCEP($D, \{CEP_{d_1}(\mathbf{e}), \ldots, CEP_{d_m}(\mathbf{e})\}$);
22 else
23     $\mathbf{D}_I \leftarrow$ decisions that can be made first;
24     foreach $D$ in $\mathbf{D}_I$ do
25         $DAN_D \leftarrow$ prioritize ($DAN$, $D$);
26         $\{P_D(\mathbf{e}), CEP_D(\mathbf{e})\} \leftarrow$ evaluateDAN($DAN_D$, $\mathbf{e}$);
27     $P(\mathbf{e}) \leftarrow P_D(\mathbf{e})$ for an arbitrary decision $D$;
28     $OD \leftarrow$ new decision variable (meta-decision), with one value for each decision $D$ in $\mathbf{D}_I$;
29     $CEP(\mathbf{e}) \leftarrow$ optimalCEP($OD, \{CEP_{D_1}(\mathbf{e}), \ldots, CEP_{D_m}(\mathbf{e})\}$);
30 return $\{P(\mathbf{e}), CEP(\mathbf{e})\}$
```

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 the probability $P(\mathbf{e})$ and a CEP, and the recursion continues backwards. At the end of the recursion 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 A.3.

Every node in the decomposition tree (Fig. 6) corresponds to a node in a decision tree (Fig. 7). This 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

We have implemented Algorithm 1 in OpenMarkov (see `www.openmarkov.org`), an open-source tool for probabilistic graphical models [18], such as Bayesian networks, influence diagrams (IDs), Markov influence diagrams (MIDs), partially-observable Markov decision processes (POMDPs), DANs, etc. Models having several criteria can be evaluated by joining them into a single criterion; for those having two criteria it is also possible to perform a CEA. OpenMarkov's tutorial explains how to build and evaluate these models using OpenMarkov's graphical user interface.

