Data & Knowledge Engineering 70 (2011) 167–181

Contents lists available at ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

Dealing with complex queries in decision-support systems

J.A. Fernández del Pozo, C. Bielza *

Departamento de Inteligencia Artificial, Facultad de Informática, Universidad Politécnica de Madrid, Boadilla del Monte, Madrid 28660, Spain

ARTICLE INFO

Article history:
Received 19 March 2009
Received in revised form 8 October 2010
Accepted 8 October 2010
Available online 30 October 2010

Keywords:
Decision-support systems
Query system
Information systems
Explanations
Influence diagrams
Medicine

ABSTRACT

In decision-making problems under uncertainty, a decision table consists of a set of attributes indicating what is the optimal decision (response) within the different scenarios defined by the attributes. We recently introduced a method to give explanations of these responses. In this paper, the method is extended. To do this, it is combined with a query system to answer expert questions about the preferred action for a given instantiation of decision table attributes. The main difficulty is to accurately answer queries associated with incomplete instantiations. Incomplete instantiations are the result of the evaluation of a partial model outputting decision tables that only include a subset of the whole problem, leading to uncertain responses. Our proposal establishes an automatic and interactive dialogue between the decision-support system and the expert to elicit information from the expert to reduce uncertainty. Typically, the process involves learning a Bayesian network structure from a relevant part of the decision table and computing some interesting conditional probabilities that are revised accordingly.

1. Introduction

1.1. Decision tables, explanations and queries

Under uncertainty, a modern and useful decision-theoretic model is the *influence diagram* [17]. It consists of an acyclic directed graph with associated probabilities and utilities, respectively modeling the uncertainties and preferences tied in with the stated problem. Nowadays this probabilistic graphical model is frequently adopted as a basis for constructing decision-support systems (DSSs). The results of evaluating an influence diagram are *decision tables* containing the optimal decision alternatives, policies or responses. Thus, for every decision, there is an associated decision table with the best alternative, i.e. the alternative with the maximum expected utility for every combination of relevant variables (usually called *attributes* within this context) that are observable before the decision is made. The evaluation algorithm determines which of the observable variables are relevant. These variables are outcomes of random variables and/or other past decisions.

A decision table may have millions of rows and typically more than twenty columns leading to enormous data sets for storage and analysis. Expert DSS users demand such an analysis on mainly two grounds. First, DSS decision tables provide the best decision-making recommendations. However, experts may find such recommendations hard to accept if they come without any *explanation* whatsoever of why the proposed decisions are optimal. Unexplained responses are not good enough for expert users since DSSs operate on a model that is an approximation of the real world. The importance of explanations has been reported in the literature, see e.g. [9,12,13]. Thus, for example, in health-care problems, usually involving difficult trade-offs between the treatment benefits and risks, practitioners may use decision tables to determine the best patient treatment recommendations. For this purpose, they need to understand the underlying reasons or implicit rules.

In medical DSSs, clinical practice guidelines assemble the relevant knowledge gathered through literature review, meta-analysis, expert consensus, etc., and operationalize this information as informal, text documents. This makes the gathered

* Corresponding author.
E-mail addresses: jafernandez@fi.upm.es (J.A. Fernández del Pozo), mcbielza@fi.upm.es (C. Bielza).

0169-023X/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.datak.2010.10.006

information difficult to interpret automatically and the decision-making process hard to guide. Shiffman and Greenes [19] propose translating guideline knowledge into decision table-based rule sets. Shiffman [18] proposes augmenting decision tables by layers, storing collateral information in slots at various levels beneath the logic layer of the conventional decision table. Information relates to table cells, rows and columns. It may include how tests are performed, the benefits/risks of the recommended strategies, costs, literature citations, etc., to help understand the domain. All these decision tables are different than ours. Our knowledge base is the model (influence diagram) and its evaluation, stored in the decision tables. The model (graph with probabilistic dependencies and probability and utility information) is built from clinical practice guidelines, data and expert input. Also, there is no uncertainty in clinical guidelines. Influence diagrams are based on subjective probabilities and utilities, and support learning and reasoning with uncertainty and preferences.

In [6] we introduced *KBM2L* lists to find explanations. The main idea stems from how computers manage multidimensional matrices: computer memory stores and manages these matrices as linear arrays, and each position is a function of the order chosen for the matrix dimensions. KBM2L lists are new list-based structures that optimize this order by putting equal responses in consecutive positions, yielding the target explanations and simultaneously achieving compact storage. These lists implicitly include the probability and utility models, they are simple, and have no added complex layers.

Not only do expert users employ decision tables as a knowledge base (KB) for explanations; they also *query* the DSS about which is the best recommendation for a given set of attributes in different ways. This is the second reason for decision table analysis. In a typical session, experts interact with DSSs to:

(A) formulate a query in the KB domain;
(B) translate the query into the KB formalism;
(C) implement the response retrieval;
(D) build the response efficiently;
(E) communicate the response(s) and/or suggest improvements, and wait for user feedback.

For (A) and (B), we distinguish between two groups of queries (closed/open) depending on whether or not the whole set of attributes is instantiated. A closed query is a specific and well-defined query entered by users that know all the attribute information. An open query is less specific, as it includes attribute values that are undefined either because they are hard or expensive to obtain or they are unreliable. Martinez et al. [15] give a similar classification for GIS (geographical information systems), although they focus on data efficient updating and access from a physical point of view (merely as a database), rather than from a logical point of view (as a KB).

(C) to (E) may be troublesome, especially for open queries, due to imprecise response retrieval failing to satisfy users. Additionally, the DSS may not include the whole decision table, because an exhaustive evaluation of the decision-making problem can be too costly. In this case there will be no response at all. Worse still, both situations could apply at the same time, demanding a methodology to undertake tasks (C)–(E) dealing with ambiguity and ignorance about the response.

1.2. Example: Optimal treatment of gastric non-Hodgkin lymphoma

Let us illustrate these ideas with the following clinical problem. It is a real health-care decision-making problem regarding the optimal treatment of non-Hodgkin lymphoma of the stomach.

Primary gastric non-Hodgkin lymphoma, *gastric NHL* for short, is a relatively rare disorder, accounting for about 5% of gastric tumors. This disorder is caused by a chronic infection by the *Helicobacter pylori* bacterium [5]. Treatment consists of a combination of antibiotics, chemotherapy, radiotherapy and surgery.

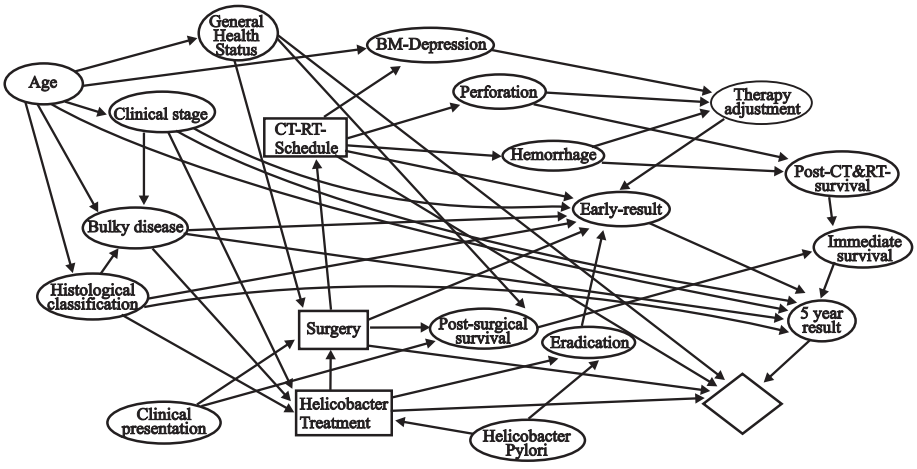

Fig. 1. Influence diagram for the treatment of gastric NHL.

A number of influence diagrams have been constructed and validated [14]. These models are only meant to be used for patients with histologically confirmed gastric NHL. We have taken the most complex version with three decision nodes. This influence diagram is shown in Fig. 1, and is briefly discussed in the following. The first of the decision nodes, HELICOBACTER-TREATMENT (HT), corresponds to the decision to prescribe antibiotics against *H. pylori*. The second decision concerns carrying out SURGERY (S). The possibilities are either curative surgery, involving the complete removal of the stomach and locoregional tumor mass; palliative surgery, i.e. partial removal of the stomach and tumor; or no surgery. The last decision, CT-RT-SCHEDULE (CTRTS), is concerned with the selection of chemotherapy (Chemo), radiotherapy (Radio), chemotherapy followed by radiotherapy (Ch.Next.Rad), or none.

The influence diagram model consists of 17 chance nodes (ellipses), one value node (diamond), three decision nodes (rectangles) and 42 arcs. Nodes to the left of the decision nodes (see Fig. 1) concern pretreatment information. Nodes to the right of the decision nodes are posttreatment nodes. Variables with their associated domains are listed in Table 1. See [14] for further details on the model. Bielza et al. [1] detail the use of KBM2L lists to gain a better understanding of the treatment basis of the gastric NHL model.

The gastric NHL influence diagram evaluation outputs three decision tables, one for each decision variable, each containing the optimal treatment for each combination of attributes in the tables.

Let us take the first decision table concerning the HT decision. It contains four attributes (CS, BD, HC, and HP), and the expected utility of each treatment alternative HT = No/Yes. To illustrate likely user queries, suppose a user queries the DSS about patients with the following configurations:

$\mathbf{Q}_0$: HC = Low.Grade, HP = Present, CS = I and BD = Yes

$\mathbf{OQ}_1$: HP = Absent, CS = I and BD = Yes

$\mathbf{OQ}_2$: CS = II2.

We will look at all the discussed queries in this paper. In the first case, $\mathbf{Q}_0$, the query is closed since the four attributes are instantiated. The question is about a patient that has a good histological classification (HC = Low.Grade), a favorable prognosis (CS = I), the *H. pylori* bacterium (HP = Present), and a big tumor (BD = Yes). Unless this query corresponds precisely to an unsolved part of the problem, the response should be easy to retrieve.

In the second case, $\mathbf{OQ}_1$, the query is open because the doctor has not yet performed a biopsy to ascertain the HISTOLOGICAL-CLASSIFICATION (HC). This could perhaps be due to the high cost of the biopsy.

In the third case, $\mathbf{OQ}_2$, the query is even more open, specifying only a medium clinical stage (CS = II2) for the patient. However, the user may be interested in finding out which treatment patients like these should receive. Responses are not expected to be easy to retrieve now. There are many possible alternatives, where users will find it unsatisfactory if different and perhaps unknown responses are retrieved. Therefore, strategies should be developed to assure user satisfaction. One possibility is table reordering to provide more precise answers. Another is sophisticated prediction procedures to infer the unknown responses from (somehow) *close* known responses or by having the user intervene at some steps to reduce response uncertainty.

Table 1
Gastric NHL variables with their possible values.

Variable	Possible values
HELICOBACTER-TREATMENT (HT)	No, Yes
SURGERY (S)	None, Curative, Palliative
CT-RT-SCHEDULE (CTRTS)	None, Radio, Chemo, Ch.Next.Rad
GENERAL-HEALTH-STATUS (GHS)	Poor, Average, Good
CLINICAL-STAGE (CS)	I, II1, II2, III, IV
BULKY-DISEASE (BD)	Yes, No
HISTOLOGICAL-CLASSIFICATION (HC)	Low.Grade, High.Grade
HELICOBACTER-PYLORI (HP)	Absent, Present
CLINICAL-PRESENTATION (CP)	None, Hemorrhage, Perforation, Obstruction
AGE	v10.19, v20.29, v30.39, v40.44, v45.49, v50.54 v55.59, v60.64, v65.69, v70.79, v80.89, GE90
ERADICATION	No, Yes
BM.DEPRESSION (BONE MARROW)	No, Yes
PERFORATION	No, Yes
HEMORRHAGE	No, Yes
THERAPY.ADJUSTMENT	No, Yes
POST.CT-RT.SURVIVAL	No, Yes
POST.SURGICAL.SURVIVAL	No, Yes
IMMEDIATE.SURVIVAL	No, Yes
EARLY.RESULT	CR (complete remission), PR (partial remission), NC (no change), PD (progressive disease)
5.YEAR.RESULT	ALIVE, DEAD

1.3. Outline

In this paper, we propose a query system based on the KBM2L framework to deal with these complex situations. Unlike database management systems that operate with facts, DSSs must provide explanations besides efficiently retrieving the query response information [10]. Thus, our KBM2L framework provides not only an efficient and satisfactory query response retrieval but also an informed response explanation. It is not our aim to develop clinical practice guidelines, but to provide a DSS with a user interface capable of performing complex queries involving more than just accessing a clinical protocol database or document.

The paper is organised as follows. Section 2 outlines the technique of KBM2L lists. Section 3 describes the query complexity and shows how to deal with a closed query. Section 4 tackles less specific and more complex open queries. The proposal combines decision tables that have been compacted using KBM2L lists with learning, information access and information retrieval processes. We give several examples applied to the non-Hodgkin lymphoma problem. Section 5 contains the conclusions and suggests further research.

2. KBM2L lists

2.1. Basics

A decision table output by evaluating an influence diagram is a set of attributes that determines the optimal policy. Besides all the attribute configurations, a decision table includes the response or optimal alternative associated with each configuration. A *base* is defined as a vector with elements equal to the attributes in a specific order. Given a base, an *index* is a vector whose elements are the attribute values, interpreted as the coordinates with respect to that base. With a fixed order of attributes with discrete domains, a decision table can be viewed as a multidimensional matrix.

We can map this multidimensional matrix to a linear array or list like sequential memory allocation in computers [11]. Given a cell of the table of $n+1$ attributes with index $\mathbf{c} = (c_0, c_1, \ldots, c_n)$, we define the *access function* $f: \mathcal{R}^{n+1} \to \mathcal{R}$, such that

$$f(c_0, c_1, \ldots, c_n) = c_0 \prod_{i=1}^{n} D_i + c_1 \prod_{i=2}^{n} D_i + \cdots + c_n = q, \tag{1}$$

where q is the **c**-offset with respect to the first element of the table in a given base, and D_i denotes the cardinality of the ith attribute domain for $i = 0, 1, \ldots, n$. The access function f can also be written more compactly as

$$f(c_0, c_1, \ldots, c_n) = \sum_{i=0}^{n} c_i w_i, \tag{2}$$

where $w_i = \prod_{j=i+1}^{n} D_j = w_{i+1} D_{i+1}$ is called the *weight* of the ith attribute, $i = 0, 1, \ldots, n$ and $w_n = 1$. The vector of weights is $\mathbf{w} = (w_0, w_1, \ldots w_n)$. Thus, index notation and offset notation are equivalent, and are related to each other by function f, defined by Eq. (1) or (2). Without loss of generality, suppose the possible outcomes of c_i are 0,1,2,…$D_i - 1$ and, hence, the possible values for q are $0, 1, 2, \ldots, w_0 D_0 - 1$.

With a view to shortening the list output by the decision table, we find that some sets of *consecutive* cells often lead to the *same* optimal alternative or response. The number of such consecutive cells represents the knowledge granularity of optimal decisions. As a result, a new compact list can be constructed. This list will only store one index (or offset) per set of equal alternatives. We will choose the last index (offset), as the representative of this set. This last index (offset), together with the shared optimal alternative that represents a set of cases, is called an *item*. The resulting list of items is called a *KBM2L* list. KBM2L stands for a "Knowledge Base Matrix to List" representation [6].

An item is denoted by ⟨*index*, *alternative*| or, equivalently, ⟨*offset*, *alternative*|, where ' ' reflects that the item offsets increase monotonously, and '|' reflects granularity.

2.2. Response explanations

Consider a set of indices representing an item. Since the indices are ordered, this set will range from an index I_{inf} to I_{sup}, corresponding to the endpoints (minimum and maximum) of the item. This set of indices has a *fixed* part, representing the index components common to all the item cases (with the same values), and a *variable* part, where the values of the attributes corresponding to the indices are not shared. Both parts can be derived from the indices I_{inf} and I_{sup}, e.g., the fixed part is obtained by taking the logical AND $I_{\text{inf}} \wedge I_{\text{sup}}$.

These concepts pave the way for automatically generating explanations for decisions. The fact that the attribute values in the fixed part of items are equal somehow *explains* why the optimal alternative is also the same across items. Hence, the set of attributes of the fixed part can be interpreted as explaining why the alternative is optimal. The attributes in the variable part of an item are irrelevant for decision making.

2.3. Implementation and optimization

In [6] we implemented the process of building a KBM2L list from a decision table. In actual fact, different bases may contain the same table knowledge but the list of items may vary from one base to another. Good KBM2L lists search for bases with few items, grouping identical responses as far as possible into consecutive offsets. A base that minimises the number of items will bring up the granular knowledge, and incidentally have minimum memory requirements. Searching for solutions in the possible attribute permutation space with or without (even more complex) a fixed domain order is known to be an NP-hard combinatorial optimization problem [7]. We have implemented a genetic and a variable neighborhood algorithm to guide this search. An example with the NHL problem is given in the next section.

Sometimes the decision table is too large to be fully evaluated. In this case, a not necessarily exhaustive set of subproblems is solved instead. Each subproblem is the result of instantiating some random variables. This implies that there will be unknown optimal alternatives for some attribute combinations, i.e. for combinations associated with unsolved subproblems. This is not a problem for the KBM2L construction process which also operates with unknown responses. First, it evaluates all the subproblems sequentially or concurrently. Then, the resulting partial decision tables are sequentially added to the KBM2L list by means of a learning mechanism that optimizes the list before processing the next partial table. Each stage of this additive process improves the item synthesis and facilitates future additions [6].

Example. KBM2L optimization.