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 intervention. When the willingness to pay is very low, more precisely $\lambda <$ €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 directly applying 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 amounts to adding three nodes, which takes around one minute in OpenMarkov's graphical user

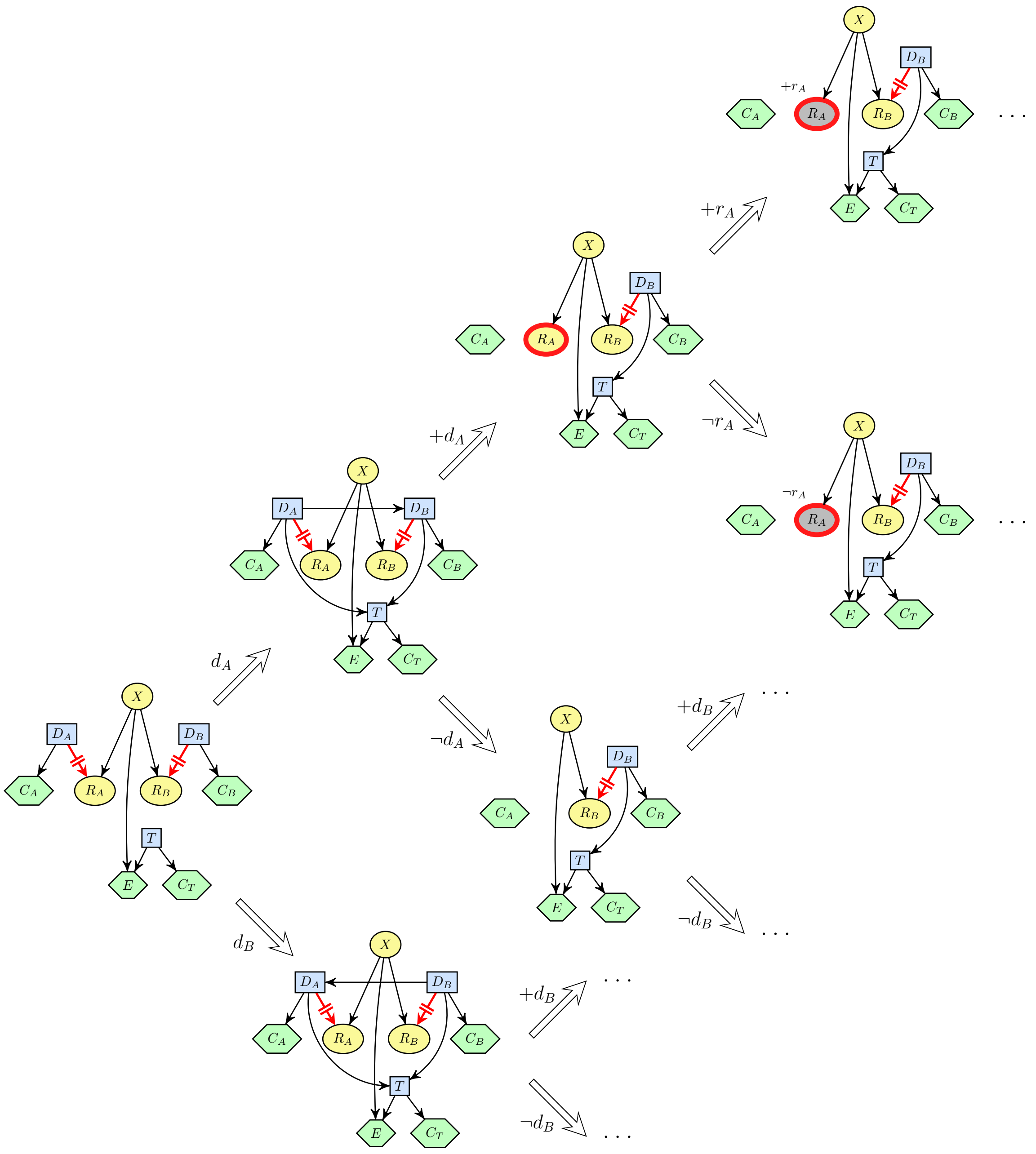


Figure 6: Recursive decomposition of the DAN for the 2-test problem. Every node in this tree corresponds to an invocation of evaluateDAN (Algorithm 1). In the DANs, chance nodes having associated evidence are colored in gray. Legend: X = disease; D_A = decision about test A; R_A = result of test A; C_A = cost of test A; T = decision about therapy; E = effectiveness; C_T = cost of therapy.

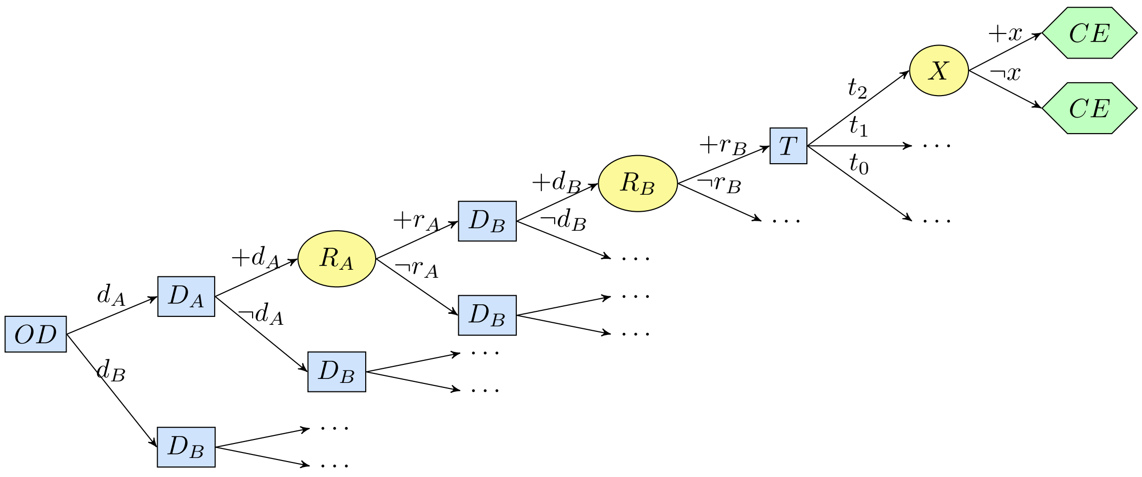
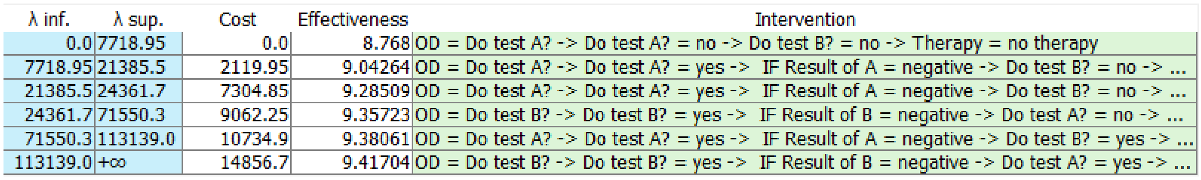


Figure 7: Decision tree generated from the DAN for the 2-test problem. Every node in this tree corresponds to a node in the decomposition tree (Fig. 6), i.e., to an invocation of evaluateDAN. The "meta-decision" node OD (order of the decisions) determines which decision to make first. Legend: CE = cost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



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.

interface, while adding a new test in a decision tree without embedded decisions, as recommended by Kuntz and Weinstein [19], is a daunting task for $n > 2$. 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 [11]. The CEA version of Algorithm 2 in that paper, which is much more efficient, 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 (even with embedded decision nodes) have more than 6 million leaves for $n = 7$ and more than 100 million for $n = 8$, as explained in 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 [31, 10]. 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 [25]—while the DAN recommends doing an endoscopic ultrasound test (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.

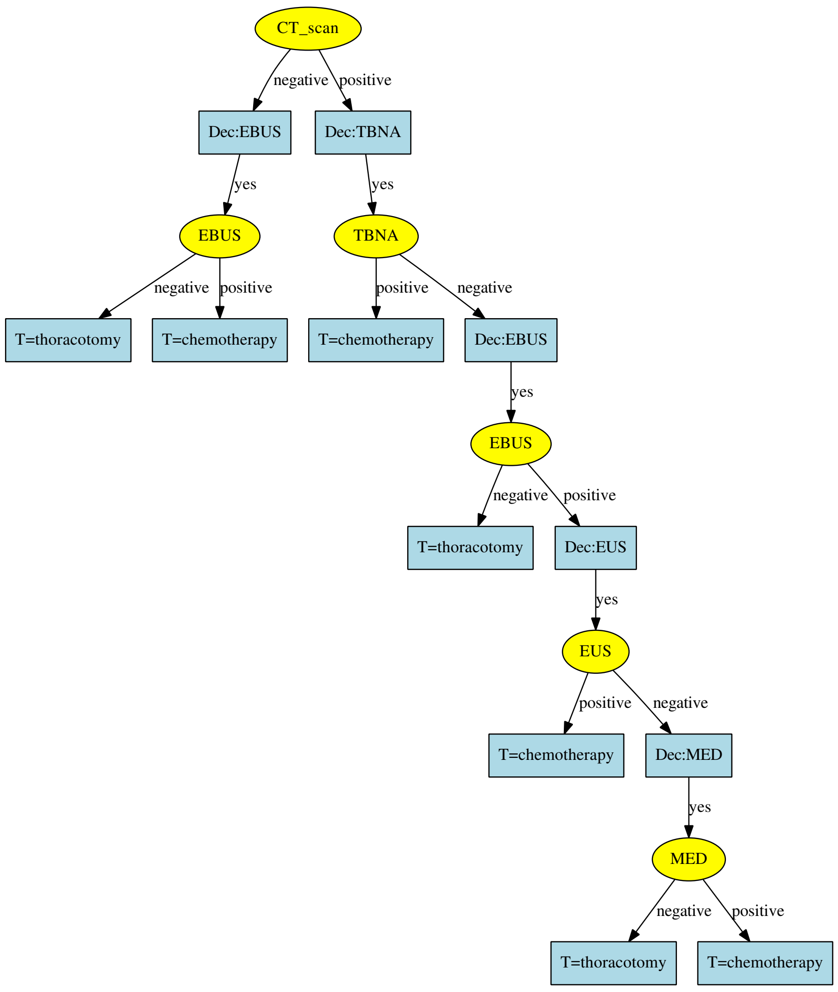


Figure 9: Optimal intervention for $\lambda = €30{,}000/\text{QALY}$ obtained from then DAN MEDIASTINET (cf. Fig. 4).

The website `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 [2]. It would be necessary to build a tree containing one decision node, at the root, with an outgoing branch for each intervention, as proposed by Kuntz and Weinstein [19]. In Appendix A.1 we show that when no test is available ($n = 0$), there are 3 interventions, 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 [33]. It is likely that the shortlist contained the optimal intervention, but it is not necessarily so. Similarly, the approach by Quaglini et al. [29] orders the tests from the least invasive to the most invasive (a total ordering) and only considers the do-nothing strategy and those strategies in which a test is done when the previous one was positive. Additionally, in the 2-test problem shown above that approach is inapplicable because both tests are non-invasive (no one has a cost in QALYs).