Let us briefly explain the KBM2L list optimization process for the gastric NHL problem. Let us use the first decision table concerning the HT decision (*Helicobacter* treatment) with 40 cases resulting from the influence diagram evaluation. This table has four attributes (CS, BD, HC, and HP) determining the expected utility of each alternative HT = No/Yes, see Table 2 (left-hand side). Each row represents a case from the combinatorial space induced by the Cartesian product of the attribute domains. For each attribute configuration, we get the optimal alternative of the HT decision, i.e. the alternative with the maximum expected utility. For example, 407.89 (in the last row) is the expected utility of alternative *No* for the HT decision when CS is *IV*, BD is *No*, HC is *High.Grade* and HP is *Present*. Since the expected utility is 397.89 for the alternative *Yes*, *No* is the optimal decision alternative in this case.

The first step to synthesize knowledge from this decision table is to build a KBM2L list. We use the attributes in the same order, i.e. base [CS, BD, HC, HP], see Table 2 (right-hand side). The KBM2L list has 17 items. Each item represents its last case and the optimal alternative (0 = No, 1 = Yes). Remember that we transform the table by grouping together equal adjacent responses as items.

Next, we undertake a two-step KBM2L optimization. First, we optimize the attribute permutation space, which yields a new base [HP, HC, CS, BD] and five items, see Table 3 (top). Second, we optimize a larger space, where both the attributes and their domains are permutable. This means, for example, that during table row ordering, an attribute A with a binary domain {0,1} can induce rows with A ordered as either [0,1] (denoted $A_{[0,1]}$) or [1,0] (denoted $A_{[1,0]}$). Note that we must declare the relative permutation on attribute domains as the subindex of the corresponding attribute. In our example, the domain permutation is allowed to have only two items, which is really optimal. Then the optimal base is $B^* = [HC_{[0,1]}, HP_{[1,0]}, CS_{[4,1,2,0,3]}, BD_{[0,1]}]$, see Table 3 (bottom).

The initial base assumes the natural ordering of both attributes and domains. Therefore, the three encoded bases are:

# Items	Base
17	$[CS_{[0,1,2,3,4]}, BD_{[0,1]}, HC_{[0,1]}, HP_{[0,1]}]$
5	$[HP_{[0,1]}, HC_{[0,1]}, CS_{[0,1,2,3,4]}, BD_{[0,1]}]$
2	$B^* = [HC_{[0,1]}, HP_{[1,0]}, CS_{[4,1,2,0,3]}, BD_{[0,1]}]$

Finally, let us look at the explanation of the optimal policies for the HT decision. Alternative HT = No (first item) has no explanation, whereas the explanation for alternative HT = Yes (second item) is as follows. The DSS recommends the prescription of antibiotics as the *Helicobacter* treatment if HISTOLOGICAL-CLASSIFICATION (HC) is Low.Grade, HELICOBACTER-PYLORI (HP) is Present and CLINICAL-STAGE (CS) is not III. This corresponds to the fixed part of the set of indices in the second item.

The expert may or may not agree with the DSS, but the KBM2L provides some reasons for this recommendation. Also, note that the KBM2L refines the knowledge because an irrelevant attribute has been identified (BULKY-DISEASE (BD)), and an attribute domain has been constrained (CLINICAL-STAGE (CS) is not III, i.e. it is I, II1, II2, or IV).

This example illustrates how the decision table and even the initial KBM2L list are not efficient enough to show the structure of knowledge represented by the DSS after evaluating the influence diagram. However, the optimal list sorts the attributes and domains in order to state their relevance providing explanations. □

3. Queries and KBM2L lists

3.1. Complexity of queries

Queries are stated as attribute instantiations. Therefore, they are related to the KBM2L index and employ multidimensional point access methods [21]. The DSS is expected to return a response stating the optimal policy using a small subset of the KB. However, an added difficulty is that the optimal policy may be unknown.

Table 2
Optimal decision table transformed into a KBM2L list for the Helicobacter treatment (HT) decision variable. Attribute labels: for CS: 0 = I, 1 = II1, 2 = II2, 3 = III, 4 = IV; for BD: 0 = Yes, 1 = No; for HC: 0 = Low.Grade, 1 = High.Grade; for HP: 0 = Absent, 1 = Present; for HT: 0 = No, 1 = Yes.

Case	Attributes				Exp.Utility for HT = No and Yes	OptDec for HT	Item	Base				HT
	CS	BD	HC	HP				(CS	BD	HC	HP)	
1	0	0	0	0	819.49>809.49	No	1	0	0	0	0	0
2	0	0	0	1	789.02<809.49	Yes	2	0	0	0	1	1
3	0	0	1	0	573.96>563.96	No	–	–	–	–	–	–
4	0	0	1	1	573.96>563.96	No	–	–	–	–	–	–
5	0	1	0	0	887.71>877.71	No	3	0	1	0	0	0
6	0	1	0	1	866.88<877.71	Yes	4	0	1	0	1	1
7	0	1	1	0	729.80>719.80	No	–	–	–	–	–	–
8	0	1	1	1	729.80>719.80	No	–	–	–	–	–	–
9	1	0	0	0	714.05>704.05	No	5	1	0	0	0	0
10	1	0	0	1	685.66<704.05	Yes	6	1	0	0	1	1
11	1	0	1	0	505.82>495.82	No	–	–	–	–	–	–
12	1	0	1	1	505.82>495.82	No	–	–	–	–	–	–
13	1	1	0	0	836.01>826.01	No	7	1	1	0	0	0
14	1	1	0	1	797.42<826.01	Yes	8	1	1	0	1	1
15	1	1	1	0	616.04>606.04	No	–	–	–	–	–	–
16	1	1	1	1	616.04>606.04	No	–	–	–	–	–	–
17	2	0	0	0	419.72>409.72	No	9	2	0	0	0	0
18	2	0	0	1	395.05<409.72	Yes	10	2	0	0	1	1
19	2	0	1	0	323.26>313.26	No	–	–	–	–	–	–
20	2	0	1	1	323.26>313.26	No	–	–	–	–	–	–
21	2	1	0	0	551.89>541.89	No	11	2	1	0	0	0
22	2	1	0	1	537.61<541.89	Yes	12	2	1	0	1	1
23	2	1	1	0	460.88<450.88	No	–	–	–	–	–	–
24	2	1	1	1	460.88>450.88	No	–	–	–	–	–	–
25	3	0	0	0	257.90>247.90	No	–	–	–	–	–	–
26	3	0	0	1	252.82>247.90	No	–	–	–	–	–	–
27	3	0	1	0	248.20>238.20	No	–	–	–	–	–	–
28	3	0	1	1	248.19>238.20	No	–	–	–	–	–	–
29	3	1	0	0	324.55>314.55	No	–	–	–	–	–	–
30	3	1	0	1	343.46>314.55	No	–	–	–	–	–	–
31	3	1	1	0	378.92>368.92	No	–	–	–	–	–	–
32	3	1	1	1	378.92>368.92	No	–	–	–	–	–	–
33	4	0	0	0	307.88>297.88	No	13	4	0	0	0	0
34	4	0	0	1	269.28<297.88	Yes	14	4	0	0	1	1
35	4	0	1	0	265.14>255.13	No	–	–	–	–	–	–
36	4	0	1	1	265.13>255.13	No	–	–	–	–	–	–
37	4	1	0	0	408.35>398.35	No	15	4	1	0	0	0
38	4	1	0	1	376.15<398.35	Yes	16	4	1	0	1	1
39	4	1	1	0	407.89>397.89	No	–	–	–	–	–	–
40	4	1	1	1	407.89>397.89	No	17	4	1	1	1	0

Let us explain this point in further detail. As mentioned earlier, the exhaustive evaluation of the decision-making problem may be too costly in terms of time and memory requirements. The future decision-making policy domain grows exponentially with time and very quickly becomes intractable. We cannot compute the first decisions because we do not have tractable

Table 3
Two-step KBM2L optimization: i) in the attribute permutation space and, ii), in the attribute and domain permutation space. Attribute labels: for CS: 0 = I, 1 = II1, 2 = II2, 3 = III, 4 = IV; for BD: 0 = Yes, 1 = No; for HC: 0 = Low.Grade, 1 = High.Grade; for HP: 0 = Absent, 1 = Present; for HT: 0 = No, 1 = Yes.

First step	Item	Size	Base				HT
			(HP	HC	CS	BD)	
	1	20	0	0	4	1	0
	2	6	0	1	2	1	1
	3	2	0	1	3	1	0
	4	2	0	1	4	1	1
	5	10	1	1	4	1	0
Second step	Item	Size	Base B^*				HT
			($HC_{[0,1]}$	$HP_{[1,0]}$	$CS_{[4,1,2,0,3]}$	$BD_{[0,1]}$)	
	1	8	0	0	3	1	0
	2	32	1	1	4	1	1

representations of policies for future decisions. This is called the curse of time horizon [20]. In a complex DSS for neonatal jaundice management with five decision nodes, the size of the decision table associated with the last decision was 119.4 MB [8]. Some possible ways of getting tractable policy domain sizes are to try to identify variables that are irrelevant for future decisions or to consider variables as irrelevant by assuming that their states will not be remembered.