In contrast, the approach we propose in this paper considers all the possible strategies, including some that might not be obvious. Thus, the first three strategies obtained for the 2-test problem in Section 3 are straightforward, but the others are not. The fourth and the sixth may be counterintuitive because common sense says that cheap tests should be done before expensive ones—at least it is not usual in medicine to do first an expensive test and then a cheaper one with lower sensitivity and specificity. The fifth intervention is also counterintuitive because it does both tests—in any order—even when the first one is negative. Therefore some decision analysts, including ourselves, might have overlooked those strategies when building a model with completely ordered decisions. 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 EBUS should be done before EUS or vice versa, which made him feel uncomfortable. As explained above, this question was 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 [1]. 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 infeasible 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 [3], 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 [7, 11]. For this reason even when a problem can be modeled with an ID, the DAN representation is cleaner and more elegant.

Artificial intelligence researchers, who are in general familiar with Bayesian networks and IDs, will find it easy to build DANs. On the contrary, most health economists, used to building decision trees and implementing them in Excel, TreeAge, or R, would need to learn a new representation formalism, based on conditional independences. However, the effort will be small for someone acquainted with the fundamentals of decision analysis, and the possibility of solving easily much larger problems will soon compensate for the time invested. Additionally, the availability of an open-source tool for DANs avoids the need to acquire software licenses for building and evaluating decision trees, and the effort of understanding and implementing the algorithms for DANs.[3]

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 around one minute in OpenMarkov, as mentioned above, 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 [11]), which can take profit of tensor-processing libraries that run on GPUs and TPUs.

In Appendix A.4 we compare DANs with decision circuits [5, 6, 22], a formalism that, to our

[3]In fact, OpenMarkov can automatically convert a DAN into a decision tree (if the tree is too big, the user can select the depth of the expansion, up to the limit allowed by the available working memory). 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.

knowledge, has never been used to solve real-world problems.

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 [1] and for IDs [3], and those for unicriterion DANs [11]. 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 or the compilation of DANs into decision circuits. Another line is to design sensitivity analysis options and explanation facilities similar to those existing for IDs [20]. A third line is to develop Markov DANs, which will be able to model the evolution of diseases in discrete time; these networks will combine the advantages of DANs with those of Markov IDs [12]. And fourth, we might try to compact the interventions returned by the algorithms [28, 23].

Acknowledgments

We thank the anonymous reviewers for their comments. This work has been supported by grants TIN2016-77206-R and PID2019-110686RB-I00 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 the decision trees for the n-test problem

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 a 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 solving this problem is hard for two tests and unfeasible 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.[4]

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.[5] The solution proposed by Kuntz and Weinstein [19] 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 by 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

[4]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.

[5]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 [2].

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

Appendix A.2. Basic algorithms for CEA

In this section we present an algorithm for deterministic CEA as well the auxiliary algorithms invoked by Algorithm 1. They were all introduced to evaluate decision trees with embedded decision nodes [1] and later used for evaluating influence diagrams [3]. We reproduce them here for the sake of completeness.

A deterministic CEA takes as arguments a set of interventions, each one having known cost and effectiveness, and returns the non-dominated interventions, discarding those dominated by others (single dominance or extended dominance) [34]; it also returns a set of thresholds, i.e., the ICERs defined by Equation 4. A slightly more efficient method is given by Algorithm 2. 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. The output is structured as a CEP. An example can be found in [1].

The method instantiate, invoked in lines 11 and 18 of Algorithm 1, 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 $D'_i \to Y$ 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;

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

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 3. 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 [1].

Line 21 of Algorithm 1 invokes the method optimalCEP, given by Algorithm 4. 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 sub-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 contiguous intervals having the same cost, effectiveness, and intervention. Those thresholds were generated in branches of the tree that later turned out to be suboptimal—see again [1] for an example.

Finally, the method prioritize, invoked in line 25 of Algorithm 1 with a DAN and a decision node D

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

as arguments, is implemented by just drawing links to the other candidates to be the first decision, i.e., the nodes in $\mathbf{D}_I$.

Appendix A.3. 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.3.1. Decomposition of the DAN

Every invocation of the Algorithm 1 takes two arguments: a DAN and a set of findings, denoted by $\mathbf{e}$ (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 (as explained in footnote 2); 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 $D_A \to D_B$ and $D_A \to T$; 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 $D_A \to D_B$ 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 $D_A \to R_A$. This DAN is subsequently decomposed as in the previous case, and also the DAN containing the link $D_B \to D_A$ is evaluated in the same way.