Another alternative is to resort to solving a set of subproblems instead of the global problem, where each subproblem is the result of instantiating some random variables. In the context of influence diagrams, Cortazar et al. [4] explain how to do this. This subproblem set may not be exhaustive, implying unknown optimal policies for some attribute combinations, i.e. policies associated with unsolved subproblems. This is common in large decision-making problems. The process of evaluating the large diagram is therefore divided by instantiating some attributes and solving the reduced influence diagrams. At the end, the partial results are collected and composed incrementally in a KBM2L structure.

In Section 2.3 we already mentioned that the KBM2L construction process also operates with unknown policies and subproblems. Therefore, we distinguish not only between closed and open queries, but also between known and unknown or uncertain responses. The different scenarios are analysed in the following sections. They apply to tasks (C)–(E) mentioned in the Introduction.

3.2. Closed queries

As far as closed queries are concerned, since the whole set of attributes is instantiated, the DSS only needs to look for the corresponding KBM2L list element and retrieve the optimal policy, which is presented to the user as the response. The mechanism is formalized as follows.

Let $A_0,...,A_n$ be the attributes and $\Omega_0,...,\Omega_n$ their respective domains. Ω_R denotes the response domain. Let $\mathbf{Q}$ denote a closed query, i.e. $\mathbf{Q} = (a_0, ..., a_n), a_i \in \Omega_i, i = 0, ..., n$. Suppose the optimal KBM2L list has been achieved with respect to base $B = [A_0,...,A_n]$, and $\mathbf{w}$ is its respective weight vector. If this list has h items, then the list is $\langle q_0, r_0 | \langle q_1, r_1 | \cdots \langle q_{h-1}, r_{h-1} |$, where $0 \le q_i \le w_0 D_0 - 1$, $q_i < q_{i+1} \forall i$ (offsets), $r_i \in \Omega_R, r_i \neq r_{i+1} \forall i$ (policies). The response retrieval procedure (task (C) above) consists of projecting $\mathbf{Q}$ into the offset space $\{0,1,...,w_0 D_0 - 1\}$ and deriving the optimal alternative from the KBM2L list. Namely, if $\langle \cdot, \cdot \rangle$ denotes the scalar product, we compute $\langle \mathbf{Q}, \mathbf{w} \rangle = f(\mathbf{Q}) = q$, and whenever $q \in (q_{i-1}, q_i]$, then the response is r_i. This response is efficiently retrieved with $O(nlog(h))$ complexity using a binary search procedure on the KBM2L.

If r_i is unknown, then an efficient solution is to call the influence diagram evaluation and solve the respective subproblem that makes r_i known. Finally, response r_i displayed by the DSS to the expert may be further completed by asking for an explanation.

Example. Closed query.

To illustrate the formulation of a closed query and the explanation of the respective response, suppose the expert queries the DSS about a patient with the following configuration:

$\mathbf{Q}_0$:HC = Low.Grade, HP = Present, CS = I and BD = Yes.

The KBM2L list using the optimal base B^* is given by

$$\langle (\text{HC} = 0, \text{HP} = 0, \text{CS} = 3, \text{BD} = 1), 1 | \langle (\text{HC} = 1, \text{HP} = 1, \text{CS} = 4, \text{BD} = 1), 0 |,$$

expressed in index notation or $\langle (7,1) |$ $\langle (39,0) |$ expressed in offset notation. Then, $\mathbf{Q}_0 = (0,0,1,0)$ and $\mathbf{w}_0 = (10,5,2,1)$, and the response is derived from $\langle \mathbf{Q}_0, \mathbf{w}_0 \rangle = \langle (0,0,1,0), (10,5,2,1) \rangle = 2$. Finally, offset 2 belongs to the first item, between offsets 0 and 7, and then the DSS suggests HT = Yes. The explanation is obviously the one given earlier: because HISTOLOGICAL-CLASSIFICATION (HC) is *Low.Grade*, HELICOBACTER-PYLORI (HP) is *Present* and CLINICAL-STAGE (CS) is not *III*, the DSS recommends the prescription of antibiotics as the *Helicobacter* treatment. □

Therefore, we have explained how to perform tasks (C)–(E) mentioned in the Introduction, where there is no need for improvements or feedback in (E).

4. Open queries

We have seen that the expert is an agent that queries the DSS about the optimal policy for the decision-making problem. Expert and DSS enter into a dialogue consisting of queries, responses and explanations. For closed queries, the expert receives definite and accurate responses. Responses to open queries are not so straightforward due to expert imprecision. Not all attributes are instantiated. Possible reasons are the unreliability of some attribute values, missing knowledge, high retrieval cost or simply an interest in making a complex query concerning the whole range of some attributes. In medical settings, for example, physicians often have access to administrative data, like sex, age, etc., but may have no access to (or no confidence in) attributes like the first treatment received or some test results. Also, they may be interested in asking for all possible patient weight intervals. Thus, an open query would be $\mathbf{OQ} = (*, a_1, *, ..., a_n)$, where $*$ denotes non-instantiated attribute values.

In principle, the DSS looks up the respective KBM2L list elements and retrieves the optimal policy (or policies) to be presented to the user as the response. Suppose that the optimal KBM2L list has been achieved with respect to base $B = [A_0,...,A_n]$, and the

query is open with respect to attributes i and j, i.e. the query is $\mathbf{OQ} = (a_0, a_1, ..., *_i, ..., *_j, ..., a_n)$. Actually, $\mathbf{OQ}$ is a set of closed queries $\mathbf{Q}_i$, namely,

$$\mathbf{OQ} = \{(a_0, a_1, ..., x, ..., y, ..., a_n) : x \in \Omega_i, y \in \Omega_j\} = \bigcup_{i=1}^{D_i \times D_j} \mathbf{Q}_i.$$

Then the response retrieval procedure would consist of applying the technique described earlier to each $\mathbf{Q}_i$, computing $\langle \mathbf{Q}_i, \mathbf{w} \rangle = f(\mathbf{Q}_i) = p_i$, using Eq. (2), to give an offset set $P = \{p_1, p_2, ..., p_{D_i \times Dj}\}$ with the respective responses $S = \{s_1, s_2, ..., s_{D_i \times Dj}\}$.

Example. Open query.

Suppose again that we have the HT decision table and base B^* as in the previous example. Let us take the following open query $\mathbf{OQ}_1 = (*, \text{Absent}, \text{I}, \text{Yes})$, i.e. the expert is interested in finding out the optimal decision when

$\mathbf{OQ}_1$: HP = Absent, BD = Yes and CS = I.

This patient has a favorable prognosis: CLINICAL-STAGE (CS) = I; does not have the *H. pylori* bacterium: HELICOBACTER-PYLORI (HP) = Absent; but does have a big tumor: BULKY-DISEASE (BD) = Yes. This is an open query because the doctor has not yet performed a biopsy to confirm the HISTOLOGICAL-CLASSIFICATION (HC). Note that, according to its weight in the optimal base B^*, the unknown attribute HC is the most relevant. This open query is equivalent to the set of closed queries $\mathbf{Q}_1, \mathbf{Q}_2$, see Table 4.

Here $\mathbf{w}$ is (20, 10, 2, 1). Both closed queries are cases from the second item (see Table 3) located at offsets 10 (for $\mathbf{Q}_1$) and 30 (for $\mathbf{Q}_2$), and the response is unambiguously $S = \{\text{HT} = \text{No}\}$. □

To implement a general procedure we will not enumerate all closed queries. An efficient mechanism for performing open queries will be based on the concept of *operative bases* and the structure of the KBM2L list, see the following discussion.

There are four possible situations depending on the type of elements in S:

(i) all s_i are equal and known;
(ii) all s_i are unknown;
(iii) there are at least two different values among s_i and they are known;
(iv) there are at least two different values among s_i but some s_i are unknown.

Each situation is analysed in the following.

4.1. The KBM2L knows the response. Operative bases

Situation (i) implies an accurate response (s_i), and the DSS requires no further interaction with the expert. All open attributes are irrelevant. To be helpful for the expert the other situations involve several possible responses and/or uncertain responses requiring refinement. It is here where tasks (D) and (E) listed in the Introduction play an important role.

The information for the expert comprises two sets P and S, jointly involving simple KB records. They have been extracted from the optimal base. Note that this base is the best base for the *whole KB*, both minimising storage requirements and maximising knowledge explanation performance. But this base may not be so useful with respect to the *part* of the KB concerning the open query.

Attribute weight changes depending on its position within a base, and the further to the right the position, the smaller the weight is. We propose moving open query attributes towards positions further to the right. The query is unchanged, but its attribute order implies the use of another base, where open attributes are moved towards the positions with the smallest weights. Semantically speaking, this shift also agrees with the idea of consigning open attributes to less important positions as the query appears to indicate, since the expert has not assigned a value to and does not show any significant interest in these attributes. The new base will be called *operative* base. This base yields an operative KBM2L list to implement the open query. There are several possible operative bases, all of which are valid. After testing a number of these bases in linear time, we can choose the one requiring the least computational effort for the change of base or knowledge transposition.

A base change may be interpreted as a change in the query and response points of view. For the DSS, the optimal base represents a good organisation of the whole KB content. For experts, the operative base provides an organised view of the responses to their open query, as *consecutive records*. This base will be optimal for explaining the responses. This illustrates how a

Table 4
Set of closed queries for the open query $\mathbf{OQ}_1$: HP = Absent, BD = Yes and CS = I.

Closed queries		Offset comput.
$\mathbf{Q}_1$	HELICOBACTER-PYLORI (HP) = Absent, BULKY-DISEASE (BD) = Yes, CLINICAL-STAGE (CS) = I and HISTOLOGICAL-CLASSIFICATION (HC) = Low.Grade	$\langle (0,1,0,0), (20,10,2,1) \rangle = 10 = p_1$
$\mathbf{Q}_2$	HELICOBACTER-PYLORI (HP) = Absent, BULKY-DISEASE (BD) = Yes, CLINICAL-STAGE (CS) = I and HISTOLOGICAL-CLASSIFICATION (HC) = High.Grade	$\langle (1,1,0,0), (20,10,2,1) \rangle = 30 = p_2$

query may be difficult in one base and easier in another base. It bears a resemblance to human domains, where a query is simple for an expert but hard for non-experts. Experts have a good problem description: a good conceptual model, few relevant facts, and a good information organisation meaning that they can analyse and explain facts. Indeed, this is why they are experts.

Now, working on the operative base, the new offset set P' will include consecutive p_is and we can introduce some distance-based rules in the offset space to make more accurate recommendations to the expert.

Example. Open query (continued).

Now we try to illustrate the use of the operative base. The optimal base in the previous example, where we performed the open query $\mathbf{OQ}_1$, was

$$B^* = \left[\mathrm{HC}_{[0,1]}, \mathrm{HP}_{[1,0]}, \mathrm{CS}_{[4,1,2,0,3]}, \mathrm{BD}_{[0,1]}\right].$$

This simple query has one non-instantiated attribute, i.e. HC. To obtain the query response, we need to move this attribute to the right-most position in the base (low weight). One (operative) base meeting that condition is

$$B^1_{op} = \left[\mathrm{HP}_{[1,0]}, \mathrm{CS}_{[4,1,2,0,3]}, \mathrm{BD}_{[0,1]}, \mathrm{HC}_{[0,1]}\right],$$

with the same ordering of domains as B^*. Tables 5 and 6 show respectively the KBM2L list expressed in optimal and operative bases. The list in the operative base is 16 items long. This list manages to show all the target cases in a sequence such that the offset ranges consecutively from 36 to 37. The response is always HT = No. The open attribute HC is irrelevant in the context of that query. Remember from the previous example that when using the optimal base B^*, the offsets of the target cases were not consecutive. They were 10 and 30 (see Table 4). □

4.2. The KBM2L does not know the response at all. Predicting the response in a neighborhood

Situation (ii) may seem surprising since the DSS is being queried about something that is unknown. This is because the associated subproblems have not been evaluated. Nevertheless, we will try to provide a solution. Specifically, situation (ii) may be solved by predicting that the response is associated with the nearest offset to P' based on the Euclidean distance in the offset space, i.e. in $\{0,1,...,w_0D_0-1\}$, as follows. Let p^l, p^u be respectively the minimum and maximum offsets included in P'. Suppose q^l is the maximum offset with known policy (say r) that precedes p^l in the operative KBM2L list. Likewise, q^u is the minimum offset with known policy (say s) that follows p^u in the operative KBM2L list. All records that match (p',s), with $p' \in P', s \in S$, belong to the same item in the operative KBM2L list, whereas offsets q^l and q^u are located in adjoining items. Then, we compute $d_1 = |p^l - q^l|$ and $d_2 = |p^u - q^u|$. If $d_1 < d_2$, then the response is r; otherwise the response is s.

This result is only the first step in the inference process run on the KBM2L list and is therefore an approximation. The inference process run on an incomplete KBM2L list would be performed taking into account a set of bases from a small neighborhood of the operative base to confirm the result. Therefore, a second step conducts the above distance analysis using some bases close to the operative base.

Example. Open query and unknown response.

To illustrate this scenario, suppose that we have run a partial evaluation of the decision model. For this reason, we do not know the optimal decision for all cases such that HP = Absent and BD = Yes in the HT table. The open query is $\mathbf{OQ}_1$, as in the previous examples

$$\mathrm{HP} = \text{Absent},\ \mathrm{BD} = \text{Yes and } \mathrm{CS} = \text{I}.$$

Due to the unknown responses (denoted as -1), the KBM2L list is now different. Table 7 shows this list when using the same operative base B^1_{op} as in the last example (Table 6). The open query matches item 25. We have $p^l = 36$, $p^u = 37$, $q^l = 35$, $r =$ No, $q^u = 38$ and $s =$ No. Therefore, we have $d_1 = |p^l - q^l| = 1$ and $d_2 = |p^u - q^u| = 1$, and the response is HT = No.

In a further step, we conduct the same distance analysis using some bases close to the operative base in the sense defined in [6]. Suppose we take the base $B^2_{op} = [\mathrm{CS}_{[4,1,2,0,3]}, \mathrm{HP}_{[1,0]}, \mathrm{BD}_{[0,1]}, \mathrm{HC}_{[0,1]}]$. Note that only the heaviest two attributes CS and HP are swapped in this

Table 5
KBM2L list using the optimal base B^*.

Item	Description in optimal base B^*					Offset
	HP	HC	CS	BD	HT	
1	⟨ (Present,	Low.Grade,	I,	No),	Yes |	7
2	⟨ (Absent,	High.Grade,	III,	No),	No |	39

Table 6
KBM2L list using an operative base B^1_{op}.

Item	Description in operative base B^1_{op}					Offset
	HP	CS	BD	HC	HT	
1	⟨ (Present,	IV,	Yes,	Low.Grade),	Yes |	0
2	⟨ (Present,	IV,	Yes,	High.Grade),	No |	1
3	⟨ (Present,	IV,	No,	Low.Grade),	Yes |	2
4	⟨ (Present,	IV,	No,	High.Grade),	No |	3
5	⟨ (Present,	II1,	Yes,	Low.Grade),	Yes |	4
6	⟨ (Present,	II1,	Yes,	High.Grade),	No |	5
7	⟨ (Present,	II1,	No,	Low.Grade),	Yes |	6
8	⟨ (Present,	II1,	No,	High.Grade),	No |	7
9	⟨ (Present,	II2,	Yes,	Low.Grade),	Yes |	8
10	⟨ (Present,	II2,	Yes,	High.Grade),	No |	9
11	⟨ (Present,	II2,	No,	Low.Grade),	Yes |	10
12	⟨ (Present,	II2,	No,	High.Grade),	No |	11
13	⟨ (Present,	I,	Yes,	Low.Grade),	Yes |	12
14	⟨ (Present,	I,	Yes,	High.Grade),	No |	13
15	⟨ (Present,	I,	No,	Low.Grade),	Yes |	14
16	⟨ (Absent,	III,	No,	High.Grade),	No |	39

new base. The open query determines $p^l = 28$ and $p^u = 29$ in item 22, see Table 8. Now $q^l = 27$, $r = \text{No}$, $q^u = 30$ and $s = \text{No}$. Therefore, we have $d_1 = d_2 = 1$, equal by chance to B^1_{op}, and the response is $\text{HT} = \text{No}$.

We could check other bases, like

$$\left[\text{CS}_{[1,4,2,0,3]}, \text{HP}_{[1,0]}, \text{BD}_{[0,1]}, \text{HC}_{[0,1]}\right], \left[\text{CS}_{[2,1,4,0,3]}, \text{HP}_{[1,0]}, \text{BD}_{[0,1]}, \text{HC}_{[0,1]}\right],$$
$$\left[\text{CS}_{[0,1,2,4,3]}, \text{HP}_{[1,0]}, \text{BD}_{[0,1]}, \text{HC}_{[0,1]}\right], \left[\text{CS}_{[3,1,2,0,4]}, \text{HP}_{[1,0]}, \text{BD}_{[0,1]}, \text{HC}_{[0,1]}\right],$$
$$\left[\text{CS}_{[3,0,2,1,4]}, \text{HP}_{[1,0]}, \text{BD}_{[0,1]}, \text{HC}_{[0,1]}\right], \left[\text{HP}_{[0,1]}, \text{CS}_{[4,1,2,0,3]}, \text{BD}_{[0,1]}, \text{HC}_{[0,1]}\right],$$
$$\left[\text{HP}_{[0,1]}, \text{CS}_{[3,0,2,1,4]}, \text{BD}_{[0,1]}, \text{HC}_{[0,1]}\right], \ldots$$

This process can confirm the response value. For instance, we can check an open query on N bases close to the operative base. If N_k bases suggest that the response is s_k, then the DSS gives us a probability N_k/N for s_k, and we can either choose the mode of the distribution of the different responses or solve the ambiguity as in situation (iii) using Algorithm $\mathcal{A}1$, see Section 4.3. □

Table 7
KBM2L list using operative base B^1_{op}. No cases such that $\text{HP} = \text{Absent}$ and $\text{BD} = \text{Yes}$ have been evaluated (unknown responses, denoted as -1).