Appendix A.3.2. 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 3) 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 4), which performs a deterministic analysis (with Algorithm 2) 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.4. DANs vs. decision circuits

Decision circuits were developed as a method for evaluating influence diagrams [5]—in the same way as arithmetic circuits were developed for Bayesian networks [9]—and later used for solving asymmetric problems [6]. As a representation formalism, decision circuits suffer from the same problem as decision trees: the size of the model. According to Proposition 2 in [22], the decision circuit for the n-test problem has $(c+1)^n$ decision nodes, plus many sum, product, and terminal nodes. Thus, the graph in Figure 7 in that paper, with $n = c = 2$, has 9 decision nodes and a total of 95 nodes. Building a decision circuit for a problem like MEDIASTINET is unfeasible in practice, even if there were a software tool for this task. In contrast, the DAN for the n-test problem only needs $3(n+1)$ nodes, regardless of the number of outcomes of each test, i.e., 9 nodes in total for $n = 2$.

With respect to efficiency, the computational complexity of solving the n-test problem with decision circuits is $O(n(c+1)^{n+1})$, as stated in Theorem 5 of [22], while the time complexity of the algorithms for DANs is $O(n!)$. It would be possible to reduce this complexity to exponential, at the expense of increasing the space complexity, i.e., using more working memory, by detecting the DANs that appear repeatedly during the recursive decomposition of the original network (see Figure 6); this is essentially the same as applying coalescence in decision trees and decision circuits.

Given the impossibility of manually building decision circuits for non-trivial problems, it might be interesting to develop an algorithm for converting DANs into decision circuits and another algorithm for evaluating decision circuits having two criteria, cost and effectiveness.

References

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

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

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

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

[5] Bhattacharjya, D., Shachter, R.D., 2007. Evaluating influence diagrams with decision circuits, in: Parr, R., van der Gaag, L.C. (Eds.), Proceedings of the Twenty-third Conference on Uncertainty in Artificial Intelligence (UAI'07), AUAI Press, Corvallis, OR. p. 9–16.

[6] Bhattacharjya, D., Shachter, R.D., 2012. Formulating asymmetric decision problems as decision circuits. Decision Analysis 9, 138–145.

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

[8] Boutilier, C., 2005. The influence of influence diagrams on artificial intelligence. Decision Analysis 2, 229–231.

[9] Darwiche, A., 2003. A differential approach to inference in Bayesian networks. Journal of the ACM 50, 280–305.

[10] de Cock, E., Miravitlles, M., González-Juanatey, J.R., Azanza-Perea, J.R., 2007. 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, 97–107.

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

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

[13] Drummond, M.F., Sculpher, M.J., Claxton, K., Stoddart, G.L., Torrance, G.W., 2015. Methods for the Economic Evaluation of Health Care Programmes. 4th ed., Oxford University Press, Oxford, UK.

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

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

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

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

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

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

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

[21] León, D., 2011. A Probabilistic Graphical Model for Total Knee Arthroplasty. Master's thesis. Dept. Artificial Intelligence, UNED. Madrid, Spain.

[22] Liu, W., Shachter, R.D., 2015. Complexity of the exact solution to the test sequencing problem, in: Proceedings of the Thirty-first Conference on Uncertainty in Artificial Intelligence (UAI'15), AUAI Press, Amsterdam, Netherlands. p. 494–503.

[23] Luque, M., Arias, M., Díez, F.J., 2017. Synthesis of strategies in influence diagrams, in: Proceedings of the Thirty-third Conference on Uncertainty in Artificial Intelligence (UAI'17), AUAI Press, Corvallis, OR. p. 1–9. URL: http://auai.org/uai2017/proceedings/papers/134.pdf.

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

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

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

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

[28] Prada, L., 2014. Optimization of Policy Trees for Inluence Diagrams. Master's thesis. Dept. Artificial Intelligence, UNED. Madrid, Spain.

[29] Quaglini, S., Dazzi, L., Stefanelli, M., Barosi, G., Marchetti, M., 1999. Decision support systems in health economics. Topics in Health Information Management 20, 16–30.

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

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

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

[33] Walker, S., Girardin, F., McKenna, C., Ball, S., Nixon, J., Plein, S., Greenwood, J.P., Sculpher, M., 2013. 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, 873–881.

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

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