Item	Description in operative base B^1_{op}					Offset
	HP	CS	BD	HC	HT	
1	⟨ (Present,	IV,	Yes,	Low.Grade),	Yes |	0
2	⟨ (Present,	IV,	Yes,	High.Grade),	No |	1
3	⟨ (Present,	IV,	No,	Low.Grade),	Yes |	2
4	⟨ (Present,	IV,	No,	High.Grade),	No |	3
5	⟨ (Present,	II1,	Yes,	Low.Grade),	Yes |	4
6	⟨ (Present,	II1,	Yes,	High.Grade),	No |	5
7	⟨ (Present,	II1,	No,	Low.Grade),	Yes |	6
8	⟨ (Present,	II1,	No,	High.Grade),	No |	7
9	⟨ (Present,	II2,	Yes,	Low.Grade),	Yes |	8
10	⟨ (Present,	II2,	Yes,	High.Grade),	No |	9
11	⟨ (Present,	II2,	No,	Low.Grade),	Yes |	10
12	⟨ (Present,	II2,	No,	High.Grade),	No |	11
13	⟨ (Present,	I,	Yes,	Low.Grade),	Yes |	12
14	⟨ (Present,	I,	Yes,	High.Grade),	No |	13
15	⟨ (Present,	I,	No,	Low.Grade),	Yes |	14
16	⟨ (Present,	III,	No,	High.Grade),	No |	19
17	⟨ (Absent,	IV,	Yes,	High.Grade),	−1 |	21
18	⟨ (Absent,	IV,	No,	High.Grade),	No |	23
19	⟨ (Absent,	II1,	Yes,	High.Grade),	−1 |	25
20	⟨ (Absent,	II1,	No,	High.Grade),	No |	27
21	⟨ (Absent,	II2,	Yes,	High.Grade),	−1 |	29
22	⟨ (Absent,	II2,	No,	High.Grade),	No |	31
23	⟨ (Absent,	I,	Yes,	High.Grade),	−1 |	33
24	⟨ (Absent,	I,	No,	High.Grade),	No |	35
25	⟨ (Absent,	I,	Yes,	High.Grade),	−1 |	37
26	⟨ (Absent,	III,	No,	High.Grade),	No |	39

Table 8
KBM2L list using operative base B^2_{op}. No cases such that HP = Absent and BD = Yes have been evaluated (unknown responses, denoted as −1).

Item	Description in operative base B^2_{op}					Offset
	CS	HP	BD	HC	HT	
1:20	…					
21	⟨ (I,	Present,	No,	High.Grade),	No |	27
22	⟨ (I,	Absent,	Yes,	High.Grade),	−1 |	29
23	⟨ (III,	Present,	No,	High.Grade),	No |	35
24:25	…					

Finally, the first and last items of the list are two special cases in inference terms. In those cases, not all of the offsets q^l or q^u are defined. Then the best strategy is to evaluate the model because the list does not contain enough information to support queries.

4.3. The KBM2L knows the response, but the response is ambiguous. A dialogue between DSS and expert

Situation (iii) presents different policy values in S. We may give an immediate answer to the expert based on statistics over the policy value distribution (median, mode, etc.). However, more intelligent responses will be preferred. As a first proposal, say Algorithm $\mathcal{A}1$, the DSS asks the expert to instantiate the open attribute further to the left with respect to the optimal base. It will be the most efficient attribute for reducing response uncertainty. That is, it will have the greatest weight of all the open attributes, meaning that it is more likely to belong to the fixed part of the item indices, which is what explains a fixed policy. Thus, the further to the left the attribute is, the more likely the query is to lead to fewer different responses. If necessary, the DSS would prompt for the open attribute second furthest to the left and so on. This is a sound approach for problems with many attributes but with *few* open attributes.

Example. Open query and different responses (I).

The KBM2L list containing 5 items in the base $[HP_{[0,1]}, HC_{[0,1]}, CS_{[0,1,2,3,4]}, BD_{[0,1]}]$ (see Table 3) illustrates Algorithm $\mathcal{A}1$ (Section 4.3). The dialogue is driven by the following steps, where the above base is used to provide answers. Table 9 summarizes the dialogue. □

For *many* open attributes, say more than 10, we have enough information to make automatic inferences via a learning process. Thus, we propose focusing once again on the operative KBM2L, since it is easier to retrieve responses and learn the probabilistic relationships among the attributes *and* the policy from the records of interest. The structure to be learned is a Bayesian network (BN) (see e.g. [16]), as it has a clear semantics for performing many inference tasks. Then, the resulting structure will provide a basis for starting a DSS/expert dialogue to lend insight into the problem and refine the query in the light of new evidence until the response satisfies the expert.

For the sake of simplicity, let $\mathbf{X}\subset\mathcal{R}^{n_1}, \mathbf{Y}\subset\mathcal{R}^{n_2}, n_1 + n_2 = n$ denote, respectively, instantiated and non-instantiated attributes of the open query. Our Algorithm $\mathcal{A}2$ includes the following steps.

Algorithm $\mathcal{A}2$.

S0. Initialize $\mathbf{X}^0 = \mathbf{X}, \mathbf{Y}^0 = \mathbf{Y}$.
S1. DSS extracts data (records) matching $\mathbf{X}^0$ from the operative KBM2L.
S2. DSS learns a BN from data (structure and conditional probabilities).
S3. DSS computes $P\left(R = r|\mathbf{X}^0\right), \forall r\in\Omega_R$ on the BN. From this probability distribution, the expert fixes a decision criterion, usually the mode, to choose among rs. Let m^0 be this value. This is evidence to be propagated through the network.
S4. DSS computes $P\left(Y_j = y_t|R = m^0, \mathbf{X}^0\right), \forall j = 1, \ldots, n_2, \forall y_t\in\Omega_{Y_j}$ on the BN. From these probability distributions, the expert fixes a decision criterion, usually the distribution of minimum variance, to choose among Y_js. Let $\tilde{\mathbf{Y}}^0$ be the resulting vector of Y_js, with coordinates given by expert instantiations, like, e.g., the Y_j mode.
S5. Extend vector $\mathbf{X}^0$ as $\mathbf{X}^1 = \mathbf{X}^0\cup\tilde{\mathbf{Y}}^0$. Set $\mathbf{Y}^1 = \mathbf{Y}^0\backslash\tilde{\mathbf{Y}}^0$.

Table 9
Dialogue driven by Algorithm $\mathcal{A}1$ (Section 4.3). Query $\mathbf{OQ}_2$ is open, specifying only a medium clinical stage (CS = II2) for the patient.

	Expert	DSS
1	States a query $\mathbf{OQ}_2$ about HT (No/Yes?) when CLINICAL-STAGE (CS) = II2	Answers with the (ambiguous) response {No,Yes}
2	Wants a more definite response	Suggests to know the two most important attributes according to the optimal base B^*, i.e. HP and HC
3	Specifies her most preferred, attribute, HP, and formulates a second query that extends $\mathbf{OQ}_2$ with HP = Present	Answers {No,Yes} again, and the dialogue continues
4	States a third query, which extends $\mathbf{OQ}_2$ with HC = High.Grade	Answers {No} and the dialogue finishes

Steps *S3* and *S4* are repeated until the expert is satisfied or $\mathbf{Y}^j$ has few components, and Algorithm $\mathcal{A}1$ (Section 4.3) is called to continue. If the algorithm stops at $\mathbf{X}^j$, then m^j is the response returned. Experts can always revise decisions made at *S3* and *S4* of the last iteration whenever they do not agree with the current outputs. The DSS will be on the lookout for and warn experts about probabilities conditioned to impossible events (registered in the DSS).

Expert decision criteria at Steps *S3* and *S4* might be different. They are expert choices. Using the criteria suggested earlier: (a) at Step *S3*, we choose the most likely response given the instantiated attribute set for the query; and (b) at Step *S4*, we choose the attribute(s) in which we have more confidence or the attribute(s) fluctuating less than a fixed threshold. Then, they are instantiated in the query as their mode, giving rise to a new, more accurate, query. Later steps allow continuous probability updating.

A BN is learned at Step *S2* via a structure learning algorithm [2], where the standard is the K2 algorithm [3]. K2 is a greedy parent search algorithm using a Bayesian score to rank different structures. Thus, nodes are assumed to be ordered and K2 returns the most probable parents for each node given the data. The algorithm works on quite reasonable data sizes, as in the context we propose. In fact, we have even tried with 20 open attributes involving one million cases.

DSS explanations are mediated by the BN and its probabilities, firstly giving information about the response and, secondly, about the likelihood of each open attribute, given the expert's chosen response. This support for both responses and queries allow experts to re-state and improve their query until it is better defined, leading to more accurate answers. This is task (E) mentioned in the Introduction.

Example. Open query and different responses (II).

For this scenario we use a more complex decision table than in the previous examples because Algorithm $\mathcal{A}1$ (Section 4.3) will be good enough to support the open query on a small table, like the HT table. So, we use the S table about the decision on surgery in the NHL problem (i.e. the second decision to be made). The open query is $\mathbf{OQ}_1$ again:

$$\text{HP} = \text{Absent},\ \text{BD} = \text{Yes and CS} = \text{I}.$$

The S table consists of seven attributes: the HT table attributes that we used before, plus decision HT and the random variables GHS and CP. Therefore, the non-instantiated attributes in this open query are GHS, HT, HC, and CP. The initial base is $B_{ini} = [\text{GHS}_{[0,1,2]}, \text{HT}_{[0,1]}, \text{CS}_{[0,1,2,3,4]}, \text{BD}_{[0,1]}, \text{HC}_{[0,1]}, \text{HP}_{[0,1]}, \text{CP}_{[0,1,2,3]}]$, and the KBM2L list returns 385 items with 960 cases and 2 alternatives, S = None and S = Curative. The optimal base is $B_{fin} = [\text{GHS}_{[0,1,2]}, \text{CP}_{[0,1,2,3]}, \text{CS}_{[0,1,2,3,4]}, \text{BD}_{[0,1]}, \text{HC}_{[0,1]}, \text{HP}_{[0,1]}, \text{HT}_{[0,1]}]$, and the KBM2L shows 27 items.

All the responses to the query are known but they are different, and this does not satisfy the expert. Therefore, Algorithm $\mathcal{A}2$ is applied. We choose any operative base, i.e. a base having the non-instantiated attributes moved to the right.

First, a BN model is learnt from the data involved in the query, i.e. 97 cases (Steps *S1* and *S2*). We have used the GeNIe free software, available at http://genie.sis.pitt.edu/. The model is shown in Fig. 2, showing the dependencies among variables. Note that five variables (CS, BD, HP, HC, and HT) are not connected to the S (Surgery) variable. Thus, these variables are irrelevant to the query. At this point, $\mathbf{X}^0 = (\text{HP, BD, CS})$, instantiated as the query indicates, and $P\left(s|\mathbf{X}^0\right)$ is the probability shown in Fig. 2 after propagating the evidence given by $\mathbf{X}^0$. This probability distribution is 0.5827 and 0.4173 for the values S = None and S = Curative, respectively. We are already at Step *S3*, and S = None is the mode of the previous distribution, which is chosen as m^0.

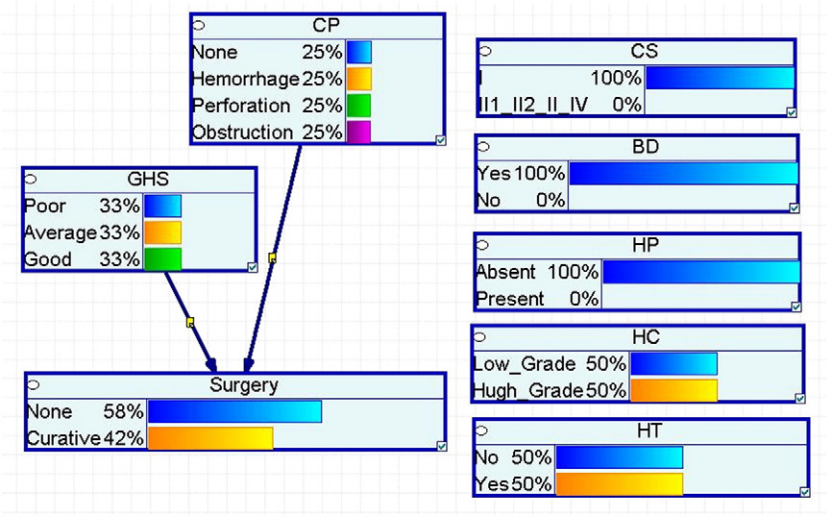

Fig. 2. Bayesian network for the S (Surgery) decision for the treatment of gastric NHL. Posterior distributions of each node given $\mathbf{OQ}_1$, i.e. HP = Absent, BD = Yes and CS = I. This corresponds with Steps *S1*, *S2* and *S3* of Algorithm $\mathcal{A}2$.

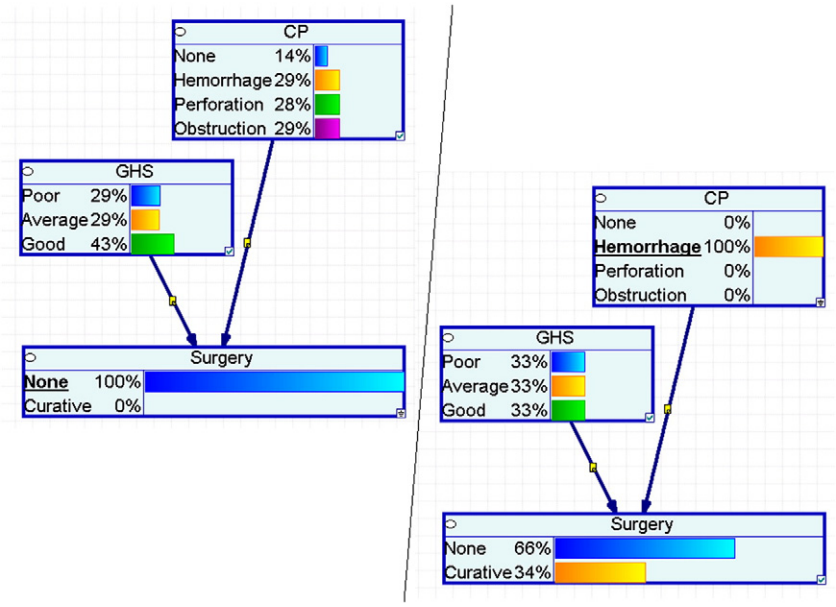

Fig. 3. Bayesian networks for the S (Surgery) decision for the treatment of gastric NHL. Left: Step *S4* of Algorithm $\mathcal{A}2$, with posterior distributions of CP and GHS after adding S = None as evidence. Right: Step *S5* extends the evidence with CP = Hemorrhage and a new cycle starts. Step *S3* computes posterior distributions of S and GHS given this new evidence.

At Step *S4* we propagate this new value as evidence, computing the probability of each attribute given m^0 and $\mathbf{X}^0$. The result is shown in Fig. 3 (left-hand side). The probability distributions are updated from (1/3, 1/3, 1/3) for GHS and (1/4, 1/4, 1/4, 1/4) for CP to (0.2870, 0.2856, 0.4274) for GHS and (0.1440, 0.2857, 0.2848, 0.2855) for CP. The expert chooses variable CP with variance 0.0049, smaller than the variance of GHS which is 0.0066. Its mode is CP = Hemorrhage.

At Step *S5*, vector $\mathbf{X}^0$ is extended with a new dimension given by the CP attribute. At this point, $\mathbf{X}^1 = (\text{HP}, \text{BD}, \text{CS}, \text{CP})$, instantiated as mentioned earlier, and a new cycle starts. $P\left(s|\mathbf{X}^1\right)$ is the probability shown in Fig. 3 (right-hand side) after propagating the evidence given by $\mathbf{X}^1$. This probability distribution is 0.6648 and 0.3352 for the values S = None and S = Curative, respectively. S = None is the mode of the previous distribution, which is chosen as m^1.

We propagate this new value as evidence, computing the probability of GHS given m^1 and $\mathbf{X}^1$. The result is shown in Fig. 4 (left-hand side). The probability distribution is updated as (0.0012, 0.4994, 0.4994) for the values GHS = Poor, GHS = Average and GHS = Good, respectively.

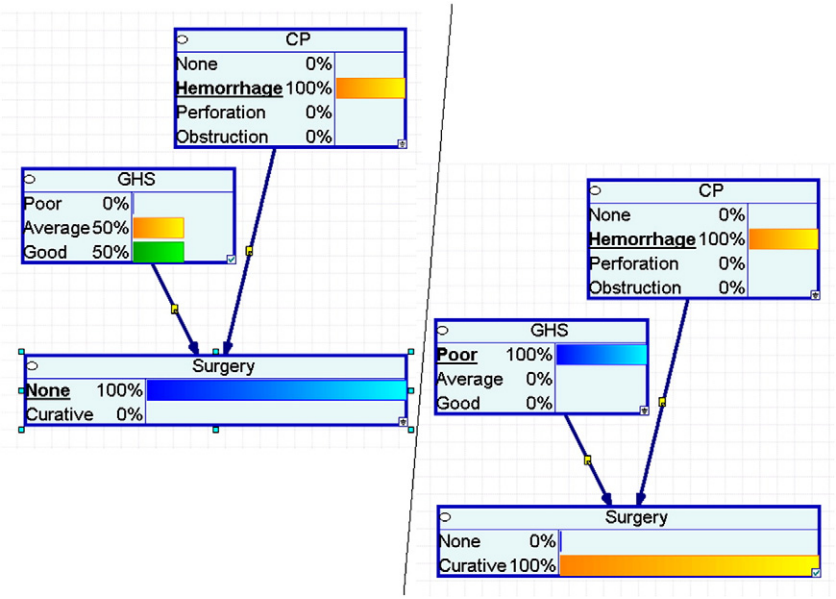

Fig. 4. Bayesian networks for the S (Surgery) decision for the treatment of gastric NHL. Left: Step *S4* of Algorithm $\mathcal{A}2$, after adding S = None as evidence. Right: Step *S5* extends the evidence with GHS = Poor, and the posterior distribution of S given this evidence is shown.

Vector $\mathbf{X}^1$ is extended with a new dimension given by the GHS attribute, where the result is $\mathbf{X}^2 = (\text{HP}, \text{BD}, \text{CS}, \text{CP}, \text{GHS})$. The dialogue continues asking the expert whether she believes that the patient has GHS = Poor. In this case, the network suggests a curative surgery (with probability 0.9976), see Fig. 4 (right-hand side). If, instead, the expert had chosen the mode of GHS, i.e. GHS = Average and GHS = Good, then the network would suggest that surgery should not be performed (not shown).

In short, when HP = Absent, BD = Yes and CS = I, the DSS suggests that surgery should not be performed unless GHS = Poor, where curative surgery would be mandatory. This is what we learn from this dialog. □

4.4. The KBM2L faces uncertain and ambiguous responses

Situation (iv) also presents different response values in S, some of which may, however, be unknown. Unknown policies are the result of having a system that can legitimately be termed knowledge based [10] due to its significant size and is impossible to solve completely. Experts play an important role in deciding which part of the problem should be solved, stating where their interest lies.

Situation (iv) can be solved by reducing it to situations (ii) and (iii). We split the problem into these two parts and apply the procedures mentioned earlier. The outcome could possibly be situation (iii) again, if the response to situation (ii) is different from the response to situation (iii). Thus, the procedures to solve situation (iii) would be reiterated, i.e. Algorithms $\mathcal{A}1$ (Section 4.3) and $\mathcal{A}2$, would be applied.

5. Conclusions and further research

A decision model builds on guidelines, probabilities, utilities, probabilistic relationships, among other sources of information. Decision tables are the result of evaluating a decision model, taking into account that information. Their extraordinarily large size motivated us to analyse them. The aim was to save memory space and, more interestingly, retrieve knowledge (to understand DSS suggestions). In our previous paper we managed to achieve both aims. Moreover, by analysing the items—groups of cases from the decision tables with the same optimal alternative, we could get an explanation of that optimal alternative, the similarity among several alternatives, attribute relevance… Therefore, KBM2L lists managed to solve space savings, optimization and explanations of decision tables.

In this paper, the KBM2L framework is extended to deal with queries, one of the most important DSS facilities. The user enters any query into the DSS, assuming that the model (influence diagram) has been validated. This is more than a simple database query, which matches a single rule. Nowadays, influence diagrams do not have any such facility, and users have to deal with huge decision tables from which it is almost impossible to extract useful/concise information.

Open queries and situations with imprecise/uncertain responses are specially difficult. These are the cases that we solve in this paper. The focus is on guiding the user toward what variable to query to get a more accurate and convincing response. The optimal and operative bases allow the records involved in a query to be organised from different perspectives. General queries leading to imprecise responses are addressed via an attributes–policy relationship learning process, where interaction with experts is required to arrive at a satisfactory response.

Our approach provides the KB definite exploitation for the DSS. As opposed to only listing the influence diagram outputs, we report on improvements in space savings, knowledge extraction as rules, explanations of optimal policies and satisfactory answers to complex queries.

Despite the power of our iterative scheme of progressive knowledge elicitation, a possible field of future research would focus on enabling queries with constrained rather than non-instantiated attributes, covering initial beliefs about the attributes. Also, more effort could be employed in determining good operative bases if there is more than one. Two criteria could be: minimum computational effort to output the new KBM2L and minimum item fragmentation. Finally, rather than directly allowing the expert to choose a decision criterion in Algorithm $\mathcal{A}2$, we could first implement a search within the tree of possible sessions, i.e. possible $r-y-r-y\cdots$ (response r and instantiated attributes y) sequences. This would filter out possibilities that will not satisfy expert expectations, facilitating choices.

Acknowledgments

Research partially supported by grants from the Spanish Ministry of Science and Innovation (TIN2007-62626 and Consolider Ingenio 2010-CSD2007-00018). Thanks to Peter Lucas for valuable support with the medical problem. We are also grateful to the referees for their valuable remarks that have definitely helped to improve the manuscript.

References

[1] C. Bielza, J.A. Fernández del Pozo, P.J.F. Lucas, Explaining clinical decisions by extracting regularity patterns, Decision Support Systems 44 (2008) 397–408.
[2] W.L. Buntine, A guide to the literature on learning probabilistic networks from data, IEEE Transactions on Knowledge and Data Engineering 8 (1996) 195–210.
[3] G.F. Cooper, E. Herskovits, A Bayesian method for the induction of probabilistic networks from data, Machine Learning 9 (1992) 309–347.
[4] G. Cortazar, E.S. Schwartz, M. Salinas, J.E. Smith, S. Axsater, K.J. Ezawa, Evidence propagation and value of evidence on influence diagrams, Operations Research 46 (1998) 73–83.
[5] S. Eidt, M. Stolte, R. Fischer, *Helicobacter pylori* gastritis and primary gastric non-Hodgkin's lymphomas, Journal of Clinical Pathology 47 (1994) 436–439.

[6] J.A. Fernández del Pozo, C. Bielza, M. Gómez, A list-based compact representation for large decision tables management, European Journal of Operational Research 160 (2005) 638–662.
[7] M.R. Garey, D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman and Company, New York, 1979.
[8] 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.
[9] M.S. Gönül, D. Önkal, M. Lawrence, The effects of structural characteristics of explanations on use of a DSS, Decision Support Systems 42 (2006) 1481–1493.
[10] M. Henrion, J.S. Breese, E.J. Horvitz, Decision analysis and expert systems, Artificial Intelligence Magazine (1991) 64–91.
[11] D.E. Knuth, The Art of Computer Programming, Vol. 1: Fundamental Algorithms, Addison-Wesley, Reading, 1968.
[12] C. Lacave, F.J. Díez, A review of explanation methods for heuristic expert systems, Knowledge Engineering Review 19 (2004) 133–146.
[13] 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 37 (2007) 952–965.
[14] P. Lucas, H. Boot, B. Taal, Computer-based decision-support in the management of primary gastric non-Hodgkin lymphoma, Methods of Information in Medicine 37 (1998) 206–219.
[15] C. Martinez, A. Panholzer, H. Prodinger, Partial match queries in relaxed multidimensional search trees, Algorithmica 29 (2001) 181–204.
[16] J. Pearl, Probabilistic Reasoning in Intelligent Systems, Morgan Kaufmann, San Mateo, CA, 1988.
[17] R.D. Shachter, Evaluating influence diagrams, Operations Research 34 (1986) 871–882.
[18] R.N. Shiffman, Representation of clinical practice guidelines in conventional and augmented decision tables, Journal of the American Medical Informatics Association 4 (1997) 382–393.
[19] R.N. Shiffman, R.A. Greenes, Improving clinical guidelines with logic and decision table techniques: application to hepatitis immunization recommendations, Medical Decision Making 14 (1994) 245–254.
[20] N. Søndberg-Jeppesen, F.V. Jensen, A PGM framework for recursive modeling of players in simple sequential Bayesian games, International Journal of Approximate Reasoning 51 (2010) 587–599.
[21] B. Yu, T. Bailey, Processing partially specified queries over high-dimensional databases, Data & Knowledge Engineering 62 (2007) 177–197.

Concha Bielza received the M.S. degree in mathematics from Complutense University of Madrid, Madrid, Spain, in 1989 and the Ph.D. degree in computer science from the Universidad Politécnica de Madrid, Madrid, in 1996. She is currently a Full Professor of statistics and operations research with the Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid. Her research interests are primarily in the areas of probabilistic graphical models, decision analysis, metaheuristics for optimization, data mining, classification models, and real applications, like biomedicine, bioinformatics and neuroscience. Her research has appeared in journals like Management Science, Computers and Operations Research, Statistics and Computing, the European Journal of Operational Research, Decision-Support Systems, Naval Research Logistics, the Journal of the Operational Research Society, Medical Decision Making, Methods of Information in Medicine, IEEE Transactions on SMC, International Journal of Systems Science, Bioinformatics, Briefings in Bioinformatics, Journal of Statistical Software, Journal of Heuristics, Intelligent Data Analysis, Developmental Neurobiology, Neuroinformatics, IEEE Transactions on Signal Processing, and Expert Systems with Applications as well as chapters of many books.

Juan A. Fernandez del Pozo received his MS degree in Computer Science in 1999 and PhD in Computer Science in 2006 from Universidad Politécnica de Madrid (UPM), Madrid (Spain). He is currently Associate Professor of Statistics and Operations Research at School of Computer Science and member of the Computational Intelligence Group at the UPM. His research interest includes decision analysis and intelligent decision-support systems based on influence diagrams and Bayesian networks that perform knowledge acquisition in huge decision tables, knowledge discovery and data mining on models' outputs for explanation synthesis and sensitivity analysis. He is also interested in optimization based on evolutionary algorithm and classification models. He is collaborating with several Spanish Foundations in modeling the service quality and life quality on social service environments. His articles have appeared in various academic journals including: Springer Lecture Notes in Computer Science, Journal of Operational Research, Computers & Operations Research, Decision-Support Systems, Expert Systems with Applications, Medical Decision Making. His teaching interests include Statistics, Decision-Support Systems and Operations Research.